# C8051F85x/86x

**Low-Cost 8-bit MCU Family with up to 8 kB of Flash**

## Memory
- Up to 8 kB flash
- Flash is in-system programmable in 512-Byte sectors
- Up to 512 Bytes RAM (256 + 256)

## 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

## 12-Bit Analog-to-Digital Converter
- Up to 16 input channels
- Up to 200 ksps 12-bit mode or 800 ksps 10-bit mode
- Internal VREF or external VREF supported

## Internal Low-Power Oscillator
- Calibrated to 24.5 MHz
- Low supply current
- ±2% accuracy over supply and temperature

## Internal Low-Frequency Oscillator
- 80 kHz nominal operation
- Low supply current
- Independent clock source for watchdog timer

## 2 Analog Comparators
- Programmable hysteresis and response time
- Configurable as interrupt or reset source
- Low current

## General-Purpose I/O
- Up to 18 pins
- 5 V-Tolerant
- Crossbar-enabled

## High-Speed CIP-51 µC Core
- Efficient, pipelined instruction architecture
- Up to 25 MIPS throughput with 25 MHz clock
- Uses standard 8051 instruction set
- Expanded interrupt handler

## Communication Peripherals
- UART
- $I^2C$ / SMBus™
- SPI™

## Timer/Counters and PWM
- 4 General-Purpose 16-bit Timer/Counters
- 16-bit Programmable Counter Array (PCA) with three channels of PWM, capture/compare, or frequency output capability, and hardware kill/safe state capability

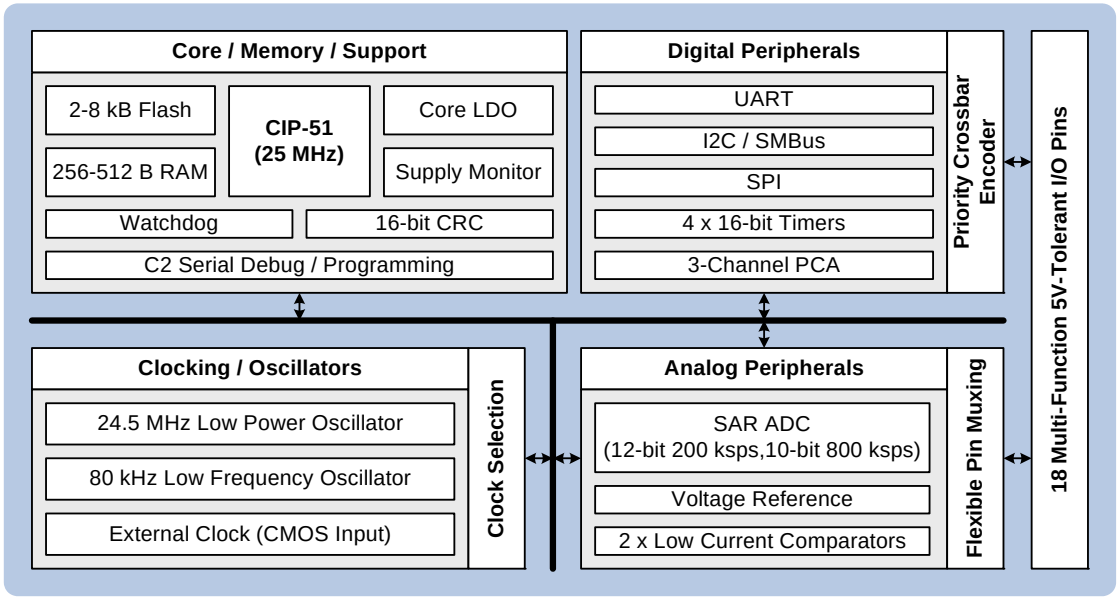
## Additional Support Peripherals
- Independent watchdog timer clocked from LFO
- 16-bit CRC engine

## Unique Identifier
- 32-bit unique key for each device

## Supply Voltage
- 2.2 to 3.6 V

## Package Options
- 16-pin SOIC
- 20-pin QFN, 3 x 3 mm
- 24-pin QSOP
- Available in die form
- Qualified to AEC-Q100 Standards

## Temperature Ranges:
- –40 to +125 °C (-Ix) and –40 to +85 °C (-Gx)





Copyright © 2018 by Silicon Laboratories

## Table of Contents

**1. Electrical Specifications ..... 8**
1.1. Electrical Characteristics ..... 8
1.2. Typical Performance Curves ..... 19
1.2.1. Operating Supply Current ..... 19
1.2.2. ADC Supply Current ..... 20
1.2.3. Port I/O Output Drive ..... 21
1.3. Thermal Conditions ..... 21
1.4. Absolute Maximum Ratings ..... 22
**2. System Overview ..... 23**
2.1. Power ..... 25
2.1.1. LDO ..... 25
2.1.2. Voltage Supply Monitor (VMON0) ..... 25
2.1.3. Device Power Modes ..... 25
2.2. I/O ..... 26
2.2.1. General Features ..... 26
2.2.2. Crossbar ..... 26
2.3. Clocking ..... 27
2.4. Counters/Timers and PWM ..... 27
2.4.1. Programmable Counter Array (PCA0) ..... 27
2.4.2. Timers (Timer 0, Timer 1, Timer 2 and Timer 3) ..... 27
2.4.3. Watchdog Timer (WDT0) ..... 27
2.5. Communications and other Digital Peripherals ..... 28
2.5.1. Universal Asynchronous Receiver/Transmitter (UART0) ..... 28
2.5.2. Serial Peripheral Interface (SPI0) ..... 28
2.5.3. System Management Bus / I2C (SMBus0) ..... 28
2.5.4. 16/32-bit CRC (CRC0) ..... 28
2.6. Analog Peripherals ..... 30
2.6.1. 12-Bit Analog-to-Digital Converter (ADC0) ..... 30
2.6.2. Low Current Comparators (CMP0, CMP1) ..... 30
2.7. Reset Sources ..... 31
2.8. On-Chip Debugging ..... 31
**3. Pin Definitions ..... 32**
3.1. C8051F850/1/2/3/4/5 QSOP24 Pin Definitions ..... 32
3.2. C8051F850/1/2/3/4/5 QFN20 Pin Definitions ..... 36
3.3. C8051F860/1/2/3/4/5 SOIC16 Pin Definitions ..... 39
**4. Ordering Information ..... 42**
**5. QSOP-24 Package Specifications ..... 45**
**6. QFN-20 Package Specifications ..... 47**
**7. SOIC-16 Package Specifications ..... 50**
**8. Memory Organization ..... 52**
8.1. Program Memory ..... 53
8.1.1. MOVX Instruction and Program Memory ..... 53
8.2. Data Memory ..... 53

8.2.1. Internal RAM ..... 53
8.2.2. External RAM ..... 54
8.2.3. Special Function Registers ..... 55
**9. Special Function Register Memory Map ..... 56**
**10. Flash Memory ..... 61**
10.1. Security Options ..... 61
10.2. Programming the Flash Memory ..... 63
10.2.1. Flash Lock and Key Functions ..... 63
10.2.2. Flash Erase Procedure ..... 63
10.2.3. Flash Write Procedure ..... 63
10.3. Non-Volatile Data Storage ..... 64
10.4. Flash Write and Erase Guidelines ..... 64
10.4.1. Voltage Supply Maintenance and the Supply Monitor ..... 64
10.4.2. PSWE Maintenance ..... 65
10.4.3. System Clock ..... 65
10.5. Flash Control Registers ..... 66
**11. Device Identification and Unique Identifier ..... 68**
11.1. Device Identification Registers ..... 69
**12. Interrupts ..... 72**
12.1. MCU Interrupt Sources and Vectors ..... 72
12.1.1. Interrupt Priorities ..... 72
12.1.2. Interrupt Latency ..... 72
12.2. Interrupt Control Registers ..... 75
**13. Power Management and Internal Regulator ..... 82**
13.1. Power Modes ..... 82
13.1.1. Idle Mode ..... 82
13.1.2. Stop Mode ..... 83
13.2. LDO Regulator ..... 83
13.3. Power Control Registers ..... 83
13.4. LDO Control Registers ..... 84
**14. Analog-to-Digital Converter (ADC0) ..... 85**
14.1. ADC0 Analog Multiplexer ..... 86
14.2. ADC Operation ..... 88
14.2.1. Starting a Conversion ..... 88
14.2.2. Tracking Modes ..... 88
14.2.3. Burst Mode ..... 89
14.2.4. Settling Time Requirements ..... 90
14.2.5. Gain Setting ..... 91
14.3. 8-Bit Mode ..... 91
14.4. 12-Bit Mode ..... 91
14.5. Power Considerations ..... 92
14.6. Output Code Formatting ..... 94
14.7. Programmable Window Detector ..... 95
14.7.1. Window Detector In Single-Ended Mode ..... 95
14.8. Voltage and Ground Reference Options ..... 97

14.8.1. External Voltage Reference ..... 97
14.8.2. Internal Voltage Reference ..... 97
14.8.3. Analog Ground Reference ..... 97
14.9. Temperature Sensor ..... 98
14.9.1. Calibration ..... 98
14.10. ADC Control Registers ..... 99
**15. CIP-51 Microcontroller Core ..... 113**
15.1. Performance ..... 113
15.2. Programming and Debugging Support ..... 114
15.3. Instruction Set ..... 114
15.3.1. Instruction and CPU Timing ..... 114
15.4. CPU Core Registers ..... 119
**16. Clock Sources and Selection (HFOSC0, LFOSC0, and EXTCLK) ..... 125**
16.1. Programmable High-Frequency Oscillator ..... 125
16.2. Programmable Low-Frequency Oscillator ..... 125
16.2.1. Calibrating the Internal L-F Oscillator ..... 125
16.3. External Clock ..... 126
16.4. Clock Selection ..... 126
16.5. High Frequency Oscillator Control Registers ..... 127
16.6. Low Frequency Oscillator Control Registers ..... 128
16.7. Clock Selection Control Registers ..... 129
**17. Comparators (CMP0 and CMP1) ..... 130**
17.1. System Connectivity ..... 130
17.2. Functional Description ..... 133
17.3. Comparator Control Registers ..... 134
**18. Cyclic Redundancy Check Unit (CRC0) ..... 140**
18.1. CRC Algorithm ..... 140
18.2. Preparing for a CRC Calculation ..... 142
18.3. Performing a CRC Calculation ..... 142
18.4. Accessing the CRC0 Result ..... 142
18.5. CRC0 Bit Reverse Feature ..... 142
18.6. CRC Control Registers ..... 143
**19. External Interrupts (INT0 and INT1) ..... 149**
19.1. External Interrupt Control Registers ..... 150
**20. Programmable Counter Array (PCA0) ..... 152**
20.1. PCA Counter/Timer ..... 153
20.2. PCA0 Interrupt Sources ..... 153
20.3. Capture/Compare Modules ..... 154
20.3.1. Output Polarity ..... 154
20.3.2. Edge-Triggered Capture Mode ..... 155
20.3.3. Software Timer (Compare) Mode ..... 156
20.3.4. High-Speed Output Mode ..... 157
20.3.5. Frequency Output Mode ..... 158
20.4. PWM Waveform Generation ..... 159
20.4.1. Edge Aligned PWM ..... 159

20.4.2. Center Aligned PWM ..... 161
20.4.3. 8 to11-bit Pulse Width Modulator Modes ..... 163
20.4.4. 16-Bit Pulse Width Modulator Mode ..... 164
20.5. Comparator Clear Function ..... 165
20.6. PCA Control Registers ..... 167
**21. Port I/O (Port 0, Port 1, Port 2, Crossbar, and Port Match) ..... 184**
21.1. General Port I/O Initialization ..... 185
21.2. Assigning Port I/O Pins to Analog and Digital Functions ..... 186
21.2.1. Assigning Port I/O Pins to Analog Functions ..... 186
21.2.2. Assigning Port I/O Pins to Digital Functions ..... 186
21.2.3. Assigning Port I/O Pins to Fixed Digital Functions ..... 187
21.3. Priority Crossbar Decoder ..... 188
21.4. Port I/O Modes of Operation ..... 191
21.4.1. Configuring Port Pins For Analog Modes ..... 191
21.4.2. Configuring Port Pins For Digital Modes ..... 191
21.4.3. Port Drive Strength ..... 192
21.5. Port Match ..... 192
21.6. Direct Read/Write Access to Port I/O Pins ..... 192
21.7. Port I/O and Pin Configuration Control Registers ..... 193
**22. Reset Sources and Supply Monitor ..... 211**
22.1. Power-On Reset ..... 212
22.2. Power-Fail Reset / Supply Monitor ..... 213
22.3. Enabling the VDD Monitor ..... 213
22.4. External Reset ..... 214
22.5. Missing Clock Detector Reset ..... 214
22.6. Comparator0 Reset ..... 214
22.7. Watchdog Timer Reset ..... 214
22.8. Flash Error Reset ..... 214
22.9. Software Reset ..... 214
22.10. Reset Sources Control Registers ..... 215
22.11. Supply Monitor Control Registers ..... 216
**23. Serial Peripheral Interface (SPI0) ..... 217**
23.1. Signal Descriptions ..... 218
23.1.1. Master Out, Slave In (MOSI) ..... 218
23.1.2. Master In, Slave Out (MISO) ..... 218
23.1.3. Serial Clock (SCK) ..... 218
23.1.4. Slave Select (NSS) ..... 218
23.2. SPI0 Master Mode Operation ..... 219
23.3. SPI0 Slave Mode Operation ..... 221
23.4. SPI0 Interrupt Sources ..... 221
23.5. Serial Clock Phase and Polarity ..... 221
23.6. SPI Special Function Registers ..... 223
23.7. SPI Control Registers ..... 227
**24. System Management Bus / I2C (SMBus0) ..... 233**
24.1. Supporting Documents ..... 234

24.2. SMBus Configuration ..... 234
24.3. SMBus Operation ..... 234
24.3.1. Transmitter vs. Receiver ..... 235
24.3.2. Arbitration ..... 235
24.3.3. Clock Low Extension ..... 235
24.3.4. SCL Low Timeout ..... 235
24.3.5. SCL High (SMBus Free) Timeout ..... 236
24.4. Using the SMBus ..... 236
24.4.1. SMBus Configuration Register ..... 236
24.4.2. SMBus Pin Swap ..... 238
24.4.3. SMBus Timing Control ..... 238
24.4.4. SMB0CN Control Register ..... 238
24.4.5. Hardware Slave Address Recognition ..... 240
24.4.6. Data Register ..... 241
24.5. SMBus Transfer Modes ..... 242
24.5.1. Write Sequence (Master) ..... 242
24.5.2. Read Sequence (Master) ..... 243
24.5.3. Write Sequence (Slave) ..... 244
24.5.4. Read Sequence (Slave) ..... 245
24.6. SMBus Status Decoding ..... 245
24.7. I2C / SMBus Control Registers ..... 251
**25. Timers (Timer0, Timer1, Timer2 and Timer3) ..... 259**
25.1. Timer 0 and Timer 1 ..... 261
25.1.1. Mode 0: 13-bit Counter/Timer ..... 262
25.1.2. Mode 1: 16-bit Counter/Timer ..... 263
25.1.3. Mode 2: 8-bit Counter/Timer with Auto-Reload ..... 264
25.1.4. Mode 3: Two 8-bit Counter/Timers (Timer 0 Only) ..... 265
25.2. Timer 2 and Timer 3 ..... 266
25.2.1. 16-bit Timer with Auto-Reload ..... 266
25.2.2. 8-bit Timers with Auto-Reload ..... 267
25.2.3. Capture Mode ..... 268
25.3. Timer Control Registers ..... 269
**26. Universal Asynchronous Receiver/Transmitter (UART0) ..... 289**
26.1. Enhanced Baud Rate Generation ..... 289
26.2. Operational Modes ..... 291
26.2.1. 8-Bit UART ..... 291
26.2.2. 9-Bit UART ..... 292
26.3. Multiprocessor Communications ..... 293
26.4. UART Control Registers ..... 295
**27. Watchdog Timer (WDT0) ..... 298**
27.1. Features ..... 298
27.2. Enabling / Resetting the WDT ..... 299
27.3. Disabling the WDT ..... 299
27.4. Disabling the WDT Lockout ..... 299
27.5. Setting the WDT Interval ..... 299

27.6. Synchronization ..... 300
27.7. Watchdog Timer Control Registers ..... 301
**28. Revision-Specific Behavior ..... 303**
28.1. Revision Identification ..... 303
28.2. Temperature Sensor Offset and Slope ..... 305
28.3. Flash Endurance ..... 305
28.4. Latch-Up Performance ..... 305
28.5. Unique Identifier ..... 305
**29. C2 Interface ..... 306**
29.1. C2 Pin Sharing ..... 306
29.2. C2 Interface Registers ..... 307
**Document Change List ..... 312**

# 1. Electrical Specifications

## 1.1. Electrical Characteristics

All electrical parameters in all tables are specified under the conditions listed in Table 1.1, unless stated otherwise.

**Table 1.1. Recommended Operating Conditions**

| Parameter | Symbol | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| Operating Supply Voltage on VDD | $V_{DD}$ | | 2.2 | — | 3.6 | V |
| System Clock Frequency | $f_{SYSCLK}$ | | 0 | — | 25 | MHz |
| Operating Ambient Temperature | $T_A$ | Commercial Grade Devices (-GM, -GS, -GU) | –40 | — | 85 | °C |
| | | Industrial Grade Devices (-IM, -IS, -IU) | –40 | — | 125 | °C |
| **Note:** All voltages with respect to GND | | | | | | |

**Table 1.2. Power Consumption**

| Parameter | Symbol | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| **Digital Core Supply Current (–Gx Devices, -40°C to +85°C)** | | | | | | |
| Normal Mode—Full speed with code executing from flash | $I_{DD}$ | $F_{SYSCLK}$ = 24.5 MHz[2] | — | 4.45 | 4.85 | mA |
| | | $F_{SYSCLK}$ = 1.53 MHz[2] | — | 915 | 1150 | µA |
| | | $F_{SYSCLK}$ = 80 kHz[3], $T_A$ = 25 °C | — | 250 | 290 | µA |
| | | $F_{SYSCLK}$ = 80 kHz[3] | — | 250 | 380 | µA |
| Idle Mode—Core halted with peripherals running | $I_{DD}$ | $F_{SYSCLK}$ = 24.5 MHz[2] | — | 2.05 | 2.3 | mA |
| | | $F_{SYSCLK}$ = 1.53 MHz[2] | — | 550 | 700 | µA |
| | | $F_{SYSCLK}$ = 80 kHz[3], $T_A$ = 25 °C | — | 125 | 130 | µA |
| | | $F_{SYSCLK}$ = 80 kHz[3] | — | 125 | 200 | µA |
| Stop Mode—Core halted and all clocks stopped, Supply monitor off. | $I_{DD}$ | Internal LDO ON, $T_A$ = 25 °C | — | 105 | 120 | µA |
| | | Internal LDO ON | — | 105 | 170 | µA |
| | | Internal LDO OFF | — | 0.2 | — | µA |

**Notes:**
1. Currents are additive. For example, where $I_{DD}$ is specified and the mode is not mutually exclusive, enabling the functions increases supply current by the specified amount.
2. Includes supply current from internal regulator, supply monitor, and High Frequency Oscillator.
3. Includes supply current from internal regulator, supply monitor, and Low Frequency Oscillator.
4. ADC0 always-on power excludes internal reference supply current.
5. The internal reference is enabled as-needed when operating the ADC in burst mode to save power.

**Table 1.2. Power Consumption (Continued)**

| Parameter | Symbol | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| **Digital Core Supply Current (–Ix Devices, -40°C to +125°C)** | | | | | | |
| Normal Mode—Full speed with code executing from flash | $I_{DD}$ | $F_{SYSCLK}$ = 24.5 MHz[2] | — | 4.45 | 5.25 | mA |
| | | $F_{SYSCLK}$ = 1.53 MHz[2] | — | 915 | 1600 | µA |
| | | $F_{SYSCLK}$ = 80 kHz[3], $T_A$ = 25 °C | — | 250 | 290 | µA |
| | | $F_{SYSCLK}$ = 80 kHz[3] | — | 250 | 725 | µA |
| Idle Mode—Core halted with peripherals running | $I_{DD}$ | $F_{SYSCLK}$ = 24.5 MHz[2] | — | 2.05 | 2.6 | mA |
| | | $F_{SYSCLK}$ = 1.53 MHz[2] | — | 550 | 1000 | µA |
| | | $F_{SYSCLK}$ = 80 kHz[3], $T_A$ = 25 °C | — | 125 | 130 | µA |
| | | $F_{SYSCLK}$ = 80 kHz[3] | — | 125 | 550 | µA |
| Stop Mode—Core halted and all clocks stopped, Supply monitor off. | $I_{DD}$ | Internal LDO ON, $T_A$ = 25 °C | — | 105 | 120 | µA |
| | | Internal LDO ON | — | 105 | 270 | µA |
| | | Internal LDO OFF | — | 0.2 | — | µA |
| **Analog Peripheral Supply Currents (Both –Gx and –Ix Devices)** | | | | | | |
| High-Frequency Oscillator | $I_{HFOSC}$ | Operating at 24.5 MHz, $T_A$ = 25 °C | — | 155 | — | µA |
| Low-Frequency Oscillator | $I_{LFOSC}$ | Operating at 80 kHz, $T_A$ = 25 °C | — | 3.5 | — | µA |
| ADC0 Always-on[4] | $I_{ADC}$ | 800 ksps, 10-bit conversions or 200 ksps, 12-bit conversions Normal bias settings $V_{DD}$ = 3.0 V | — | 845 | 1200 | µA |
| | | 250 ksps, 10-bit conversions or 62.5 ksps 12-bit conversions Low power bias settings $V_{DD}$ = 3.0 V | — | 425 | 580 | µA |
| ADC0 Burst Mode, 10-bit single conversions, external reference | $I_{ADC}$ | 200 ksps, $V_{DD}$ = 3.0 V | — | 370 | — | µA |
| | | 100 ksps, $V_{DD}$ = 3.0 V | — | 185 | — | µA |
| | | 10 ksps, $V_{DD}$ = 3.0 V | — | 19 | — | µA |

**Notes:**

1. Currents are additive. For example, where $I_{DD}$ is specified and the mode is not mutually exclusive, enabling the functions increases supply current by the specified amount.
2. Includes supply current from internal regulator, supply monitor, and High Frequency Oscillator.
3. Includes supply current from internal regulator, supply monitor, and Low Frequency Oscillator.
4. ADC0 always-on power excludes internal reference supply current.
5. The internal reference is enabled as-needed when operating the ADC in burst mode to save power.

**Table 1.2. Power Consumption (Continued)**

| Parameter | Symbol | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| ADC0 Burst Mode, 10-bit single conversions, internal reference, Low power bias settings | $I_{ADC}$ | 200 ksps, $V_{DD}$ = 3.0 V | — | 490 | — | µA |
| | | 100 ksps, $V_{DD}$ = 3.0 V | — | 245 | — | µA |
| | | 10 ksps, $V_{DD}$ = 3.0 V | — | 23 | — | µA |
| ADC0 Burst Mode, 12-bit single conversions, external reference | $I_{ADC}$ | 100 ksps, $V_{DD}$ = 3.0 V | — | 530 | — | µA |
| | | 50 ksps, $V_{DD}$ = 3.0 V | — | 265 | — | µA |
| | | 10 ksps, $V_{DD}$ = 3.0 V | — | 53 | — | µA |
| ADC0 Burst Mode, 12-bit single conversions, internal reference | $I_{ADC}$ | 100 ksps, $V_{DD}$ = 3.0 V, Normal bias | — | 950 | — | µA |
| | | 50 ksps, $V_{DD}$ = 3.0 V, Low power bias | — | 420 | — | µA |
| | | 10 ksps, $V_{DD}$ = 3.0 V, Low power bias | — | 85 | — | µA |
| Internal ADC0 Reference, Always-on[5] | $I_{IREF}$ | Normal Power Mode | — | 680 | 790 | µA |
| | | Low Power Mode | — | 160 | 210 | µA |
| Temperature Sensor | $I_{TSENSE}$ | | — | 75 | 120 | µA |
| Comparator 0 (CMP0), Comparator 1 (CMP1) | $I_{CMP}$ | CPnMD = 11 | — | 0.5 | — | µA |
| | | CPnMD = 10 | — | 3 | — | µA |
| | | CPnMD = 01 | — | 10 | — | µA |
| | | CPnMD = 00 | — | 25 | — | µA |
| Voltage Supply Monitor (VMON0) | $I_{VMON}$ | | — | 15 | 20 | µA |

**Notes:**
1. Currents are additive. For example, where $I_{DD}$ is specified and the mode is not mutually exclusive, enabling the functions increases supply current by the specified amount.
2. Includes supply current from internal regulator, supply monitor, and High Frequency Oscillator.
3. Includes supply current from internal regulator, supply monitor, and Low Frequency Oscillator.
4. ADC0 always-on power excludes internal reference supply current.
5. The internal reference is enabled as-needed when operating the ADC in burst mode to save power.

**Table 1.3. Reset and Supply Monitor**

| Parameter | Symbol | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $V_{DD}$ Supply Monitor Threshold | $V_{VDDM}$ | | 1.85 | 1.95 | 2.1 | V |
| Power-On Reset (POR) Threshold | $V_{POR}$ | Rising Voltage on $V_{DD}$ | — | 1.4 | — | V |
| | | Falling Voltage on $V_{DD}$ | 0.75 | — | 1.36 | V |
| $V_{DD}$ Ramp Time | $t_{RMP}$ | Time to $V_{DD} \geq 2.2$ V | 10 | — | — | µs |
| Reset Delay from POR | $t_{POR}$ | Relative to $V_{DD} \geq V_{POR}$ | 3 | 10 | 31 | ms |
| Reset Delay from non-POR source | $t_{RST}$ | Time between release of reset source and code execution | — | 39 | — | µs |
| $\overline{RST}$ Low Time to Generate Reset | $t_{RSTL}$ | | 15 | — | — | µs |
| Missing Clock Detector Response Time (final rising edge to reset) | $t_{MCD}$ | $F_{SYSCLK} > 1$ MHz | — | 0.625 | 1.2 | ms |
| Missing Clock Detector Trigger Frequency | $F_{MCD}$ | | — | 7.5 | 13.5 | kHz |
| $V_{DD}$ Supply Monitor Turn-On Time | $t_{MON}$ | | — | 2 | — | µs |

**Table 1.4. Flash Memory**

| Parameter | Symbol | Test Condition | Min | Typ | Max | Units |
|---|---|---|---|---|---|---|
| Write Time[1,2] | $t_{WRITE}$ | One Byte, $F_{SYSCLK}$ = 24.5 MHz | 19 | 20 | 21 | µs |
| Erase Time[1,2] | $t_{ERASE}$ | One Page, $F_{SYSCLK}$ = 24.5 MHz | 5.2 | 5.35 | 5.5 | ms |
| $V_{DD}$ Voltage During Programming[3] | $V_{PROG}$ | | 2.2 | — | 3.6 | V |
| Endurance (Write/Erase Cycles) | $N_{WE}$ | | 20k | 100k | — | Cycles |

**Notes:**
1. Does not include sequencing time before and after the write/erase operation, which may be multiple SYSCLK cycles.
2. The internal High-Frequency Oscillator has a programmable output frequency using the OSCICL register, which is factory programmed to 24.5 MHz. If user firmware adjusts the oscillator speed, it must be between 22 and 25 MHz during any flash write or erase operation. It is recommended to write the OSCICL register back to its reset value when writing or erasing flash.
3. Flash can be safely programmed at any voltage above the supply monitor threshold ($V_{VDDM}$).
4. Data Retention Information is published in the Quarterly Quality and Reliability Report.

**Table 1.5. Internal Oscillators**

| Parameter | Symbol | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| **High Frequency Oscillator (24.5 MHz)** | | | | | | |
| Oscillator Frequency | $f_{HFOSC}$ | Full Temperature and Supply Range | 24 | 24.5 | 25 | MHz |
| Power Supply Sensitivity | $PSS_{HFOSC}$ | $T_A$ = 25 °C | — | 0.5 | — | %/V |
| Temperature Sensitivity | $TS_{HFOSC}$ | $V_{DD}$ = 3.0 V | — | 40 | — | ppm/°C |
| **Low Frequency Oscillator (80 kHz)** | | | | | | |
| Oscillator Frequency | $f_{LFOSC}$ | Full Temperature and Supply Range | 75 | 80 | 85 | kHz |
| Power Supply Sensitivity | $PSS_{LFOSC}$ | $T_A$ = 25 °C | — | 0.05 | — | %/V |
| Temperature Sensitivity | $TS_{LFOSC}$ | $V_{DD}$ = 3.0 V | — | 65 | — | ppm/°C |

**Table 1.6. External Clock Input**

| Parameter | Symbol | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| External Input CMOS Clock Frequency (at EXTCLK pin) | $f_{CMOS}$ | | 0 | — | 25 | MHz |
| External Input CMOS Clock High Time | $t_{CMOSH}$ | | 18 | — | — | ns |
| External Input CMOS Clock Low Time | $t_{CMOSL}$ | | 18 | — | — | ns |

**Table 1.7. ADC**

| Parameter | Symbol | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| Resolution | $N_{bits}$ | 12 Bit Mode | | 12 | | Bits |
| | | 10 Bit Mode | | 10 | | Bits |
| Throughput Rate (High Speed Mode) | $f_S$ | 12 Bit Mode | — | — | 200 | ksps |
| | | 10 Bit Mode | — | — | 800 | ksps |
| Throughput Rate (Low Power Mode) | $f_S$ | 12 Bit Mode | — | — | 62.5 | ksps |
| | | 10 Bit Mode | — | — | 250 | ksps |
| Tracking Time | $t_{TRK}$ | High Speed Mode | 230 | — | — | ns |
| | | Low Power Mode | 450 | — | — | ns |
| Power-On Time | $t_{PWR}$ | | 1.2 | — | — | µs |
| SAR Clock Frequency | $f_{SAR}$ | High Speed Mode, Reference is 2.4 V internal | — | — | 6.25 | MHz |
| | | High Speed Mode, Reference is not 2.4 V internal | — | — | 12.5 | MHz |
| | | Low Power Mode | — | — | 4 | MHz |
| Conversion Time | $t_{CNV}$ | 10-Bit Conversion, SAR Clock = 12.25 MHz, System Clock = 24.5 MHz. | | 1.1 | | µs |
| Sample/Hold Capacitor | $C_{SAR}$ | Gain = 1 | — | 5 | — | pF |
| | | Gain = 0.5 | — | 2.5 | — | pF |
| Input Pin Capacitance | $C_{IN}$ | | — | 20 | — | pF |
| Input Mux Impedance | $R_{MUX}$ | | — | 550 | — | Ω |
| Voltage Reference Range | $V_{REF}$ | | 1 | — | $V_{DD}$ | V |
| Input Voltage Range* | $V_{IN}$ | Gain = 1 | 0 | — | $V_{REF}$ | V |
| | | Gain = 0.5 | 0 | — | $2xV_{REF}$ | V |
| Power Supply Rejection Ratio | $PSRR_{ADC}$ | | — | 70 | — | dB |
| **DC Performance** | | | | | | |
| Integral Nonlinearity | INL | 12 Bit Mode | — | ±1 | ±2.3 | LSB |
| | | 10 Bit Mode | — | ±0.2 | ±0.6 | LSB |
| Differential Nonlinearity (Guaranteed Monotonic) | DNL | 12 Bit Mode | –1 | ±0.7 | 1.9 | LSB |
| | | 10 Bit Mode | — | ±0.2 | ±0.6 | LSB |
| ***Note:** Absolute input pin voltage is limited by the $V_{DD}$ supply. | | | | | | |

**Table 1.7. ADC (Continued)**

| Parameter | Symbol | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| Offset Error | $E_{OFF}$ | 12 Bit Mode, VREF = 1.65 V | –3 | 0 | 3 | LSB |
| | | 10 Bit Mode, VREF = 1.65 V | –2 | 0 | 2 | LSB |
| Offset Temperature Coefficient | $TC_{OFF}$ | | — | 0.004 | — | LSB/°C |
| Slope Error | $E_M$ | 12 Bit Mode | — | ±0.02 | ±0.1 | % |
| | | 10 Bit Mode | — | ±0.06 | ±0.24 | % |
| **Dynamic Performance 10 kHz Sine Wave Input 1dB below full scale, Max throughput, using AGND pin** | | | | | | |
| Signal-to-Noise | SNR | 12 Bit Mode | 61 | 66 | — | dB |
| | | 10 Bit Mode | 53 | 60 | — | dB |
| Signal-to-Noise Plus Distortion | SNDR | 12 Bit Mode | 61 | 66 | — | dB |
| | | 10 Bit Mode | 53 | 60 | — | dB |
| Total Harmonic Distortion (Up to 5th Harmonic) | THD | 12 Bit Mode | — | 71 | — | dB |
| | | 10 Bit Mode | — | 70 | — | dB |
| Spurious-Free Dynamic Range | SFDR | 12 Bit Mode | — | –79 | — | dB |
| | | 10 Bit Mode | — | –74 | — | dB |

***Note:** Absolute input pin voltage is limited by the $V_{DD}$ supply.

**Table 1.8. Voltage Reference**

| Parameter | Symbol | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| **Internal Fast Settling Reference** | | | | | | |
| Output Voltage (Full Temperature and Supply Range) | $V_{REFFS}$ | 1.65 V Setting | 1.62 | 1.65 | 1.68 | V |
| | | 2.4 V Setting, $V_{DD} \geq 2.6$ V | 2.35 | 2.4 | 2.45 | V |
| Temperature Coefficient | $TC_{REFFS}$ | | — | 50 | — | ppm/°C |
| Turn-on Time | $t_{REFFS}$ | | — | — | 1.5 | µs |
| Power Supply Rejection | $PSRR_{REFFS}$ | | — | 400 | — | ppm/V |
| **External Reference** | | | | | | |
| Input Current | $I_{EXTREF}$ | Sample Rate = 800 ksps; VREF = 3.0 V | — | 5 | — | µA |

**Table 1.9. Temperature Sensor**

| Parameter | Symbol | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| Offset | $V_{OFF}$ | $T_A$ = 0 °C | — | 757 | — | mV |
| Offset Error* | $E_{OFF}$ | $T_A$ = 0 °C | — | 17 | — | mV |
| Slope | M | | — | 2.85 | — | mV/°C |
| Slope Error* | $E_M$ | | — | 70 | — | µV/°C |
| Linearity | | | — | 0.5 | — | °C |
| Turn-on Time | | | — | 1.8 | — | µs |
| ***Note:** Represents one standard deviation from the mean. | | | | | | |

**Table 1.10. Comparators**

| Parameter | Symbol | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| Response Time, CPnMD = 00 (Highest Speed) | $t_{RESP0}$ | +100 mV Differential | — | 100 | — | ns |
| | | –100 mV Differential | — | 150 | — | ns |
| Response Time, CPnMD = 11 (Lowest Power) | $t_{RESP3}$ | +100 mV Differential | — | 1.5 | — | µs |
| | | –100 mV Differential | — | 3.5 | — | µs |
| Positive Hysterisis Mode 0 (CPnMD = 00) | $HYS_{CP+}$ | CPnHYP = 00 | — | 0.4 | — | mV |
| | | CPnHYP = 01 | — | 8 | — | mV |
| | | CPnHYP = 10 | — | 16 | — | mV |
| | | CPnHYP = 11 | — | 32 | — | mV |
| Negative Hysterisis Mode 0 (CPnMD = 00) | $HYS_{CP-}$ | CPnHYN = 00 | — | -0.4 | — | mV |
| | | CPnHYN = 01 | — | –8 | — | mV |
| | | CPnHYN = 10 | — | –16 | — | mV |
| | | CPnHYN = 11 | — | –32 | — | mV |
| Positive Hysterisis Mode 1 (CPnMD = 01) | $HYS_{CP+}$ | CPnHYP = 00 | — | 0.5 | — | mV |
| | | CPnHYP = 01 | — | 6 | — | mV |
| | | CPnHYP = 10 | — | 12 | — | mV |
| | | CPnHYP = 11 | — | 24 | — | mV |
| Negative Hysterisis Mode 1 (CPnMD = 01) | $HYS_{CP-}$ | CPnHYN = 00 | — | -0.5 | — | mV |
| | | CPnHYN = 01 | — | –6 | — | mV |
| | | CPnHYN = 10 | — | –12 | — | mV |
| | | CPnHYN = 11 | — | –24 | — | mV |
| Positive Hysterisis Mode 2 (CPnMD = 10) | $HYS_{CP+}$ | CPnHYP = 00 | — | 0.7 | — | mV |
| | | CPnHYP = 01 | — | 4.5 | — | mV |
| | | CPnHYP = 10 | — | 9 | — | mV |
| | | CPnHYP = 11 | — | 18 | — | mV |
| Negative Hysterisis Mode 2 (CPnMD = 10) | $HYS_{CP-}$ | CPnHYN = 00 | — | -0.6 | — | mV |
| | | CPnHYN = 01 | — | –4.5 | — | mV |
| | | CPnHYN = 10 | — | –9 | — | mV |
| | | CPnHYN = 11 | — | –18 | — | mV |

**Table 1.10. Comparators**

| Parameter | Symbol | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| Positive Hysteresis Mode 3 (CPnMD = 11) | $HYS_{CP+}$ | CPnHYP = 00 | — | 1.5 | — | mV |
| | | CPnHYP = 01 | — | 4 | — | mV |
| | | CPnHYP = 10 | — | 8 | — | mV |
| | | CPnHYP = 11 | — | 16 | — | mV |
| Negative Hysteresis Mode 3 (CPnMD = 11) | $HYS_{CP-}$ | CPnHYN = 00 | — | -1.5 | — | mV |
| | | CPnHYN = 01 | — | –4 | — | mV |
| | | CPnHYN = 10 | — | –8 | — | mV |
| | | CPnHYN = 11 | — | –16 | — | mV |
| Input Range (CP+ or CP–) | $V_{IN}$ | | -0.25 | — | $V_{DD}$+0.25 | V |
| Input Pin Capacitance | $C_{CP}$ | | — | 7.5 | — | pF |
| Common-Mode Rejection Ratio | $CMRR_{CP}$ | | — | 70 | — | dB |
| Power Supply Rejection Ratio | $PSRR_{CP}$ | | — | 72 | — | dB |
| Input Offset Voltage | $V_{OFF}$ | $T_A$ = 25 °C | -10 | 0 | 10 | mV |
| Input Offset Tempco | $TC_{OFF}$ | | — | 3.5 | — | µV/°C |

**Table 1.11. Port I/O**

| Parameter | Symbol | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| Output High Voltage (High Drive) | $V_{OH}$ | $I_{OH}$ = –3 mA | $V_{DD}$ – 0.7 | — | — | V |
| Output Low Voltage (High Drive) | $V_{OL}$ | $I_{OL}$ = 8.5 mA | — | — | 0.6 | V |
| Output High Voltage (Low Drive) | $V_{OH}$ | $I_{OH}$ = –1 mA | $V_{DD}$ – 0.7 | — | — | V |
| Output Low Voltage (Low Drive) | $V_{OL}$ | $I_{OL}$ = 1.4 mA | — | — | 0.6 | V |
| Input High Voltage | $V_{IH}$ | | $V_{DD}$ – 0.6 | — | — | V |
| Input Low Voltage | $V_{IL}$ | | — | — | 0.6 | V |
| Pin Capacitance | $C_{IO}$ | | — | 7 | — | pF |
| Weak Pull-Up Current ($V_{IN}$ = 0 V) | $I_{PU}$ | $V_{DD}$ = 3.6 | –30 | –20 | –10 | µA |
| Input Leakage (Pullups off or Analog) | $I_{LK}$ | GND ≤ $V_{IN}$ ≤ $V_{DD}$ | –1.1 | — | 1.1 | µA |
| Input Leakage Current with $V_{IN}$ above $V_{DD}$ | $I_{LK}$ | $V_{DD}$ < $V_{IN}$ < $V_{DD}$+2.0 V | 0 | 5 | 150 | µA |

## 1.2. Typical Performance Curves

### 1.2.1. Operating Supply Current

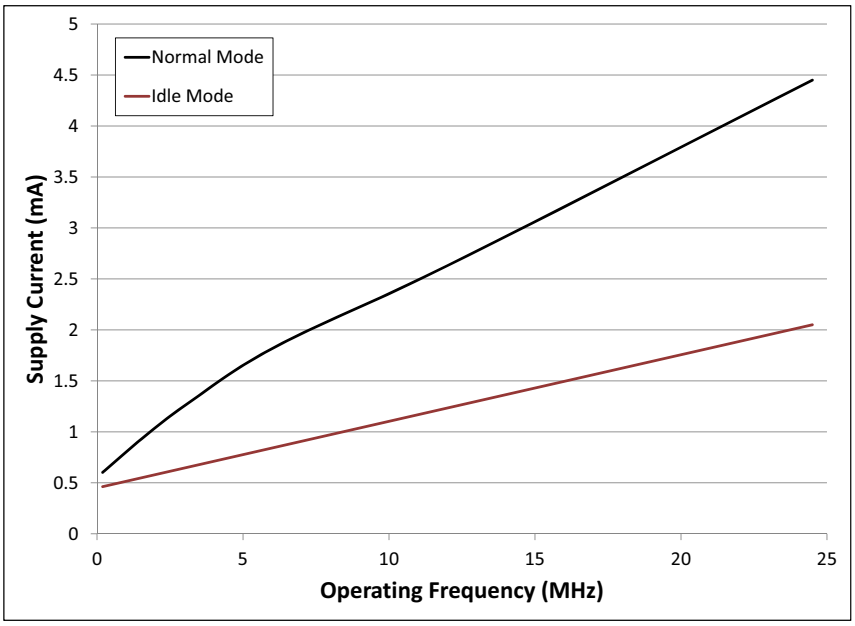



**Figure 1.1. Typical Operating Current Running From 24.5 MHz Internal Oscillator**

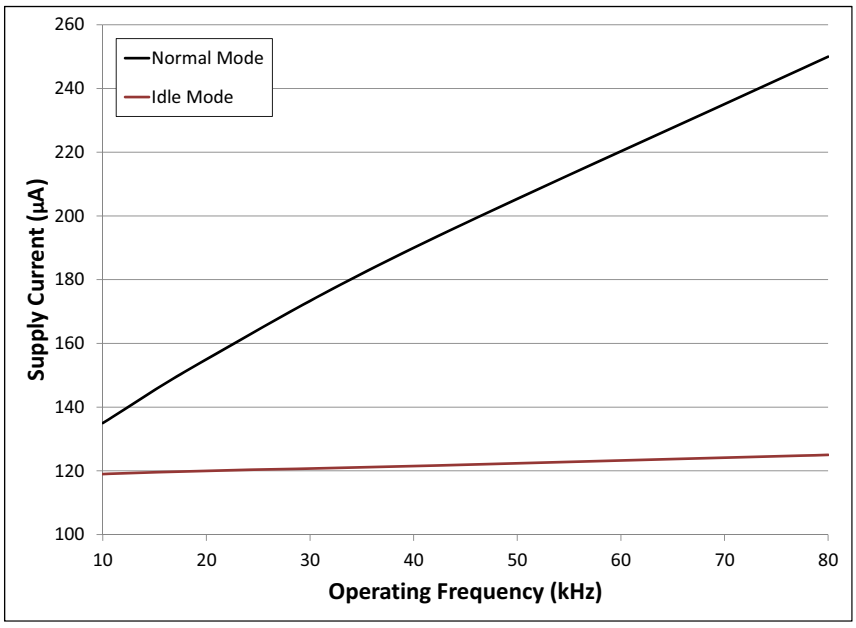



**Figure 1.2. Typical Operating Current Running From 80 kHz Internal Oscillator**

### 1.2.2. ADC Supply Current

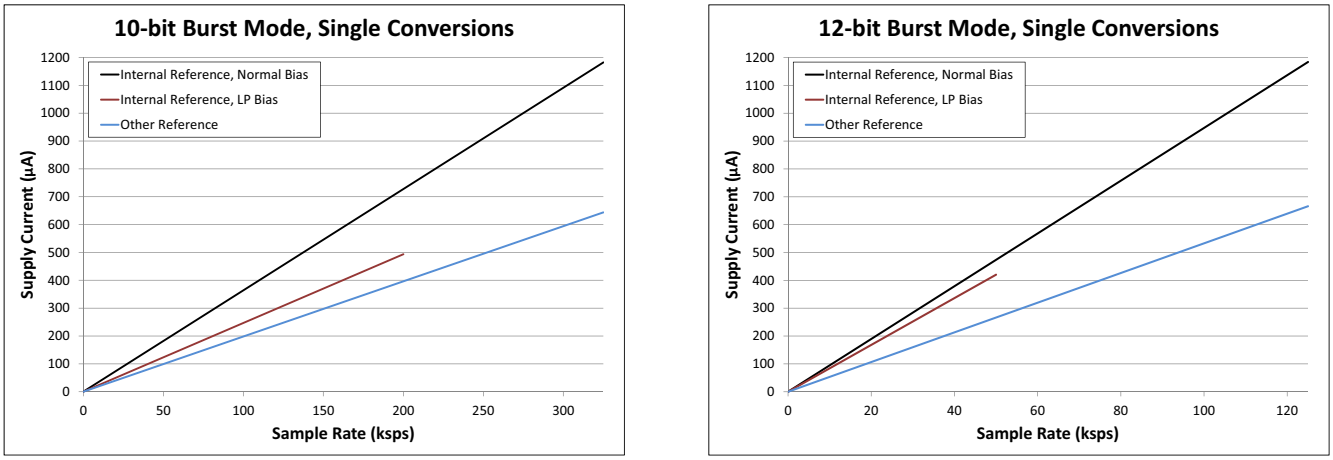



**Figure 1.3. Typical ADC and Internal Reference Power Consumption in Burst Mode**

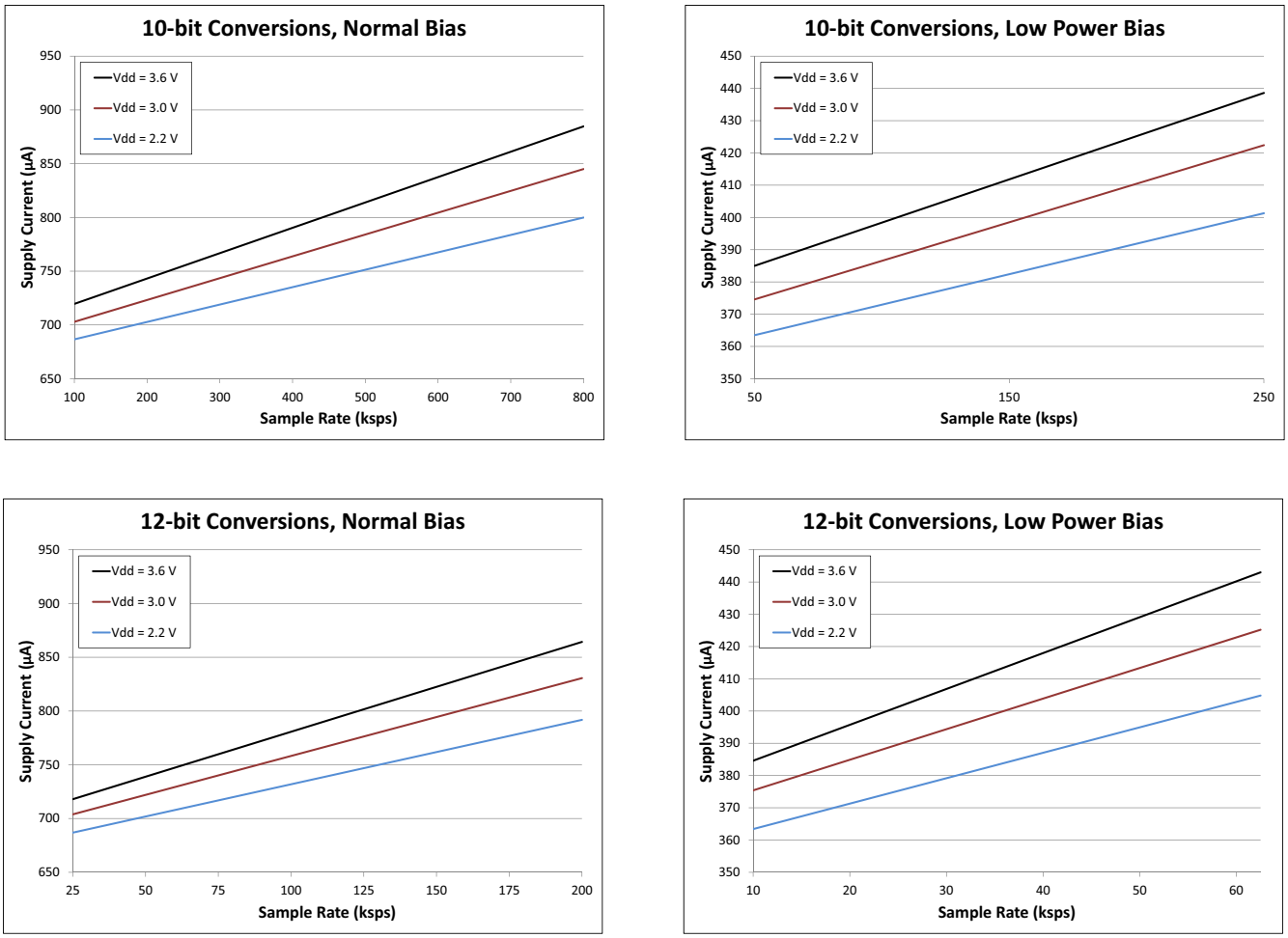



**Figure 1.4. Typical ADC Power Consumption in Normal (Always-On) Mode**

### 1.2.3. Port I/O Output Drive

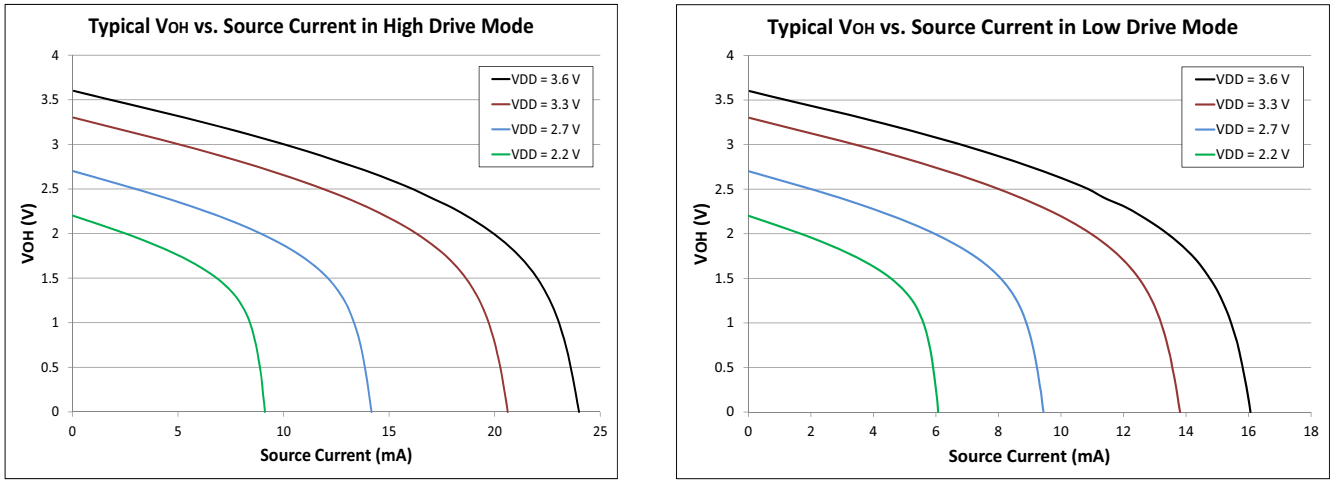



**Figure 1.5. Typical $V_{OH}$ vs. Source Current**

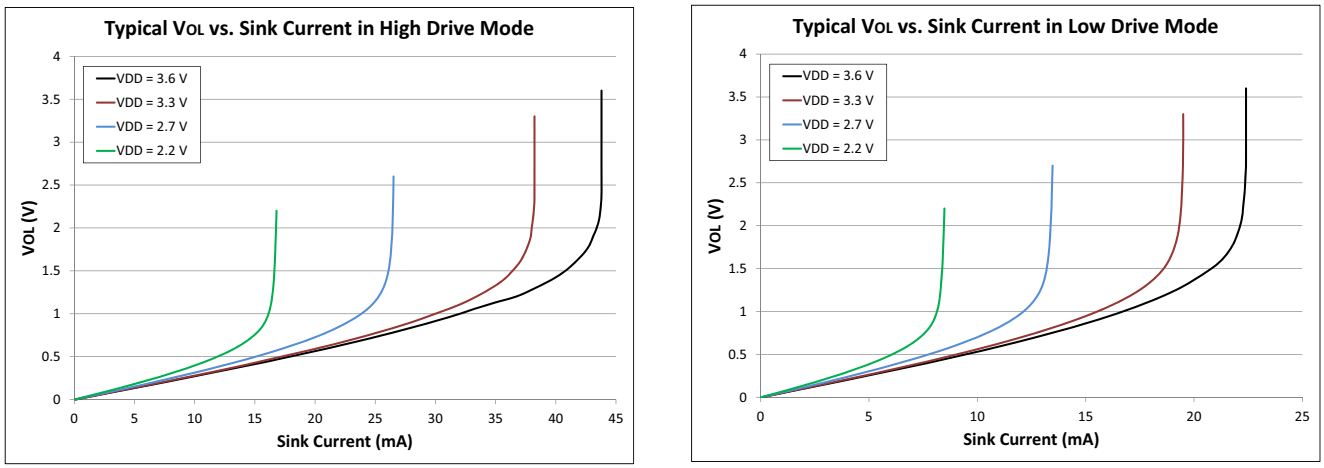



**Figure 1.6. Typical $V_{OL}$ vs. Sink Current**

## 1.3. Thermal Conditions

**Table 1.12. Thermal Conditions**

| Parameter | Symbol | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| Thermal Resistance* | $\theta_{JA}$ | SOIC-16 Packages | — | 70 | — | °C/W |
| | | QFN-20 Packages | — | 60 | — | °C/W |
| | | QSOP-24 Packages | — | 65 | — | °C/W |
| ***Note:** Thermal resistance assumes a multi-layer PCB with any exposed pad soldered to a PCB pad. | | | | | | |

## 1.4. Absolute Maximum Ratings

Stresses above those listed under Table 1.13 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.

For more information on the available quality and reliability data, see the Quality and Reliability Monitor Report at http://www.silabs.com/support/quality/pages/default.aspx.

**Table 1.13. Absolute Maximum Ratings**

| Parameter | Symbol | Test Condition | Min | Max | Unit |
|---|---|---|---|---|---|
| Ambient Temperature Under Bias | $T_{BIAS}$ | | –55 | 125 | °C |
| Storage Temperature | $T_{STG}$ | | –65 | 150 | °C |
| Voltage on $V_{DD}$ | $V_{DD}$ | | GND–0.3 | 4.2 | V |
| Voltage on I/O pins or $\overline{RST}$ | $V_{IN}$ | $V_{DD} \geq 3.3$ V | GND–0.3 | 5.8 | V |
| | | $V_{DD} < 3.3$ V | GND–0.3 | $V_{DD}$+2.5 | V |
| Total Current Sunk into Supply Pin | $I_{VDD}$ | | — | 400 | mA |
| Total Current Sourced out of Ground Pin | $I_{GND}$ | | 400 | — | mA |
| Current Sourced or Sunk by Any I/O Pin or $\overline{RST}$ | $I_{PIO}$ | | -100 | 100 | mA |
| Operating Junction Temperature | $T_J$ | Commercial Grade Devices (-GM, -GS, -GU) | –40 | 105 | °C |
| | | Industrial Grade Devices (-IM, -IS, -IU) | –40 | 125 | °C |
| **Note:** Exposure to maximum rating conditions for extended periods may affect device reliability. | | | | | |

## 2. System Overview

The C8051F85x/86x device family are fully integrated, mixed-signal system-on-a-chip MCUs. Highlighted features are listed below. Refer to Table 4.1 for specific product feature selection and part ordering numbers.

- **Core:**
  - Pipelined CIP-51 Core
  - Fully compatible with standard 8051 instruction set
  - 70% of instructions execute in 1-2 clock cycles
  - 25 MHz maximum operating frequency
- **Memory:**
  - 2-8 kB flash; in-system programmable in 512-byte sectors
  - 512 bytes RAM (including 256 bytes standard 8051 RAM and 256 bytes on-chip XRAM)
- **Power:**
  - Internal low drop-out (LDO) regulator for CPU core voltage
  - Power-on reset circuit and brownout detectors
- **I/O: Up to 18 total multifunction I/O pins:**
  - All pins 5 V tolerant under bias
  - Flexible peripheral crossbar for peripheral routing
  - 5 mA source, 12.5 mA sink allows direct drive of LEDs
- **Clock Sources:**
  - Low-power internal oscillator: 24.5 MHz ±2%
  - Low-frequency internal oscillator: 80 kHz
  - External CMOS clock option
- **Timers/Counters and PWM:**
  - 3-channel Programmable Counter Array (PCA) supporting PWM, capture/compare and frequency output modes
  - 4x 16-bit general-purpose timers
  - Independent watchdog timer, clocked from low frequency oscillator
- **Communications and Other Digital Peripherals:**
  - UART
  - SPI™
  - $I^2C$ / SMBus™
  - 16-bit CRC Unit, supporting automatic CRC of flash at 256-byte boundaries
- **Analog:**
  - 12-Bit Analog-to-Digital Converter (ADC)
  - 2 x Low-Current Comparators
- **On-Chip Debugging**

With on-chip power-on reset, voltage supply monitor, watchdog timer, and clock oscillator, the C8051F85x/86x devices are truly standalone system-on-a-chip solutions. The flash memory is reprogrammable in-circuit, providing non-volatile data storage and allowing field upgrades of the firmware.

The on-chip debugging interface (C2) allows non-intrusive (uses no on-chip resources), full speed, in-circuit debugging using the production MCU installed in the final application. This debug logic supports inspection and modification of memory and registers, setting breakpoints, single stepping, and run and halt commands. All analog and digital peripherals are fully functional while debugging.

Each device is specified for 2.2 to 3.6 V operation, and are available in 20-pin QFN, 16-pin SOIC or 24-pin QSOP packages. All package options are lead-free and RoHS compliant. The device is available in two temperature grades: -40 to +85 °C or –40 to +125 °C. See Table 4.1 for ordering information. A block diagram is included in Figure 2.1.

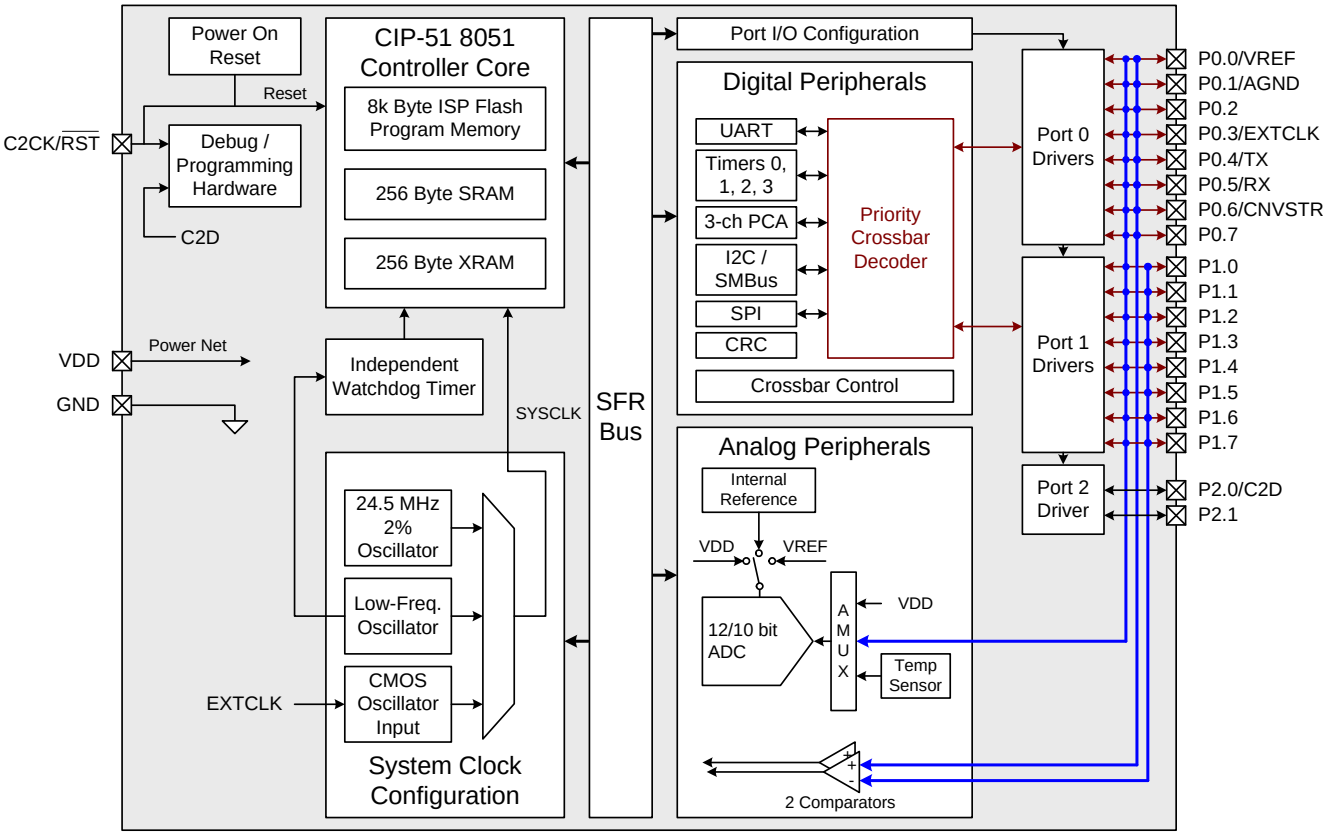



**Figure 2.1. C8051F85x/86x Family Block Diagram (QSOP-24 Shown)**

## 2.1. Power

### 2.1.1. LDO

The C8051F85x/86x devices include an internal regulator to regulate the supply voltage down the core operating voltage of 1.8 V. This LDO consumes little power, but can be shut down in the power-saving Stop mode.

### 2.1.2. Voltage Supply Monitor (VMON0)

The C8051F85x/86x devices include a voltage supply monitor which allows devices to function in known, safe operating condition without the need for external hardware.

The supply monitor module includes the following features:

- Holds the device in reset if the main VDD supply drops below the VDD Reset threshold.

### 2.1.3. Device Power Modes

The C8051F85x/86x devices feature three low power modes in addition to normal operating mode, allowing the designer to save power when the core is not in use. All power modes are detailed in Table 2.1.

**Table 2.1. C8051F85x/86x Power Modes**

| Mode | Description | Mode Entrance | Mode Exit |
|---|---|---|---|
| Normal | Core and peripherals operating at full speed | | |
| Idle | ■ Core halted<br>■ Peripherals operate at full speed | Set IDLE bit in PCON | Any enabled interrupt or reset source |
| Stop | ■ All clocks stopped<br>■ Core LDO and (optionally) comparators still running<br>■ Pins retain state | Clear STOPCF in REG0MD and<br>Set STOP bit in PCON | Device reset |
| Shutdown | ■ All clocks stopped<br>■ Core LDO and all analog circuits shut down<br>■ Pins retain state | Set STOPCF in REG0MD and<br>Set STOP bit in PCON | Device reset |

In addition, the user may choose to lower the clock speed in Normal and Idle modes to save power when the CPU requirements allow for lower speed.

#### 2.1.3.1. Normal Mode

Normal mode encompasses the typical full-speed operation. The power consumption of the device in this mode will vary depending on the system clock speed and any analog peripherals that are enabled.

#### 2.1.3.2. Idle Mode

Setting the IDLE bit in PCON causes the hardware to halt the CPU and enter idle mode as soon as the instruction that sets the bit completes execution. All internal registers and memory maintain their original data. All analog and digital peripherals can remain active during idle mode.

Idle mode is terminated when an enabled interrupt is asserted or a reset occurs. The assertion of an enabled interrupt will cause the IDLE bit 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.

#### 2.1.3.3. Stop Mode (Regulator On)

Setting the STOP bit in PCON when STOPCF in REG0CN is clear causes the controller core to enter stop mode as soon as the instruction that sets the bit completes execution. In stop mode the internal oscillator, CPU, and all digital peripherals are stopped. Each analog peripheral may be shut down individually prior to entering stop mode. Stop mode can only be terminated by an internal or external reset.

#### 2.1.3.4. Shutdown Mode (Regulator Off)

Shutdown mode is an extension of the normal stop mode operation. Setting the STOP bit in PCON when STOPCF in REG0CN is also set causes the controller core to enter shutdown mode as soon as the instruction that sets the bit completes execution, and then the internal regulator is powered down. In shutdown mode, all core functions, memories and peripherals are powered off. An external pin reset or power-on reset is required to exit shutdown mode.

## 2.2. I/O

### 2.2.1. General Features

The C8051F85x/86x ports have the following features:

- Push-pull or open-drain output modes and analog or digital modes.
- Port Match allows the device to recognize a change on a port pin value and wake from idle mode or generate an interrupt.
- Internal pull-up resistors can be globally enabled or disabled.
- Two external interrupts provide unique interrupt vectors for monitoring time-critical events.
- Above-rail tolerance allows 5 V interface when device is powered.

### 2.2.2. Crossbar

The C8051F85x/86x devices have a digital peripheral crossbar with the following features:

- Flexible peripheral assignment to port pins.
- Pins can be individually skipped to move peripherals as needed for design or layout considerations.

The crossbar has a fixed priority for each I/O function and assigns these functions to the port pins. When a digital resource is selected, the least-significant unassigned port pin is assigned to that resource. 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. This provides some flexibility when designing a system: pins involved with sensitive analog measurements can be moved away from digital I/O and peripherals can be moved around the chip as needed to ease layout constraints.

## 2.3. Clocking

The C8051F85x/86x devices have two internal oscillators and the option to use an external CMOS input at a pin as the system clock. A programmable divider allows the user to internally run the system clock at a slower rate than the selected oscillator if desired.

## 2.4. Counters/Timers and PWM

### 2.4.1. Programmable Counter Array (PCA0)

The C8051F85x/86x devices include a three-channel, 16-bit Programmable Counter Array with the following features:

- 16-bit time base.
- Programmable clock divisor and clock source selection.
- Three independently-configurable channels.
- 8, 9, 10, 11 and 16-bit PWM modes (center or edge-aligned operation).
- Output polarity control.
- Frequency output mode.
- Capture on rising, falling or any edge.
- Compare function for arbitrary waveform generation.
- Software timer (internal compare) mode.
- Can accept hardware “kill” signal from comparator 0.

### 2.4.2. Timers (Timer 0, Timer 1, Timer 2 and Timer 3)

Timers include the following features:

- Timer 0 and Timer 1 are standard 8051 timers, supporting backwards-compatibility with firmware and hardware.
- Timer 2 and Timer 3 can each operate as 16-bit auto-reload or two independent 8-bit auto-reload timers, and include pin or LFO clock capture capabilities.

### 2.4.3. Watchdog Timer (WDT0)

The watchdog timer includes a 16-bit timer with a programmable reset period. The registers are protected from inadvertent access by an independent lock and key interface.

The Watchdog Timer has the following features:

- Programmable timeout interval.
- Runs from the low frequency oscillator.
- Lock-out feature to prevent any modification until a system reset.

## 2.5. Communications and other Digital Peripherals

### 2.5.1. Universal Asynchronous Receiver/Transmitter (UART0)

The UART uses two signals (TX and RX) and a predetermined fixed baud rate to provide asynchronous communications with other devices.

The UART module provides the following features:

- Asynchronous transmissions and receptions.
- Baud rates up to SYSCLK / 2 (transmit) or SYSCLK / 8 (receive).
- 8- or 9-bit data.
- Automatic start and stop generation.

### 2.5.2. Serial Peripheral Interface (SPI0)

SPI is a 3- or 4-wire communication interface that includes a clock, input data, output data, and an optional select signal.

The SPI module includes the following features:

- Supports 3- or 4-wire master or slave modes.
- Supports external clock frequencies up to SYSCLK / 2 in master mode and SYSCLK / 10 in slave mode.
- Support for all clock phase and polarity modes.
- 8-bit programmable clock rate.
- Support for multiple masters on the same data lines.

### 2.5.3. System Management Bus / I2C (SMBus0)

The SMBus interface is a two-wire, bi-directional serial bus compatible with both I2C and SMBus protocols. The two clock and data signals operate in open-drain mode with external pull-ups to support automatic bus arbitration.

Reads and writes to the interface are byte-oriented with the SMBus interface autonomously controlling the serial transfer of the data. Data can be transferred at up to 1/8th of the system clock as a master or slave, which can be faster than allowed by the SMBus / I2C specification, depending on the clock source 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 module includes the following features:

- Standard (up to 100 kbps) and Fast (400 kbps) transfer speeds.
- Support for master, slave, and multi-master modes.
- Hardware synchronization and arbitration for multi-master mode.
- Clock low extending (clock stretching) to interface with faster masters.
- Hardware support for 7-bit slave and general call address recognition.
- Firmware support for 10-bit slave address decoding.
- Ability to inhibit all slave states.
- Programmable data setup/hold times.

### 2.5.4. 16/32-bit CRC (CRC0)

The CRC module is designed to provide hardware calculations for flash memory verification and communications protocols. The CRC module supports the standard CCITT-16 16-bit polynomial (0x1021), and includes the following features:

- Support for four CCITT-16 polynomial.

- Byte-level bit reversal.
- Automatic CRC of flash contents on one or more 256-byte blocks.
- Initial seed selection of 0x0000 or 0xFFFF.

## 2.6. Analog Peripherals

### 2.6.1. 12-Bit Analog-to-Digital Converter (ADC0)

The ADC0 module on C8051F85x/86x devices is a Successive Approximation Register (SAR) Analog to Digital Converter (ADC). The key features of the ADC module are:

- Single-ended 12-bit and 10-bit modes.
- Supports an output update rate of 200 ksps samples per second in 12-bit mode or 800 ksps samples per second in 10-bit mode.
- Operation in low power modes at lower conversion speeds.
- Selectable asynchronous hardware conversion trigger.
- Output data window comparator allows automatic range checking.
- Support for Burst Mode, which produces one set of accumulated data per conversion-start trigger with programmable power-on settling and tracking time.
- Conversion complete and window compare interrupts supported.
- Flexible output data formatting.
- Includes an internal fast-settling reference with two levels (1.65 V and 2.4 V) and support for external reference and signal ground.

### 2.6.2. Low Current Comparators (CMP0, CMP1)

The comparators take two analog input voltages and output the relationship between these voltages (less than or greater than) as a digital signal. The Low Power Comparator module includes the following features:

- Multiple sources for the positive and negative poles, including VDD, VREF, and I/O pins.
- Two outputs are available: a digital synchronous latched output and a digital asynchronous raw output.
- Programmable hysteresis and response time.
- Falling or rising edge interrupt options on the comparator output.
- Provide “kill” signal to PCA module.
- Comparator 0 can be used to reset the device.

## 2.7. 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:

- The core halts program execution.
- Module registers are initialized to their defined reset values unless the bits reset only with a power-on reset.
- External port pins are forced to a known state.
- Interrupts and timers are disabled.

All registers are reset to the predefined values noted in the register descriptions unless the bits only reset with a power-on reset. The contents of RAM are unaffected during a reset; any previously stored data is preserved as long as power is not lost.

The Port I/O latches are reset to 1 in open-drain mode. Weak pullups are enabled during and after the reset. For VDD Supply Monitor and power-on 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 the internal low-power oscillator. The Watchdog Timer is enabled with the Low Frequency Oscillator (LFO0) as its clock source. Program execution begins at location 0x0000.

## 2.8. On-Chip Debugging

The C8051F85x/86x 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.

# 3. Pin Definitions

## 3.1. C8051F850/1/2/3/4/5 QSOP24 Pin Definitions

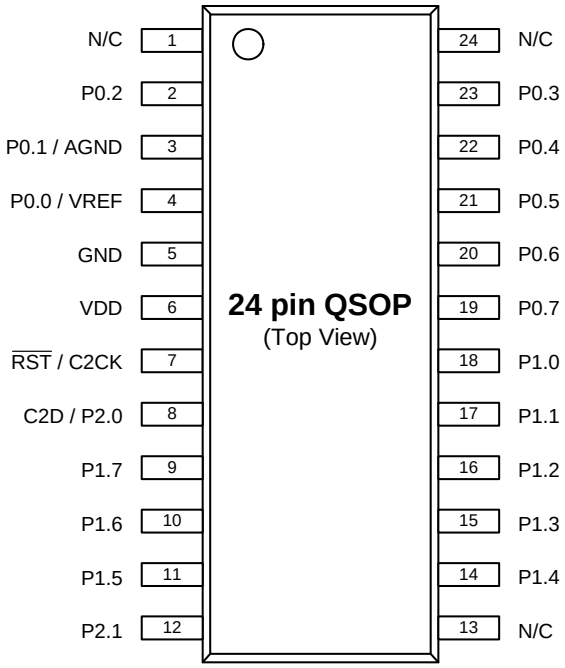



**Figure 3.1. C8051F850/1/2/3/4/5-GU and C8051F850/1/2/3/4/5-IU Pinout**

**Table 3.1. Pin Definitions for C8051F850/1/2/3/4/5-GU and C8051F850/1/2/3/4/5-IU**

| Pin Name | Type | Pin Numbers | Crossbar Capability | Additional Digital Functions | Analog Functions |
|---|---|---|---|---|---|
| GND | Ground | 5 | | | |
| VDD | Power | 6 | | | |
| RST / C2CK | Active-low Reset / C2 Debug Clock | 7 | | | |

**Table 3.1. Pin Definitions for C8051F850/1/2/3/4/5-GU and C8051F850/1/2/3/4/5-IU**

| Pin Name | Type | Pin Numbers | Crossbar Capability | Additional Digital Functions | Analog Functions |
|---|---|---|---|---|---|
| P0.0 | Standard I/O | 4 | Yes | P0MAT.0<br>INT0.0<br>INT1.0 | ADC0.0<br>CP0P.0<br>CP0N.0<br>VREF |
| P0.1 | Standard I/O | 3 | Yes | P0MAT.1<br>INT0.1<br>INT1.1 | ADC0.1<br>CP0P.1<br>CP0N.1<br>AGND |
| P0.2 | Standard I/O | 2 | Yes | P0MAT.2<br>INT0.2<br>INT1.2 | ADC0.2<br>CP0P.2<br>CP0N.2 |
| P0.3 /<br>EXTCLK | Standard I/O /<br>External CMOS Clock Input | 23 | Yes | P0MAT.3<br>EXTCLK<br>INT0.3<br>INT1.3 | ADC0.3<br>CP0P.3<br>CP0N.3 |
| P0.4 | Standard I/O | 22 | Yes | P0MAT.4<br>INT0.4<br>INT1.4 | ADC0.4<br>CP0P.4<br>CP0N.4 |
| P0.5 | Standard I/O | 21 | Yes | P0MAT.5<br>INT0.5<br>INT1.5 | ADC0.5<br>CP0P.5<br>CP0N.5 |
| P0.6 | Standard I/O | 20 | Yes | P0MAT.6<br>CNVSTR<br>INT0.6<br>INT1.6 | ADC0.6<br>CP0P.6<br>CP0N.6 |
| P0.7 | Standard I/O | 19 | Yes | P0MAT.7<br>INT0.7<br>INT1.7 | ADC0.7<br>CP0P.7<br>CP0N.7 |

**Table 3.1. Pin Definitions for C8051F850/1/2/3/4/5-GU and C8051F850/1/2/3/4/5-IU**

| Pin Name | Type | Pin Numbers | Crossbar Capability | Additional Digital Functions | Analog Functions |
|---|---|---|---|---|---|
| P1.0 | Standard I/O | 18 | Yes | P1MAT.0 | ADC0.8<br>CP1P.0<br>CP1N.0 |
| P1.1 | Standard I/O | 17 | Yes | P1MAT.1 | ADC0.9<br>CP1P.1<br>CP1N.1 |
| P1.2 | Standard I/O | 16 | Yes | P1MAT.2 | ADC0.10<br>CP1P.2<br>CP1N.2 |
| P1.3 | Standard I/O | 15 | Yes | P1MAT.3 | ADC0.11<br>CP1P.3<br>CP1N.3 |
| P1.4 | Standard I/O | 14 | Yes | P1MAT.4 | ADC0.12<br>CP1P.4<br>CP1N.4 |
| P1.5 | Standard I/O | 11 | Yes | P1MAT.5 | ADC0.13<br>CP1P.5<br>CP1N.5 |
| P1.6 | Standard I/O | 10 | Yes | P1MAT.6 | ADC0.14<br>CP1P.6<br>CP1N.6 |
| P1.7 | Standard I/O | 9 | Yes | P1MAT.7 | ADC0.15<br>CP1P.7<br>CP1N.7 |
| P2.0 /<br>C2D | Standard I/O /<br>C2 Debug Data | 8 | | | |
| P2.1 | Standard I/O | 12 | | | |

**Table 3.1. Pin Definitions for C8051F850/1/2/3/4/5-GU and C8051F850/1/2/3/4/5-IU**

| Pin Name | Type | Pin Numbers | Crossbar Capability | Additional Digital Functions | Analog Functions |
|---|---|---|---|---|---|
| N/C | No Connection | 1<br>13<br>24 | | | |

## 3.2. C8051F850/1/2/3/4/5 QFN20 Pin Definitions

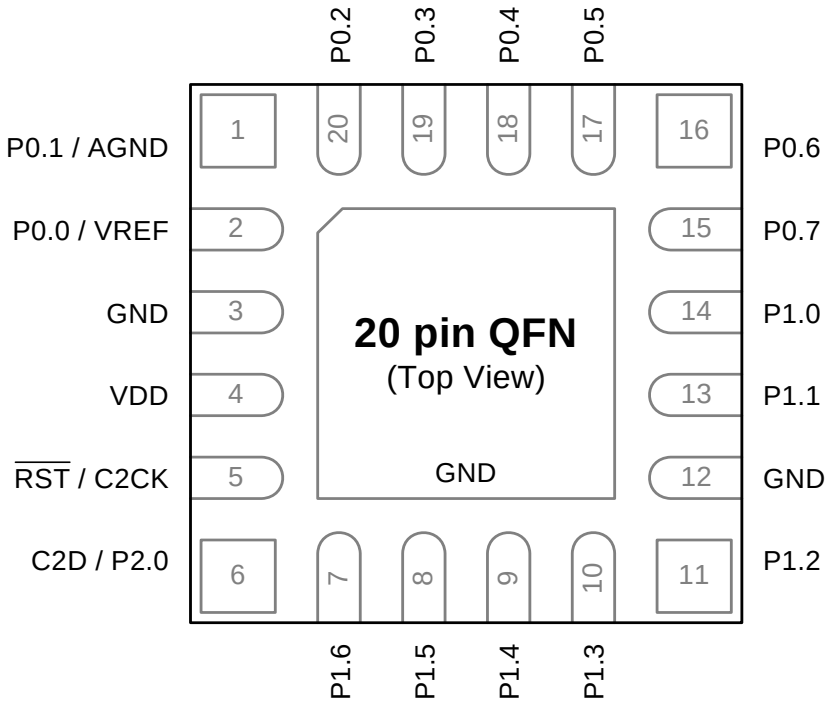



**Figure 3.2. C8051F850/1/2/3/4/5-GM and C8051F850/1/2/3/4/5-IM Pinout**

**Table 3.2. Pin Definitions for C8051F850/1/2/3/4/5-GM and C8051F850/1/2/3/4/5-IM**

| Pin Name | Type | Pin Numbers | Crossbar Capability | Additional Digital Functions | Analog Functions |
|---|---|---|---|---|---|
| GND | Ground | Center<br>3<br>12 | | | |
| VDD | Power | 4 | | | |
| RST / C2CK | Active-low Reset / C2 Debug Clock | 5 | | | |

**Table 3.2. Pin Definitions for C8051F850/1/2/3/4/5-GM and C8051F850/1/2/3/4/5-IM**

| Pin Name | Type | Pin Numbers | Crossbar Capability | Additional Digital Functions | Analog Functions |
|---|---|---|---|---|---|
| P0.0 | Standard I/O | 2 | Yes | P0MAT.0<br>INT0.0<br>INT1.0 | ADC0.0<br>CP0P.0<br>CP0N.0<br>VREF |
| P0.1 | Standard I/O | 1 | Yes | P0MAT.1<br>INT0.1<br>INT1.1 | ADC0.1<br>CP0P.1<br>CP0N.1<br>AGND |
| P0.2 | Standard I/O | 20 | Yes | P0MAT.2<br>INT0.2<br>INT1.2 | ADC0.2<br>CP0P.2<br>CP0N.2 |
| P0.3 | Standard I/O | 19 | Yes | P0MAT.3<br>EXTCLK<br>INT0.3<br>INT1.3 | ADC0.3<br>CP0P.3<br>CP0N.3 |
| P0.4 | Standard I/O | 18 | Yes | P0MAT.4<br>INT0.4<br>INT1.4 | ADC0.4<br>CP0P.4<br>CP0N.4 |
| P0.5 | Standard I/O | 17 | Yes | P0MAT.5<br>INT0.5<br>INT1.5 | ADC0.5<br>CP0P.5<br>CP0N.5 |
| P0.6 | Standard I/O | 16 | Yes | P0MAT.6<br>CNVSTR<br>INT0.6<br>INT1.6 | ADC0.6<br>CP0P.6<br>CP0N.6 |
| P0.7 | Standard I/O | 15 | Yes | P0MAT.7<br>INT0.7<br>INT1.7 | ADC0.7<br>CP0P.7<br>CP0N.7 |

**Table 3.2. Pin Definitions for C8051F850/1/2/3/4/5-GM and C8051F850/1/2/3/4/5-IM**

| Pin Name | Type | Pin Numbers | Crossbar Capability | Additional Digital Functions | Analog Functions |
|---|---|---|---|---|---|
| P1.0 | Standard I/O | 14 | Yes | P1MAT.0 | ADC0.8<br>CP1P.0<br>CP1N.0 |
| P1.1 | Standard I/O | 13 | Yes | P1MAT.1 | ADC0.9<br>CP1P.1<br>CP1N.1 |
| P1.2 | Standard I/O | 11 | Yes | P1MAT.2 | ADC0.10<br>CP1P.2<br>CP1N.2 |
| P1.3 | Standard I/O | 10 | Yes | P1MAT.3 | ADC0.11<br>CP1P.3<br>CP1N.3 |
| P1.4 | Standard I/O | 9 | Yes | P1MAT.4 | ADC0.12<br>CP1P.4<br>CP1N.4 |
| P1.5 | Standard I/O | 8 | Yes | P1MAT.5 | ADC0.13<br>CP1P.5<br>CP1N.5 |
| P1.6 | Standard I/O | 7 | Yes | P1MAT.6 | ADC0.14<br>CP1P.6<br>CP1N.6 |
| P2.0 /<br>C2D | Standard I/O /<br>C2 Debug Data | 6 | | | |

## 3.3. C8051F860/1/2/3/4/5 SOIC16 Pin Definitions

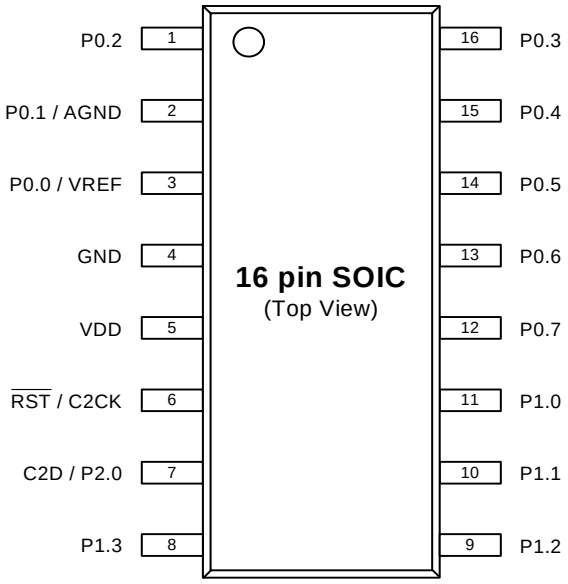



**Figure 3.3. C8051F860/1/2/3/4/5-GS and C8051F860/1/2/3/4/5-IS Pinout**

**Table 3.3. Pin Definitions for C8051F860/1/2/3/4/5-GS and C8051F860/1/2/3/4/5-IS**

| Pin Name | Type | Pin Numbers | Crossbar Capability | Additional Digital Functions | Analog Functions |
|---|---|---|---|---|---|
| GND | Ground | 4 | | | |
| VDD | Power | 5 | | | |
| $\overline{\text{RST}}$ / C2CK | Active-low Reset / C2 Debug Clock | 6 | | | |
| P0.0 | Standard I/O | 3 | Yes | P0MAT.0<br>INT0.0<br>INT1.0 | ADC0.0<br>CP0P.0<br>CP0N.0 |

**Table 3.3. Pin Definitions for C8051F860/1/2/3/4/5-GS and C8051F860/1/2/3/4/5-IS**

| Pin Name | Type | Pin Numbers | Crossbar Capability | Additional Digital Functions | Analog Functions |
|---|---|---|---|---|---|
| P0.1 | Standard I/O | 2 | Yes | P0MAT.1<br>INT0.1<br>INT1.1 | ADC0.1<br>CP0P.1<br>CP0N.1 |
| P0.2 | Standard I/O | 1 | Yes | P0MAT.2<br>INT0.2<br>INT1.2 | ADC0.2<br>CP0P.2<br>CP0N.2 |
| P0.3 /<br>EXTCLK | Standard I/O /<br>External CMOS Clock Input | 16 | Yes | P0MAT.3<br>EXTCLK<br>INT0.3<br>INT1.3 | ADC0.3<br>CP0P.3<br>CP0N.3 |
| P0.4 | Standard I/O | 15 | Yes | P0MAT.4<br>INT0.4<br>INT1.4 | ADC0.4<br>CP0P.4<br>CP0N.4 |
| P0.5 | Standard I/O | 14 | Yes | P0MAT.5<br>INT0.5<br>INT1.5 | ADC0.5<br>CP0P.5<br>CP0N.5 |
| P0.6 | Standard I/O | 13 | Yes | P0MAT.6<br>CNVSTR<br>INT0.6<br>INT1.6 | ADC0.6<br>CP1P.0<br>CP1N.0 |
| P0.7 | Standard I/O | 12 | Yes | P0MAT.7<br>INT0.7<br>INT1.7 | ADC0.7<br>CP1P.1<br>CP1N.1 |
| P1.0 | Standard I/O | 11 | Yes | P1MAT.0 | ADC0.8<br>CP1P.2<br>CP1N.2 |

**Table 3.3. Pin Definitions for C8051F860/1/2/3/4/5-GS and C8051F860/1/2/3/4/5-IS**

| Pin Name | Type | Pin Numbers | Crossbar Capability | Additional Digital Functions | Analog Functions |
|---|---|---|---|---|---|
| P1.1 | Standard I/O | 10 | Yes | P1MAT.1 | ADC0.9<br>CP1P.3<br>CP1N.3 |
| P1.2 | Standard I/O | 9 | Yes | P1MAT.2 | ADC0.10<br>CP1P.4<br>CP1N.4 |
| P1.3 | Standard I/O | 8 | Yes | P1MAT.3 | ADC0.11<br>CP1P.5<br>CP1N.5 |
| P2.0 /<br>C2D | Standard I/O /<br>C2 Debug Data | 7 | | | |

## 4. Ordering Information

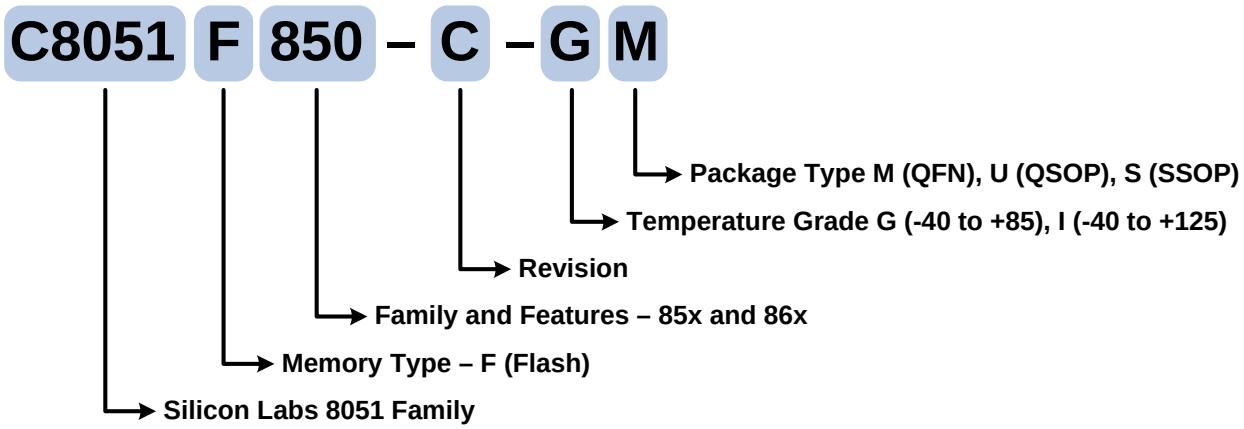



**Figure 4.1. C8051F85x/86x Part Numbering**

All C8051F85x/86x family members have the following features:

- CIP-51 Core running up to 25 MHz
- Two Internal Oscillators (24.5 MHz and 80 kHz)
- I2C/SMBus
- SPI
- UART
- 3-Channel Programmable Counter Array (PWM, Clock Generation, Capture/Compare)
- 4 16-bit Timers
- 2 Analog Comparators
- 16-bit CRC Unit

In addition to these features, each part number in the C8051F85x/86x family has a set of features that vary across the product line. The product selection guide in Table 4.1 shows the features available on each family member.

All devices in Table 4.1 are also available in an industrial version. For the industrial version, the -G in the ordering part number is replaced with -I. For example, the industrial version of the C8051F850-C-GM is the C8051F850-C-IM.

**Table 4.1. Product Selection Guide**

| Ordering Part Number | Flash Memory (kB) | RAM (Bytes) | Digital Port I/Os (Total) | Number of ADC0 Channels | I/O with Comparator 0/1 Inputs | Pb-free (RoHS Compliant) | AEC-Q100 Qualified | Temperature Range | Package |
|---|---|---|---|---|---|---|---|---|---|
| C8051F850-C-GM | 8 | 512 | 16 | 15 | 15 | ✓ | ✓ | -40 to 85 °C | QFN-20 |
| C8051F850-C-GU | 8 | 512 | 18 | 16 | 16 | ✓ | ✓ | -40 to 85 °C | QSOP-24 |
| C8051F851-C-GM | 4 | 512 | 16 | 15 | 15 | ✓ | ✓ | -40 to 85 °C | QFN-20 |
| C8051F851-C-GU | 4 | 512 | 18 | 16 | 16 | ✓ | ✓ | -40 to 85 °C | QSOP-24 |
| C8051F852-C-GM | 2 | 256 | 16 | 15 | 15 | ✓ | ✓ | -40 to 85 °C | QFN-20 |
| C8051F852-C-GU | 2 | 256 | 18 | 16 | 16 | ✓ | ✓ | -40 to 85 °C | QSOP-24 |
| C8051F853-C-GM | 8 | 512 | 16 | — | 15 | ✓ | ✓ | -40 to 85 °C | QFN-20 |
| C8051F853-C-GU | 8 | 512 | 18 | — | 16 | ✓ | ✓ | -40 to 85 °C | QSOP-24 |
| C8051F854-C-GM | 4 | 512 | 16 | — | 15 | ✓ | ✓ | -40 to 85 °C | QFN-20 |
| C8051F854-C-GU | 4 | 512 | 18 | — | 16 | ✓ | ✓ | -40 to 85 °C | QSOP-24 |
| C8051F855-C-GM | 2 | 256 | 16 | — | 15 | ✓ | ✓ | -40 to 85 °C | QFN-20 |
| C8051F855-C-GU | 2 | 256 | 18 | — | 16 | ✓ | ✓ | -40 to 85 °C | QSOP-24 |
| C8051F860-C-GS | 8 | 512 | 13 | 12 | 12 | ✓ | ✓ | -40 to 85 °C | SOIC-16 |
| C8051F861-C-GS | 4 | 512 | 13 | 12 | 12 | ✓ | ✓ | -40 to 85 °C | SOIC-16 |
| C8051F862-C-GS | 2 | 256 | 13 | 12 | 12 | ✓ | ✓ | -40 to 85 °C | SOIC-16 |
| C8051F863-C-GS | 8 | 512 | 13 | — | 12 | ✓ | ✓ | -40 to 85 °C | SOIC-16 |
| C8051F864-C-GS | 4 | 512 | 13 | — | 12 | ✓ | ✓ | -40 to 85 °C | SOIC-16 |
| C8051F865-C-GS | 2 | 256 | 13 | — | 12 | ✓ | ✓ | -40 to 85 °C | SOIC-16 |

**Table 4.1. Product Selection Guide**

| Ordering Part Number | Flash Memory (kB) | RAM (Bytes) | Digital Port I/Os (Total) | Number of ADC0 Channels | I/O with Comparator 0/1 Inputs | Pb-free (RoHS Compliant) | AEC-Q100 Qualified | Temperature Range | Package |
|---|---|---|---|---|---|---|---|---|---|
| -IM, -IU and -IS extended temperature range devices (-40 to 125 °C) are also available. | | | | | | | | | |

## 5. QSOP-24 Package Specifications

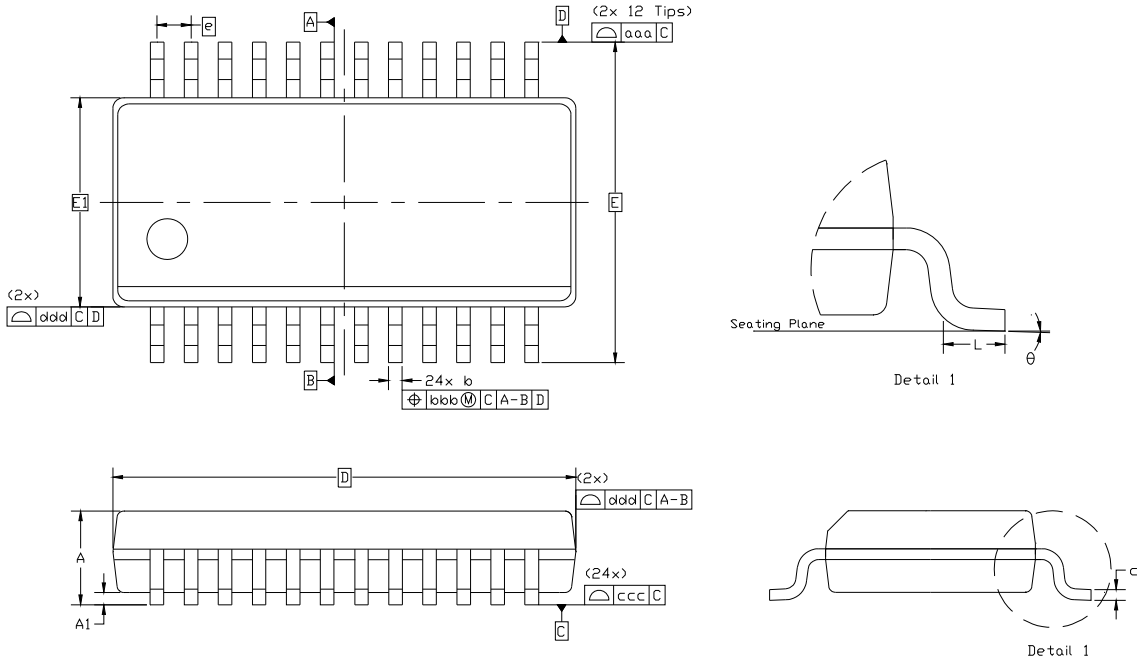



**Figure 5.1. QSOP-24 Package Drawing**

**Table 5.1. QSOP-24 Package Dimensions**

| Dimension | Min | Nom | Max |
|---|---|---|---|
| A | — | — | 1.75 |
| A1 | 0.10 | — | 0.25 |
| b | 0.20 | — | 0.30 |
| c | 0.10 | — | 0.25 |
| D | 8.65 BSC | | |
| E | 6.00 BSC | | |
| E1 | 3.90 BSC | | |

| Dimension | Min | Nom | Max |
|---|---|---|---|
| e | 0.635 BSC | | |
| L | 0.40 | — | 1.27 |
| θ | 0º | — | 8º |
| aaa | 0.20 | | |
| bbb | 0.18 | | |
| ccc | 0.10 | | |
| ddd | 0.10 | | |

**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-137, variation AE.
4. Recommended card reflow profile is per the JEDEC/IPC J-STD-020 specification for Small Body Components.

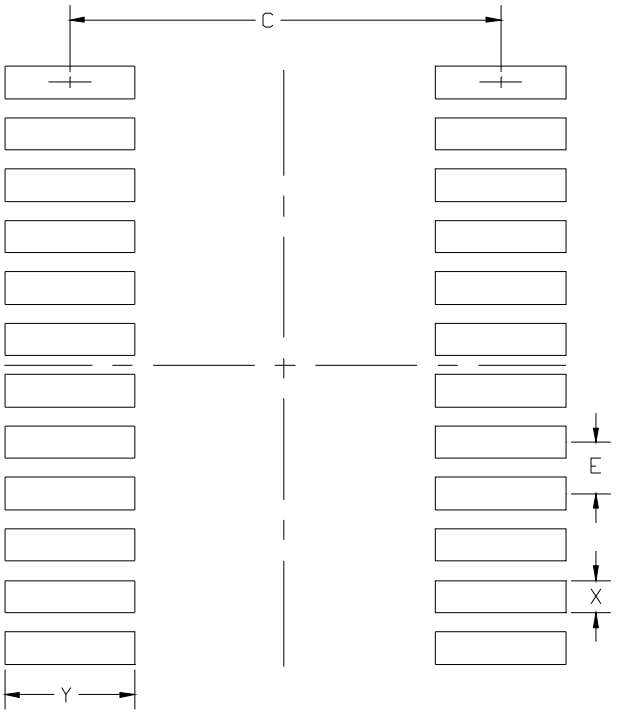



**Figure 5.2. QSOP-24 PCB Land Pattern**

**Table 5.2. QSOP-24 PCB Land Pattern Dimensions**

| 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**

3. 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**

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

**Card Assembly**

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

## 6. QFN-20 Package Specifications

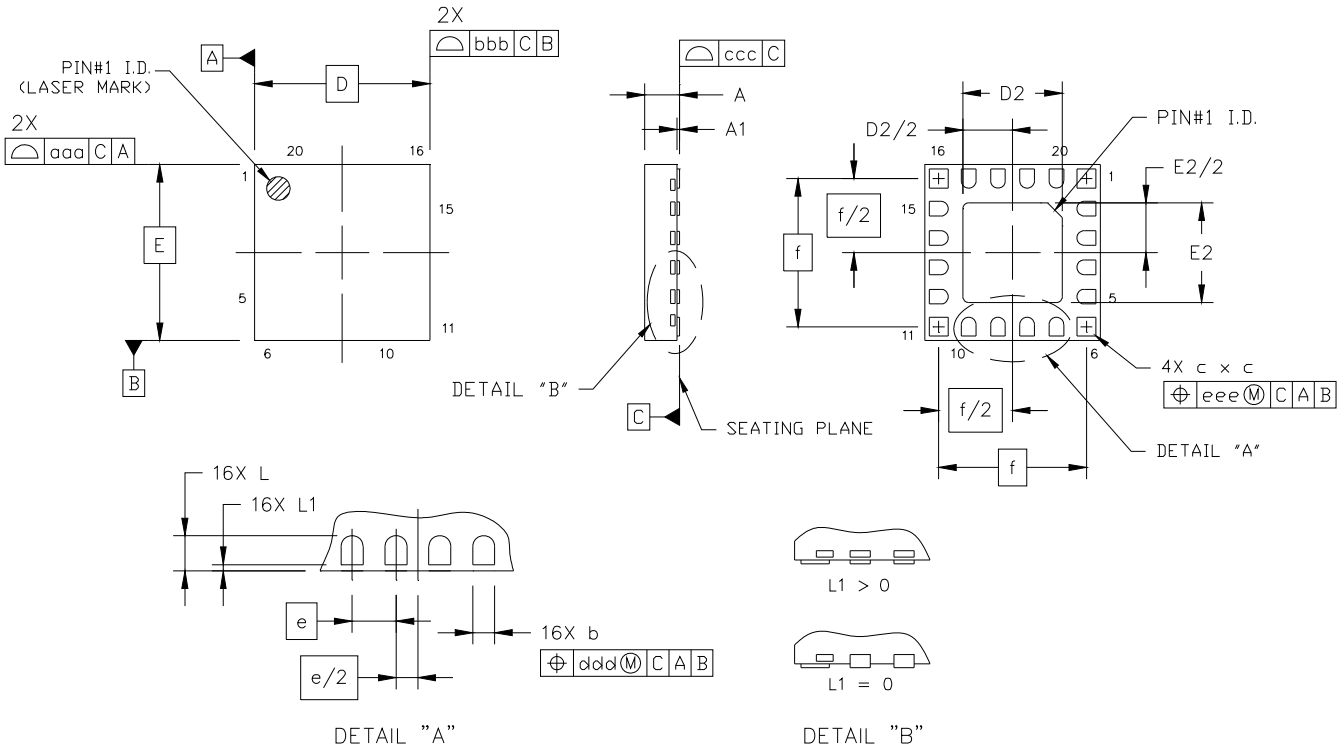



**Figure 6.1. QFN-20 Package Drawing**

**Table 6.1. QFN-20 Package Dimensions**

| Symbol | Millimeters | | |
|---|---|---|---|
| | Min | Nom | Max |
| A | 0.70 | 0.75 | 0.80 |
| A1 | 0.00 | 0.02 | 0.05 |
| b | 0.20 | 0.25 | 0.30 |
| c | 0.25 | 0.30 | 0.35 |
| D | | 3.00 BSC | |
| D2 | 1.6 | 1.70 | 1.8 |
| e | | 0.50 BSC | |
| E | | 3.00 BSC | |
| E2 | 1.6 | 1.70 | 1.8 |

| Symbol | Millimeters | | |
|---|---|---|---|
| | Min | Nom | Max |
| f | | 2.53 BSC | |
| L | 0.3 | 0.40 | 0.5 |
| L1 | 0.00 | — | 0.10 |
| aaa | — | — | 0.05 |
| bbb | — | — | 0.05 |
| ccc | — | — | 0.08 |
| ddd | — | — | 0.10 |
| eee | — | — | 0.10 |

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

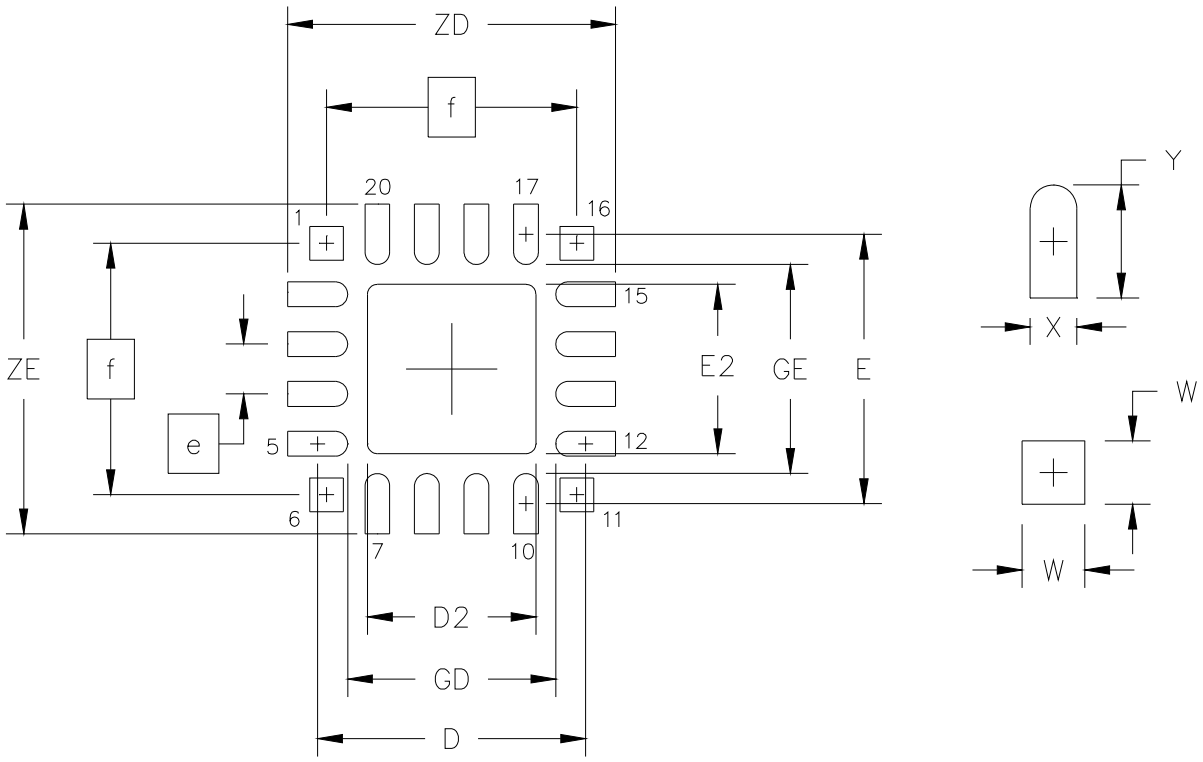



**Figure 6.2. QFN-20 Landing Diagram**

**Table 6.2. QFN-20 Landing Diagram Dimensions**

| Symbol | Millimeters | |
|---|---|---|
| | 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 BSC | |
| GD | 2.10 | — |

| Symbol | Millimeters | |
|---|---|---|
| | Min | Max |
| 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.

**Notes: 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.

**Notes: 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 the 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.

**Notes: 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.

## 7. SOIC-16 Package Specifications

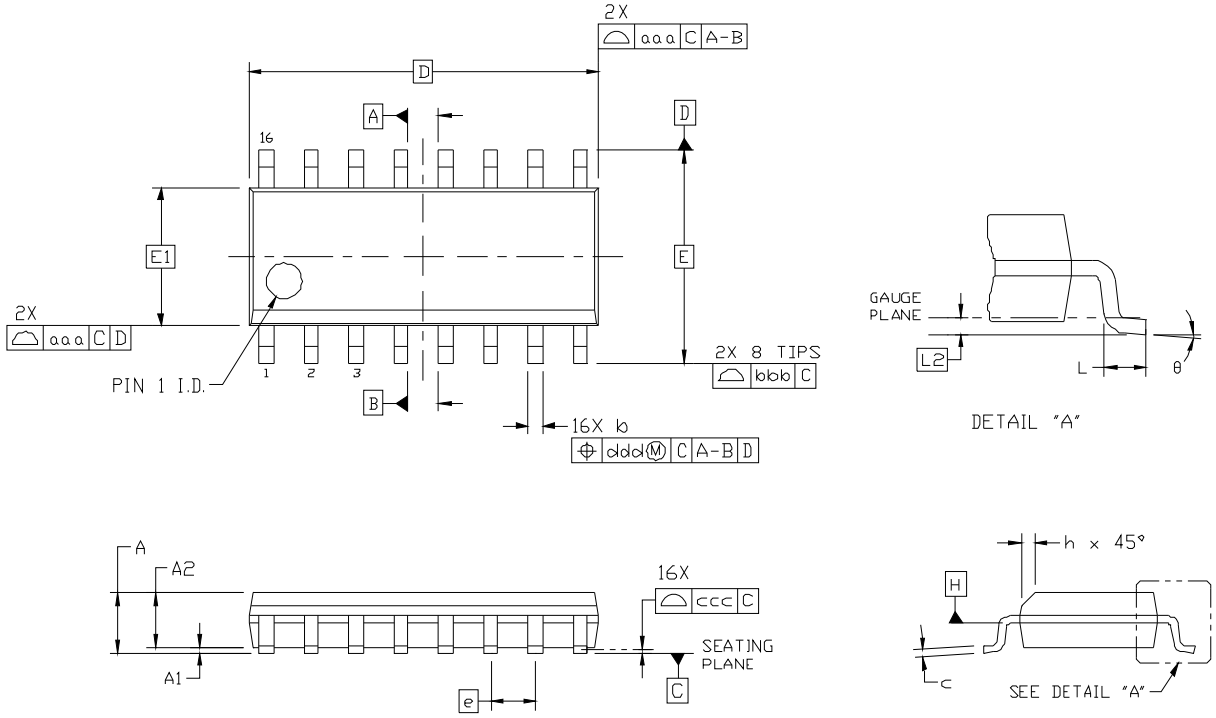



**Figure 7.1. SOIC-16 Package Drawing**

**Table 7.1. SOIC-16 Package Dimensions**

| Dimension | Min | Nom | Max |
|---|---|---|---|
| A | — | | 1.75 |
| A1 | 0.10 | | 0.25 |
| A2 | 1.25 | | — |
| b | 0.31 | | 0.51 |
| c | 0.17 | | 0.25 |
| D | | 9.90 BSC | |
| E | | 6.00 BSC | |
| E1 | | 3.90 BSC | |
| e | | 1.27 BSC | |

| Dimension | Min | Nom | Max |
|---|---|---|---|
| L | 0.40 | | 1.27 |
| L2 | | 0.25 BSC | |
| h | 0.25 | | 0.50 |
| θ | 0º | | 8º |
| aaa | | 0.10 | |
| bbb | | 0.20 | |
| ccc | | 0.10 | |
| ddd | | 0.25 | |

**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 MS-012, Variation AC.
4. Recommended card reflow profile is per the JEDEC/IPC J-STD-020 specification for Small Body Components.

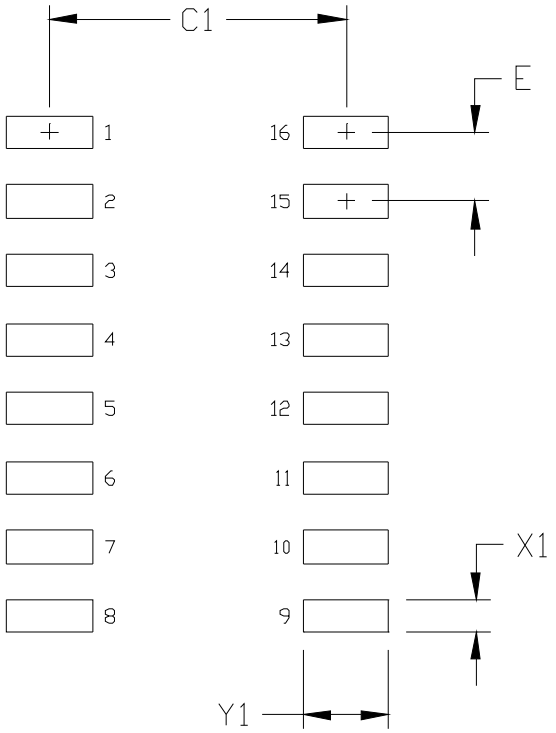



**Figure 7.2. SOIC-16 PCB Land Pattern**

**Table 7.2. SOIC-16 PCB Land Pattern Dimensions**

| Dimension | Feature | (mm) |
|---|---|---|
| C1 | Pad Column Spacing | 5.40 |
| E | Pad Row Pitch | 1.27 |
| X1 | Pad Width | 0.60 |
| Y1 | Pad Length | 1.55 |

**Notes:**
General

1. All dimensions shown are in millimeters (mm) unless otherwise noted.
2. This Land Pattern Design is based on IPC-7351 pattern SOIC127P600X165-16N for Density Level B (Median Land Protrusion).
3. All feature sizes shown are at Maximum Material Condition (MMC) and a card fabrication tolerance of 0.05 mm is assumed.

## 8. 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 C8051F85x/86x device family is shown in Figure 8.1.

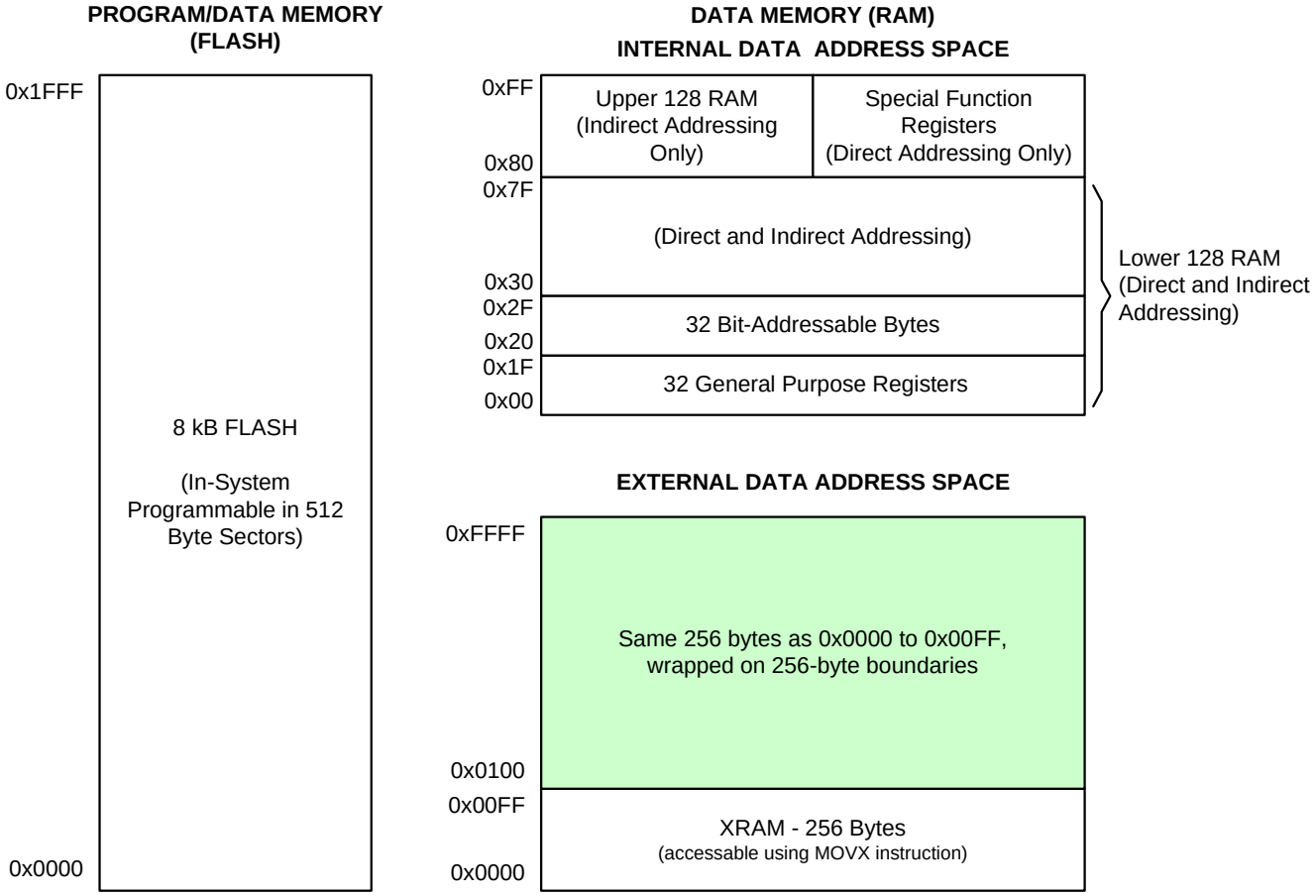



**Figure 8.1. C8051F85x/86x Memory Map (8 kB flash version shown)**

## 8.1. Program Memory

The CIP-51 core has a 64 kB program memory space. The C8051F85x/86x family implements 8 kB, 4 kB or 2 kB of this program memory space as in-system, re-programmable flash memory. The last address in the flash block (0x1FFF on 8 kB devices, 0x0FFF on 4 kB devices and 0x07FF on 2 kB devices) serves as a security lock byte for the device, and provides read, write and erase protection. Addresses above the lock byte within the 64 kB address space are reserved.

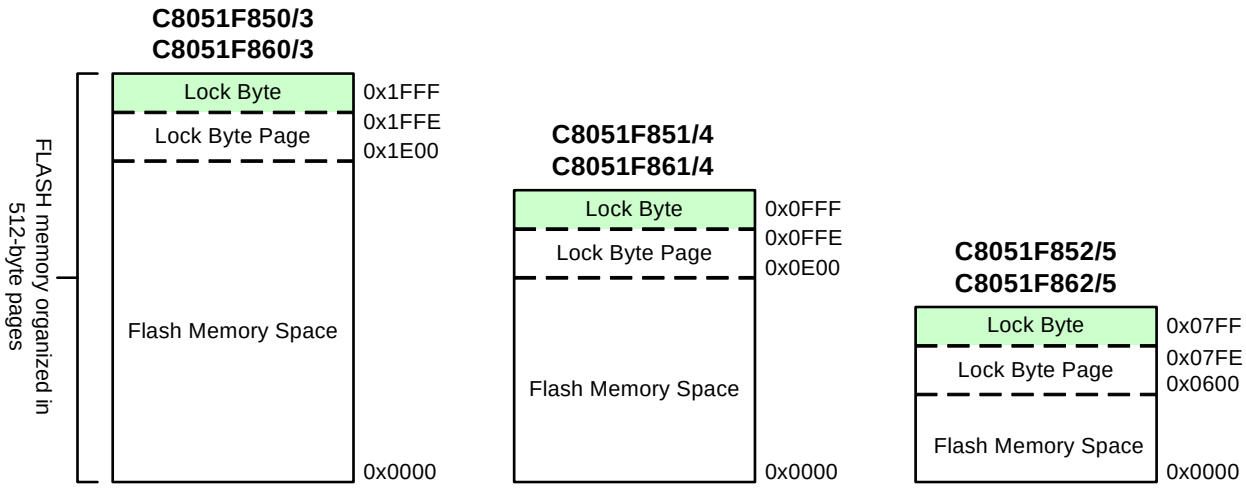



**Figure 8.2. Flash Program Memory Map**

### 8.1.1. MOVX Instruction and Program Memory

The MOVX instruction in an 8051 device is typically used to access external data memory. On the C8051F85x/86x 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 C8051F85x/86x to update program code and use the program memory space for non-volatile data storage. Refer to Section “10. Flash Memory” on page 61 for further details.

## 8.2. Data Memory

The C8051F85x/86x device family includes up to 512 bytes of RAM data memory. 256 bytes of this memory is mapped into the internal RAM space of the 8051. On devices with 512 bytes total RAM, 256 additional bytes of memory are available as on-chip “external” memory. The data memory map is shown in Figure 8.1 for reference.

### 8.2.1. 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 8.1 illustrates the data memory organization of the C8051F85x/86x.

Revision C C8051F852/5 and C8051F862/5 devices implement the upper four bytes of internal RAM as a 32-bit Unique Identifier. More information can be found in “Device Identification and Unique Identifier” on page 68.

#### 8.2.1.1. 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 (PSW) register, RS0 and RS1, select the active register bank. This allows fast context switching when entering subroutines and interrupt service routines. Indirect addressing modes use registers R0 and R1 as index registers.

#### 8.2.1.2. 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.

#### 8.2.1.3. 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.

### 8.2.2. External RAM

On devices with 512 bytes total 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. Note: The 16-bit MOVX instruction is also used for writes to the flash memory. See Section “10. Flash Memory” on page 61 for details. The MOVX instruction accesses XRAM by default.

For a 16-bit MOVX operation (@DPTR), the upper 8 bits of the 16-bit external data memory address word are "don't cares". As a result, addresses 0x0000 through 0x00FF are mapped modulo style over the entire 64 k external data memory address range. For example, the XRAM byte at address 0x0000 is shadowed at addresses 0x0100, 0x0200, 0x0300, 0x0400, etc.

Revision C C8051F850/1/3/4 and C8051F860/1/3/4 devices implement the upper four bytes of external RAM as a 32-bit Unique Identifier. More information can be found in “Device Identification and Unique Identifier” on page 68.

### 8.2.3. Special Function Registers

The direct-access data memory locations from 0x80 to 0xFF constitute the special function registers (SFRs). The SFRs provide control and data exchange with the CIP-51's resources and peripherals. The CIP-51 duplicates the SFRs found in a typical 8051 implementation as well as implementing additional SFRs used to configure and access the sub-systems unique to the MCU. This allows the addition of new functionality while retaining compatibility with the MCS-51™ instruction set.

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.

## 9. Special Function Register Memory Map

This section details the special function register memory map for the C8051F85x/86x devices.

**Table 9.1. Special Function Register (SFR) Memory Map**

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| F8 | SPI0CN | PCA0L | PCA0H | PCA0CPL0 | PCA0CPH0 | P0MAT | P0MASK | VDM0CN |
| F0 | B | P0MDIN | P1MDIN | EIP1 | - | - | PRTDRV | PCA0PWM |
| E8 | ADC0CN0 | PCA0CPL1 | PCA0CPH1 | PCA0CPL2 | PCA0CPH2 | P1MAT | P1MASK | RSTSRC |
| E0 | ACC | XBR0 | XBR1 | XBR2 | IT01CF | - | EIE1 | - |
| D8 | PCA0CN | PCA0MD | PCA0CPM0 | PCA0CPM1 | PCA0CPM2 | CRC0IN | CRC0DAT | ADC0PWR |
| D0 | PSW | REF0CN | CRC0AUTO | CRC0CNT | P0SKIP | P1SKIP | SMB0ADM | SMB0ADR |
| C8 | TMR2CN | REG0CN | TMR2RLL | TMR2RLH | TMR2L | TMR2H | CRC0CN | CRC0FLIP |
| C0 | SMB0CN | SMB0CF | SMB0DAT | ADC0GTL | ADC0GTH | ADC0LTL | ADC0LTH | OSCICL |
| B8 | IP | ADC0TK | - | ADC0MX | ADC0CF | ADC0L | ADC0H | CPT1CN |
| B0 | - | OSCLCN | ADC0CN1 | ADC0AC | - | DEVICEID | REVID | FLKEY |
| A8 | IE | CLKSEL | CPT1MX | CPT1MD | SMB0TC | DERIVID | - | - |
| A0 | P2 | SPI0CFG | SPI0CKR | SPI0DAT | P0MDOUT | P1MDOUT | P2MDOUT | - |
| 98 | SCON0 | SBUF0 | - | CPT0CN | PCA0CLR | CPT0MD | PCA0CENT | CPT0MX |
| 90 | P1 | TMR3CN | TMR3RLL | TMR3RLH | TMR3L | TMR3H | PCA0POL | WDTCN |
| 88 | TCON | TMOD | TL0 | TL1 | TH0 | TH1 | CKCON | PSCTL |
| 80 | P0 | SP | DPL | DPH | - | - | - | PCON |
| | 0(8) | 1(9) | 2(A) | 3(B) | 4(C) | 5(D) | 6(E) | 7(F) |

(bit addressable)

**Table 9.2. Special Function Registers**

| Register | Address | Register Description | Page |
|---|---|---|---|
| ACC | 0xE0 | Accumulator | 122 |
| ADC0AC | 0xB3 | ADC0 Accumulator Configuration | 102 |
| ADC0CF | 0xBC | ADC0 Configuration | 101 |
| ADC0CN0 | 0xE8 | ADC0 Control 0 | 99 |
| ADC0CN1 | 0xB2 | ADC0 Control 1 | 100 |
| ADC0GTH | 0xC4 | ADC0 Greater-Than High Byte | 107 |
| ADC0GTL | 0xC3 | ADC0 Greater-Than Low Byte | 108 |
| ADC0H | 0xBE | ADC0 Data Word High Byte | 105 |

**Table 9.2. Special Function Registers (Continued)**

| Register | Address | Register Description | Page |
|---|---|---|---|
| ADC0L | 0xBD | ADC0 Data Word Low Byte | 106 |
| ADC0LTH | 0xC6 | ADC0 Less-Than High Byte | 109 |
| ADC0LTL | 0xC5 | ADC0 Less-Than Low Byte | 110 |
| ADC0MX | 0xBB | ADC0 Multiplexer Selection | 111 |
| ADC0PWR | 0xDF | ADC0 Power Control | 103 |
| ADC0TK | 0xB9 | ADC0 Burst Mode Track Time | 104 |
| B | 0xF0 | B Register | 123 |
| CKCON | 0x8E | Clock Control | 269 |
| CLKSEL | 0xA9 | Clock Selection | 129 |
| CPT0CN | 0x9B | Comparator 0 Control | 134 |
| CPT0MD | 0x9D | Comparator 0 Mode | 135 |
| CPT0MX | 0x9F | Comparator 0 Multiplexer Selection | 136 |
| CPT1CN | 0xBF | Comparator 1 Control | 137 |
| CPT1MD | 0xAB | Comparator 1 Mode | 138 |
| CPT1MX | 0xAA | Comparator 1 Multiplexer Selection | 139 |
| CRC0AUTO | 0xD2 | CRC0 Automatic Control | 146 |
| CRC0CN | 0xCE | CRC0 Control | 143 |
| CRC0CNT | 0xD3 | CRC0 Automatic Flash Sector Count | 147 |
| CRC0DAT | 0xDE | CRC0 Data Output | 145 |
| CRC0FLIP | 0xCF | CRC0 Bit Flip | 148 |
| CRC0IN | 0xDD | CRC0 Data Input | 144 |
| DERIVID | 0xAD | Derivative Identification | 70 |
| DEVICEID | 0xB5 | Device Identification | 69 |
| DPH | 0x83 | Data Pointer Low | 120 |
| DPL | 0x82 | Data Pointer High | 119 |
| EIE1 | 0xE6 | Extended Interrupt Enable 1 | 78 |
| EIP1 | 0xF3 | Extended Interrupt Priority 1 | 80 |
| FLKEY | 0xB7 | Flash Lock and Key | 67 |

**Table 9.2. Special Function Registers (Continued)**

| Register | Address | Register Description | Page |
|---|---|---|---|
| IE | 0xA8 | Interrupt Enable | 75 |
| IP | 0xB8 | Interrupt Priority | 77 |
| IT01CF | 0xE4 | INT0 / INT1 Configuration | 150 |
| OSCICL | 0xC7 | High Frequency Oscillator Calibration | 127 |
| OSCLCN | 0xB1 | Low Frequency Oscillator Control | 128 |
| P0 | 0x80 | Port 0 Pin Latch | 199 |
| P0MASK | 0xFE | Port 0 Mask | 197 |
| P0MAT | 0xFD | Port 0 Match | 198 |
| P0MDIN | 0xF1 | Port 0 Input Mode | 200 |
| P0MDOUT | 0xA4 | Port 0 Output Mode | 201 |
| P0SKIP | 0xD4 | Port 0 Skip | 202 |
| P1 | 0x90 | Port 1 Pin Latch | 205 |
| P1MASK | 0xEE | Port 1 Mask | 203 |
| P1MAT | 0xED | Port 1 Match | 204 |
| P1MDIN | 0xF2 | Port 1 Input Mode | 206 |
| P1MDOUT | 0xA5 | Port 1 Output Mode | 207 |
| P1SKIP | 0xD5 | Port 1 Skip | 208 |
| P2 | 0xA0 | Port 2 Pin Latch | 209 |
| P2MDOUT | 0xA6 | Port 2 Output Mode | 210 |
| PCA0CENT | 0x9E | PCA Center Alignment Enable | 177 |
| PCA0CLR | 0x9C | PCA Comparator Clear Control | 170 |
| PCA0CN | 0xD8 | PCA Control | 167 |
| PCA0CPH0 | 0xFC | PCA Capture Module High Byte 0 | 175 |
| PCA0CPH1 | 0xEA | PCA Capture Module High Byte 1 | 181 |
| PCA0CPH2 | 0xEC | PCA Capture Module High Byte 2 | 183 |
| PCA0CPL0 | 0xFB | PCA Capture Module Low Byte 0 | 174 |
| PCA0CPL1 | 0xE9 | PCA Capture Module Low Byte 1 | 180 |
| PCA0CPL2 | 0xEB | PCA Capture Module Low Byte 2 | 182 |

**Table 9.2. Special Function Registers (Continued)**

| Register | Address | Register Description | Page |
|---|---|---|---|
| PCA0CPM0 | 0xDA | PCA Capture/Compare Mode 0 | 171 |
| PCA0CPM1 | 0xDB | PCA Capture/Compare Mode 1 | 178 |
| PCA0CPM2 | 0xDC | PCA Capture/Compare Mode 1 | 179 |
| PCA0H | 0xFA | PCA Counter/Timer Low Byte | 173 |
| PCA0L | 0xF9 | PCA Counter/Timer High Byte | 172 |
| PCA0MD | 0xD9 | PCA Mode | 168 |
| PCA0POL | 0x96 | PCA Output Polarity | 176 |
| PCA0PWM | 0xF7 | PCA PWM Configuration | 169 |
| PCON | 0x87 | Power Control | 83 |
| PRTDRV | 0xF6 | Port Drive Strength | 196 |
| PSCTL | 0x8F | Program Store Control | 66 |
| PSW | 0xD0 | Program Status Word | 124 |
| REF0CN | 0xD1 | Voltage Reference Control | 112 |
| REG0CN | 0xC9 | Voltage Regulator Control | 84 |
| REVID | 0xB6 | Revision Identification | 71 |
| RSTSRC | 0xEF | Reset Source | 215 |
| SBUF0 | 0x99 | UART0 Serial Port Data Buffer | 297 |
| SCON0 | 0x98 | UART0 Serial Port Control | 295 |
| SMB0ADM | 0xD6 | SMBus0 Slave Address Mask | 257 |
| SMB0ADR | 0xD7 | SMBus0 Slave Address | 256 |
| SMB0CF | 0xC1 | SMBus0 Configuration | 251 |
| SMB0CN | 0xC0 | SMBus0 Control | 254 |
| SMB0DAT | 0xC2 | SMBus0 Data | 258 |
| SMB0TC | 0xAC | SMBus0 Timing and Pin Control | 253 |
| SP | 0x81 | Stack Pointer | 121 |
| SPI0CFG | 0xA1 | SPI0 Configuration | 227 |
| SPI0CKR | 0xA2 | SPI0 Clock Control | 231 |
| SPI0CN | 0xF8 | SPI0 Control | 229 |

**Table 9.2. Special Function Registers (Continued)**

| Register | Address | Register Description | Page |
|---|---|---|---|
| SPI0DAT | 0xA3 | SPI0 Data | 232 |
| TCON | 0x88 | Timer 0/1 Control | 271 |
| TH0 | 0x8C | Timer 0 High Byte | 275 |
| TH1 | 0x8D | Timer 1 High Byte | 276 |
| TL0 | 0x8A | Timer 0 Low Byte | 273 |
| TL1 | 0x8B | Timer 1 Low Byte | 274 |
| TMOD | 0x89 | Timer 0/1 Mode | 272 |
| TMR2CN | 0xC8 | Timer 2 Control | 277 |
| TMR2H | 0xCD | Timer 2 High Byte | 282 |
| TMR2L | 0xCC | Timer 2 Low Byte | 281 |
| TMR2RLH | 0xCB | Timer 2 Reload High Byte | 280 |
| TMR2RLL | 0xCA | Timer 2 Reload Low Byte | 279 |
| TMR3CN | 0x91 | Timer 3 Control | 283 |
| TMR3H | 0x95 | Timer 3 High Byte | 288 |
| TMR3L | 0x94 | Timer 3 Low Byte | 287 |
| TMR3RLH | 0x93 | Timer 3 Reload High Byte | 286 |
| TMR3RLL | 0x92 | Timer 3 Reload Low Byte | 285 |
| VDM0CN | 0xFF | Supply Monitor Control | 216 |
| WDTCN | 0x97 | Watchdog Timer Control | 301 |
| XBR0 | 0xE1 | Port I/O Crossbar 0 | 193 |
| XBR1 | 0xE2 | Port I/O Crossbar 1 | 194 |
| XBR2 | 0xE3 | Port I/O Crossbar 2 | 195 |

# 10. Flash Memory

On-chip, re-programmable flash memory is included for program code and non-volatile data storage. The flash memory is organized in 512-byte pages. It can be erased and written through the C2 interface or from firmware by overloading the MOVX instruction. Any individual byte in flash memory must only be written once between page erase operations.

## 10.1. 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 in 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 "8. Memory Organization" on page 52 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 1's complement number represented by the Security Lock Byte. **Note that the page containing the flash Security Lock Byte is unlocked when no other flash pages are locked (all bits of the Lock Byte are '1') and locked when any other flash pages are locked (any bit of the Lock Byte is '0').** An example is shown in Figure 10.1.

| | |
|---|---|
| Security Lock Byte: | 11111101b |
| 1s Complement: | 00000010b |
| Flash pages locked: | 3 (First two flash pages + Lock Byte Page) |

**Figure 10.1. Security Byte Decoding**

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 10.1 summarizes the flash security features of the C8051F85x/86x devices.

**Table 10.1. 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 | Flash Error Reset | Permitted |
| Read or Write page containing Lock Byte (if no pages are locked) | Permitted | Permitted | N/A |
| Read or Write page containing Lock Byte (if any page is locked) | Not Permitted | Flash Error Reset | Permitted |

**Table 10.1. Flash Security Summary (Continued)**

| | | | |
|---|---|---|---|
| Read contents of Lock Byte (if no pages are locked) | Permitted | Permitted | N/A |
| Read contents of Lock Byte (if any page is locked) | Not Permitted | Flash Error Reset | Permitted |
| Erase page containing Lock Byte (if no pages are locked) | Permitted | Permitted | N/A |
| Erase page containing Lock Byte—Unlock all pages (if any page is locked) | C2 Device Erase Only | Flash Error Reset | Flash Error Reset |
| Lock additional pages (change 1s to 0s in the Lock Byte) | Not Permitted | Flash Error Reset | Flash Error Reset |
| Unlock individual pages (change 0s to 1s in the Lock Byte) | Not Permitted | Flash Error Reset | Flash Error Reset |
| Read, Write or Erase Reserved Area | Not Permitted | Flash Error Reset | Flash Error Reset |

C2 Device Erase—Erases all flash pages including the page containing the Lock Byte.

Flash Error Reset —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.

## 10.2. Programming the Flash Memory

Writes to flash memory clear bits from logic 1 to logic 0, and can be performed on single byte locations. Flash erasures set bits back to logic 1, and occur only on full pages. The write and erase operations are automatically timed by hardware for proper execution; data polling to determine the end of the write/erase operation is not required. Code execution is stalled during a flash write/erase operation.

The simplest means of programming the flash memory is through the C2 interface using programming tools provided by Silicon Labs or a third party vendor. This is the only means for programming a non-initialized device.

To ensure the integrity of flash contents, it is strongly recommended that the on-chip supply monitor be enabled in any system that includes code that writes and/or erases flash memory from software.

### 10.2.1. 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.

### 10.2.2. Flash Erase Procedure

The flash memory can be programmed by software using the MOVX write instruction with the address and data byte to be programmed provided as normal operands. Before writing to flash memory using MOVX, flash write operations must be enabled by: (1) setting the PSWE Program Store Write Enable bit in the PSCTL register to logic 1 (this directs the MOVX writes to target flash memory); and (2) Writing the flash key codes in sequence to the Flash Lock register (FLKEY). The PSWE bit remains set until cleared by software.

A write to flash memory can clear bits to logic 0 but cannot set them; only an erase operation can set bits to logic 1 in flash. **A byte location to be programmed should be erased before a new value is written.** Erase operation applies to an entire page (setting all bytes in the page to 0xFF). To erase an entire page, perform the following steps:

1. Disable interrupts (recommended).
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.

### 10.2.3. Flash Write Procedure

Flash bytes are programmed by software with the following sequence:

1. Disable interrupts (recommended).
2. Erase the flash page containing the target location, as described in Section 10.2.2.
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 desired

page.

8. Clear the PSWE bit.

Steps 5–7 must be repeated for each byte to be written. After flash writes are complete, PSWE should be cleared so that MOVX instructions do not target program memory.

## 10.3. Non-Volatile Data Storage

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

## 10.4. Flash Write and Erase Guidelines

Any system which contains routines which write or erase flash memory from software involves some risk that the write or erase routines will execute unintentionally if the CPU is operating outside its specified operating range of supply voltage, 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, hardware restricts flash writes and erasures when the supply monitor is not active and selected as a reset source. As the monitor is enabled and selected as a reset source by default, it is recommended that systems writing or erasing flash simply maintain the default state.

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

### 10.4.1. Voltage Supply Maintenance and the Supply 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 supply rise time specification is met. If the system cannot meet this rise time specification, then add an external supply brownout circuit to the RST pin of the device that holds the device in reset until the voltage supply reaches the lower limit, and re-asserts RST if the supply drops below the low supply limit.
3. Do not disable the supply monitor. If the supply monitor must be disabled in the system, firmware should be added to the startup routine to enable the on-chip supply monitor and enable the supply 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 may 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 supply monitor and enabling the supply monitor as a reset source. Code examples showing this can be found in “AN201: Writing to Flash From Firmware", available from the Silicon Laboratories web site. **Note that the supply monitor must be enabled and enabled as a reset source when writing or erasing flash memory. A flash error reset will occur if either condition is not met.**
4. As an added precaution if the supply monitor is ever disabled, explicitly enable the supply monitor and enable the supply monitor as a reset source inside the functions that write and erase flash memory. The supply 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. "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.

### 10.4.2. PSWE Maintenance

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

### 10.4.3. System Clock

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

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

## 10.5. Flash Control Registers

### Register 10.1. PSCTL: Program Store Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Reserved | | | | | | PSEE | PSWE |
| Type | R | | | | | | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0x8F** | | | | | | | | |

**Table 10.2. PSCTL Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:2 | Reserved | Must write reset value. |
| 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. |

## Register 10.2. FLKEY: Flash Lock and Key

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | FLKEY | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xB7** | | | | | | | | |

**Table 10.3. FLKEY Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | FLKEY | **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 are disabled until the next reset. |

## 11. Device Identification and Unique Identifier

The C8051F85x/86x has SFRs that identify the device family, derivative, and revision. These SFRs can be read by firmware at runtime to determine the capabilities of the MCU that is executing code. This allows the same firmware image to run on MCUs with different memory sizes and peripherals, and dynamically change functionality to suit the capabilities of that MCU.

In addition to the device identification registers, a 32-bit unique identifier (UID) is pre-programmed into all Revision C and later devices. The UID resides in the last four bytes of XRAM (C8051F850/1/3/4 and C8051F860/1/3/4) or RAM (C8051F852/5 and C8051F862/5). For devices with the UID in RAM, the UID can be read by firmware using indirect data accesses. For devices with the UID in XRAM, the UID can be read by firmware using MOVX instructions. The UID can also be read through the debug port for all devices.

Firmware can overwrite the UID during normal operation, and the bytes in memory will be automatically reinitialized with the UID value after any device reset. Firmware using this area of memory should always initialize the memory to a known value, as any previous data stored at these locations will be overwritten and not retained through a reset.

**Table 11.1. UID Implementation Information**

| Device | Memory Segment | Addresses |
|---|---|---|
| C8051F850<br>C8051F851<br>C8051F853<br>C8051F854<br>C8051F860<br>C8051F861<br>C8051F863<br>C8051F864 | XRAM | **(MSB)** 0x00FF, 0x00FE, 0x00FD, 0x00FC **(LSB)** |
| C8051F852<br>C8051F855<br>C8051F862<br>C8051F865 | RAM (indirect) | **(MSB)** 0xFF, 0xFE, 0xFD, 0xFC **(LSB)** |

## 11.1. Device Identification Registers

### Register 11.1. DEVICEID: Device Identification

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | DEVICEID | | | | | | | |
| Type | R | | | | | | | |
| Reset | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xB5** | | | | | | | | |

**Table 11.2. DEVICEID Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | DEVICEID | **Device ID.**<br>This read-only register returns the 8-bit device ID: 0x30 (C8051F85x/86x). |

## Register 11.2. DERIVID: Derivative Identification

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | DERIVID | | | | | | | |
| Type | R | | | | | | | |
| Reset | X | X | X | X | X | X | X | X |
| **SFR Address: 0xAD** | | | | | | | | |

**Table 11.3. DERIVID Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | DERIVID | **Derivative ID.**<br>This read-only register returns the 8-bit derivative ID, which can be used by firmware to identify which device in the product family the code is executing on. The '{R}' tag in the part numbers below indicates the device revision letter in the ordering code.<br>0xD0: C8051F850-{R}-GU<br>0xD1: C8051F851-{R}-GU<br>0xD2: C8051F852-{R}-GU<br>0xD3: C8051F853-{R}-GU<br>0xD4: C8051F854-{R}-GU<br>0xD5: C8051F855-{R}-GU<br>0xE0: C8051F860-{R}-GS<br>0xE1: C8051F861-{R}-GS<br>0xE2: C8051F862-{R}-GS<br>0xE3: C8051F863-{R}-GS<br>0xE4: C8051F864-{R}-GS<br>0xE5: C8051F865-{R}-GS<br>0xF0: C8051F850-{R}-GM<br>0xF1: C8051F851-{R}-GM<br>0xF2: C8051F852-{R}-GM<br>0xF3: C8051F853-{R}-GM<br>0xF4: C8051F854-{R}-GM<br>0xF5: C8051F855-{R}-GM |

## Register 11.3. REVID: Revision Identifcation

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | REVID | | | | | | | |
| Type | R | | | | | | | |
| Reset | X | X | X | X | X | X | X | X |
| **SFR Address: 0xB6** | | | | | | | | |

**Table 11.4. REVID Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | REVID | **Revision ID.**<br>This read-only register returns the 8-bit revision ID.<br>00000000: Revision A<br>00000001: Revision B<br>00000010: Revision C<br>00000011-11111111: Reserved. |

# 12. Interrupts

The C8051F85x/86x includes an extended interrupt system supporting multiple interrupt sources with two priority levels. The allocation of interrupt sources between on-chip peripherals and external input pins varies according to the specific version of the device. Each interrupt source has one or more associated interrupt-pending flag(s) located in an SFR. When a peripheral or external source meets a valid interrupt condition, the associated interrupt-pending flag is set to logic 1.

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

Each interrupt source can be individually enabled or disabled through the use of an associated interrupt enable bit in an SFR (IE and EIE1). However, interrupts must first be globally enabled by setting the EA bit in the IE register 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.

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

## 12.1. MCU Interrupt Sources and Vectors

The C8051F85x/86x MCUs support interrupt sources for each peripheral on the device. Software can simulate an interrupt by setting any interrupt-pending flag to logic 1. If interrupts are enabled for the flag, an interrupt request will be generated and the CPU will vector to the ISR address associated with the interrupt-pending flag. MCU interrupt sources, associated vector addresses, priority order and control bits are summarized in Table 12.1. Refer to the datasheet section associated with a particular on-chip peripheral for information regarding valid interrupt conditions for the peripheral and the behavior of its interrupt-pending flag(s).

### 12.1.1. Interrupt Priorities

Each interrupt source can be individually programmed to one of two priority levels: low or high. A low priority interrupt service routine can be preempted by a high priority interrupt. A high priority interrupt cannot be preempted. Each interrupt has an associated interrupt priority bit in an SFR (IP or EIP1) used to configure its priority level. Low priority is the default. If two interrupts are recognized simultaneously, the interrupt with the higher priority is serviced first. If both interrupts have the same priority level, a fixed priority order is used to arbitrate, given in Table 12.1.

### 12.1.2. 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 5 system clock cycles: 1 clock cycle to detect the interrupt and 4 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 18 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 4 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. If more than one interrupt is pending when the CPU exits an ISR, the CPU will service the next highest priority interrupt that is pending.

**Table 12.1. Interrupt Summary**

| Interrupt Source | Interrupt Vector | Priority Order | Pending Flags | Bit addressable? | Cleared by HW? | Enable Flag |
|---|---|---|---|---|---|---|
| Reset | 0x0000 | Top | None | N/A | N/A | Always Enabled |
| External Interrupt 0 ($\overline{INT0}$) | 0x0003 | 0 | IE0 (TCON.1) | Y | Y | EX0 (IE.0) |
| Timer 0 Overflow | 0x000B | 1 | TF0 (TCON.5) | Y | Y | ET0 (IE.1) |
| External Interrupt 1 ($\overline{INT1}$) | 0x0013 | 2 | IE1 (TCON.3) | Y | Y | EX1 (IE.2) |
| Timer 1 Overflow | 0x001B | 3 | TF1 (TCON.7) | Y | Y | ET1 (IE.3) |
| UART0 | 0x0023 | 4 | RI (SCON0.0)<br>TI (SCON0.1) | Y | N | ES0 (IE.4) |
| Timer 2 Overflow | 0x002B | 5 | TF2H (TMR2CN.7)<br>TF2L (TMR2CN.6) | Y | N | ET2 (IE.5) |
| SPI0 | 0x0033 | 6 | SPIF (SPI0CN.7)<br>WCOL (SPI0CN.6)<br>MODF (SPI0CN.5)<br>RXOVRN (SPI0CN.4) | Y | N | ESPI0 (IE.6) |
| SMB0 | 0x003B | 7 | SI (SMB0CN.0) | Y | N | ESMB0 (EIE1.0) |
| Port Match | 0x0043 | 8 | None | N/A | N/A | EMAT (EIE1.1) |
| ADC0 Window Compare | 0x004B | 9 | ADWINT (ADC0CN.3) | Y | N | EWADC0 (EIE1.2) |
| ADC0 Conversion Complete | 0x0053 | 10 | ADINT (ADC0CN.5) | Y | N | EADC0 (EIE1.3) |
| Programmable Counter Array | 0x005B | 11 | CF (PCA0CN.7)<br>CCFn (PCA0CN.n)<br>COVF (PCA0PWM.6) | Y | N | EPCA0 (EIE1.4) |
| Comparator0 | 0x0063 | 12 | CPFIF (CPT0CN.4)<br>CPRIF (CPT0CN.5) | N | N | ECP0 (EIE1.5) |
| Comparator1 | 0x006B | 13 | CPFIF (CPT1CN.4)<br>CPRIF (CPT1CN.5) | N | N | ECP1 (EIE1.6) |
| Timer 3 Overflow | 0x0073 | 14 | TF3H (TMR3CN.7)<br>TF3L (TMR3CN.6) | N | N | ET3 (EIE1.7) |

## 12.2. Interrupt Control Registers

### Register 12.1. IE: Interrupt Enable

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | EA | ESPI0 | ET2 | ES0 | ET1 | EX1 | ET0 | EX0 |
| Type | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xA8 (bit-addressable)** | | | | | | | | |

**Table 12.2. IE Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7 | EA | **Enable All Interrupts.**<br>Globally enables/disables all interrupts and 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 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 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. |

**Table 12.2. IE Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 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 INT0 input. |

## Register 12.2. IP: Interrupt Priority

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Reserved | PSPI0 | PT2 | PS0 | PT1 | PX1 | PT0 | PX0 |
| Type | R | RW | RW | RW | RW | RW | RW | RW |
| Reset | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| SFR Address: 0xB8 (bit-addressable) | | | | | | | | |

**Table 12.3. IP Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7 | Reserved | Must write reset value. |
| 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. |

## Register 12.3. EIE1: Extended Interrupt Enable 1

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ET3 | ECP1 | ECP0 | EPCA0 | EADC0 | EWADC0 | EMAT | ESMB0 |
| Type | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xE6** | | | | | | | | |

**Table 12.4. EIE1 Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7 | ET3 | **Enable Timer 3 Interrupt.**<br>This bit sets the masking of the Timer 3 interrupt.<br>0: Disable Timer 3 interrupts.<br>1: Enable interrupt requests generated by the TF3L or TF3H flags. |
| 6 | ECP1 | **Enable Comparator1 (CP1) Interrupt.**<br>This bit sets the masking of the CP1 interrupt.<br>0: Disable CP1 interrupts.<br>1: Enable interrupt requests generated by the comparator 1 CPRIF or CPFIF flags. |
| 5 | ECP0 | **Enable Comparator0 (CP0) Interrupt.**<br>This bit sets the masking of the CP0 interrupt.<br>0: Disable CP0 interrupts.<br>1: Enable interrupt requests generated by the comparator 0 CPRIF or CPFIF 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 ADINT 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 (ADWINT). |
| 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. |

**Table 12.4. EIE1 Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 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. |

## Register 12.4. EIP1: Extended Interrupt Priority 1

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | PT3 | PCP1 | PCP0 | PPCA0 | PADC0 | PWADC0 | PMAT | PSMB0 |
| Type | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xF3** | | | | | | | | |

**Table 12.5. EIP1 Register Bit Descriptions**

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

**Table 12.5. EIP1 Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 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. |

# 13. Power Management and Internal Regulator

All internal circuitry on the C8051F85x/86x devices draws power from the VDD supply pin. Circuits with external connections (I/O pins, analog muxes) are powered directly from the VDD supply voltage, while most of the internal circuitry is supplied by an on-chip LDO regulator. The regulator output is fully internal to the device, and is available also as an ADC input or reference source for the comparators and ADC.

The devices support the standard 8051 power modes: idle and stop. For further power savings in stop mode, the internal LDO regulator may be disabled, shutting down the majority of the power nets on the device.

Although the C8051F85x/86x has idle and stop modes available, more control over the device power can be achieved by enabling/disabling individual peripherals as needed. Each analog peripheral can be disabled when not in use and placed in low power mode. Digital peripherals, such as timers and serial buses, have their clocks gated off and draw little power when they are not in use.

## 13.1. Power Modes

Idle mode halts the CPU while leaving the peripherals and clocks active. In stop mode, the CPU is halted, all interrupts and timers are inactive, and the internal oscillator is stopped (analog peripherals remain in their selected states; the external oscillator is not affected). Since clocks are running in Idle mode, power consumption is dependent upon the system clock frequency and the number of peripherals left in active mode before entering Idle. Stop mode consumes the least power because the majority of the device is shut down with no clocks active. The Power Control Register (PCON) is used to control the C8051F85x/86x's Stop and Idle power management modes.

### 13.1.1. Idle Mode

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

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

**Note:** If the instruction following the write of the IDLE bit is a single-byte instruction and an interrupt occurs during the execution phase of the instruction that sets the IDLE bit, the CPU may not wake from Idle mode when a future interrupt occurs. Therefore, instructions that set the IDLE bit should be followed by an instruction that has two or more opcode bytes, for example:

```
// in 'C':
PCON |= 0x01;                  // set IDLE bit
PCON = PCON;                   // ... followed by a 3-cycle dummy instruction

; in assembly:
ORL PCON, #01h                 ; set IDLE bit
MOV PCON, PCON                 ; ... followed by a 3-cycle dummy instruction
```

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.

### 13.1.2. Stop Mode

Setting the Stop Mode Select bit (PCON.1) causes the controller core to enter stop mode as soon as the instruction that sets the bit completes execution. Before entering stop mode, the system clock must be sourced by the internal high-frequency oscillator. In stop mode the internal oscillator, CPU, and all digital peripherals are stopped; the state of the external oscillator circuit is not affected. Each analog peripheral (including the external oscillator circuit) may be shut down individually prior to entering stop mode. Stop mode can only be terminated by an internal or external reset. On reset, the device 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.

## 13.2. LDO Regulator

C8051F85x/86x devices include an internal regulator that regulates the internal core and logic supply. Under default conditions, the internal regulator will remain on when the device enters STOP mode. This allows any enabled reset source to generate a reset for the device and bring the device out of STOP mode. For additional power savings, the STOPCF bit can be used to shut down the regulator and the internal power network of the device when the part enters STOP mode. When STOPCF is set to 1, the $\overline{\text{RST}}$ pin and a full power cycle of the device are the only methods of generating a reset.

## 13.3. Power Control Registers

**Register 13.1. PCON: Power Control**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | GF | | | | | | STOP | IDLE |
| Type | RW | | | | | | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0x87** | | | | | | | | |

**Table 13.1. PCON Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:2 | GF | **General Purpose Flags 5-0.**<br>These are general purpose flags for use under software control. |
| 1 | STOP | **Stop Mode Select.**<br>Setting this bit will place the CIP-51 in Stop mode. This bit will always be read as 0. |
| 0 | IDLE | **Idle Mode Select.**<br>Setting this bit will place the CIP-51 in Idle mode. This bit will always be read as 0. |

## 13.4. LDO Control Registers

### Register 13.2. REG0CN: Voltage Regulator Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Reserved | | | | STOPCF | Reserved | | |
| Type | R | | | | RW | R | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xC9** | | | | | | | | |

**Table 13.2. REG0CN Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:4 | Reserved | Must write reset value. |
| 3 | STOPCF | **Stop Mode Configuration.**<br>This bit configures the regulator's behavior when the device enters stop mode.<br>0: Regulator is still active in stop mode. Any enabled reset source will reset the device.<br>1: Regulator is shut down in stop mode. Only the $\overline{\text{RST}}$ pin or power cycle can reset the device. |
| 2:0 | Reserved | Must write reset value. |

## 14. Analog-to-Digital Converter (ADC0)

The ADC is a successive-approximation-register (SAR) ADC with 12-, 10-, and 8-bit modes, integrated track-and-hold and a programmable window detector. These different modes allow the user to trade off speed for resolution. ADC0 also has an autonomous low-power burst mode which can automatically enable ADC0, capture and accumulate samples, then place ADC0 in a low power shutdown mode without CPU intervention. It also has a 16-bit accumulator that can automatically oversample and average the ADC results.

The ADC is fully configurable under software control via several registers. The ADC0 operates in single-ended mode and may be configured to measure different signals using the analog multiplexer. The voltage reference for the ADC is selectable between internal and external reference sources.

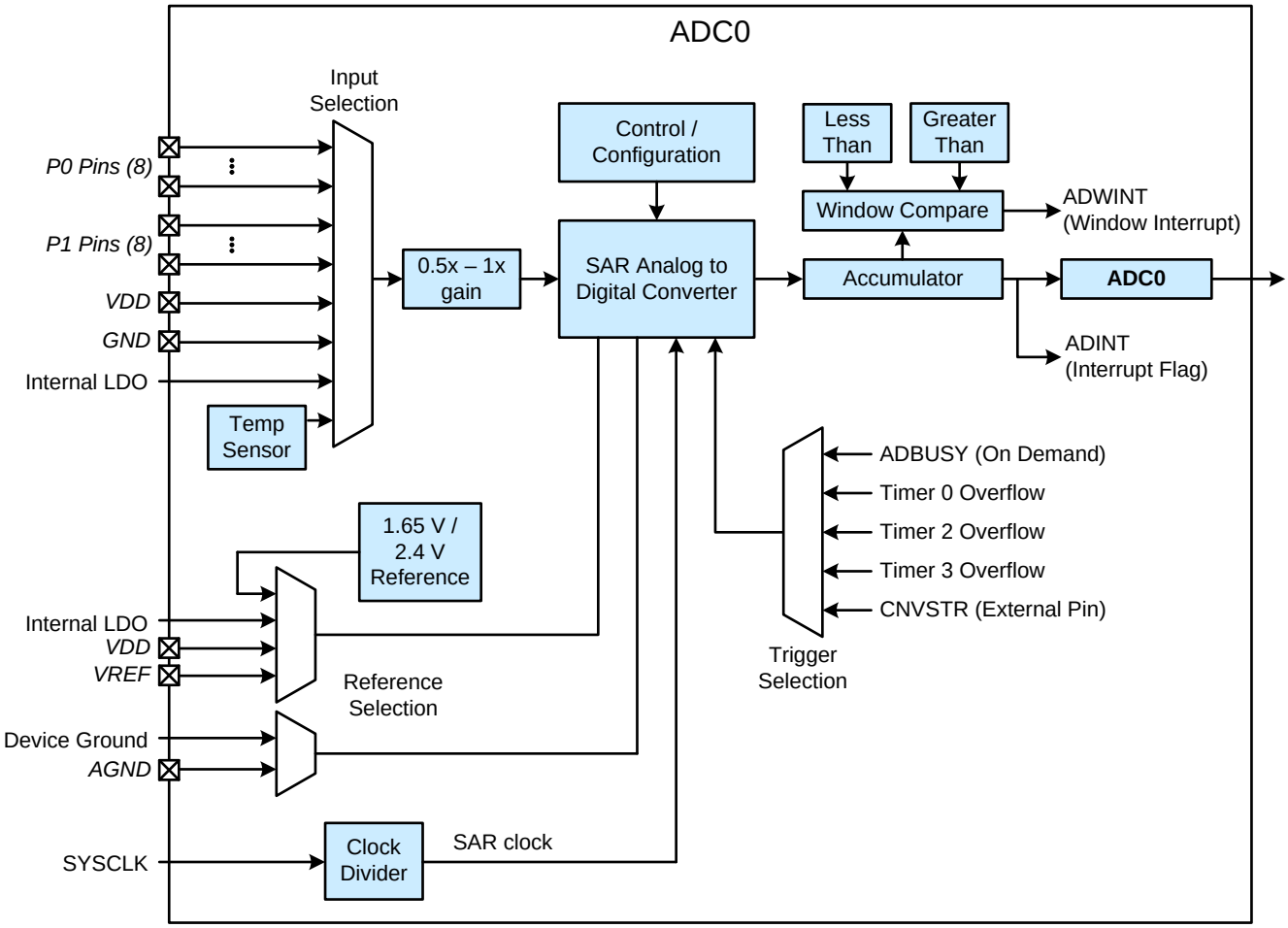



**Figure 14.1. ADC0 Functional Block Diagram**

## 14.1. ADC0 Analog Multiplexer

ADC0 on C8051F85x/86x has an analog multiplexer capable of selecting any pin on ports P0 and P1 (up to 16 total), the on-chip temperature sensor, the internal regulated supply, the VDD supply, or GND. ADC0 input channels are selected using the ADC0MX register.

**Table 14.1. ADC0 Input Multiplexer Channels**

| ADC0MX setting | Signal Name | QSOP24 Pin Name | QFN20 Pin Name | SOIC16 Pin Name |
|---|---|---|---|---|
| 00000 | ADC0.0 | P0.0 | P0.0 | P0.0 |
| 00001 | ADC0.1 | P0.1 | P0.1 | P0.1 |
| 00010 | ADC0.2 | P0.2 | P0.2 | P0.2 |
| 00011 | ADC0.3 | P0.3 | P0.3 | P0.3 |
| 00100 | ADC0.4 | P0.4 | P0.4 | P0.4 |
| 00101 | ADC0.5 | P0.5 | P0.5 | P0.5 |
| 00110 | ADC0.6 | P0.6 | P0.6 | P0.6 |
| 00111 | ADC0.7 | P0.7 | P0.7 | P0.7 |
| 01000 | ADC0.8 | P1.0 | P1.0 | P1.0 |
| 01001 | ADC0.9 | P1.1 | P1.1 | P1.1 |
| 01010 | ADC0.10 | P1.2 | P1.2 | P1.2 |
| 01011 | ADC0.11 | P1.3 | P1.3 | P1.3 |
| 01100 | ADC0.12 | P1.4 | P1.4 | Reserved |
| 01101 | ADC0.13 | P1.5 | P1.5 | Reserved |
| 01110 | ADC0.14 | P1.6 | P1.6 | Reserved |
| 01111 | ADC0.15 | P1.7 | Reserved | Reserved |
| 10000 | Temp Sensor | Internal Temperature Sensor | | |
| 10001 | LDO | Internal 1.8 V LDO Output | | |
| 10010 | VDD | VDD Supply Pin | | |
| 10011 | GND | GND Supply Pin | | |
| 10100-11111 | None | No connection | | |

Important note about ADC0 input configuration: Port pins selected as ADC0 inputs should be configured as analog inputs, and should be skipped by the 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.

## 14.2. ADC Operation

The ADC is clocked by an adjustable conversion clock (SARCLK). SARCLK is a divided version of the selected system clock when burst mode is disabled (ADBMEN = 0), or a divided version of the high-frequency oscillator when burst mode is enabled (ADBMEN = 1). The clock divide value is determined by the ADSC bits in the ADC0CF register. In most applications, SARCLK should be adjusted to operate as fast as possible, without exceeding the maximum electrical specifications. The SARCLK does not directly determine sampling times or sampling rates.

### 14.2.1. Starting a Conversion

A conversion can be initiated in many ways, depending on the programmed states of the ADC0 Start of Conversion Mode field (ADCM) in register ADC0CN0. Conversions may be initiated by one of the following:

1. Writing a 1 to the ADBUSY bit of register ADC0CN0 (software-triggered)
2. A timer overflow (see the ADC0CN0 register and the timer section for timer options)
3. A rising edge on the CNVSTR input signal (external pin-triggered)

Writing a 1 to ADBUSY provides software control of ADC0 whereby conversions are performed "on-demand". All other trigger sources occur autonomous to code execution. When the conversion is complete, the ADC posts the result to its output register and sets the ADC interrupt flag (ADINT). ADINT may be used to trigger a system interrupts, if enabled, or polled by firmware.

During conversion, the ADBUSY bit is set to logic 1 and reset to logic 0 when the conversion is complete. However, when polling for ADC conversion completions, the ADC0 interrupt flag (ADINT) should be used instead of the ADBUSY bit. Converted data is available in the ADC0 data registers, ADC0H:ADC0L, when the conversion is complete.

**Important Note About Using CNVSTR:** When the CNVSTR input is used as the ADC0 conversion source, the associated port pin should be skipped in the crossbar settings.

### 14.2.2. 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 the electrical specifications tables. The ADTM bit in register ADC0CN0 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. A conversion will begin immediately when the start-of-conversion trigger occurs.

When the ADTM bit is logic 1, each conversion is preceded by a tracking period of 4 SAR clocks (after the start-of-conversion signal) for any internal (non-CNVSTR) conversion trigger source. When the CNVSTR signal is used to initiate conversions with ADTM set to 1, ADC0 tracks only when CNVSTR is low; conversion begins on the rising edge of CNVSTR (see Figure 14.2). Setting ADTM to 1 is primarily useful when AMUX settings are frequently changed and conversions are started using the ADBUSY bit.

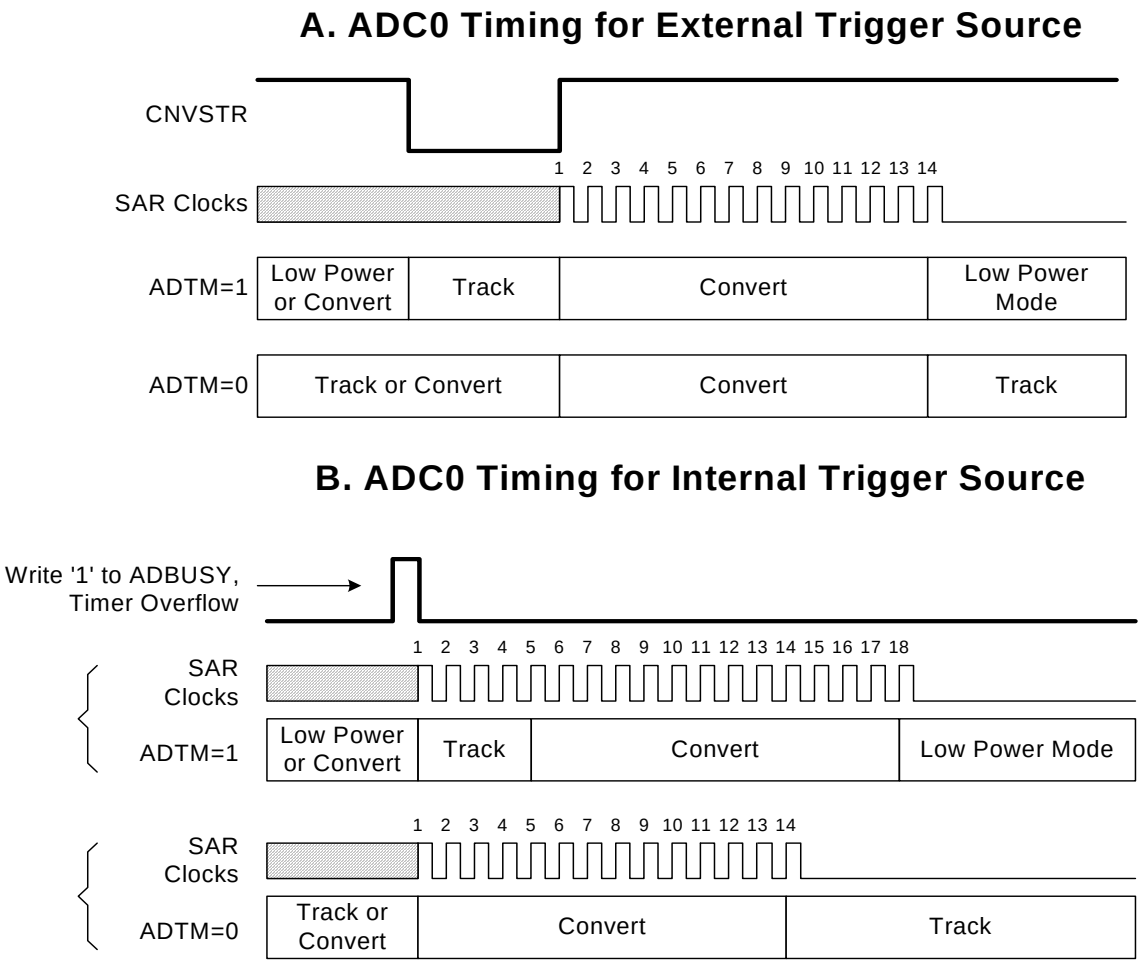



**Figure 14.2. 10-Bit ADC Track and Conversion Example Timing (ADBMEN = 0)**

### 14.2.3. 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 samples using the internal low-power high-frequency oscillator, 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. 80 kHz).

Burst Mode is enabled by setting ADBMEN to logic 1. When in Burst Mode, ADEN controls the ADC0 idle power state (i.e. the state ADC0 enters when not tracking or performing conversions). If ADEN is set to logic 0, ADC0 is powered down after each burst. If ADEN 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 ADPWR bits. Otherwise, ADC0 will start tracking and converting immediately. Figure 14.3 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 (ADINT) 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 ADPWR and ADTK. Settling time requirements may need adjustment in some applications. Refer to “14.2.4. Settling Time Requirements” on page 90 for more details.

**Notes:**

- Setting ADTM to 1 will insert an additional 4 SAR clocks of tracking before each conversion, regardless of the settings of ADPWR and ADTK.
- 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. The ADC will ignore convert start signals which arrive before a burst is finished.

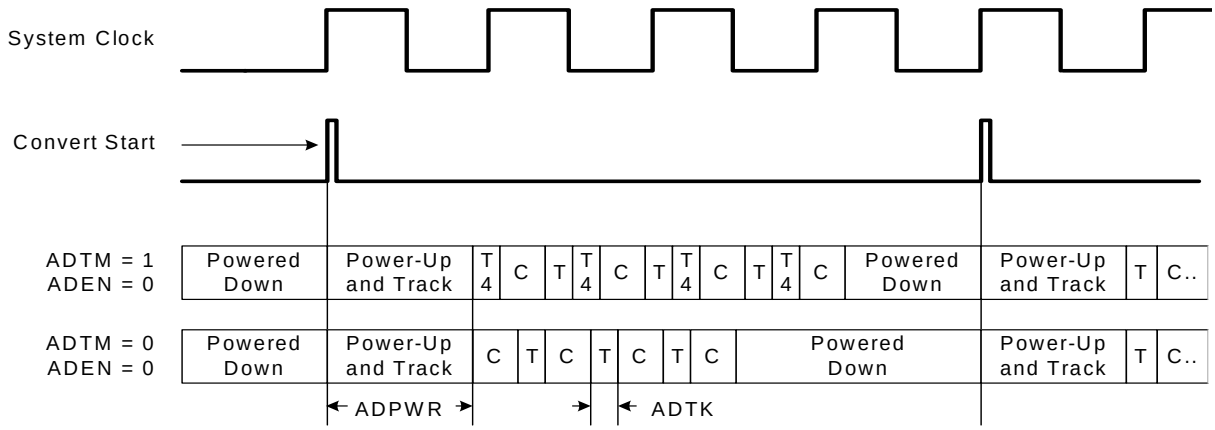



**Figure 14.3. Burst Mode Tracking Example with Repeat Count Set to 4**

### 14.2.4. 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 when ADTM is set to 1, four SAR clocks are used for tracking at the start of every conversion. Large external source impedance will increase the required tracking time.

Figure 14.4 shows the equivalent ADC0 input circuit. The required ADC0 settling time for a given settling accuracy (SA) may be approximated by Equation 14.1. When measuring any internal source, $R_{TOTAL}$ reduces to $R_{MUX}$. See the electrical specification tables 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 14.1. ADC0 Settling Time Requirements**

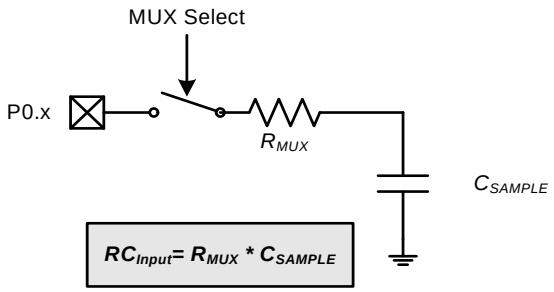
Where:

*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 (8/10/12).




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

**Figure 14.4. ADC0 Equivalent Input Circuits**

### 14.2.5. 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 VREF. In 0.5x mode, the full-scale reading of the ADC occurs when the input voltage is VREF x 2. The 0.5x gain setting can be useful to obtain a higher input voltage range when using a small VREF voltage, or to measure input voltages that are between VREF and VDD. Gain settings for the ADC are controlled by the ADGN bit in register ADC0CF. Note that even with a gain setting of 0.5, voltages above the supply rail cannot be measured directly by the ADC.

## 14.3. 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 fewer SAR clock cycles than a 10-bit conversion. The two LSBs of a conversion are always 00 in this mode, and the ADC0L register will always read back 0x00.

## 14.4. 12-Bit Mode

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. This reconfiguration cancels any matching errors and enables the converter to achieve 12-bit linearity performance to go along with its 12-bit resolution.

The 12-bit mode is enabled by setting the AD12BE bit in register ADC0AC to logic 1 and configuring the ADC in burst mode (ADBMEN = 1) for four or more conversions. The conversion can be initiated using any of the conversion start sources, 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.

The AD12SM bit in register ADC0TK controls when the ADC will track and sample the input signal. When AD12SM is set to 1, the selected input signal will be tracked before the first conversion of a set and held internally during all four conversions. When AD12SM is cleared to 0, the ADC will track and sample the selected input before each of the four conversions in a set. When maximum throughput (180-200 ksps) is

needed, it is recommended that AD12SM be set to 1 and ADTK to 0x3F, and that the ADC be placed in always-on mode (ADEN = 1). For sample rates under 180 ksps, or when accumulating multiple samples, AD12SM should normally be cleared to 0, and ADTK should be configured to provide the appropriate settling time for the subsequent conversions.

## 14.5. Power Considerations

The ADC has several power-saving features which can help the user optimize power consumption according to the needs of the application. The most efficient way to use the ADC for slower sample rates is by using burst mode. Burst mode dynamically controls power to the ADC and (if used) the internal voltage reference. By completely powering off these circuits when the ADC is not tracking or converting, the average supply current required for lower sampling rates is reduced significantly.

The ADC also provides low power options that allow reduction in operating current when operating at low SAR clock frequencies or with longer tracking times. The internal common-mode buffer can be configured for low power mode by setting the ADLPM bit in ADC0PWR to 1. Two other fields in the ADC0PWR register (ADBIAS and ADMXLP) may be used together to adjust the power consumed by the ADC and its multiplexer and reference buffers, respectively. In general, these options are used together, when operating with a SAR conversion clock frequency of 4 MHz.

**Table 14.2. ADC0 Optimal Power Configuration (8- and 10-bit Mode)**

| Required Throughput | Reference Source | Mode Configuration | SAR Clock Speed | Other Register Field Settings |
|---|---|---|---|---|
| 325-800 ksps | Any | Always-On<br>(ADEN = 1<br>ADBMEN = 0) | 12.25 MHz<br>(ADSC = 1) | ADC0PWR = 0x40<br>ADC0TK = N/A<br>ADRPT = 0 |
| 0-325 ksps | External | Burst Mode<br>(ADEN = 0<br>ADBMEN = 1) | 12.25 MHz<br>(ADSC = 1) | ADC0PWR = 0x44<br>ADC0TK = 0x3A<br>ADRPT = 0 |
| 250-325 ksps | Internal | Burst Mode<br>(ADEN = 0<br>ADBMEN = 1) | 12.25 MHz<br>(ADSC = 1) | ADC0PWR = 0x44<br>ADC0TK = 0x3A<br>ADRPT = 0 |
| 200-250 ksps | Internal | Always-On<br>(ADEN = 1<br>ADBMEN = 0) | 4.08 MHz<br>(ADSC = 5) | ADC0PWR = 0xF0<br>ADC0TK = N/A<br>ADRPT = 0 |
| 0-200 ksps | Internal | Burst Mode<br>(ADEN = 0<br>ADBMEN = 1) | 4.08 MHz<br>(ADSC = 5) | ADC0PWR = 0xF4<br>ADC0TK = 0x34<br>ADRPT = 0 |

**Notes:**
1. For always-on configuration, ADSC settings assume SYSCLK is the internal 24.5 MHz high-frequency oscillator. Adjust ADSC as needed if using a different source for SYSCLK.
2. ADRPT reflects the minimum setting for this bit field. When using the ADC in Burst Mode, up to 64 samples may be auto-accumulated per conversion start by adjusting ADRPT.

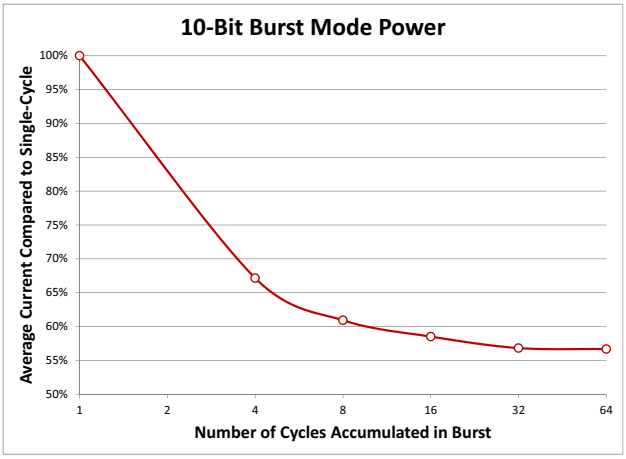
**Table 14.3. ADC0 Optimal Power Configuration (12-bit Mode)**

| Required Throughput | Reference Source | Mode Configuration | SAR Clock Speed | Other Register Field Settings |
|---|---|---|---|---|
| 180-200 ksps | Any | Always-On + Burst Mode (ADEN = 1 ADBMEN = 1) | 12.25 MHz (ADSC = 1) | ADC0PWR = 0x40 ADC0TK = 0xBF ADRPT = 1 |
| 125-180 ksps | Any | Always-On + Burst Mode (ADEN = 1 ADBMEN = 1) | 12.25 MHz (ADSC = 1) | ADC0PWR = 0x40 ADC0TK = 0x3A ADRPT = 1 |
| 0-125 ksps | External | Burst Mode (ADEN = 0 ADBMEN = 1) | 12.25 MHz (ADSC = 1) | ADC0PWR = 0x44 ADC0TK = 0x3A ADRPT = 1 |
| 50-125 ksps | Internal | Burst Mode (ADEN = 0 ADBMEN = 1) | 12.25 MHz (ADSC = 1) | ADC0PWR = 0x44 ADC0TK = 0x3A ADRPT = 1 |
| 0-50 ksps | Internal | Burst Mode (ADEN = 0 ADBMEN = 1) | 4.08 MHz (ADSC = 5) | ADC0PWR = 0xF4 ADC0TK = 0x34 ADRPT = 1 |

**Note:** ADRPT reflects the minimum setting for this bit field. When using the ADC in Burst Mode, up to 64 samples may be auto-accumulated per conversion trigger by adjusting ADRPT.

For applications where burst mode is used to automatically accumulate multiple results, additional supply current savings can be realized. The length of time the ADC is active during each burst contains power-up time at the beginning of the burst as well as the conversion time required for each conversion in the burst. The power-on time is only required at the beginning of each burst. When compared with single-sample bursts to collect the same number of conversions, multi-sample bursts will consume significantly less power. For example, performing an eight-cycle burst of 10-bt conversions consumes about 61% of the power required to perform those same eight samples in single-cycle bursts. For 12-bit conversions, an eight-cycle burst results in about 85% of the equivalent single-cycle bursts. Figure 14.5 shows this relationship for the different burst cycle lengths.

See the Electrical Characteristics chapter for details on power consumption and the maximum clock frequencies allowed in each mode.



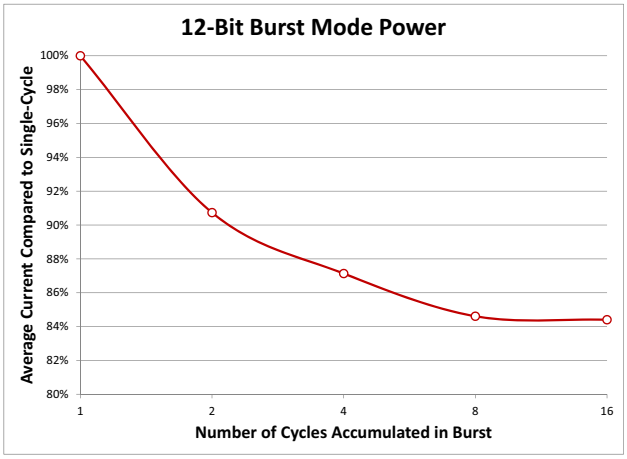




**Figure 14.5. Burst Mode Accumulation Power Savings**

## 14.6. 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 ADSJST field. When the repeat count is set to 1 in 10-bit mode, 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<br>ADC0H:ADC0L (ADSJST = 000) | Left-Justified<br>ADC0H:ADC0L (ADSJST = 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 ADRPT bits in the ADC0AC register. When a repeat count is higher than 1, the ADC output must be right-justified (ADSJST = 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 ADSJST 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 |

## 14.7. 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 (ADWINT in register ADC0CN0) 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.

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

Figure 14.6 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 ADWINT 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 ADWINT 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 14.7 shows an example using left-justified data with the same comparison values.

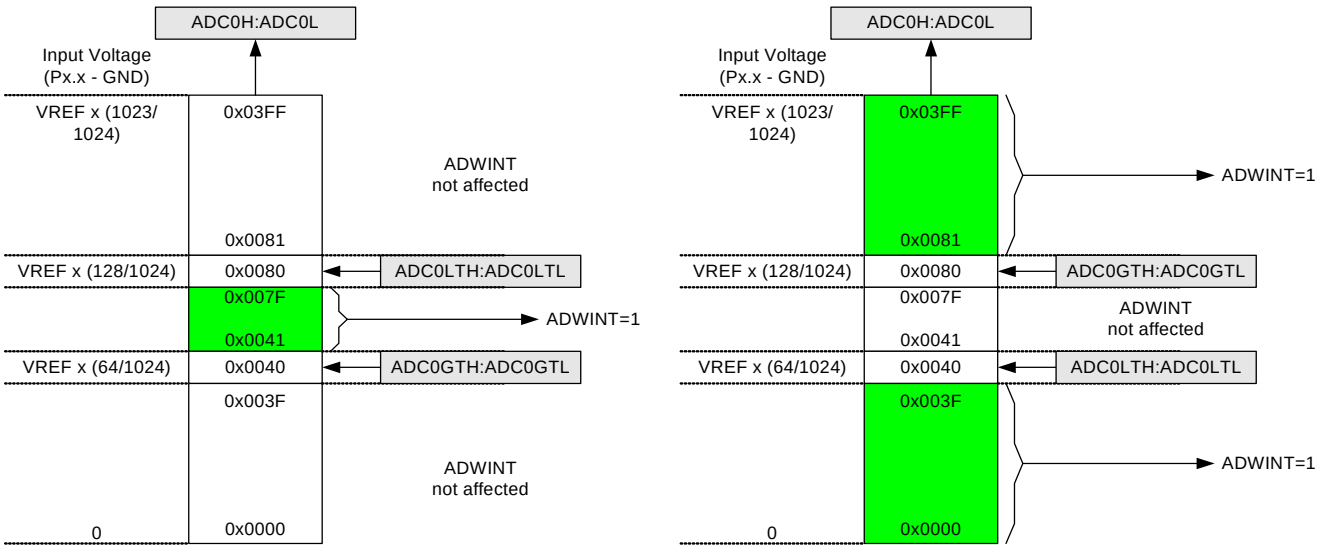



**Figure 14.6. ADC Window Compare Example: Right-Justified Single-Ended Data**

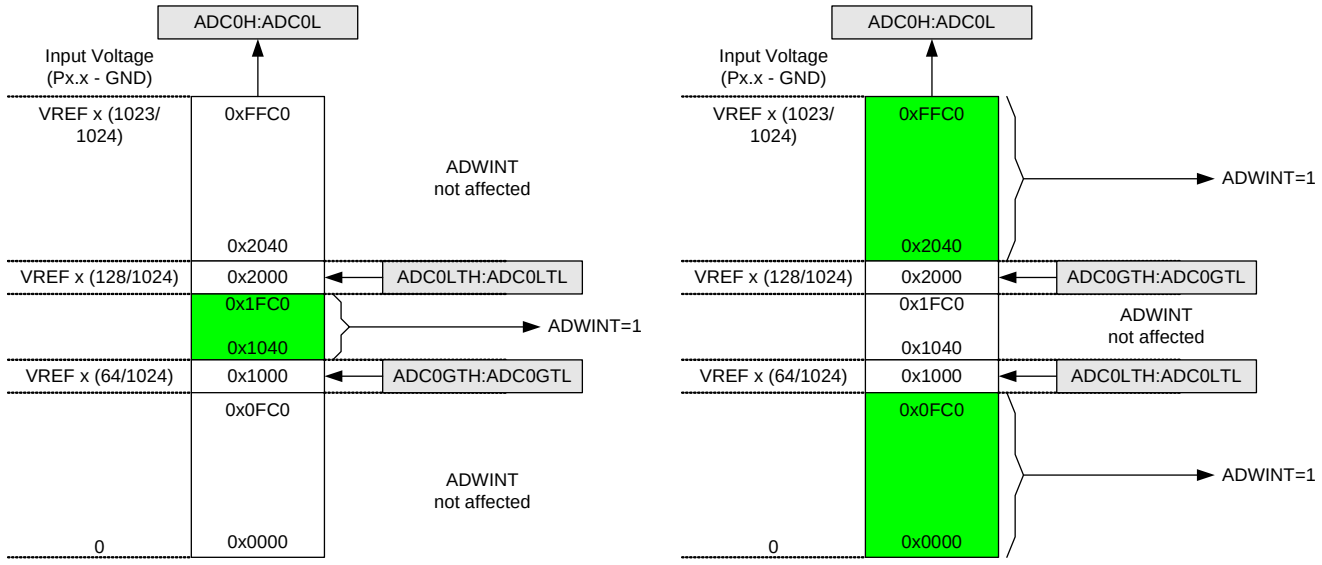



**Figure 14.7. ADC Window Compare Example: Left-Justified Single-Ended Data**

## 14.8. Voltage and Ground Reference Options

The voltage reference multiplexer is configurable to use an externally connected voltage reference, the internal voltage reference, or one of two power supply voltages. 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 (AGND).

The voltage and ground reference options are configured using the REF0CN register.

**Important Note About the VREF and AGND Inputs:** Port pins are used as the external VREF and AGND inputs. When using an external voltage reference, VREF should be configured as an analog input and skipped by the digital crossbar. When using AGND as the ground reference to ADC0, AGND should be configured as an analog input and skipped by the Digital Crossbar.

### 14.8.1. External Voltage Reference

To use an external voltage reference, REFSL 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 4.7uF in parallel with a 0.1uF capacitor is recommended.

### 14.8.2. Internal Voltage Reference

For applications requiring the maximum number of port I/O pins, or very short VREF turn-on time, the high-speed reference will be the best internal reference option to choose. The internal reference is selected by setting REFSL to 11. When selected, the internal reference will be automatically enabled/disabled on an as-needed basis by the ADC. The reference can be set to one of two voltage values: 1.65 V or 2.4 V, depending on the value of the IREFLVL bit.

For applications with a non-varying power supply voltage, using the power supply as the voltage reference can provide the ADC with added dynamic range at the cost of reduced power supply noise rejection. To use the external supply pin (VDD) or the 1.8 V regulated digital supply voltage as the reference source, REFSL should be set to 01 or 10, respectively.

Internal reference sources are not routed to the VREF pin, and do not require external capacitors. The electrical specifications tables detail SAR clock and throughput limitations for each reference source.

### 14.8.3. 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 the ADC during both the tracking/sampling and the conversion periods is taken from the AGND pin. Any external sensors sampled by the ADC should be referenced to the AGND pin. If an external voltage reference is used, the 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 GNDSL 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 GNDSL 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 GNDSL bit.

## 14.9. Temperature Sensor

An on-chip temperature sensor is included, 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 14.8. 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. 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 the electrical specification tables for the slope and offset parameters of the temperature sensor.

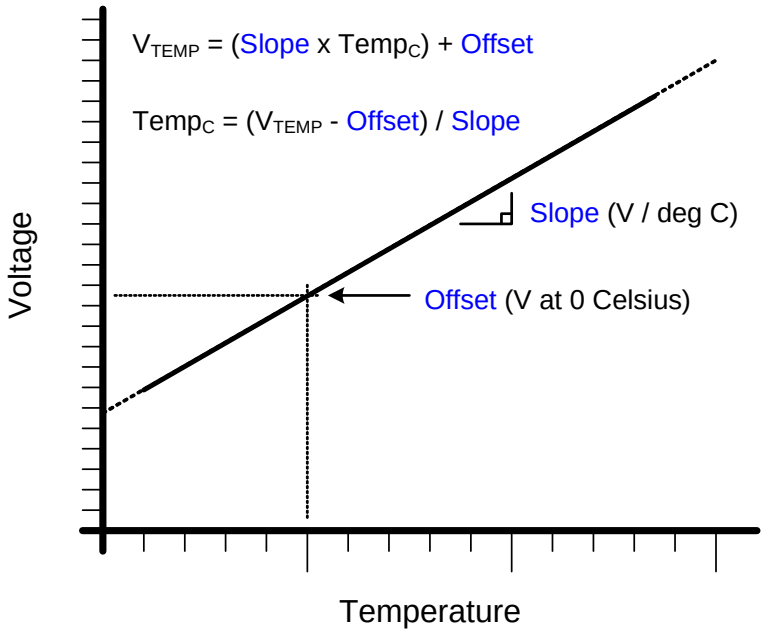



**Figure 14.8. Temperature Sensor Transfer Function**

### 14.9.1. Calibration

The uncalibrated temperature sensor output is extremely linear and suitable for relative temperature measurements. 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 ADC input.
4. Calculate the offset characteristics, and store this value in non-volatile memory for use with subsequent temperature sensor measurements.

## 14.10. ADC Control Registers

### Register 14.1. ADC0CN0: ADC0 Control 0

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ADEN | ADBMEN | ADINT | ADBUSY | ADWINT | ADCM | | |
| Type | RW | RW | RW | RW | RW | RW | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xE8 (bit-addressable)** | | | | | | | | |

**Table 14.4. ADC0CN0 Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7 | ADEN | **Enable.**<br>0: ADC0 Disabled (low-power shutdown).<br>1: ADC0 Enabled (active and ready for data conversions). |
| 6 | ADBMEN | **Burst Mode Enable.**<br>0: ADC0 Burst Mode Disabled.<br>1: ADC0 Burst Mode Enabled. |
| 5 | ADINT | **Conversion Complete Interrupt Flag.**<br>Set by hardware upon completion of a data conversion (ADBMEN=0), or a burst of conversions (ADBMEN=1). Can trigger an interrupt. Must be cleared by software. |
| 4 | ADBUSY | **ADC Busy.**<br>Writing 1 to this bit initiates an ADC conversion when ADC0CM = 000. This bit should not be polled to indicate when a conversion is complete. Instead, the ADINT bit should be used when polling for conversion completion. |
| 3 | ADWINT | **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 | ADCM | **Start of Conversion Mode Select.**<br>Specifies the ADC0 start of conversion source. All remaining bit combinations are reserved.<br>000: ADC0 conversion initiated on write of 1 to ADBUSY.<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>100: ADC0 conversion initiated on rising edge of CNVSTR.<br>101-111: Reserved. |

## Register 14.2. ADC0CN1: ADC0 Control 1

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Reserved | | | | | | | ADCMBE |
| Type | R | | | | | | | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xB2** | | | | | | | | |

**Table 14.5. ADC0CN1 Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:1 | Reserved | Must write reset value. |
| 0 | ADCMBE | **Common Mode Buffer Enable.**<br>0: Disable the common mode buffer. This setting should be used only if the tracking time of the signal is greater than 1.5 us.<br>1: Enable the common mode buffer. This setting should be used in most cases, and will give the best dynamic ADC performance. The common mode buffer must be enabled if signal tracking time is less than or equal to 1.5 us. |

## Register 14.3. ADC0CF: ADC0 Configuration

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ADSC | | | | | AD8BE | ADTM | ADGN |
| Type | RW | | | | | RW | RW | RW |
| Reset | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| **SFR Address: 0xBC** | | | | | | | | |

**Table 14.6. ADC0CF Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:3 | ADSC | **SAR Clock Divider.**<br>This field sets the ADC clock divider value. It should be configured to be as close to the maximum SAR clock speed as the datasheet will allow. The SAR clock frequency is given by the following equation:<br>$F_{CLKSAR} = \frac{F_{ADCCLK}}{ADSC + 1}$<br>$F_{ADCCLK}$ is equal to the selected SYSCLK when ADBMEN is 0 and the high-frequency oscillator when ADBMEN is 1. |
| 2 | AD8BE | **8-Bit Mode Enable.**<br>0: ADC0 operates in 10-bit or 12-bit mode (normal operation).<br>1: ADC0 operates in 8-bit mode. |
| 1 | ADTM | **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 4 SAR clock cycles following the start-of-conversion signal. The ADC is allowed to track during this time. |
| 0 | ADGN | **Gain Control.**<br>0: The on-chip PGA gain is 0.5.<br>1: The on-chip PGA gain is 1. |

## Register 14.4. ADC0AC: ADC0 Accumulator Configuration

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | AD12BE | ADAE | ADSJST | | | ADRPT | | |
| Type | RW | RW | RW | | | RW | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xB3** | | | | | | | | |

**Table 14.7. ADC0AC Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7 | AD12BE | **12-Bit Mode Enable.**<br>Enables 12-bit Mode. In 12-bit mode, the ADC throughput is reduced by a factor of 4.<br>0: 12-bit Mode Disabled.<br>1: 12-bit Mode Enabled. |
| 6 | ADAE | **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. |
| 5:3 | ADSJST | **Accumulator Shift and Justify.**<br>Specifies the format of data read from ADC0H:ADC0L. All remaining bit combinations are reserved.<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>101-111: Reserved. |
| 2:0 | ADRPT | **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 (not used in 12-bit mode).<br>001: Perform and Accumulate 4 conversions (1 conversion in 12-bit mode).<br>010: Perform and Accumulate 8 conversions (2 conversions in 12-bit mode).<br>011: Perform and Accumulate 16 conversions (4 conversions in 12-bit mode).<br>100: Perform and Accumulate 32 conversions (8 conversions in 12-bit mode).<br>101: Perform and Accumulate 64 conversions (16 conversions in 12-bit mode).<br>110-111: Reserved. |

## Register 14.5. ADC0PWR: ADC0 Power Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ADBIAS | | ADMXLP | ADLPM | ADPWR | | | |
| Type | RW | | RW | RW | RW | | | |
| Reset | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| **SFR Address: 0xDF** | | | | | | | | |

**Table 14.8. ADC0PWR Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:6 | ADBIAS | **Bias Power Select.**<br>This field can be used to adjust the ADC's power consumption based on the conversion speed. Higher bias currents allow for faster conversion times.<br>00: Select bias current mode 0. Recommended to use modes 1, 2, or 3.<br>01: Select bias current mode 1 (SARCLK <= 16 MHz).<br>10: Select bias current mode 2.<br>11: Select bias current mode 3 (SARCLK <= 4 MHz). |
| 5 | ADMXLP | **Mux and Reference Low Power Mode Enable.**<br>Enables low power mode operation for the multiplexer and voltage reference buffers.<br>0: Low power mode disabled.<br>1: Low power mode enabled (SAR clock < 4 MHz). |
| 4 | ADLPM | **Low Power Mode Enable.**<br>This bit can be used to reduce power to the ADC's internal common mode buffer. It can be set to 1 to reduce power when tracking times in the application are longer (slower sample rates).<br>0: Disable low power mode.<br>1: Enable low power mode (requires extended tracking time). |
| 3:0 | ADPWR | **Burst Mode Power Up Time.**<br>This field sets the time delay allowed for the ADC to power up from a low power state. When ADTM is set, an additional 4 SARCLKs are added to this time.<br>$T_{PWRTIME} = \frac{8 \times ADPWR}{F_{HFOSC}}$ |

## Register 14.6. ADC0TK: ADC0 Burst Mode Track Time

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | AD12SM | Reserved | ADTK | | | | | |
| Type | RW | RW | RW | | | | | |
| Reset | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 |
| **SFR Address: 0xB9** | | | | | | | | |

**Table 14.9. ADC0TK Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7 | AD12SM | **12-Bit Sampling Mode.**<br>This bit controls the way that the ADC samples the input when in 12-bit mode. When the ADC is configured for multiple 12-bit conversions in burst mode, the AD12SM bit should be cleared to 0.<br>0: The ADC will re-track and sample the input four times during a 12-bit conversion.<br>1: The ADC will sample the input once at the beginning of each 12-bit conversion. The ADTK field can be set to 63 to maximize throughput. |
| 6 | Reserved | Must write reset value. |
| 5:0 | ADTK | **Burst Mode Tracking Time.**<br>This field sets the time delay between consecutive conversions performed in Burst Mode. When ADTM is set, an additional 4 SARCLKs are added to this time.<br>$T_{BMTK} = \frac{64 - ADTK}{F_{HFOSC}}$<br>The Burst Mode track delay is not inserted prior to the first conversion. The required tracking time for the first conversion should be defined with the ADPWR field. |

## Register 14.7. ADC0H: ADC0 Data Word High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ADC0H | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xBE** | | | | | | | | |

**Table 14.10. ADC0H Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | ADC0H | **Data Word High Byte.**<br>When read, this register returns the most significant byte of the 16-bit ADC0 accumulator formatted according to the settings in ADSJST. The register may also be written to set the upper byte of the 16-bit ADC0 accumulator. |
| **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. | | |

## Register 14.8. ADC0L: ADC0 Data Word Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ADC0L | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xBD** | | | | | | | | |

**Table 14.11. ADC0L Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | ADC0L | **Data Word Low Byte.**<br>When read, this register returns the least significant byte of the 16-bit ADC0 accumulator, formatted according to the settings in ADSJST. The register may also be written, to set the lower byte of the 16-bit ADC0 accumulator. |
| **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. | | |

## Register 14.9. ADC0GTH: ADC0 Greater-Than High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ADC0GTH | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **SFR Address: 0xC4** | | | | | | | | |

**Table 14.12. ADC0GTH Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | ADC0GTH | **Greater-Than High Byte.**<br>Most Significant Byte of the 16-bit Greater-Than window compare register. |

## Register 14.10. ADC0GTL: ADC0 Greater-Than Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ADC0GTL | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **SFR Address: 0xC3** | | | | | | | | |

**Table 14.13. ADC0GTL Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | ADC0GTL | **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. | | |

## Register 14.11. ADC0LTH: ADC0 Less-Than High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ADC0LTH | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xC6** | | | | | | | | |

**Table 14.14. ADC0LTH Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | ADC0LTH | **Less-Than High Byte.**<br>Most Significant Byte of the 16-bit Less-Than window compare register. |

## Register 14.12. ADC0LTL: ADC0 Less-Than Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ADC0LTL | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xC5** | | | | | | | | |

**Table 14.15. ADC0LTL Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | ADC0LTL | **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. | | |

## Register 14.13. ADC0MX: ADC0 Multiplexer Selection

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Reserved | | | ADC0MX | | | | |
| Type | R | | | RW | | | | |
| Reset | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| **SFR Address: 0xBB** | | | | | | | | |

**Table 14.16. ADC0MX Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:5 | Reserved | Must write reset value. |
| 4:0 | ADC0MX | **AMUX0 Positive Input Selection.**<br>Selects the positive input channel for ADC0. For reserved bit combinations, no input is selected.<br>00000: ADC0.0<br>00001: ADC0.1<br>00010: ADC0.2<br>00011: ADC0.3<br>00100: ADC0.4<br>00101: ADC0.5<br>00110: ADC0.6<br>00111: ADC0.7<br>01000: ADC0.8<br>01001: ADC0.9<br>01010: ADC0.10<br>01011: ADC0.11<br>01100: ADC0.12<br>01101: ADC0.13<br>01110: ADC0.14<br>01111: ADC0.15<br>10000: Temperature sensor.<br>10001: Internal LDO regulator output.<br>10010: VDD<br>10011: GND<br>10100-11111: Reserved. |

## Register 14.14. REF0CN: Voltage Reference Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | IREFLVL | Reserved | GNDSL | REFSL | | TEMPE | Reserved | |
| Type | RW | R | RW | RW | | RW | R | |
| Reset | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| **SFR Address: 0xD1** | | | | | | | | |

**Table 14.17. REF0CN Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7 | IREFLVL | **Internal Voltage Reference Level.**<br>Sets the voltage level for the internal reference source.<br>0: The internal reference operates at 1.65 V nominal.<br>1: The internal reference operates at 2.4 V nominal. |
| 6 | Reserved | Must write reset value. |
| 5 | GNDSL | **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 AGND pin. |
| 4:3 | REFSL | **Voltage Reference Select.**<br>Selects the ADC0 voltage reference.<br>00: The ADC0 voltage reference is the 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 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 | Reserved | Must write reset value. |

# 15. CIP-51 Microcontroller Core

The C8051F85x/86x uses 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 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 15.1 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

## 15.1. Performance

The CIP-51 employs a pipelined architecture that greatly increases its instruction throughput over the standard 8051 architecture. 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.

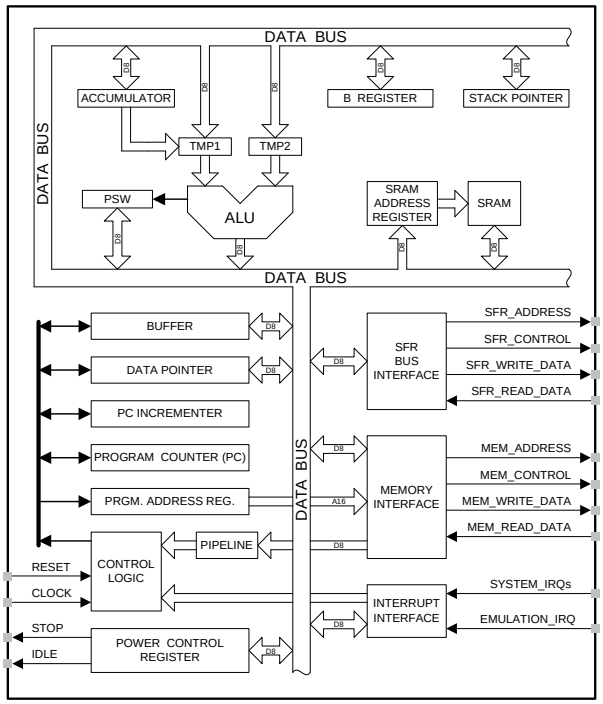



**Figure 15.1. CIP-51 Block Diagram**

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 |

## 15.2. 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.

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.

## 15.3. 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.

### 15.3.1. 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 15.1 is the CIP-51 Instruction Set Summary, which includes the mnemonic, number of bytes, and number of clock cycles for each instruction.

**Table 15.1. 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 |

**Table 15.1. CIP-51 Instruction Set Summary (Continued)**

| Mnemonic | Description | Bytes | Clock Cycles |
|---|---|---|---|
| XRL A, Rn | Exclusive-OR Register to A | 1 | 1 |
| XRL A, direct | Exclusive-OR direct byte to A | 2 | 2 |
| XRL A, @Ri | Exclusive-OR indirect RAM to A | 1 | 2 |
| XRL A, #data | Exclusive-OR immediate to A | 2 | 2 |
| XRL direct, A | Exclusive-OR A to direct byte | 2 | 2 |
| XRL direct, #data | Exclusive-OR immediate to direct byte | 3 | 3 |
| CLR A | Clear A | 1 | 1 |
| CPL A | Complement A | 1 | 1 |
| RL A | Rotate A left | 1 | 1 |
| RLC A | Rotate A left through Carry | 1 | 1 |
| RR A | Rotate A right | 1 | 1 |
| RRC A | Rotate A right through Carry | 1 | 1 |
| SWAP A | Swap nibbles of A | 1 | 1 |
| **Data Transfer** | | | |
| MOV A, Rn | Move Register to A | 1 | 1 |
| MOV A, direct | Move direct byte to A | 2 | 2 |
| MOV A, @Ri | Move indirect RAM to A | 1 | 2 |
| MOV A, #data | Move immediate to A | 2 | 2 |
| MOV Rn, A | Move A to Register | 1 | 1 |
| MOV Rn, direct | Move direct byte to Register | 2 | 2 |
| MOV Rn, #data | Move immediate to Register | 2 | 2 |
| MOV direct, A | Move A to direct byte | 2 | 2 |
| MOV direct, Rn | Move Register to direct byte | 2 | 2 |
| MOV direct, direct | Move direct byte to direct byte | 3 | 3 |
| MOV direct, @Ri | Move indirect RAM to direct byte | 2 | 2 |
| MOV direct, #data | Move immediate to direct byte | 3 | 3 |
| MOV @Ri, A | Move A to indirect RAM | 1 | 2 |
| MOV @Ri, direct | Move direct byte to indirect RAM | 2 | 2 |
| MOV @Ri, #data | Move immediate to indirect RAM | 2 | 2 |
| MOV DPTR, #data16 | Load DPTR with 16-bit constant | 3 | 3 |
| MOVC A, @A+DPTR | Move code byte relative DPTR to A | 1 | 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 |

**Table 15.1. CIP-51 Instruction Set Summary (Continued)**

| Mnemonic | Description | Bytes | Clock Cycles |
|---|---|---|---|
| XCH A, Rn | Exchange Register with A | 1 | 1 |
| XCH A, direct | Exchange direct byte with A | 2 | 2 |
| XCH A, @Ri | Exchange indirect RAM with A | 1 | 2 |
| XCHD A, @Ri | Exchange low nibble of indirect RAM with A | 1 | 2 |
| **Boolean Manipulation** | | | |
| CLR C | Clear Carry | 1 | 1 |
| CLR bit | Clear direct bit | 2 | 2 |
| SETB C | Set Carry | 1 | 1 |
| SETB bit | Set direct bit | 2 | 2 |
| CPL C | Complement Carry | 1 | 1 |
| CPL bit | Complement direct bit | 2 | 2 |
| ANL C, bit | AND direct bit to Carry | 2 | 2 |
| ANL C, /bit | AND complement of direct bit to Carry | 2 | 2 |
| ORL C, bit | OR direct bit to carry | 2 | 2 |
| ORL C, /bit | OR complement of direct bit to Carry | 2 | 2 |
| MOV C, bit | Move direct bit to Carry | 2 | 2 |
| MOV bit, C | Move Carry to direct bit | 2 | 2 |
| JC rel | Jump if Carry is set | 2 | 2/3 |
| 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 |

**Table 15.1. CIP-51 Instruction Set Summary (Continued)**

| Mnemonic | Description | Bytes | Clock Cycles |
|---|---|---|---|
| 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.

## 15.4. CPU Core Registers

### Register 15.1. DPL: Data Pointer Low

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | DPL | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0x82** | | | | | | | | |

**Table 15.2. DPL Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | DPL | **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. |

## Register 15.2. DPH: Data Pointer High

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | DPH | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0x83** | | | | | | | | |

**Table 15.3. DPH Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | DPH | **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. |

## Register 15.3. SP: Stack Pointer

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | SP | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| **SFR Address: 0x81** | | | | | | | | |

**Table 15.4. SP Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | SP | **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. |

## Register 15.4. ACC: Accumulator

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ACC | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xE0 (bit-addressable)** | | | | | | | | |

**Table 15.5. ACC Register Bit Descriptions**

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

## Register 15.5. B: B Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | B | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xF0 (bit-addressable)** | | | | | | | | |

**Table 15.6. B Register Bit Descriptions**

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

## Register 15.6. PSW: Program Status Word

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CY | AC | F0 | RS | | OV | F1 | PARITY |
| Type | RW | RW | RW | RW | | RW | RW | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| SFR Address: 0xD0 (bit-addressable) | | | | | | | | |

**Table 15.7. PSW Register Bit Descriptions**

| 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 | **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>1. An ADD, ADDC, or SUBB instruction causes a sign-change overflow.<br>2. A MUL instruction results in an overflow (result is greater than 255).<br>3. 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. |

# 16. Clock Sources and Selection (HFOSC0, LFOSC0, and EXTCLK)

The C8051F85x/86x devices can be clocked from the internal low power 24.5 MHz oscillator, the internal low-frequency 80 kHz oscillator, or an external CMOS clock signal at the EXTCLK pin. An adjustable clock divider allows the selected clock source to be post-scaled by powers of 2, up to a factor of 128. By default, the system clock comes up as the 24.5 MHz oscillator divided by 8.

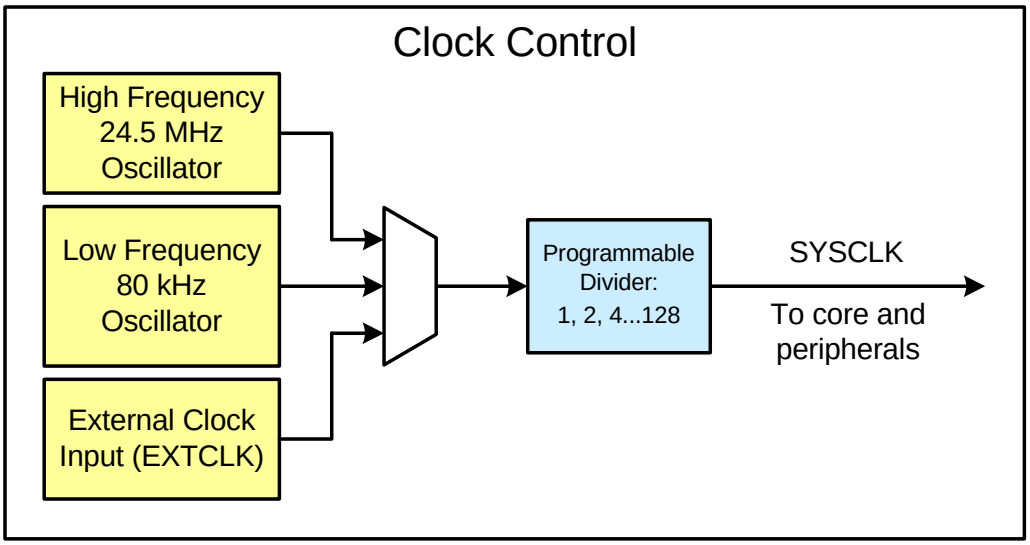



**Figure 16.1. Clocking Options**

## 16.1. Programmable High-Frequency Oscillator

All C8051F85x/86x devices include a programmable internal high-frequency oscillator that defaults as the system clock after a system reset. The oscillator is automatically enabled when it is requested. The internal oscillator period can be adjusted via the OSCICL register. On C8051F85x/86x devices, OSCICL is factory calibrated to obtain a 24.5 MHz base frequency.

## 16.2. Programmable Low-Frequency Oscillator

A programmable low-frequency internal oscillator is also included. The low-frequency oscillator is calibrated to a nominal frequency of 80 kHz. A divider at the oscillator output is capable of dividing the output clock of the module by 1, 2, 4, or 8, using the OSCLD bits in the OSCLCN register. Additionally, the OSCLF bits can be used to coarsely adjust the oscillator's output frequency.

### 16.2.1. Calibrating the Internal L-F Oscillator

Timer 3 includes a capture function that can be used to capture the oscillator frequency, when running from a known time base. When Timer 3 is configured for L-F Oscillator Capture Mode, a rising edge of the low-frequency oscillator's output will cause a capture event on the corresponding timer. As a capture event occurs, the current timer value (TMR3H:TMR3L) is copied into the timer reload registers (TMR3RLH:TMR3RLL). By recording the difference between two successive timer capture values, the low-frequency oscillator's period can be calculated. The OSCLF bits can then be adjusted to produce the desired oscillator frequency.

## 16.3. External Clock

An external CMOS clock source is also supported by the C8051F85x/86x family. The EXTCLK pin on the device serves as the external clock input when running in this mode. The EXTCLK input may also be used to clock some of the digital peripherals (e.g., Timers, PCA, etc.) while SYSCLK runs from one of the internal oscillator sources. When not selected as the SYSCLK source, the EXTCLK input is always re-synchronized to SYSCLK.

## 16.4. Clock Selection

The CLKSEL register is used to select the clock source for the system. The CLKSL field selects which oscillator source is used as the system clock, while CLKDIV controls the programmable divider. CLKSL must be set to 01b for the system clock to run from the external oscillator; however the external oscillator may still clock certain peripherals (timers, PCA) when the internal oscillator is selected as the system clock. In these cases, the external oscillator source is synchronized to the SYSCLK source. The system clock may be switched on-the-fly between any of the oscillator sources so long as the selected clock source is enabled and has settled, and CLKDIV may be changed at any time.

The internal high-frequency and low-frequency oscillators require little start-up time and may be selected as the system clock immediately following the register write which enables the oscillator. When selecting the EXTCLK pin as a clock input source, the pin should be skipped in the crossbar and configured as a digital input. Firmware should ensure that the external clock source is present or enable the missing clock detector before switching the CLKSL field.

## 16.5. High Frequency Oscillator Control Registers

### Register 16.1. OSCICL: High Frequency Oscillator Calibration

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | OSCICL | | | | | | | |
| Type | RW | | | | | | | |
| Reset | X | X | X | X | X | X | X | X |
| **SFR Address: 0xC7** | | | | | | | | |

**Table 16.1. OSCICL Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | OSCICL | **Oscillator Calibration Bits.**<br>These bits determine the internal oscillator period. When set to 00000000b, the oscillator operates at its fastest setting. When set to 11111111b, the oscillator operates at its slowest setting. The reset value is factory calibrated to generate an internal oscillator frequency of 24.5 MHz. |

## 16.6. Low Frequency Oscillator Control Registers

### Register 16.2. OSCLCN: Low Frequency Oscillator Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | OSCLEN | OSCLRDY | OSCLF | | | | OSCLD | |
| Type | RW | R | RW | | | | RW | |
| Reset | 0 | 0 | X | X | X | X | 0 | 0 |
| **SFR Address: 0xB1** | | | | | | | | |

**Table 16.2. OSCLCN Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7 | OSCLEN | **Internal L-F Oscillator Enable.**<br>This bit enables the internal low-frequency oscillator. Note that the low-frequency oscillator is automatically enabled when the watchdog timer is active.<br>0: Internal L-F Oscillator Disabled.<br>1: Internal L-F Oscillator Enabled. |
| 6 | OSCLRDY | **Internal L-F Oscillator Ready.**<br>0: Internal L-F Oscillator frequency not stabilized.<br>1: Internal L-F Oscillator frequency stabilized. |
| 5:2 | OSCLF | **Internal L-F Oscillator Frequency Control Bits.**<br>Fine-tune control bits for the Internal L-F oscillator frequency. When set to 0000b, the L-F oscillator operates at its fastest setting. When set to 1111b, the L-F oscillator operates at its slowest setting. The OSCLF bits should only be changed by firmware when the L-F oscillator is disabled (OSCLEN = 0). |
| 1:0 | OSCLD | **Internal L-F Oscillator Divider Select.**<br>00: Divide by 8 selected.<br>01: Divide by 4 selected.<br>10: Divide by 2 selected.<br>11: Divide by 1 selected. |
| **Note:** OSCLRDY is only set back to 0 in the event of a device reset or a change to the OSCLD bits. | | |

## 16.7. Clock Selection Control Registers

### Register 16.3. CLKSEL: Clock Select

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Reserved | CLKDIV | | | Reserved | | CLKSL | |
| Type | R | RW | | | R | | RW | |
| Reset | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xA9** | | | | | | | | |

**Table 16.3. CLKSEL Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7 | Reserved | Must write reset value. |
| 6:4 | CLKDIV | **Clock Source Divider.**<br>This field controls the divider applied to the clock source selected by CLKSL. The output of this divider is the system clock (SYSCLK).<br>000: SYSCLK is equal to selected clock source divided by 1.<br>001: SYSCLK is equal to selected clock source divided by 2.<br>010: SYSCLK is equal to selected clock source divided by 4.<br>011: SYSCLK is equal to selected clock source divided by 8.<br>100: SYSCLK is equal to selected clock source divided by 16.<br>101: SYSCLK is equal to selected clock source divided by 32.<br>110: SYSCLK is equal to selected clock source divided by 64.<br>111: SYSCLK is equal to selected clock source divided by 128. |
| 3:2 | Reserved | Must write reset value. |
| 1:0 | CLKSL | **Clock Source Select.**<br>Selects the system clock source.<br>00: Clock derived from the Internal High-Frequency Oscillator.<br>01: Clock derived from the External Oscillator circuit.<br>10: Clock derived from the Internal Low-Frequency Oscillator.<br>11: Reserved. |

# 17. Comparators (CMP0 and CMP1)

C8051F85x/86x devices include two on-chip programmable voltage comparators, CMP0 and CMP1. The two comparators are functionally identical, but have different connectivity within the device. A functional block diagram is shown in Figure 17.1.

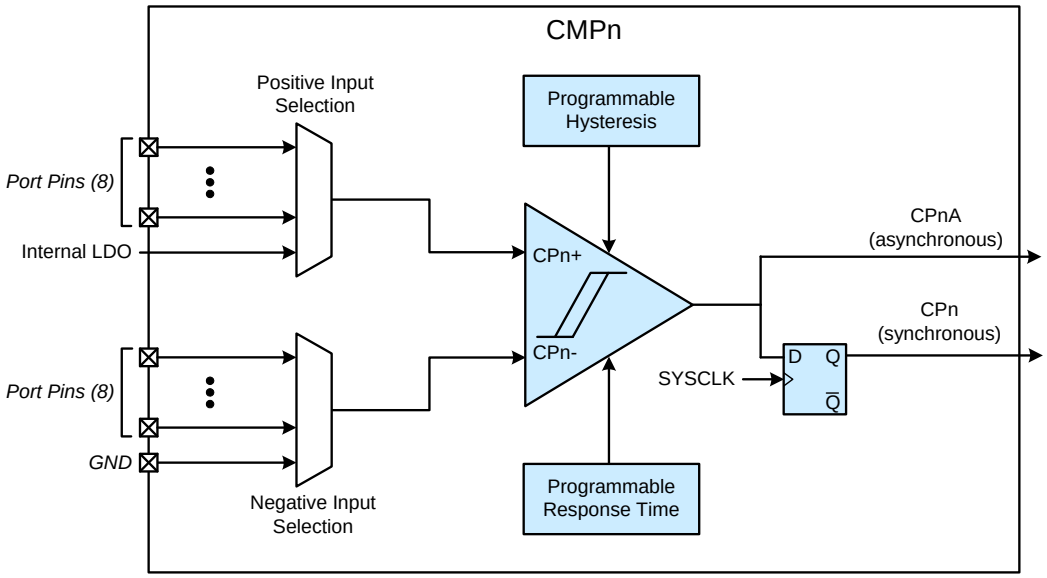



**Figure 17.1. Comparator Functional Block Diagram**

## 17.1. System Connectivity

Comparator inputs are routed to port I/O pins or internal signals using the comparator mux registers. The comparator's synchronous and asynchronous outputs can optionally be routed to port I/O pins through the port I/O crossbar. The output of either comparator may also be configured to generate a system interrupt. CMP0 may also be used as a reset source, or as a trigger to kill a PCA output channel.

The CMP0 inputs are selected in the CPT0MX register, while CPT1MX selects the CMP1 inputs. The CMXP field selects the comparator's positive input (CPnP.x); the CMXN field selects the comparator's negative input (CPnN.x). Table 17.1 through Table 17.4 detail the comparator input multiplexer options on the C8051F85x/86x family. See the port I/O crossbar sections for details on configuring comparator outputs via the digital crossbar. Comparator inputs can be externally driven from –0.25 V to ($V_{DD}$) + 0.25 V without damage or upset.

**Important Note About Comparator Inputs:** The port pins selected as comparator inputs should be configured as analog inputs in their associated port configuration register, and configured to be skipped by the crossbar.

**Table 17.1. CMP0 Positive Input Multiplexer Channels**

| CMXP Setting in Register CPT0MX | Signal Name | QSOP24 Pin Name | QFN20 Pin Name | SOIC16 Pin Name |
|---|---|---|---|---|
| 0000 | CP0P.0 | P0.0 | P0.0 | P0.0 |
| 0001 | CP0P.1 | P0.1 | P0.1 | P0.1 |
| 0010 | CP0P.2 | P0.2 | P0.2 | P0.2 |
| 0011 | CP0P.3 | P0.3 | P0.3 | P0.3 |
| 0100 | CP0P.4 | P0.4 | P0.4 | P0.4 |
| 0101 | CP0P.5 | P0.5 | P0.5 | P0.5 |
| 0110 | CP0P.6 | P0.6 | P0.6 | Reserved |
| 0111 | CP0P.7 | P0.7 | P0.7 | Reserved |
| 1000 | LDO | Internal 1.8 V LDO Output | | |
| 1001-1111 | None | No connection | | |

**Table 17.2. CMP0 Negative Input Multiplexer Channels**

| CMXN Setting in Register CPT0MX | Signal Name | QSOP24 Pin Name | QFN20 Pin Name | SOIC16 Pin Name |
|---|---|---|---|---|
| 0000 | CP0N.0 | P0.0 | P0.0 | P0.0 |
| 0001 | CP0N.1 | P0.1 | P0.1 | P0.1 |
| 0010 | CP0N.2 | P0.2 | P0.2 | P0.2 |
| 0011 | CP0N.3 | P0.3 | P0.3 | P0.3 |
| 0100 | CP0N.4 | P0.4 | P0.4 | P0.4 |
| 0101 | CP0N.5 | P0.5 | P0.5 | P0.5 |
| 0110 | CP0N.6 | P0.6 | P0.6 | Reserved |
| 0111 | CP0N.7 | P0.7 | P0.7 | Reserved |
| 1000 | GND | GND | | |
| 1001-1111 | None | No connection | | |

**Table 17.3. CMP1 Positive Input Multiplexer Channels**

| CMXP Setting in Register CPT1MX | Signal Name | QSOP24 Pin Name | QFN20 Pin Name | SOIC16 Pin Name |
|---|---|---|---|---|
| 0000 | CP1P.0 | P1.0 | P1.0 | P0.6 |
| 0001 | CP1P.1 | P1.1 | P1.1 | P0.7 |
| 0010 | CP1P.2 | P1.2 | P1.2 | P1.0 |
| 0011 | CP1P.3 | P1.3 | P1.3 | P1.1 |
| 0100 | CP1P.4 | P1.4 | P1.4 | P1.2 |
| 0101 | CP1P.5 | P1.5 | P1.5 | P1.3 |
| 0110 | CP1P.6 | P1.6 | P1.6 | Reserved |
| 0111 | CP1P.7 | P1.7 | Reserved | Reserved |
| 1000 | LDO | Internal 1.8 V LDO Output | | |
| 1001-1111 | None | No connection | | |

**Table 17.4. CMP1 Negative Input Multiplexer Channels**

| CMXN Setting in Register CPT1MX | Signal Name | QSOP24 Pin Name | QFN20 Pin Name | SOIC16 Pin Name |
|---|---|---|---|---|
| 0000 | CP1N.0 | P1.0 | P1.0 | P0.6 |
| 0001 | CP1N.1 | P1.1 | P1.1 | P0.7 |
| 0010 | CP1N.2 | P1.2 | P1.2 | P1.0 |
| 0011 | CP1N.3 | P1.3 | P1.3 | P1.1 |
| 0100 | CP1N.4 | P1.4 | P1.4 | P1.2 |
| 0101 | CP1N.5 | P1.5 | P1.5 | P1.3 |
| 0110 | CP1N.6 | P1.6 | P1.6 | Reserved |
| 0111 | CP1N.7 | P1.7 | Reserved | Reserved |
| 1000 | GND | GND | | |
| 1001-1111 | None | No connection | | |

## 17.2. Functional Description

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 (CPn), or an asynchronous “raw” output (CPnA). The asynchronous CPnA signal is available even when the system clock is not active. This allows the comparator to operate and generate an output with the device in STOP mode.

When disabled, the comparator output (if assigned to a port I/O pin via the crossbar) defaults to the logic low state, and the power supply to the comparator is turned off.

The comparator response time may be configured in software via the CPTnMD register. Selecting a longer response time reduces the comparator supply current.

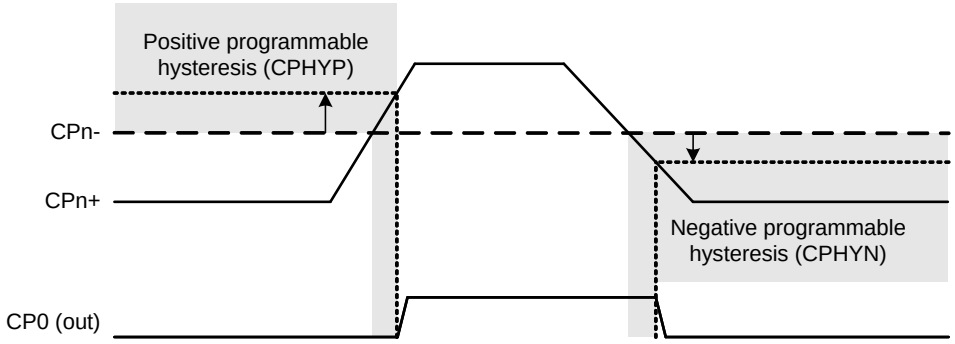



**Figure 17.2. Comparator Hysteresis Plot**

The comparator hysteresis is software-programmable via its Comparator Control register CPTnCN. The user can program both the amount of hysteresis voltage (referred to the input voltage) and the positive and negative-going symmetry of this hysteresis around the threshold voltage.

The comparator hysteresis is programmable using the CPHYN and CPHYP fields in the Comparator Control Register CPTnCN. The amount of negative hysteresis voltage is determined by the settings of the CPHYN bits. As shown in Figure 17.2, settings of 20, 10, or 5 mV (nominal) of negative hysteresis can be programmed, or negative hysteresis can be disabled. In a similar way, the amount of positive hysteresis is determined by the setting the CPHYP bits.

Comparator interrupts can be generated on both rising-edge and falling-edge output transitions. The CPFIF flag is set to logic 1 upon a comparator falling-edge occurrence, and the CPRIF flag is set to logic 1 upon the comparator rising-edge occurrence. Once set, these bits remain set until cleared by software. The comparator rising-edge interrupt mask is enabled by setting CPRIE to a logic 1. The comparator falling-edge interrupt mask is enabled by setting CPFIE to a logic 1.

The output state of the comparator can be obtained at any time by reading the CPOUT bit. The comparator is enabled by setting the CPEN bit to logic 1, and is disabled by clearing this bit to logic 0.

Note that false rising edges and falling edges can be detected when the comparator is first powered on or if changes are made to the hysteresis or response time control bits. Therefore, it is recommended that the rising-edge and falling-edge flags be explicitly cleared to logic 0 a short time after the comparator is enabled or its mode bits have been changed, before enabling comparator interrupts.

## 17.3. Comparator Control Registers

### Register 17.1. CPT0CN: Comparator 0 Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CPEN | CPOUT | CPRIF | CPFIF | CPHYP | | CPHYN | |
| Type | RW | R | RW | RW | RW | | RW | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0x9B** | | | | | | | | |

**Table 17.5. CPT0CN Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7 | CPEN | **Comparator 0 Enable Bit.**<br>0: Comparator Disabled.<br>1: Comparator Enabled. |
| 6 | CPOUT | **Comparator 0 Output State Flag.**<br>0: Voltage on CP0P < CP0N.<br>1: Voltage on CP0P > CP0N. |
| 5 | CPRIF | **Comparator 0 Rising-Edge Flag. Must be cleared by software.**<br>0: No Comparator Rising Edge has occurred since this flag was last cleared.<br>1: Comparator Rising Edge has occurred. |
| 4 | CPFIF | **Comparator 0 Falling-Edge Flag. Must be cleared by software.**<br>0: No Comparator Falling-Edge has occurred since this flag was last cleared.<br>1: Comparator Falling-Edge has occurred. |
| 3:2 | CPHYP | **Comparator 0 Positive Hysteresis Control Bits.**<br>00: Positive Hysteresis Disabled.<br>01: Positive Hysteresis = 5 mV.<br>10: Positive Hysteresis = 10 mV.<br>11: Positive Hysteresis = 20 mV. |
| 1:0 | CPHYN | **Comparator 0 Negative Hysteresis Control Bits.**<br>00: Negative Hysteresis Disabled.<br>01: Negative Hysteresis = 5 mV.<br>10: Negative Hysteresis = 10 mV.<br>11: Negative Hysteresis = 20 mV. |

## Register 17.2. CPT0MD: Comparator 0 Mode

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CPLOUT | Reserved | CPRIE | CPFIE | Reserved | | CPMD | |
| Type | RW | R | RW | RW | R | | RW | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| **SFR Address: 0x9D** | | | | | | | | |

**Table 17.6. CPT0MD Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7 | CPLOUT | **Comparator 0 Latched Output Flag.**<br>This bit represents the comparator output value at the most recent PCA counter overflow.<br>0: Comparator output was logic low at last PCA overflow.<br>1: Comparator output was logic high at last PCA overflow. |
| 6 | Reserved | Must write reset value. |
| 5 | CPRIE | **Comparator 0 Rising-Edge Interrupt Enable.**<br>0: Comparator Rising-Edge interrupt disabled.<br>1: Comparator Rising-Edge interrupt enabled. |
| 4 | CPFIE | **Comparator 0 Falling-Edge Interrupt Enable.**<br>0: Comparator Falling-Edge interrupt disabled.<br>1: Comparator Falling-Edge interrupt enabled. |
| 3:2 | Reserved | Must write reset value. |
| 1:0 | CPMD | **Comparator 0 Mode Select.**<br>These bits affect the response time and power consumption of the comparator.<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) |

## Register 17.3. CPT0MX: Comparator 0 Multiplexer Selection

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CMXN | | | | CMXP | | | |
| Type | RW | | | | RW | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **SFR Address: 0x9F** | | | | | | | | |

**Table 17.7. CPT0MX Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:4 | CMXN | **Comparator 0 Negative Input MUX Selection.**<br>0000: External pin CP0N.0<br>0001: External pin CP0N.1<br>0010: External pin CP0N.2<br>0011: External pin CP0N.3<br>0100: External pin CP0N.4<br>0101: External pin CP0N.5<br>0110: External pin CP0N.6<br>0111: External pin CP0N.7<br>1000: GND<br>1001-1111: Reserved. |
| 3:0 | CMXP | **Comparator 0 Positive Input MUX Selection.**<br>0000: External pin CP0P.0<br>0001: External pin CP0P.1<br>0010: External pin CP0P.2<br>0011: External pin CP0P.3<br>0100: External pin CP0P.4<br>0101: External pin CP0P.5<br>0110: External pin CP0P.6<br>0111: External pin CP0P.7<br>1000: Internal LDO output<br>1001-1111: Reserved. |

## Register 17.4. CPT1CN: Comparator 1 Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CPEN | CPOUT | CPRIF | CPFIF | CPHYP | | CPHYN | |
| Type | RW | R | RW | RW | RW | | RW | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xBF** | | | | | | | | |

**Table 17.8. CPT1CN Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7 | CPEN | **Comparator 1 Enable Bit.**<br>0: Comparator Disabled.<br>1: Comparator Enabled. |
| 6 | CPOUT | **Comparator 1 Output State Flag.**<br>0: Voltage on CP1P < CP1N.<br>1: Voltage on CP1P > CP1N. |
| 5 | CPRIF | **Comparator 1 Rising-Edge Flag. Must be cleared by software.**<br>0: No Comparator Rising Edge has occurred since this flag was last cleared.<br>1: Comparator Rising Edge has occurred. |
| 4 | CPFIF | **Comparator 1 Falling-Edge Flag. Must be cleared by software.**<br>0: No Comparator Falling Edge has occurred since this flag was last cleared.<br>1: Comparator Falling Edge has occurred. |
| 3:2 | CPHYP | **Comparator 1 Positive Hysteresis Control Bits.**<br>00: Positive Hysteresis Disabled.<br>01: Positive Hysteresis = 5 mV.<br>10: Positive Hysteresis = 10 mV.<br>11: Positive Hysteresis = 20 mV. |
| 1:0 | CPHYN | **Comparator 1 Negative Hysteresis Control Bits.**<br>00: Negative Hysteresis Disabled.<br>01: Negative Hysteresis = 5 mV.<br>10: Negative Hysteresis = 10 mV.<br>11: Negative Hysteresis = 20 mV. |

## Register 17.5. CPT1MD: Comparator 1 Mode

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CPLOUT | Reserved | CPRIE | CPFIE | Reserved | | CPMD | |
| Type | RW | R | RW | RW | R | | RW | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| **SFR Address: 0xAB** | | | | | | | | |

**Table 17.9. CPT1MD Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7 | CPLOUT | **Comparator 1 Latched Output Flag.**<br>This bit represents the comparator output value at the most recent PCA counter overflow.<br>0: Comparator output was logic low at last PCA overflow.<br>1: Comparator output was logic high at last PCA overflow. |
| 6 | Reserved | Must write reset value. |
| 5 | CPRIE | **Comparator 1 Rising-Edge Interrupt Enable.**<br>0: Comparator Rising-Edge interrupt disabled.<br>1: Comparator Rising-Edge interrupt enabled. |
| 4 | CPFIE | **Comparator 1 Falling-Edge Interrupt Enable.**<br>0: Comparator Falling-Edge interrupt disabled.<br>1: Comparator Falling-Edge interrupt enabled. |
| 3:2 | Reserved | Must write reset value. |
| 1:0 | CPMD | **Comparator 1 Mode Select.**<br>These bits affect the response time and power consumption of the comparator.<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) |

## Register 17.6. CPT1MX: Comparator 1 Multiplexer Selection

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CMXN | | | | CMXP | | | |
| Type | RW | | | | RW | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **SFR Address: 0xAA** | | | | | | | | |

**Table 17.10. CPT1MX Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:4 | CMXN | **Comparator 1 Negative Input MUX Selection.**<br>0000: External pin CP1N.0<br>0001: External pin CP1N.1<br>0010: External pin CP1N.2<br>0011: External pin CP1N.3<br>0100: External pin CP1N.4<br>0101: External pin CP1N.5<br>0110: External pin CP1N.6<br>0111: External pin CP1N.7<br>1000: GND<br>1001-1111: Reserved. |
| 3:0 | CMXP | **Comparator 1 Positive Input MUX Selection.**<br>0000: External pin CP1P.0<br>0001: External pin CP1P.1<br>0010: External pin CP1P.2<br>0011: External pin CP1P.3<br>0100: External pin CP1P.4<br>0101: External pin CP1P.5<br>0110: External pin CP1P.6<br>0111: External pin CP1P.7<br>1000: Internal LDO output<br>1001-1111: Reserved. |

# 18. Cyclic Redundancy Check Unit (CRC0)

C8051F85x/86x 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 CRCPNT bits and CRC0DAT register, as shown in Figure 18.1. CRC0 also has a bit reverse register for quick data manipulation.

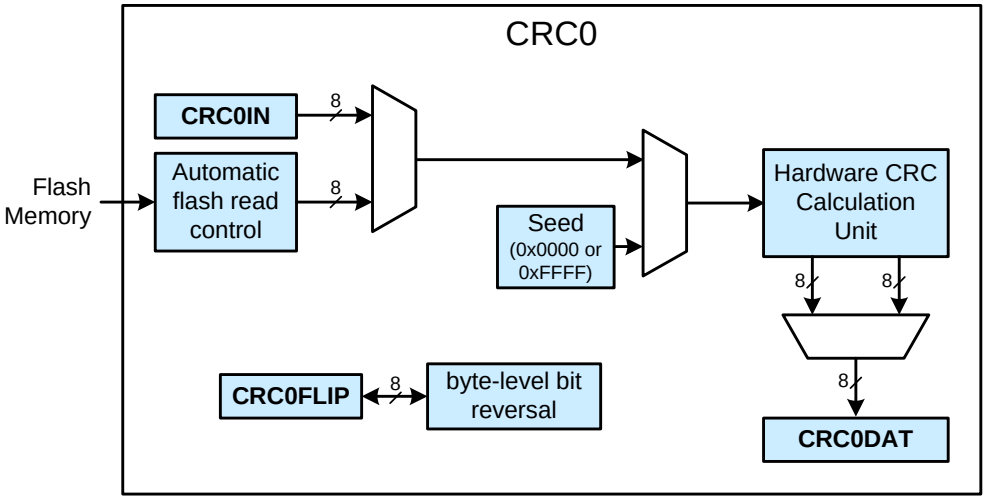



**Figure 18.1. CRC0 Block Diagram**

## 18.1. CRC Algorithm

The 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 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 18.1 lists several input values and the associated outputs using the 16-bit CRC algorithm:

**Table 18.1. Example 16-bit CRC Outputs**

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

## 18.2. 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 CRCVAL to 0 for 0x0000 or 1 for 0xFFFF).
2. Set the result to its initial value (Write 1 to CRCINIT).

## 18.3. 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 read from flash. 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 to 1 in CRC0AUTO.
4. Write the number of 256 byte blocks to perform in the CRC calculation to CRCCNT.
5. Write any value to CRC0CN (or OR its contents with 0x00) to initiate the CRC calculation. The CPU will not execute code any additional code until the CRC operation completes. See the note in the CRC0CN register definition for more information on how to properly initiate a CRC calculation.
6. Clear the AUTOEN bit in CRC0AUTO.
7. Read the CRC result.

## 18.4. Accessing the CRC0 Result

The internal CRC0 result is 16 bits. The CRCPNT 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.

## 18.5. CRC0 Bit Reverse Feature

CRC0 includes hardware to reverse the bit order of each bit in a byte as shown in Figure 18.2. 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.

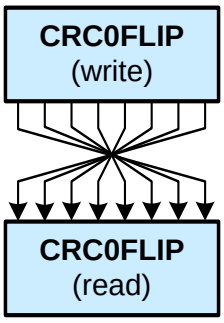



**Figure 18.2. Bit Reversal**

## 18.6. CRC Control Registers

### Register 18.1. CRC0CN: CRC0 Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Reserved | | | | CRCINIT | CRCVAL | Reserved | CRCPNT |
| Type | R | | | | RW | RW | R | RW |
| Reset | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xCE** | | | | | | | | |

**Table 18.2. CRC0CN Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:4 | Reserved | Must write reset value. |
| 3 | CRCINIT | **CRC Result Initialization Bit.**<br>Writing a 1 to this bit initializes the entire CRC result based on CRCVAL. |
| 2 | CRCVAL | **CRC Set Value Initialization Bit.**<br>This bit selects the set value of the CRC result.<br>0: CRC result is set to 0x0000 on write of 1 to CRCINIT.<br>1: CRC result is set to 0xFFFF on write of 1 to CRCINIT. |
| 1 | Reserved | Must write reset value. |
| 0 | CRCPNT | **CRC Result Pointer.**<br>Specifies the byte of the CRC result to be read/written on the next access to CRC0DAT. This bit will automatically toggle 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. | | |

## Register 18.2. CRC0IN: CRC0 Data Input

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CRC0IN | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xDD** | | | | | | | | |

**Table 18.3. CRC0IN Register Bit Descriptions**

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

## Register 18.3. CRC0DAT: CRC0 Data Output

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CRC0DAT | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xDE** | | | | | | | | |

**Table 18.4. CRC0DAT Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | CRC0DAT | **CRC 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). |
| **Note:** CRC0DAT may not be valid for one cycle after setting the CRC0INIT bit in the CRC0CN register to 1. Any time CRC0INIT is written to 1 by firmware, at least one instruction should be performed before reading CRC0DAT. | | |

## Register 18.4. CRC0AUTO: CRC0 Automatic Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | AUTOEN | Reserved | CRCST | | | | | |
| Type | RW | R | RW | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xD2** | | | | | | | | |

**Table 18.5. CRC0AUTO Register Bit Descriptions**

| 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 CRCST and continuing for CRCCNT sectors. |
| 6 | Reserved | Must write reset value. |
| 5:0 | CRCST | **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 CRCST x block_size, where block_size is 256 bytes. |

## Register 18.5. CRC0CNT: CRC0 Automatic Flash Sector Count

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CRCDN | Reserved | | CRCCNT | | | | |
| Type | R | R | | RW | | | | |
| Reset | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xD3** | | | | | | | | |

**Table 18.6. CRC0CNT Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7 | CRCDN | **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. |
| 6:5 | Reserved | Must write reset value. |
| 4:0 | CRCCNT | **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 (CRCST+CRCCNT) x Block Size - 1. The block size is 256 bytes. |

## Register 18.6. CRC0FLIP: CRC0 Bit Flip

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CRC0FLIP | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xCF** | | | | | | | | |

**Table 18.7. CRC0FLIP Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | CRC0FLIP | **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. |

## 19. External Interrupts ($\overline{INT0}$ and $\overline{INT1}$)

The C8051F85x/86x device family includes two external digital interrupt sources ($\overline{INT0}$ and $\overline{INT1}$), with dedicated interrupt sources (up to 16 additional I/O interrupts are available through the port match function). As is the case on a standard 8051 architecture, certain controls for these two interrupt sources are available in the Timer0/1 registers. Extensions to these controls which provide additional functionality on C8051F85x/86x devices are available in the IT01CF register. $\overline{INT0}$ and $\overline{INT1}$ are configurable as active high or low, edge- or level-sensitive. The IN0PL and IN1PL bits in the IT01CF register select active high or active low; the IT0 and IT1 bits in TCON select level- or edge-sensitive. The table below lists the possible configurations.

| IT0 | IN0PL | $\overline{INT0}$ 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 | $\overline{INT1}$ 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 |

$\overline{INT0}$ and $\overline{INT1}$ are assigned to port pins as defined in the IT01CF register. Note that $\overline{INT0}$ and $\overline{INT1}$ port pin assignments are independent of any crossbar assignments. $\overline{INT0}$ and $\overline{INT1}$ 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 $\overline{INT0}$ and/or $\overline{INT1}$, configure the crossbar to skip the selected pin(s).

IE0 and IE1 in the TCON register serve as the interrupt-pending flags for the $\overline{INT0}$ and $\overline{INT1}$ external interrupts, respectively. If an $\overline{INT0}$ or $\overline{INT1}$ 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.

## 19.1. External Interrupt Control Registers

### Register 19.1. IT01CF: INT0/INT1 Configuration

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | IN1PL | IN1SL | | | IN0PL | IN0SL | | |
| Type | RW | RW | | | RW | RW | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| **SFR Address: 0xE4** | | | | | | | | |

**Table 19.1. IT01CF Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7 | IN1PL | **INT1 Polarity.**<br>0: INT1 input is active low.<br>1: INT1 input is active high. |
| 6:4 | IN1SL | **INT1 Port Pin Selection Bits.**<br>These bits select which Port pin is assigned to INT1. This pin assignment is independent of the Crossbar; INT1 will monitor the assigned port pin without disturbing the peripheral that has been assigned the Port pin via the Crossbar. The Crossbar will not assign the Port pin to a peripheral if it is configured to skip the selected pin.<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 | **INT0 Polarity.**<br>0: INT0 input is active low.<br>1: INT0 input is active high. |

**Table 19.1. IT01CF Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 2:0 | IN0SL | **INT0 Port Pin Selection Bits.**<br>These bits select which Port pin is assigned to INT0. This pin assignment is independent of the Crossbar; INT0 will monitor the assigned Port pin without disturbing the peripheral that has been assigned the Port pin via the Crossbar. The Crossbar will not assign the Port pin to a peripheral if it is configured to skip the selected pin.<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 |

## 20. Programmable Counter Array (PCA0)

The Programmable Counter Array (PCA0) provides three channels of enhanced timer and PWM functionality while requiring less CPU intervention than standard counter/timers. The PCA consists of a dedicated 16-bit counter/timer and three 16-bit capture/compare modules. The counter/timer is driven by a programmable timebase that can select between seven sources: system clock, system clock divided by four, system clock divided by twelve, the external oscillator clock source divided by 8, low frequency oscillator 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. Additionally, all PWM modes support both center and edge-aligned operation. The external oscillator and LFO oscillator clock options allow the PCA to be clocked by an external oscillator or the LFO while the internal oscillator drives the system clock. 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 I/O signals have programmable polarity and Comparator 0 can optionally be used to perform a cycle-by-cycle kill operation on the PCA outputs. A PCA block diagram is shown in Figure 20.1

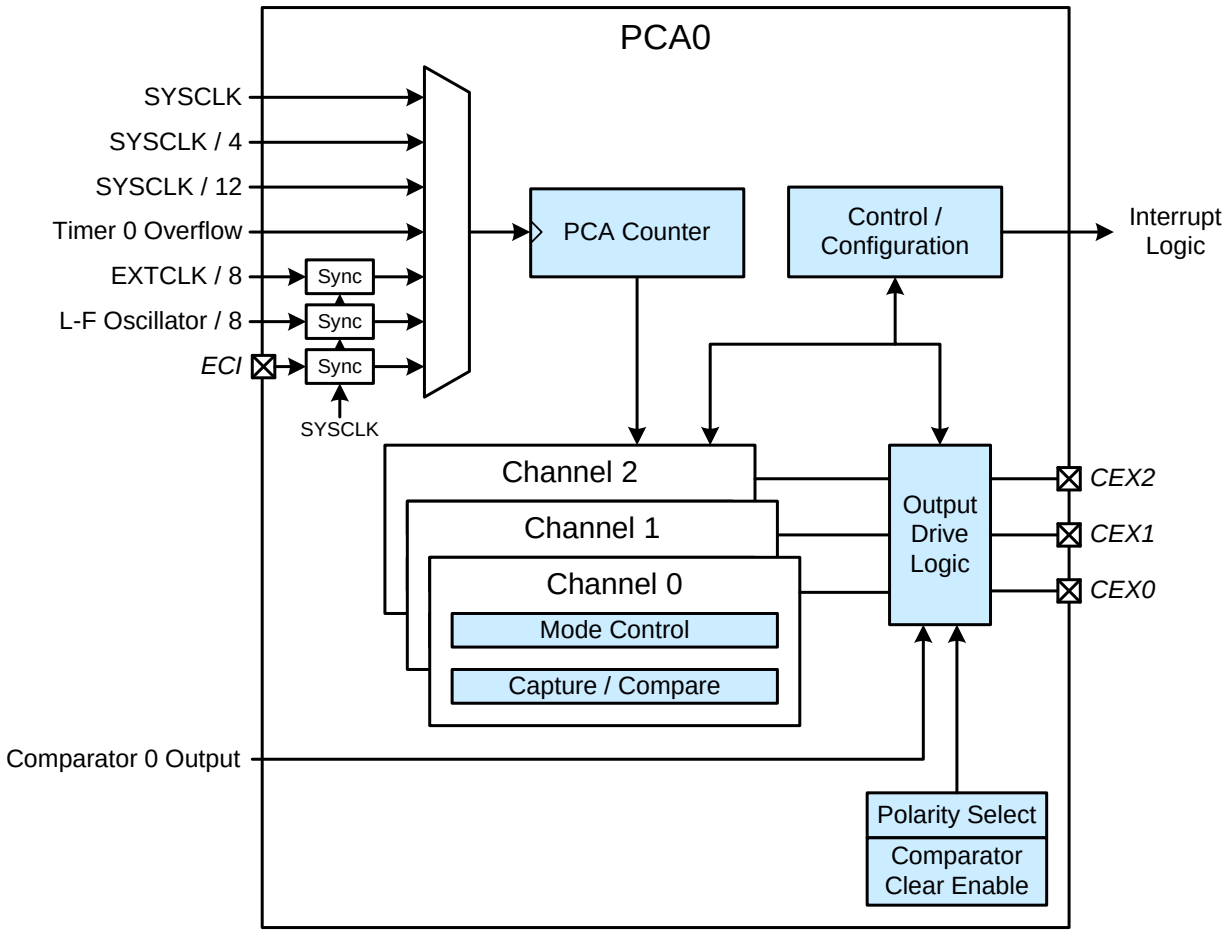



**Figure 20.1. PCA0 Block Diagram**

## 20.1. PCA Counter/Timer

The 16-bit PCA counter/timer consists of two 8-bit SFRs: PCA0L and PCA0H. PCA0H is the high byte of the 16-bit counter/timer and PCA0L is the low byte. 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 20.1.

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 20.1. PCA Timebase Input Options**

| CPS2 | CPS1 | CPS0 | Timebase |
|---|---|---|---|
| 0 | 0 | 0 | System clock divided by 12 |
| 0 | 0 | 1 | System clock divided by 4 |
| 0 | 1 | 0 | Timer 0 overflow |
| 0 | 1 | 1 | High-to-low transitions on ECI (max rate = system clock divided by 4)* |
| 1 | 0 | 0 | System clock |
| 1 | 0 | 1 | External oscillator source divided by 8* |
| 1 | 1 | 0 | Low frequency oscillator divided by 8* |
| 1 | 1 | 1 | Reserved |

***Note:** Synchronized with the system clock.

## 20.2. PCA0 Interrupt Sources

The PCA0 module shares one interrupt vector among all of its modules. There are are several 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 - 11th bit of the PCA0 counter, and the individual flags for each PCA channel (CCFn), 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.

## 20.3. 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. Table 20.2 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 20.2. 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 | 4–3 | 2–0 |
| Capture triggered by positive edge on CEXn | X | X | 1 | 0 | 0 | 0 | 0 | A | 0 | X | B | XX | XXX |
| Capture triggered by negative edge on CEXn | X | X | 0 | 1 | 0 | 0 | 0 | A | 0 | X | B | XX | XXX |
| Capture triggered by any transition on CEXn | X | X | 1 | 1 | 0 | 0 | 0 | A | 0 | X | B | XX | XXX |
| Software Timer | X | C | 0 | 0 | 1 | 0 | 0 | A | 0 | X | B | XX | XXX |
| High Speed Output | X | C | 0 | 0 | 1 | 1 | 0 | A | 0 | X | B | XX | XXX |
| Frequency Output | X | C | 0 | 0 | 0 | 1 | 1 | A | 0 | X | B | XX | XXX |
| 8-Bit Pulse Width Modulator (Note 7) | 0 | C | 0 | 0 | E | 0 | 1 | A | 0 | X | B | XX | 000 |
| 9-Bit Pulse Width Modulator (Note 7) | 0 | C | 0 | 0 | E | 0 | 1 | A | D | X | B | XX | 001 |
| 10-Bit Pulse Width Modulator (Note 7) | 0 | C | 0 | 0 | E | 0 | 1 | A | D | X | B | XX | 010 |
| 11-Bit Pulse Width Modulator (Note 7) | 0 | C | 0 | 0 | E | 0 | 1 | A | D | X | B | XX | 011 |
| 16-Bit Pulse Width Modulator | 1 | C | 0 | 0 | E | 0 | 1 | A | 0 | X | B | XX | XXX |

**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 - 11th bit overflow interrupt (Depends on setting of CLSEL).
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.

### 20.3.1. Output Polarity

The output polarity of each PCA channel is individually selectable using the PCA0POL register. By default, all output channels are configured to drive the PCA output signals (CEXn) with their internal polarity. When the CEXnPOL bit for a specific channel is set to 1, that channel's output signal will be inverted at the pin. All other properties of the channel are unaffected, and the inversion does not apply to PCA input signals. Note that changes in the PCA0POL register take effect immediately at the associated output pin.

### 20.3.2. 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.

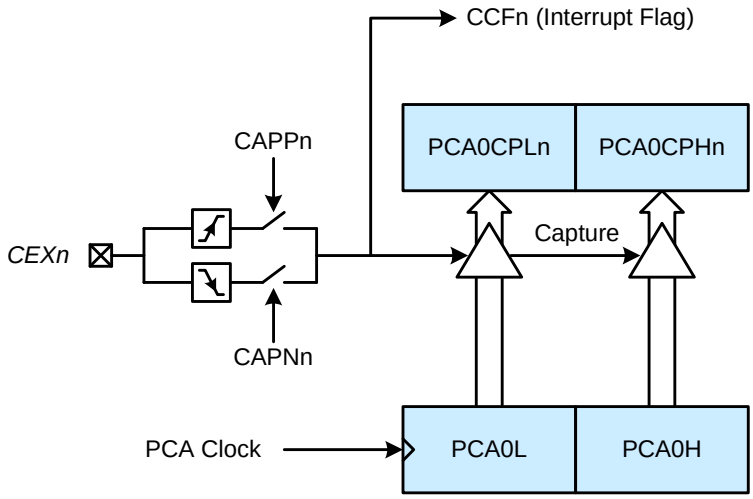



**Figure 20.2. PCA Capture Mode Diagram**

**Note:** The CEXn input signal must remain high or low for at least 2 system clock cycles to be recognized by the hardware.

### 20.3.3. 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.

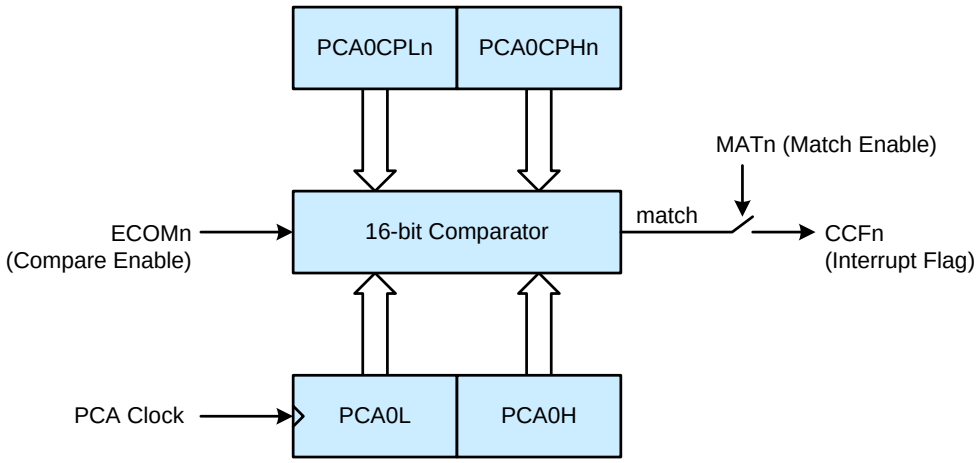



**Figure 20.3. PCA Software Timer Mode Diagram**

### 20.3.4. 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.

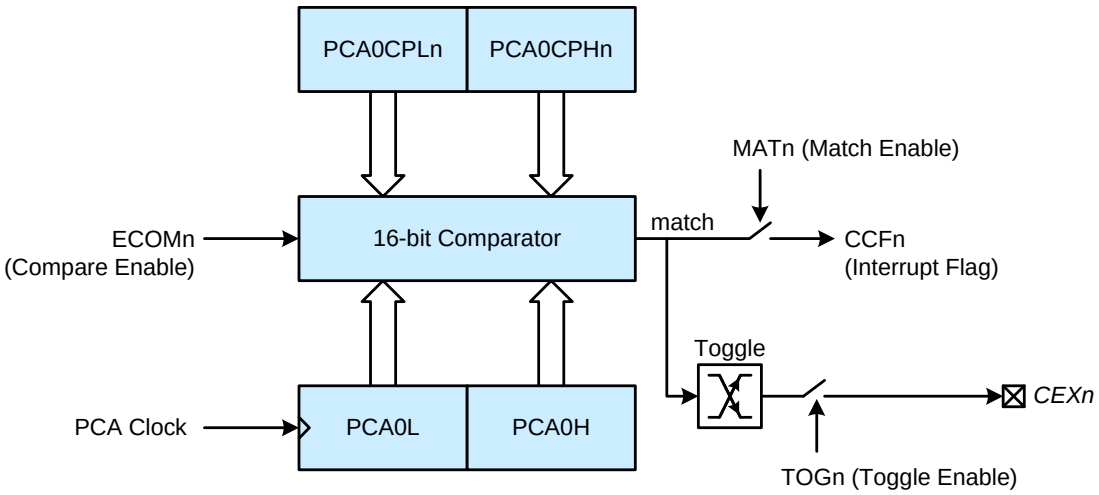



**Figure 20.4. PCA High-Speed Output Mode Diagram**

### 20.3.5. 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 20.1.

$$F_{CEXn} = \frac{F_{PCA}}{2 \times PCA0CPHn}$$

Note: A value of 0x00 in the PCA0CPHn register is equal to 256 for this equation.

**Equation 20.1. 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, n 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.

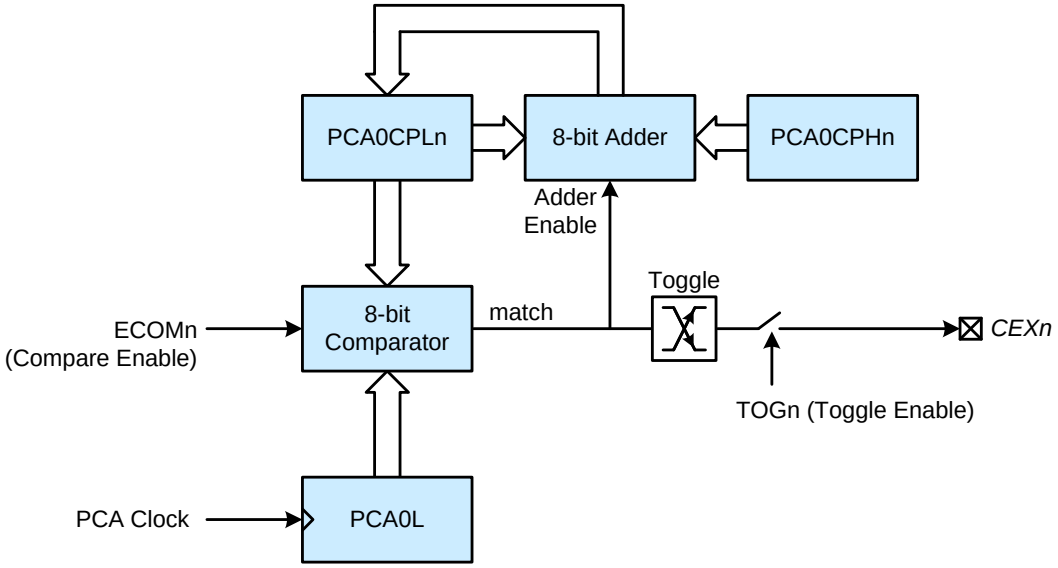



**Figure 20.5. PCA Frequency Output Mode**

## 20.4. PWM Waveform Generation

The PCA can generate edge- or center-aligned PWM waveforms with resolutions of 8, 9, 10, 11 or 16 bits. PWM resolution depends on the module setup, as specified within the individual module PCA0CPMn registers as well as the PCA0PWM register. Modules can be configured for 8-11 bit mode, or for 16-bit mode individually using the PCA0CPMn registers. All modules configured for 8-11 bit mode will have the same resolution, specified by the PCA0PWM register. When operating in one of the PWM modes, each module may be individually configured for center or edge-aligned PWM waveforms. Each channel has a single bit in the PCA0CENT register to select between the two options.

### 20.4.1. Edge Aligned PWM

When configured for edge-aligned mode, a module will generate an edge transition at two points for every $2^N$ PCA clock cycles, where N is the selected PWM resolution in bits. In edge-aligned mode, these two edges are referred to as the “match” and “overflow” edges. The polarity at the output pin is selectable, and can be inverted by setting the appropriate channel bit to ‘1’ in the PCA0POL register. Prior to inversion, a match edge sets the channel to logic high, and an overflow edge clears the channel to logic low.

The match edge occurs when the the lowest N bits of the module’s PCA0CPn register match the corresponding bits of the main PCA0 counter register. For example, with 10-bit PWM, the match edge will occur any time bits 9-0 of the PCA0CPn register match bits 9-0 of the PCA0 counter value.

The overflow edge occurs when an overflow of the PCA0 counter happens at the desired resolution. For example, with 10-bit PWM, the overflow edge will occur when bits 0-9 of the PCA0 counter transition from all 1’s to all 0’s. All modules configured for edge-aligned mode at the same resolution will align on the overflow edge of the waveforms.

An example of the PWM timing in edge-aligned mode for two channels is shown in Figure 20.6. In this example, the CEX0POL and CEX1POL bits are cleared to 0.

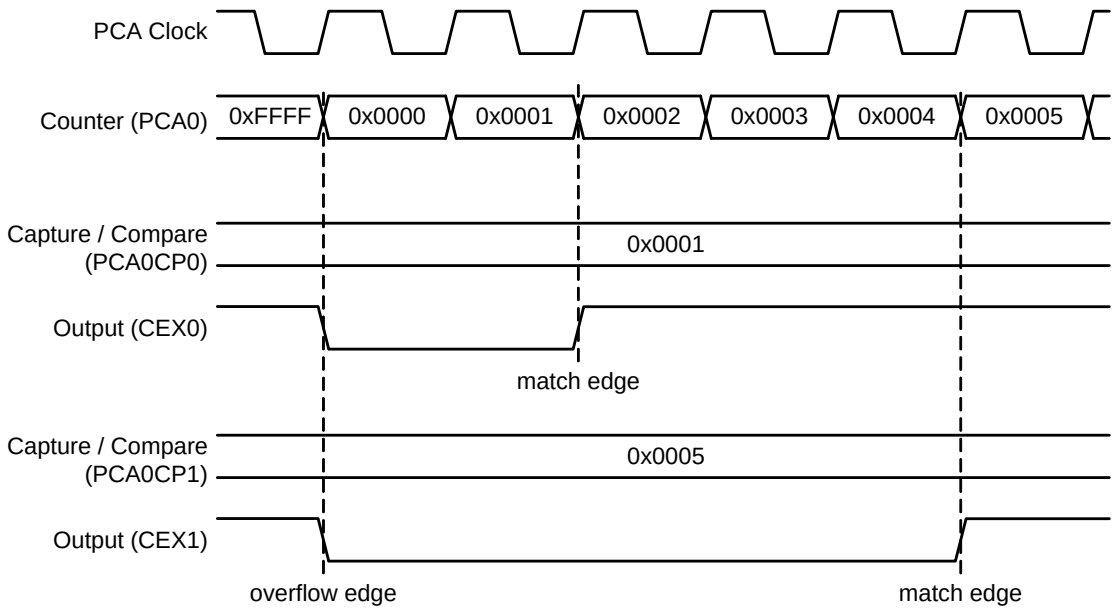



**Figure 20.6. Edge-Aligned PWM Timing**

For a given PCA resolution, the unused high bits in the PCA0 counter and the PCA0CPn compare registers are ignored, and only the used bits of the PCA0CPn register determine the duty cycle. Equation 20.2 describes the duty cycle when CEXnPOL in the PCA0POL regsiter is cleared to 0. Equation 20.3 describes the duty cycle when CEXnPOL in the PCA0POL regsiter is set to 1. A 0% duty cycle for the channel (with CEXnPOL = 0) is achieved by clearing the module’s ECOM bit to 0. This will

disable the comparison, and prevent the match edge from occuring. Note that although the PCA0CPn compare register determines the duty cycle, it is not always appropriate for firmware to update this register directly. See the sections on 8 to 11-bit and 16-bit PWM mode for additional details on adjusting duty cycle in the various modes.

$$\text{Duty Cycle} = \frac{(2^N - \text{PCA0CPn})}{2^N}$$

**Equation 20.2. N-bit Edge-Aligned PWM Duty Cycle With CEXnPOL = 0 (N = PWM resolution)**

$$\text{Duty Cycle} = \frac{\text{PCA0CPn}}{2^N}$$

**Equation 20.3. N-bit Edge-Aligned PWM Duty Cycle With CEXnPOL = 0 (N = PWM resolution)**

### 20.4.2. Center Aligned PWM

When configured for center-aligned mode, a module will generate an edge transition at two points for every $2^{(N+1)}$ PCA clock cycles, where N is the selected PWM resolution in bits. In center-aligned mode, these two edges are referred to as the “up” and “down” edges. The polarity at the output pin is selectable, and can be inverted by setting the appropriate channel bit to ‘1’ in the PCA0POL register.

The generated waveforms are centered about the points where the lower N bits of the PCA0 counter are zero. The $(N+1)^{th}$ bit in the PCA0 counter acts as a selection between up and down edges. In 16-bit mode, a special 17th bit is implemented internally for this purpose. At the center point, the (non-inverted) channel output will be low when the $(N+1)^{th}$ bit is ‘0’ and high when the $(N+1)^{th}$ bit is ‘1’, except for cases of 0% and 100% duty cycle. Prior to inversion, an up edge sets the channel to logic high, and a down edge clears the channel to logic low.

Down edges occur when the $(N+1)^{th}$ bit in the PCA0 counter is one, and a logical inversion of the value in the module’s PCA0CPn register matches the main PCA0 counter register for the lowest N bits. For example, with 10-bit PWM, the down edge will occur when the one’s complement of bits 9-0 of the PCA0CPn register match bits 9-0 of the PCA0 counter, and bit 10 of the PCA0 counter is ‘1’.

Up edges occur when the $(N+1)^{th}$ bit in the PCA0 counter is zero, and the lowest N bits of the module’s PCA0CPn register match the value of (PCA0 - 1). For example, with 10-bit PWM, the up edge will occur when bits 9-0 of the PCA0CPn register are one less than bits 9-0 of the PCA0 counter, and bit 10 of the PCA0 counter is ‘0’.

An example of the PWM timing in center-aligned mode for two channels is shown in Figure 20.7. In this example, the CEX0POL and CEX1POL bits are cleared to 0.

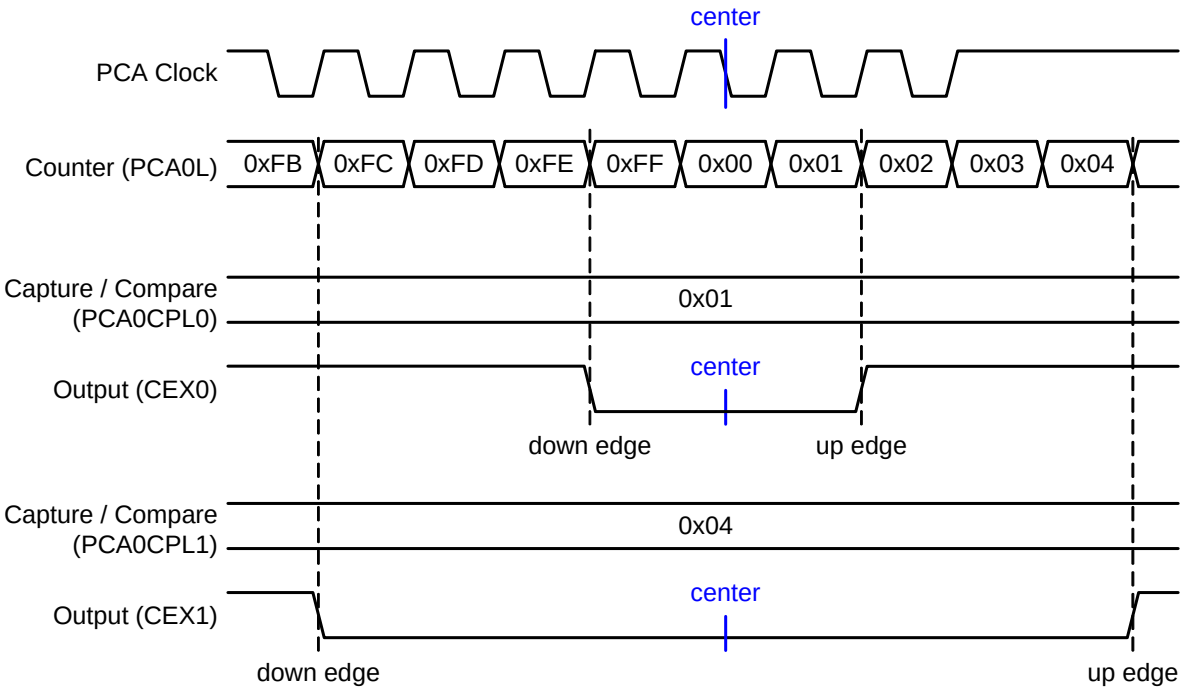



**Figure 20.7. Center-Aligned PWM Timing**

Equation 20.4 describes the duty cycle when CEXnPOL in the PCA0POL regsiter is cleared to 0. Equation 20.5 describes the duty cycle when CEXnPOL in the PCA0POL regsiter is set to 1. The equations are true only when the lowest N bits of the PCA0CPn register are not all 0's or all 1's. With CEXnPOL equal to zero, 100% duty cycle is produced when the lowest N bits of PCA0CPn are all 0, and 0% duty cycle is produced when the lowest N bits of PCA0CPn are all 1. For a given PCA resolution, the unused high bits in the PCA0 counter and the PCA0CPn compare registers are ignored, and only the used bits of the PCA0CPn register determine the duty cycle.

Note that although the PCA0CPn compare register determines the duty cycle, it is not always appropriate for firmware to update this register directly. See the sections on 8 to 11-bit and 16-bit PWM mode for additional details on adjusting duty cycle in the various modes.

$$\text{Duty Cycle} = \frac{(2^N - \text{PCA0CPn}) - \frac{1}{2}}{2^N}$$

**Equation 20.4. N-bit Center-Aligned PWM Duty Cycle With CEXnPOL = 0 (N = PWM resolution)**

$$\text{Duty Cycle} = \frac{\text{PCA0CPn} + \frac{1}{2}}{2^N}$$

**Equation 20.5. N-bit Center-Aligned PWM Duty Cycle With CEXnPOL = 1 (N = PWM resolution)**

### 20.4.3. 8 to11-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 through 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 through 11-bit PWM modes. It is important to note that all channels configured for 8 to 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. Each channel configured for a PWM mode can be individually selected to operate in edge-aligned or center-aligned mode.

#### 20.4.3.1. 8-bit Pulse Width Modulator Mode

In 8-bit PWM mode, the duty cycle is determined by the value of the low byte of the PCA0CPn register (PCA0CPLn). To adjust the duty cycle, PCA0CPLn should not normally be written directly. Instead, it is recommended to adjust the duty cycle using the high byte of the PCA0CPn register (register PCA0CPHn). This allows seamless updating of the PWM waveform, as PCA0CPLn is reloaded automatically with the value stored in PCA0CPHn during the overflow edge (in edge-aligned mode) or the up edge (in center-aligned mode).

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 a match edge or up edge occurs. The COVF flag in PCA0PWM can be used to detect the overflow (falling edge), which will occur every 256 PCA clock cycles.

#### 20.4.3.2. 9 to 11-bit Pulse Width Modulator Mode

In 9 to 11-bit PWM mode, the duty cycle is determined by the value of the least significant N bits of the PCA0CPn register, where N is the selected PWM resolution.

To adjust the duty cycle, PCA0CPn should not normally be written directly. Instead, it is recommended to adjust the duty cycle 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. This allows seamless updating of the PWM waveform, as the PCA0CPn register is reloaded automatically with the value stored in the auto-reload registers during the overflow edge (in edge-aligned mode) or the up edge (in center-aligned mode).

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 a match edge or up edge occurs. The COVF flag in PCA0PWM can be used to detect the overflow or down edge.

The 9 to 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 match edge or up edge occurs. The COVF flag in PCA0PWM can be used to detect the overflow or down edge.

**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.

### 20.4.4. 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 through 11-bit) PWM modes. The entire PCA0CP register is used to determine the duty cycle in 16-bit PWM mode.

To output a varying duty cycle, new value writes should be synchronized with the PCA CCFn match flag to ensure seamless updates.

16-Bit PWM mode is enabled by setting the ECOMn, PWMn, and PWM16n bits in the PCA0CPMn register. For a varying duty cycle, the match interrupt flag 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 match edge or up edge occurs. The CF flag in PCA0CN can be used to detect the overflow or down edge.

**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.

## 20.5. Comparator Clear Function

In 8/9/10/11/16-bit PWM modes, the comparator clear function utilizes the Comparator0 output synchronized to the system clock to clear CEXn to logic low for the current PWM cycle. This comparator clear function can be enabled for each PWM channel by setting the CPCEn bits to 1 in the PCA0CLR SFR. When the comparator clear function is disabled, CEXn is unaffected.

The asynchronous Comparator 0 output is logic high when the voltage of CP0+ is greater than CP0- and logic low when the voltage of CP0+ is less than CP0-. The polarity of the Comparator 0 output is used to clear CEXn as follows: when CPCPOL = 0, CEXn is cleared on the falling edge of the Comparator0 output (see Figure 20.8); when CPCPOL = 1, CEXn is cleared on the rising edge of the Comparator0 output (see Figure 20.9).

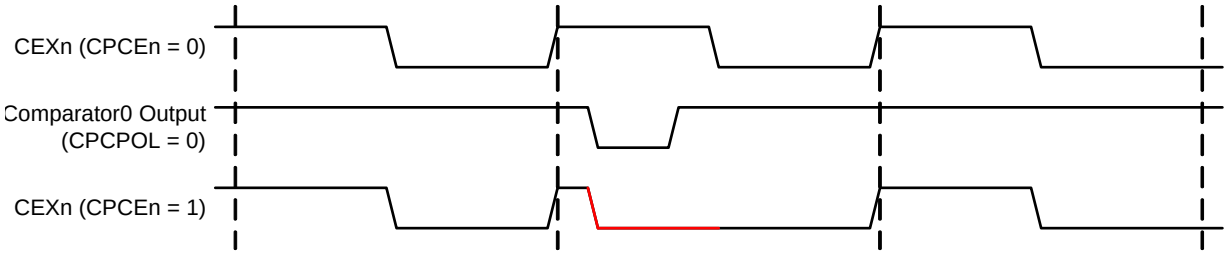



**Figure 20.8. CEXn with CPCEn = 1, CPCPOL = 0**

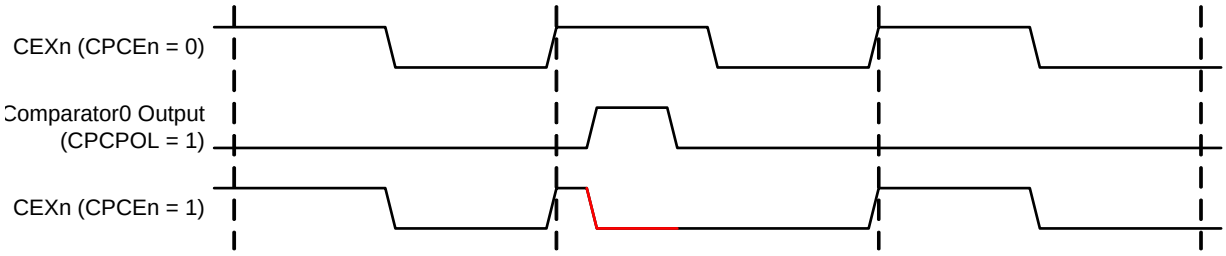



**Figure 20.9. CEXn with CPCEn = 1, CPCPOL = 1**

In the PWM cycle following the current cycle, should the Comparator 0 output remain logic low when CPCPOL = 0 or logic high when CPCPOL = 1, CEXn will continue to be cleared. See Figure 20.10 and Figure 20.11.

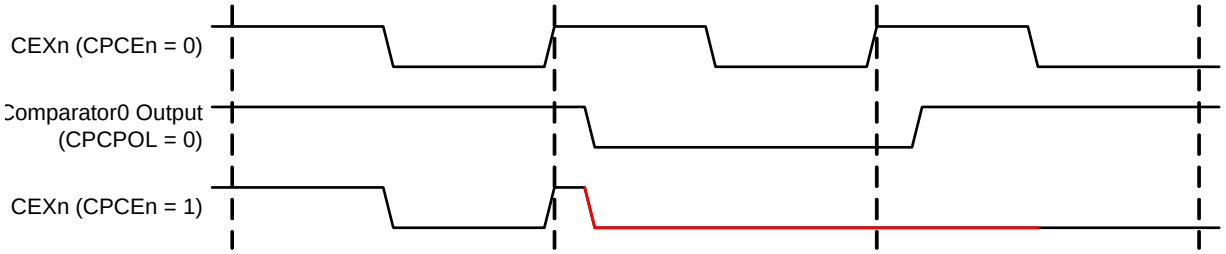



**Figure 20.10. CEXn with CPCEn = 1, CPCPOL = 0**

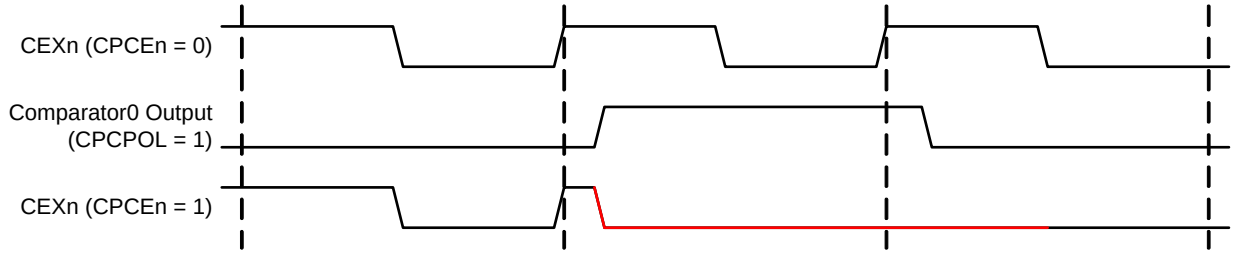



**Figure 20.11. CEXn with CPCEn = 1, CPCPOL = 1**

## 20.6. PCA Control Registers

### Register 20.1. PCA0CN: PCA Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CF | CR | Reserved | | | CCF2 | CCF1 | CCF0 |
| Type | RW | RW | R | | | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| SFR Address: 0xD8 (bit-addressable) | | | | | | | | |

**Table 20.3. PCA0CN Register Bit Descriptions**

| 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 | Reserved | Must write reset value. |
| 2 | CCF2 | **PCA Module 2 Capture/Compare Flag.**<br>This bit is set by hardware when a match or capture occurs. When the CCF2 interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by software. |
| 1 | CCF1 | **PCA Module 1 Capture/Compare Flag.**<br>This bit is set by hardware when a match or capture occurs. When the CCF1 interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by software. |
| 0 | CCF0 | **PCA Module 0 Capture/Compare Flag.**<br>This bit is set by hardware when a match or capture occurs. When the CCF0 interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by software. |

## Register 20.2. PCA0MD: PCA Mode

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CIDL | Reserved | | | CPS | | | ECF |
| Type | RW | R | | | RW | | | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xD9** | | | | | | | | |

**Table 20.4. PCA0MD Register Bit Descriptions**

| 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. |
| 6:4 | Reserved | Must write reset value. |
| 3:1 | CPS | **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: Low frequency oscillator divided by 8.<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. |

## Register 20.3. PCA0PWM: PCA PWM Configuration

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ARSEL | ECOV | COVF | Reserved | | CLSEL | | |
| Type | RW | RW | RW | R | | RW | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xF7** | | | | | | | | |

**Table 20.5. PCA0PWM Register Bit Descriptions**

| 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 to 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 to 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. |
| 4:3 | Reserved | Must write reset value. |
| 2:0 | CLSEL | **Cycle Length Select.**<br>When 16-bit PWM mode is not selected, these bits select the length of the PWM cycle. 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>000: 8 bits.<br>001: 9 bits.<br>010: 10 bits.<br>011: 11 bits.<br>100-111: Reserved. |

## Register 20.4. PCA0CLR: PCA Comparator Clear Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CPCPOL | Reserved | | | | CPCE2 | CPCE1 | CPCE0 |
| Type | RW | R | | | | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0x9C** | | | | | | | | |

**Table 20.6. PCA0CLR Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7 | CPCPOL | **Comparator Clear Polarity.**<br>Selects the polarity of the comparator result that will clear the PCA channel(s).<br>0: PCA channel(s) will be cleared when comparator result goes logic low.<br>1: PCA channel(s) will be cleared when comparator result goes logic high. |
| 6:3 | Reserved | Must write reset value. |
| 2 | CPCE2 | **Comparator Clear Enable for CEX2.**<br>Enables the comparator clear function on PCA channel 2. |
| 1 | CPCE1 | **Comparator Clear Enable for CEX1.**<br>Enables the comparator clear function on PCA channel 1. |
| 0 | CPCE0 | **Comparator Clear Enable for CEX0.**<br>Enables the comparator clear function on PCA channel 0. |

## Register 20.5. PCA0CPM0: PCA Capture/Compare Mode

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | PWM16 | ECOM | CAPP | CAPN | MAT | TOG | PWM | ECCF |
| Type | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xDA** | | | | | | | | |

**Table 20.7. PCA0CPM0 Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7 | PWM16 | **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 | ECOM | **Comparator Function Enable.**<br>This bit enables the comparator function. |
| 5 | CAPP | **Capture Positive Function Enable.**<br>This bit enables the positive edge capture capability. |
| 4 | CAPN | **Capture Negative Function Enable.**<br>This bit enables the negative edge capture capability. |
| 3 | MAT | **Match Function Enable.**<br>This bit enables the match function. When enabled, matches of the PCA counter with a module's capture/compare register cause the CCF0 bit in the PCA0MD register to be set to logic 1. |
| 2 | TOG | **Toggle Function Enable.**<br>This bit enables the toggle function. When enabled, matches of the PCA counter with the capture/compare register cause the logic level on the CEX0 pin to toggle. If the PWM bit is also set to logic 1, the module operates in Frequency Output Mode. |
| 1 | PWM | **Pulse Width Modulation Mode Enable.**<br>This bit enables the PWM function. When enabled, a pulse width modulated signal is output on the CEX0 pin. 8 to 11-bit PWM is used if PWM16 is cleared; 16-bit mode is used if PWM16 is set to logic 1. If the TOG bit is also set, the module operates in Frequency Output Mode. |
| 0 | ECCF | **Capture/Compare Flag Interrupt Enable.**<br>This bit sets the masking of the Capture/Compare Flag (CCF0) interrupt.<br>0: Disable CCF0 interrupts.<br>1: Enable a Capture/Compare Flag interrupt request when CCF0 is set. |

## Register 20.6. PCA0L: PCA Counter/Timer Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | PCA0L | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xF9** | | | | | | | | |

**Table 20.8. PCA0L Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | PCA0L | **PCA Counter/Timer Low Byte.**<br>The PCA0L register holds the low byte (LSB) of the 16-bit PCA Counter/Timer. |

## Register 20.7. PCA0H: PCA Counter/Timer High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | PCA0H | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xFA** | | | | | | | | |

**Table 20.9. PCA0H Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | PCA0H | **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. |

## Register 20.8. PCA0CPL0: PCA Capture Module Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | PCA0CPL0 | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xFB** | | | | | | | | |

**Table 20.10. PCA0CPL0 Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | PCA0CPL0 | **PCA Capture Module Low Byte.**<br>The PCA0CPL0 register holds the low byte (LSB) of the 16-bit capture module.This register address also allows access to the low byte of the corresponding PCA channels auto-reload value for 9 to 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 ECOM bit to a 0. | | |

## Register 20.9. PCA0CPH0: PCA Capture Module High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | PCA0CPH0 | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xFC** | | | | | | | | |

**Table 20.11. PCA0CPH0 Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | PCA0CPH0 | **PCA Capture Module High Byte.**<br>The PCA0CPH0 register holds the high byte (MSB) of the 16-bit capture module.This register address also allows access to the high byte of the corresponding PCA channels auto-reload value for 9 to 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 ECOM bit to a 1. | | |

## Register 20.10. PCA0POL: PCA Output Polarity

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Reserved | | | | | CEX2POL | CEX1POL | CEX0POL |
| Type | R | | | | | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0x96** | | | | | | | | |

**Table 20.12. PCA0POL Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:3 | Reserved | Must write reset value. |
| 2 | CEX2POL | **CEX2 Output Polarity.**<br>Selects the polarity of the CEX2 output channel. When this bit is modified, the change takes effect at the pin immediately.<br>0: Use default polarity.<br>1: Invert polarity. |
| 1 | CEX1POL | **CEX1 Output Polarity.**<br>Selects the polarity of the CEX1 output channel. When this bit is modified, the change takes effect at the pin immediately.<br>0: Use default polarity.<br>1: Invert polarity. |
| 0 | CEX0POL | **CEX0 Output Polarity.**<br>Selects the polarity of the CEX0 output channel. When this bit is modified, the change takes effect at the pin immediately.<br>0: Use default polarity.<br>1: Invert polarity. |

## Register 20.11. PCA0CENT: PCA Center Alignment Enable

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Reserved | | | | | CEX2CEN | CEX1CEN | CEX0CEN |
| Type | R | | | | | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0x9E** | | | | | | | | |

**Table 20.13. PCA0CENT Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:3 | Reserved | Must write reset value. |
| 2 | CEX2CEN | **CEX2 Center Alignment Enable.**<br>Selects the alignment properties of the CEX2 output channel when operated in any of the PWM modes. This bit does not affect the operation of non-PWM modes.<br>0: Edge-aligned.<br>1: Center-aligned. |
| 1 | CEX1CEN | **CEX1 Center Alignment Enable.**<br>Selects the alignment properties of the CEX1 output channel when operated in any of the PWM modes. This bit does not affect the operation of non-PWM modes.<br>0: Edge-aligned.<br>1: Center-aligned. |
| 0 | CEX0CEN | **CEX0 Center Alignment Enable.**<br>Selects the alignment properties of the CEX0 output channel when operated in any of the PWM modes. This bit does not affect the operation of non-PWM modes.<br>0: Edge-aligned.<br>1: Center-aligned. |

## Register 20.12. PCA0CPM1: PCA Capture/Compare Mode

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | PWM16 | ECOM | CAPP | CAPN | MAT | TOG | PWM | ECCF |
| Type | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xDB** | | | | | | | | |

**Table 20.14. PCA0CPM1 Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7 | PWM16 | **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 | ECOM | **Comparator Function Enable.**<br>This bit enables the comparator function. |
| 5 | CAPP | **Capture Positive Function Enable.**<br>This bit enables the positive edge capture capability. |
| 4 | CAPN | **Capture Negative Function Enable.**<br>This bit enables the negative edge capture capability. |
| 3 | MAT | **Match Function Enable.**<br>This bit enables the match function. When enabled, matches of the PCA counter with a module's capture/compare register cause the CCF1 bit in the PCA0MD register to be set to logic 1. |
| 2 | TOG | **Toggle Function Enable.**<br>This bit enables the toggle function. When enabled, matches of the PCA counter with the capture/compare register cause the logic level on the CEX1 pin to toggle. If the PWM bit is also set to logic 1, the module operates in Frequency Output Mode. |
| 1 | PWM | **Pulse Width Modulation Mode Enable.**<br>This bit enables the PWM function. When enabled, a pulse width modulated signal is output on the CEX1 pin. 8 to 11-bit PWM is used if PWM16 is cleared; 16-bit mode is used if PWM16 is set to logic 1. If the TOG bit is also set, the module operates in Frequency Output Mode. |
| 0 | ECCF | **Capture/Compare Flag Interrupt Enable.**<br>This bit sets the masking of the Capture/Compare Flag (CCF1) interrupt.<br>0: Disable CCF1 interrupts.<br>1: Enable a Capture/Compare Flag interrupt request when CCF1 is set. |

## Register 20.13. PCA0CPM2: PCA Capture/Compare Mode

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | PWM16 | ECOM | CAPP | CAPN | MAT | TOG | PWM | ECCF |
| Type | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xDC** | | | | | | | | |

**Table 20.15. PCA0CPM2 Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7 | PWM16 | **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 | ECOM | **Comparator Function Enable.**<br>This bit enables the comparator function. |
| 5 | CAPP | **Capture Positive Function Enable.**<br>This bit enables the positive edge capture capability. |
| 4 | CAPN | **Capture Negative Function Enable.**<br>This bit enables the negative edge capture capability. |
| 3 | MAT | **Match Function Enable.**<br>This bit enables the match function. When enabled, matches of the PCA counter with a module's capture/compare register cause the CCF2 bit in the PCA0MD register to be set to logic 1. |
| 2 | TOG | **Toggle Function Enable.**<br>This bit enables the toggle function. When enabled, matches of the PCA counter with the capture/compare register cause the logic level on the CEX2 pin to toggle. If the PWM bit is also set to logic 1, the module operates in Frequency Output Mode. |
| 1 | PWM | **Pulse Width Modulation Mode Enable.**<br>This bit enables the PWM function. When enabled, a pulse width modulated signal is output on the CEX2 pin. 8 to 11-bit PWM is used if PWM16 is cleared; 16-bit mode is used if PWM16 is set to logic 1. If the TOG bit is also set, the module operates in Frequency Output Mode. |
| 0 | ECCF | **Capture/Compare Flag Interrupt Enable.**<br>This bit sets the masking of the Capture/Compare Flag (CCF2) interrupt.<br>0: Disable CCF2 interrupts.<br>1: Enable a Capture/Compare Flag interrupt request when CCF2 is set. |

## Register 20.14. PCA0CPL1: PCA Capture Module Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | PCA0CPL1 | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xE9** | | | | | | | | |

**Table 20.16. PCA0CPL1 Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | PCA0CPL1 | **PCA Capture Module Low Byte.**<br>The PCA0CPL1 register holds the low byte (LSB) of the 16-bit capture module.This register address also allows access to the low byte of the corresponding PCA channels auto-reload value for 9 to 11-bit PWM mode. The ARSEL bit in register PCA0PWM controls which register is accessed. |
| **Note:** A write to this register will clear the modules ECOM bit to a 0. | | |

## Register 20.15. PCA0CPH1: PCA Capture Module High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | PCA0CPH1 | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xEA** | | | | | | | | |

**Table 20.17. PCA0CPH1 Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | PCA0CPH1 | **PCA Capture Module High Byte.**<br>The PCA0CPH1 register holds the high byte (MSB) of the 16-bit capture module.This register address also allows access to the high byte of the corresponding PCA channels auto-reload value for 9 to 11-bit PWM mode. The ARSEL bit in register PCA0PWM controls which register is accessed. |
| **Note:** A write to this register will set the modules ECOM bit to a 1. | | |

## Register 20.16. PCA0CPL2: PCA Capture Module Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | PCA0CPL2 | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xEB** | | | | | | | | |

**Table 20.18. PCA0CPL2 Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | PCA0CPL2 | **PCA Capture Module Low Byte.**<br>The PCA0CPL2 register holds the low byte (LSB) of the 16-bit capture module.This register address also allows access to the low byte of the corresponding PCA channels auto-reload value for 9 to 11-bit PWM mode. The ARSEL bit in register PCA0PWM controls which register is accessed. |
| **Note:** A write to this register will clear the modules ECOM bit to a 0. | | |

## Register 20.17. PCA0CPH2: PCA Capture Module High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | PCA0CPH2 | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xEC** | | | | | | | | |

**Table 20.19. PCA0CPH2 Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | PCA0CPH2 | **PCA Capture Module High Byte.**<br>The PCA0CPH2 register holds the high byte (MSB) of the 16-bit capture module.This register address also allows access to the high byte of the corresponding PCA channels auto-reload value for 9 to 11-bit PWM mode. The ARSEL bit in register PCA0PWM controls which register is accessed. |
| **Note:** A write to this register will set the modules ECOM bit to a 1. | | |

## 21. Port I/O (Port 0, Port 1, Port 2, Crossbar, and Port Match)

Digital and analog resources on the C8051F85x/86x family are externally available on the device's multi-purpose I/O pins. Port pins P0.0-P1.7 can be defined as general-purpose I/O (GPIO), assigned to one of the internal digital resources through the crossbar, or assigned to an analog function. Port pins P2.0 and P2.1 can be used as GPIO. Port pin P2.0 is shared with the C2 Interface Data signal (C2D). The designer has complete control over which functions are assigned, limited only by the number of physical I/O pins. This resource assignment flexibility is achieved through the use of a priority crossbar decoder. Note that the state of a port I/O pin can always be read in the corresponding port latch, regardless of the crossbar settings.

The crossbar assigns the selected internal digital resources to the I/O pins based on the Priority Decoder (Figure 21.2 and Figure 21.3). The registers XBR0, XBR1 and XBR2 are used to select internal digital functions.

The port I/O cells are configured as either push-pull or open-drain in the Port Output Mode registers (PnMDOUT, where n = 0,1). Additionally, each bank of port pins (P0, P1, and P2) has two selectable drive strength settings.

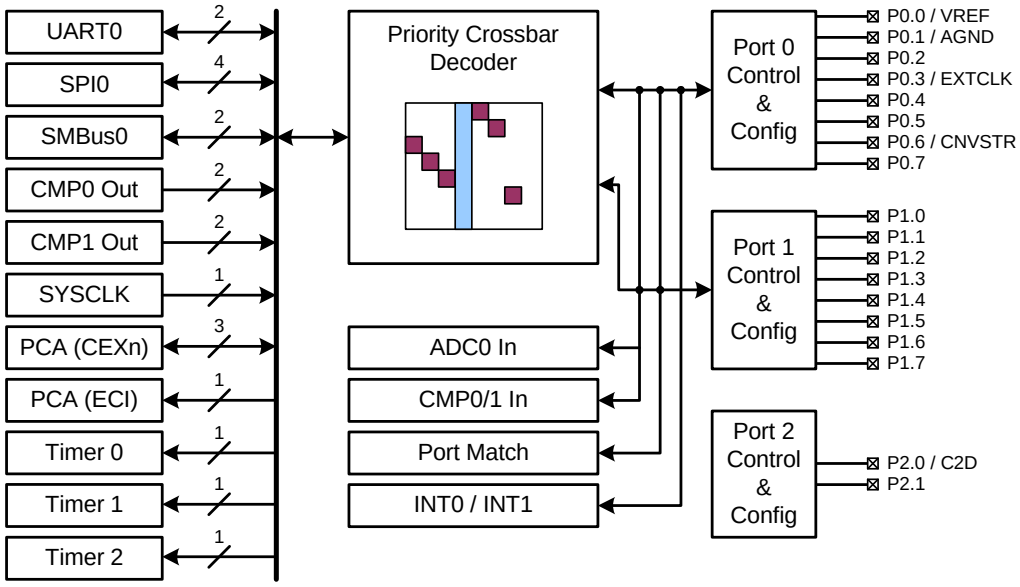



**Figure 21.1. Port I/O Functional Block Diagram**

## 21.1. General Port I/O Initialization

Port I/O initialization consists of the following steps:

1. Select the input mode (analog or digital) for all port pins, using the Port Input Mode register (PnMDIN).
2. Select the output mode (open-drain or push-pull) for all port pins, using the Port Output Mode register (PnMDOUT).
3. Select any pins to be skipped by the I/O crossbar using the Port Skip registers (PnSKIP).
4. Assign port pins to desired peripherals.
5. Enable the crossbar (XBARE = ‘1’).

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

Additionally, all analog input pins should be configured to be skipped by the crossbar (accomplished by setting the associated bits in PnSKIP). Port input mode is set in the PnMDIN register, where a ‘1’ indicates a digital input, and a ‘0’ indicates an analog input. All pins default to digital inputs on reset.

The output driver characteristics of the I/O pins are defined using the Port Output Mode registers (PnMDOUT). Each port output driver can be configured as either open drain or push-pull. This selection is required even for the digital resources selected in the XBRn registers, and is not automatic. The only exception to this is the SMBus (SDA, SCL) pins, which are configured as open-drain regardless of the PnMDOUT settings. When the WEAKPUD bit in XBR1 is ‘0’, a weak pullup is enabled for all Port I/O configured as open-drain. WEAKPUD does not affect the push-pull Port I/O. Furthermore, the weak pullup is turned off on an output that is driving a ‘0’ to avoid unnecessary power dissipation.

Registers XBR0 and XBR1 must be loaded with the appropriate values to select the digital I/O functions required by the design. Setting the XBARE bit in XBR2 to ‘1’ enables the crossbar. Until the crossbar is enabled, the external pins remain as standard port I/O (in input mode), regardless of the XBRn Register settings. For given XBRn Register settings, one can determine the I/O pin-out using the Priority Decode Table; as an alternative, Silicon Labs provides configuration utility software to determine the port I/O pin-assignments based on the crossbar register settings.

The crossbar must be enabled to use port pins as standard port I/O in output mode. Port output drivers of all crossbar pins are disabled whenever the crossbar is disabled.

## 21.2. Assigning Port I/O Pins to Analog and Digital Functions

Port I/O pins can be assigned to various analog, digital, and external interrupt functions. The port pins assigned to analog functions should be configured for analog I/O, and port pins assigned 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 require port I/O assignments. Table 21.1 shows the potential mapping of port I/O to each analog function.

**Table 21.1. Port I/O Assignment for Analog Functions**

| Analog Function | Potentially Assignable Port Pins | SFR(s) used for Assignment |
|---|---|---|
| ADC Input | P0.0 - P1.7 | ADC0MX, PnSKIP, PnMDIN |
| Comparator0 Input | P0.0 - P1.7 | CPT0MX, PnSKIP, PnMDIN |
| Comparator1 Input | P0.0 - P1.7 | CPT1MX, PnSKIP, PnMDIN |
| Voltage Reference (VREF) | P0.0 | REF0CN, PnSKIP, PnMDIN |
| Reference Ground (AGND) | P0.1 | REF0CN, PnSKIP, PnMDIN |

### 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. Table 21.2 shows all digital functions available through the crossbar and the potential mapping of port I/O to each function.

**Table 21.2. Port I/O Assignment for Digital Functions**

| Digital Function | Potentially Assignable Port Pins | SFR(s) Used for Assignment |
|---|---|---|
| UART0, SPI0, SMBus0, CP0, CP0A, CP1, CP1A, SYSCLK, PCA0 (CEX0-2 and ECI), T0, T1 or T2. | 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 pins to P0.4 and P0.5. | XBR0, XBR1, XBR2 |
| Any pin used for GPIO | P0.0 - P2.1 | P0SKIP, P1SKIP, P2SKIP |

### 21.2.3. Assigning Port I/O Pins to Fixed Digital Functions

Fixed digital functions include external clock input as well as external event trigger functions, which can be used to trigger events such as an ADC conversion, fire an interrupt or wake the device from idle mode when a transition occurs on a digital I/O pin. The fixed digital functions do not require dedicated pins and will function on both GPIO pins and pins in use by the crossbar. Fixed digital functions cannot be used on pins configured for analog I/O. Table 21.3 shows all available fixed digital functions and the potential mapping of port I/O to each function.

**Table 21.3. Port I/O Assignment for Fixed Digital Functions**

| 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 |
| Conversion Start (CNVSTR) | P0.6 | ADC0CN |
| External Clock Input (EXTCLK) | P0.3 | OSCXCN |
| Port Match | P0.0 - P1.7 | P0MASK, P0MAT<br>P1MASK, P1MAT |

## 21.3. Priority Crossbar Decoder

The priority crossbar decoder 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 P0.4 and P0.5). If a port pin is assigned, the crossbar skips that pin when assigning the next selected resource. Additionally, the crossbar will skip port pins whose associated bits in the PnSKIP registers are set. The PnSKIP registers allow software to skip port pins that are to be used for analog input, dedicated functions, or GPIO.

**Important Note on Crossbar Configuration:** If a port pin is claimed by a peripheral without use of the crossbar, its corresponding PnSKIP bit should be set. This applies to P0.0 if VREF is used, P0.1 if AGND is used, P0.3 if the EXTCLK input is enabled, P0.6 if the ADC is configured to use the external conversion start signal (CNVSTR), and any selected ADC or comparator inputs. The crossbar skips selected pins as if they were already assigned, and moves to the next unassigned pin.

Figure 21.2 shows all of the potential peripheral-to-pin assignments available to the crossbar. Note that this does not mean any peripheral can always be assigned to the highlighted pins. The actual pin assignments are determined by the priority of the enabled peripherals.

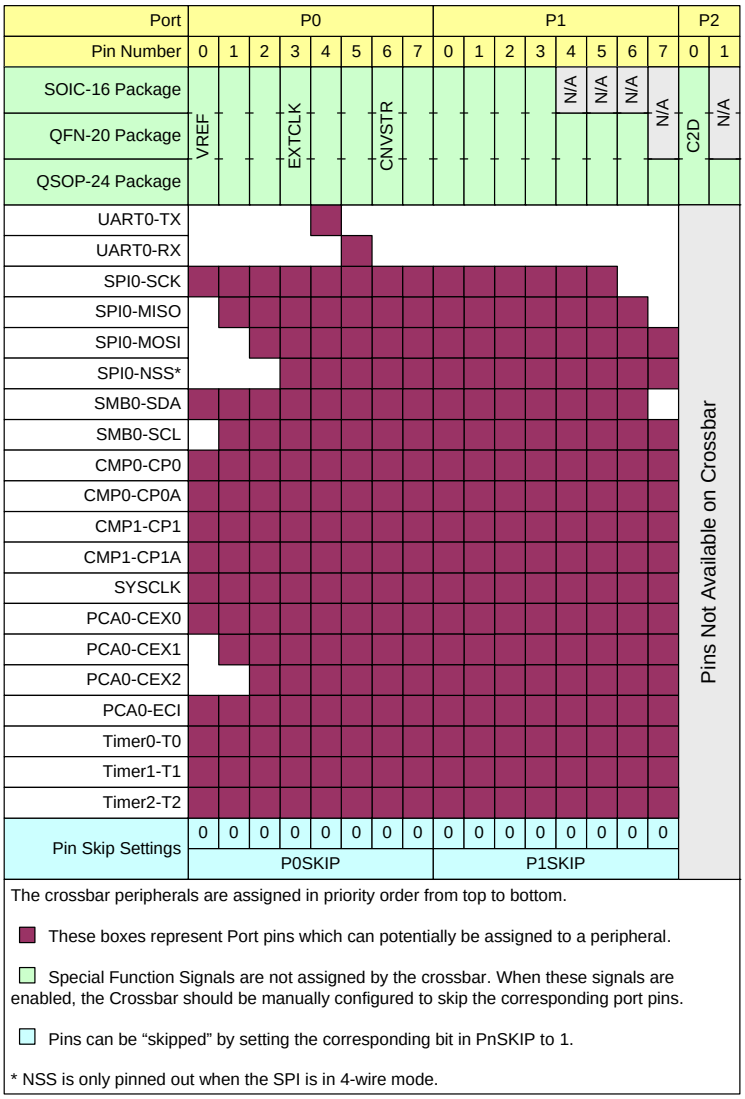



**Figure 21.2. Crossbar Priority Decoder - Possible Pin Assignments**

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 UART0 is selected, the crossbar assigns both pins associated with UART0 (TX and RX). UART0 pin assignments are fixed for bootloading purposes: UART0 TX is always assigned to P0.4; UART0 RX is always assigned to P0.5. Standard port I/Os appear contiguously after the prioritized functions have been assigned.

Figure 21.3 shows an example of the resulting pin assignments of the device with UART0 and SPI0 enabled and the EXTCLK (P0.3) pin skipped (P0SKIP = 0x08). UART0 is the highest priority and it will be assigned first. The UART0 pins can only appear on P0.4 and P0.5, so that is where it is assigned. The next-highest enabled peripheral is SPI0. P0.0, P0.1 and P0.2 are free, so SPI0 takes these three pins. The fourth pin, NSS, is routed to P0.6 because P0.3 is skipped and P0.4 and P0.5 are already occupied by the UART. The other pins on the device are available for use as general-purpose digital I/O or analog functions.

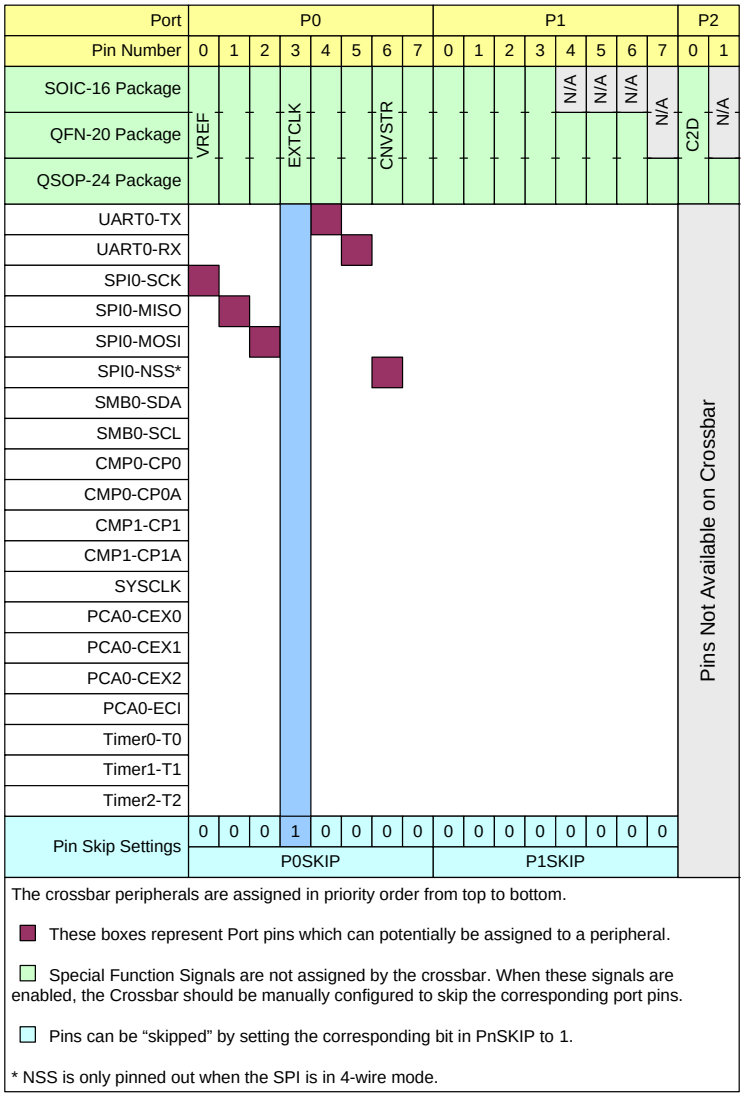



**Figure 21.3. Crossbar Priority Decoder Example**

**Note:** The SPI can be operated in either 3-wire or 4-wire modes, pending the state of the NSSMD1–NSSMD0 bits in register SPI0CN. According to the SPI mode, the NSS signal may or may not be routed to a port pin. The order in which SMBus pins are assigned is defined by the SWAP bit in the SMB0TC register.

## 21.4. Port I/O Modes of Operation

Port pins are configured by firmware as digital or analog I/O using the PnMDIN registers. On reset, all port I/O cells default to a high impedance state with weak pull-ups enabled. Until the crossbar is enabled, both the high and low port I/O drive circuits are explicitly disabled on all crossbar pins. Port pins configured as digital I/O may still be used by analog peripherals; however, this practice is not recommended and may result in measurement errors.

### 21.4.1. Configuring Port Pins For Analog Modes

Any pins to be used for analog functions should be configured for analog mode. When a pin is configured for analog I/O, its weak pullup, digital driver, and digital receiver are disabled. Port pins configured for analog functions will always read back a value of ‘0’ in the corresponding Pn Port Latch register. To configure a pin as analog, the following steps should be taken:

1. Clear the bit associated with the pin in the PnMDIN register to ‘0’. This selects analog mode for the pin.
2. Set the bit associated with the pin in the Pn register to ‘1’.
3. Skip the bit associated with the pin in the PnSKIP register to ensure the crossbar does not attempt to assign a function to the pin.

### 21.4.2. Configuring Port Pins For Digital Modes

Any pins to be used by digital peripherals 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 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 the low-side rail when the output logic value is ‘0’ and become high impedance inputs (both high 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 high-side rail to ensure the digital input is at a defined logic state. Weak pull-ups are disabled when the I/O cell is driven low to minimize power consumption, and they may be globally disabled by setting WEAKPUD to ‘1’. The user should ensure that digital I/O are always internally or externally pulled or driven to a valid logic state to minimize power consumption. 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.

**To configure a pin as digital input:**

1. Set the bit associated with the pin in the PnMDIN register to ‘1’. This selects digital mode for the pin.
2. Clear the bit associated with the pin in the PnMDOUT register to ‘0’. This configures the pin as open-drain.
3. Set the bit associated with the pin in the Pn register to ‘1’. This tells the output driver to “drive” logic high. Because the pin is configured as open-drain, the high-side driver is not active, and the pin may be used as an input.

Open-drain outputs are configured exactly as digital inputs. However, the pin may be driven low by an assigned peripheral, or by writing ‘0’ to the associated bit in the Pn register if the signal is a GPIO.

**To configure a pin as a digital, push-pull output:**

1. Set the bit associated with the pin in the PnMDIN register to ‘1’. This selects digital mode for the pin.
2. Set the bit associated with the pin in the PnMDOUT register to ‘1’. This configures the pin as push-pull.

If a digital pin is to be used as a general-purpose I/O, or with a digital function that is not part of the crossbar, the bit associated with the pin in the PnSKIP register can be set to ‘1’ to ensure the crossbar does not attempt to assign a function to the pin.

### 21.4.3. Port Drive Strength

Port drive strength can be controlled on a port-by-port basis using the PRTDRV register. Each port has a bit in PRTDRV to select the high or low drive strength setting for all pins on that port. By default, all ports are configured for high drive strength.

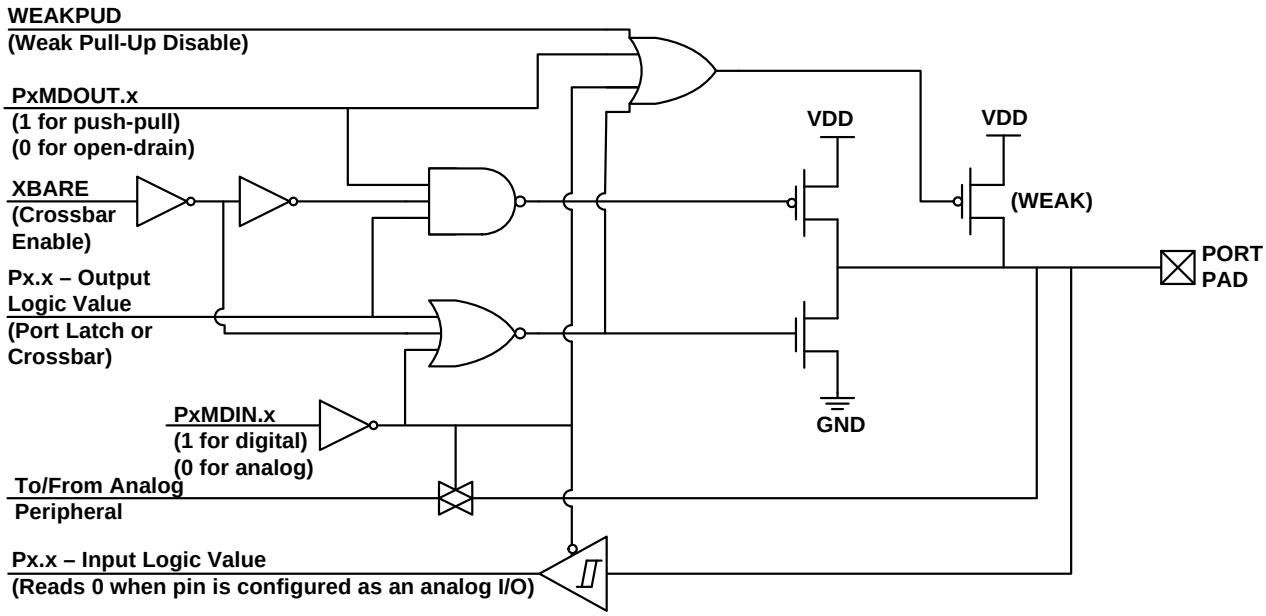



**Figure 21.4. Port I/O Cell Block Diagram**

## 21.5. Port Match

Port match functionality allows system events to be triggered by a logic value change on one or more port I/O pins. A software controlled value stored in the PnMATCH registers specifies the expected or normal logic values of the associated port pins (for example, P0MATCH.0 would correspond to P0.0). 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 the input pins regardless of the XBRn settings.

The PnMASK registers can be used to individually select which pins should be compared against the PnMATCH registers. A port mismatch event is generated if (Pn & PnMASK) does not equal (PnMATCH & PnMASK) for all ports with a PnMAT and PnMASK register.

A port mismatch event may be used to generate an interrupt or wake the device from idle mode. See the interrupts and power options chapters for more details on interrupt and wake-up sources.

## 21.6. Direct Read/Write Access to Port I/O Pins

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.

## 21.7. Port I/O and Pin Configuration Control Registers

### Register 21.1. XBR0: Port I/O Crossbar 0

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | SYSCKE | CP1AE | CP1E | CP0AE | CP0E | SMB0E | SPI0E | URT0E |
| Type | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xE1** | | | | | | | | |

**Table 21.4. XBR0 Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7 | SYSCKE | **SYSCLK Output Enable.**<br>0: SYSCLK unavailable at Port pin.<br>1: SYSCLK output routed to Port pin. |
| 6 | CP1AE | **Comparator1 Asynchronous Output Enable.**<br>0: Asynchronous CP1 unavailable at Port pin.<br>1: Asynchronous CP1 routed to Port pin. |
| 5 | CP1E | **Comparator1 Output Enable.**<br>0: CP1 unavailable at Port pin.<br>1: CP1 routed to Port pin. |
| 4 | CP0AE | **Comparator0 Asynchronous Output Enable.**<br>0: Asynchronous CP0 unavailable at Port pin.<br>1: Asynchronous CP0 routed to Port pin. |
| 3 | CP0E | **Comparator0 Output Enable.**<br>0: CP0 unavailable at Port pin.<br>1: CP0 routed to Port pin. |
| 2 | SMB0E | **SMBus0 I/O Enable.**<br>0: SMBus0 I/O unavailable at Port pins.<br>1: SMBus0 I/O routed to Port pins. |
| 1 | SPI0E | **SPI I/O Enable.**<br>0: SPI I/O unavailable at Port pins.<br>1: SPI I/O routed to Port pins. The SPI can be assigned either 3 or 4 GPIO pins. |
| 0 | URT0E | **UART I/O Output Enable.**<br>0: UART I/O unavailable at Port pin.<br>1: UART TX, RX routed to Port pins P0.4 and P0.5. |

## Register 21.2. XBR1: Port I/O Crossbar 1

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Reserved | | T2E | T1E | T0E | ECIE | PCA0ME | |
| Type | R | RW | RW | RW | RW | RW | RW | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xE2** | | | | | | | | |

**Table 21.5. XBR1 Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:6 | Reserved | Must write reset value. |
| 5 | T2E | **T2 Enable.**<br>0: T2 unavailable at Port pin.<br>1: T2 routed to Port pin. |
| 4 | T1E | **T1 Enable.**<br>0: T1 unavailable at Port pin.<br>1: T1 routed to Port pin. |
| 3 | T0E | **T0 Enable.**<br>0: T0 unavailable at Port pin.<br>1: T0 routed to Port pin. |
| 2 | ECIE | **PCA0 External Counter Input Enable.**<br>0: ECI unavailable at Port pin.<br>1: ECI routed to Port pin. |
| 1:0 | PCA0ME | **PCA Module I/O Enable Bits.**<br>00: All PCA I/O unavailable at Port pins.<br>01: CEX0 routed to Port pin.<br>10: CEX0, CEX1 routed to Port pins.<br>11: CEX0, CEX1, CEX2 routed to Port pins. |

## Register 21.3. XBR2: Port I/O Crossbar 2

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | WEAKPUD | XBARE | Reserved | | | | | |
| Type | RW | RW | R | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xE3** | | | | | | | | |

**Table 21.6. XBR2 Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7 | WEAKPUD | **Port I/O Weak Pullup Disable.**<br>0: Weak Pullups enabled (except for Ports whose I/O are configured for analog mode).<br>1: Weak Pullups disabled. |
| 6 | XBARE | **Crossbar Enable.**<br>0: Crossbar disabled.<br>1: Crossbar enabled. |
| 5:0 | Reserved | Must write reset value. |

## Register 21.4. PRTDRV: Port Drive Strength

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Reserved | | | | | P2DRV | P1DRV | P0DRV |
| Type | R | | | | | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| **SFR Address: 0xF6** | | | | | | | | |

**Table 21.7. PRTDRV Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:3 | Reserved | Must write reset value. |
| 2 | P2DRV | **Port 2 Drive Strength.**<br>0: All pins on P2 use low drive strength.<br>1: All pins on P2 use high drive strength. |
| 1 | P1DRV | **Port 1 Drive Strength.**<br>0: All pins on P1 use low drive strength.<br>1: All pins on P1 use high drive strength. |
| 0 | P0DRV | **Port 0 Drive Strength.**<br>0: All pins on P0 use low drive strength.<br>1: All pins on P0 use high drive strength. |

## Register 21.5. P0MASK: Port 0 Mask

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P0MASK | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xFE** | | | | | | | | |

**Table 21.8. P0MASK Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | P0MASK | **Port 0 Mask Value.**<br>Selects P0 pins to be compared to the corresponding bits in P0MAT.<br>0: P0.x pin logic value is ignored and will cause a port mismatch event.<br>1: P0.x pin logic value is compared to P0MAT.x. |

## Register 21.6. P0MAT: Port 0 Match

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P0MAT | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **SFR Address: 0xFD** | | | | | | | | |

**Table 21.9. P0MAT Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | P0MAT | **Port 0 Match Value.**<br>Match comparison value used on P0 pins for bits in P0MASK which are set to 1.<br>0: P0.x pin logic value is compared with logic LOW.<br>1: P0.x pin logic value is compared with logic HIGH. |

## Register 21.7. P0: Port 0 Pin Latch

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P0 | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **SFR Address: 0x80 (bit-addressable)** | | | | | | | | |

**Table 21.10. P0 Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | P0 | **Port 0 Data.**<br>Writing this register sets the port latch logic value for the associated I/O pins configured as digital I/O.<br>Reading this register returns the logic value at the pin, regardless if it is configured as output or input. |

## Register 21.8. P0MDIN: Port 0 Input Mode

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P0MDIN | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **SFR Address: 0xF1** | | | | | | | | |

**Table 21.11. P0MDIN Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | P0MDIN | **Port 0 Input Mode.**<br>Port pins configured for analog mode have their weak pullup, digital driver, and digital receiver disabled.<br>0: Corresponding P0.x pin is configured for analog mode.<br>1: Corresponding P0.x pin is configured for digital mode. |

## Register 21.9. P0MDOUT: Port 0 Output Mode

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P0MDOUT | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xA4** | | | | | | | | |

**Table 21.12. P0MDOUT Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | P0MDOUT | **Port 0 Output Mode.**<br>These bits are only applicable when the pin is configured for digital mode using the P0MDIN register.<br>0: Corresponding P0.n Output is open-drain.<br>1: Corresponding P0.n Output is push-pull. |

## Register 21.10. P0SKIP: Port 0 Skip

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P0SKIP | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xD4** | | | | | | | | |

**Table 21.13. P0SKIP Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | P0SKIP | **Port 0 Skip.**<br>These bits select port pins to be skipped by the crossbar decoder. Port pins used for analog, special functions or GPIO should be skipped.<br>0: Corresponding P0.x pin is not skipped by the crossbar.<br>1: Corresponding P0.x pin is skipped by the crossbar. |

## Register 21.11. P1MASK: Port 1 Mask

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P1MASK | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xEE** | | | | | | | | |

**Table 21.14. P1MASK Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | P1MASK | **Port 1 Mask Value.**<br>Selects P1 pins to be compared to the corresponding bits in P1MAT.<br>0: P1.x pin logic value is ignored and will cause a port mismatch event.<br>1: P1.x pin logic value is compared to P1MAT.x. |
| **Note:** Port 1 consists of 8 bits (P1.0-P1.7) on QSOP24 packages and 7 bits (P1.0-P1.6) on QFN20 packages and 4 bits (P1.0-P1.3) on SOIC16 packages. | | |

## Register 21.12. P1MAT: Port 1 Match

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P1MAT | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **SFR Address: 0xED** | | | | | | | | |

**Table 21.15. P1MAT Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | P1MAT | **Port 1 Match Value.**<br>Match comparison value used on P1 pins for bits in P1MASK which are set to 1.<br>0: P1.x pin logic value is compared with logic LOW.<br>1: P1.x pin logic value is compared with logic HIGH. |
| **Note:** Port 1 consists of 8 bits (P1.0-P1.7) on QSOP24 packages and 7 bits (P1.0-P1.6) on QFN20 packages and 4 bits (P1.0-P1.3) on SOIC16 packages. | | |

## Register 21.13. P1: Port 1 Pin Latch

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P1 | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **SFR Address: 0x90 (bit-addressable)** | | | | | | | | |

**Table 21.16. P1 Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | P1 | **Port 1 Data.**<br>Writing this register sets the port latch logic value for the associated I/O pins configured as digital I/O.<br>Reading this register returns the logic value at the pin, regardless if it is configured as output or input. |
| **Note:** Port 1 consists of 8 bits (P1.0-P1.7) on QSOP24 packages and 7 bits (P1.0-P1.6) on QFN20 packages and 4 bits (P1.0-P1.3) on SOIC16 packages. | | |

## Register 21.14. P1MDIN: Port 1 Input Mode

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P1MDIN | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **SFR Address: 0xF2** | | | | | | | | |

**Table 21.17. P1MDIN Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | P1MDIN | **Port 1 Input Mode.**<br>Port pins configured for analog mode have their weak pullup, digital driver, and digital receiver disabled.<br>0: Corresponding P1.x pin is configured for analog mode.<br>1: Corresponding P1.x pin is configured for digital mode. |
| **Note:** Port 1 consists of 8 bits (P1.0-P1.7) on QSOP24 packages and 7 bits (P1.0-P1.6) on QFN20 packages and 4 bits (P1.0-P1.3) on SOIC16 packages. | | |

## Register 21.15. P1MDOUT: Port 1 Output Mode

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P1MDOUT | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xA5** | | | | | | | | |

**Table 21.18. P1MDOUT Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | P1MDOUT | **Port 1 Output Mode.**<br>These bits are only applicable when the pin is configured for digital mode using the P1MDIN register.<br>0: Corresponding P1.n Output is open-drain.<br>1: Corresponding P1.n Output is push-pull. |
| **Note:** Port 1 consists of 8 bits (P1.0-P1.7) on QSOP24 packages and 7 bits (P1.0-P1.6) on QFN20 packages and 4 bits (P1.0-P1.3) on SOIC16 packages. | | |

## Register 21.16. P1SKIP: Port 1 Skip

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | P1SKIP | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xD5** | | | | | | | | |

**Table 21.19. P1SKIP Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | P1SKIP | **Port 1 Skip.**<br>These bits select port pins to be skipped by the crossbar decoder. Port pins used for analog, special functions or GPIO should be skipped.<br>0: Corresponding P1.x pin is not skipped by the crossbar.<br>1: Corresponding P1.x pin is skipped by the crossbar. |
| **Note:** Port 1 consists of 8 bits (P1.0-P1.7) on QSOP24 packages and 7 bits (P1.0-P1.6) on QFN20 packages and 4 bits (P1.0-P1.3) on SOIC16 packages. | | |

## Register 21.17. P2: Port 2 Pin Latch

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Reserved | | | | | | P2 | |
| Type | R | | | | | | RW | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| **SFR Address: 0xA0 (bit-addressable)** | | | | | | | | |

**Table 21.20. P2 Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:2 | Reserved | Must write reset value. |
| 1:0 | P2 | **Port 2 Data.**<br>Writing this register sets the port latch logic value for the associated I/O pins configured as digital I/O.<br>Reading this register returns the logic value at the pin, regardless if it is configured as output or input. |
| **Note:** Port 2 consists of 2 bits (P2.0-P2.1) on QSOP24 devices and 1 bit (P2.0) on QFN20 and SOIC16 packages. | | |

## Register 21.18. P2MDOUT: Port 2 Output Mode

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Reserved | | | | | | P2MDOUT | |
| Type | R | | | | | | RW | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xA6** | | | | | | | | |

**Table 21.21. P2MDOUT Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:2 | Reserved | Must write reset value. |
| 1:0 | P2MDOUT | **Port 2 Output Mode.**<br>0: Corresponding P2.n Output is open-drain.<br>1: Corresponding P2.n Output is push-pull. |
| **Note:** Port 2 consists of 2 bits (P2.0-P2.1) on QSOP24 devices and 1 bit (P2.0) on QFN20 and SOIC16 packages. | | |

## 22. Reset Sources and Supply Monitor

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

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

All SFRs are reset to the predefined values noted in the SFR detailed descriptions. The contents of internal data memory are unaffected during a reset; any previously stored data is preserved. However, since the stack pointer SFR is reset, the stack is effectively lost, even though the data on the stack is not altered.

The Port I/O latches are reset to 0xFF (all logic ones) in open-drain, low-drive mode. Weak pullups are enabled during and after the reset. For $V_{DD}$ Monitor and power-on resets, the $\overline{RST}$ pin is driven low until the device exits the reset state. Note that during a power-on event, there may be a short delay before the POR circuitry fires and the $\overline{RST}$ pin is driven low. During that time, the $\overline{RST}$ pin will be weakly pulled to the $V_{DD}$ supply pin.

On exit from the reset state, the program counter (PC) is reset, the Watchdog Timer is enabled and the system clock defaults to the internal oscillator. Program execution begins at location 0x0000.

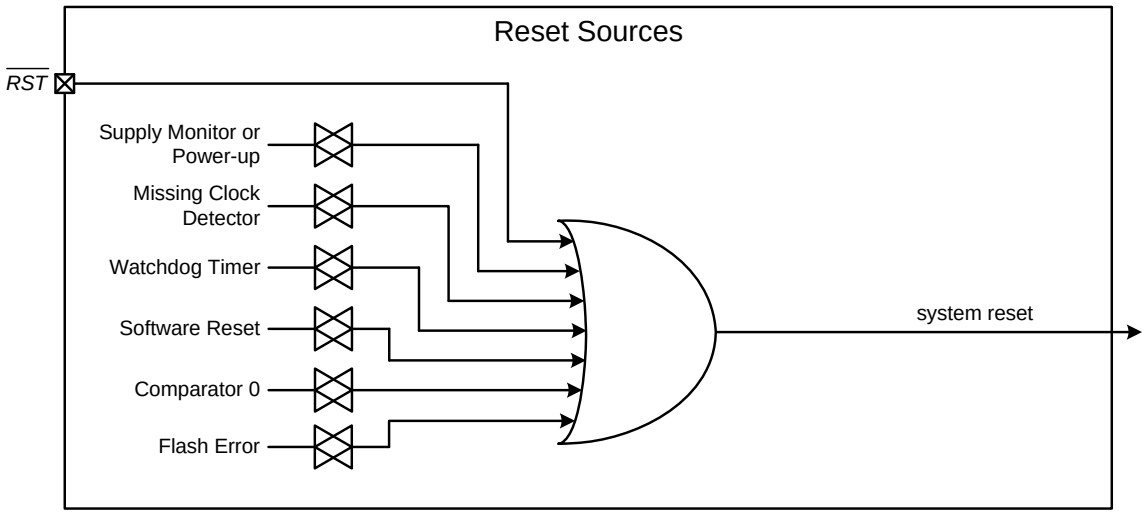



**Figure 22.1. Reset Sources**

## 22.1. Power-On Reset

During power-up, the POR circuit will fire. When POR fires, the device is held in a reset state and the $\overline{RST}$ pin is driven low until $V_{DD}$ settles above $V_{RST}$. Two delays are present during the supply ramp time. First, a delay will occur before the POR circuitry fires and pulls the $\overline{RST}$ pin low. A second delay occurs before the device is released from reset; the delay decreases as the $V_{DD}$ ramp time increases ($V_{DD}$ ramp time is defined as how fast $V_{DD}$ ramps from 0 V to $V_{RST}$). Figure 22.2. plots the power-on reset timing. For ramp times less than 1 ms, the power-on reset time ($T_{POR}$) is typically less than 0.3 ms. Additionally, the power supply must reach $V_{RST}$ before the POR circuit will release the device from reset.

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 content of internal data memory should be assumed to be undefined after a power-on reset. The $V_{DD}$ monitor is enabled following a power-on reset.

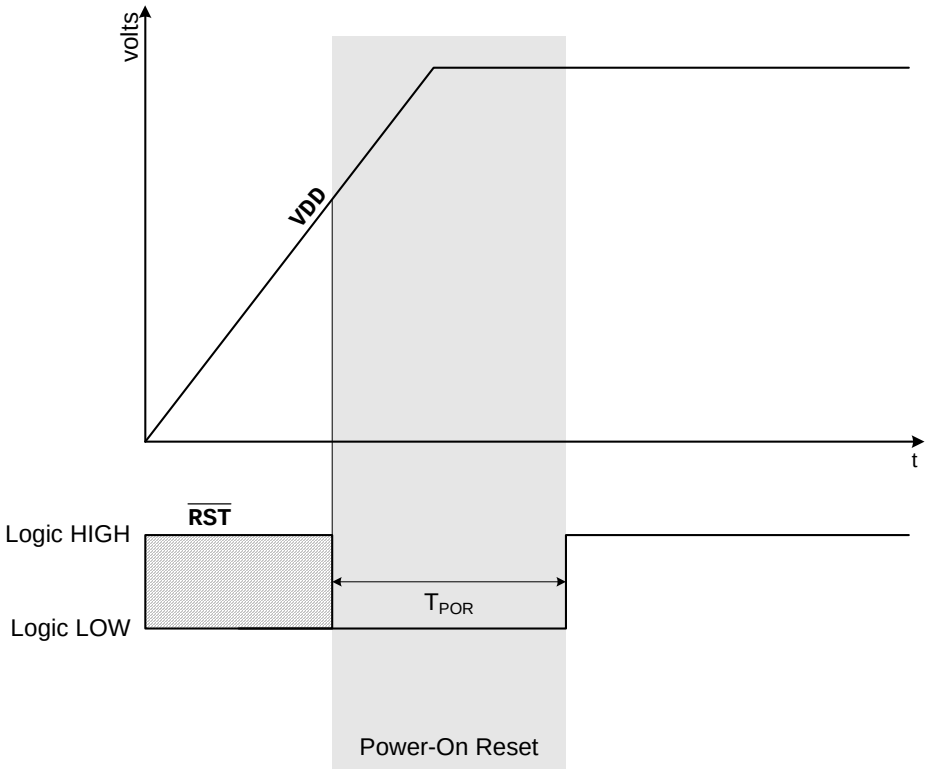



**Figure 22.2. Power-on Reset Timing**

## 22.2. Power-Fail Reset / Supply Monitor

C8051F85x/86x devices have a supply monitor that is enabled and selected as a reset source after each power-on.

The supply monitor senses the voltage on the device VDD supply and can generate a reset if the supply drops below the corresponding threshold. This monitor is enabled and enabled as a reset source after initial power-on to protect the device until VDD is an adequate and stable voltage.

When enabled and selected as a reset source, any power down transition or power irregularity that causes VDD to drop below the reset threshold will drive the $\overline{\text{RST}}$ pin low and hold the core in a reset state. When VDD returns to a level above the reset threshold, the monitor will release the core from the reset state. The reset status can then be read using the device reset sources module. After a power-fail reset, the PORF flag reads 1 and all of the other reset flags in the RSTSRC Register are indeterminate. The power-on reset delay ($t_{POR}$) is not incurred after a supply monitor reset. The contents of RAM should be presumed invalid after a VDD monitor reset.

The enable state of the VDD supply monitor and its selection as a reset source is not altered by device resets. For example, if the VDD supply monitor is de-selected as a reset source and disabled by software, and then firmware performs a software reset, the VDD supply monitor will remain disabled and de-selected after the reset.

To protect the integrity of flash contents, the VDD supply monitor must be enabled and selected as a reset source if software contains routines that erase or write flash memory. If the VDD supply monitor is not enabled, any erase or write performed on flash memory will be ignored.

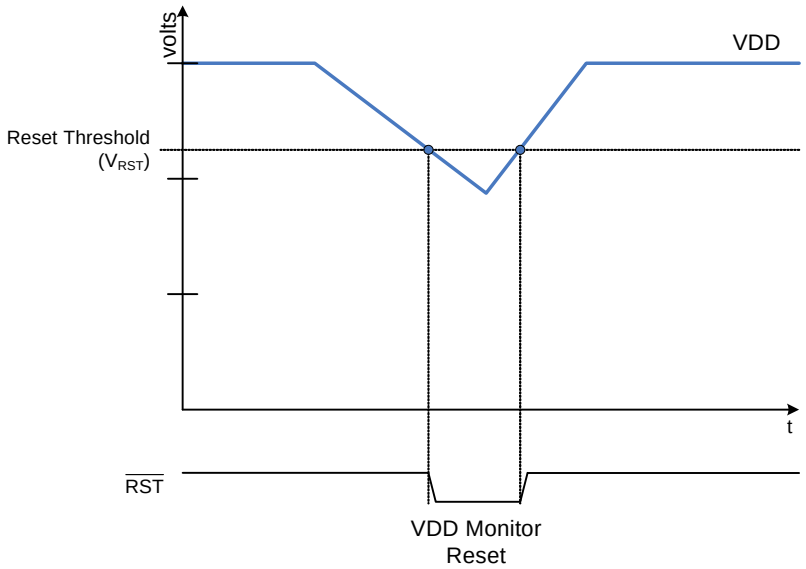



**Figure 22.3. VDD Supply Monitor Threshold**

## 22.3. Enabling the VDD Monitor

The VDD supply monitor is enabled by default. However, in systems which disable the supply monitor, it must be enabled before selecting it as a reset source. Selecting the VDD supply monitor as a reset source before it has stabilized may generate a system reset. In systems where this reset would be undesirable, a delay should be introduced between enabling the VDD supply monitor and selecting it as a reset source. No delay should be introduced in systems where software contains routines that erase or write flash memory. The procedure for enabling the VDD supply monitor and selecting it as a reset source is:

1. Enable the VDD supply monitor (VMONEN = 1).
2. Wait for the VDD supply monitor to stabilize (optional).
3. Enable the VDD monitor as a reset source in the RSTSRC register.

## 22.4. 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. The PINRSF flag is set on exit from an external reset.

## 22.5. 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 the MCD time window, the one-shot will time out and generate a reset. After a MCD reset, the MCDRSF flag will read 1, signifying the MCD as the reset source; otherwise, this bit reads 0. Writing a 1 to the MCDRSF bit enables the Missing Clock Detector; writing a 0 disables it. The state of the $\overline{\text{RST}}$ pin is unaffected by this reset.

## 22.6. Comparator0 Reset

Comparator0 can be configured as a reset source by writing a 1 to the C0RSEF flag. 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 will read 1 signifying Comparator0 as the reset source; otherwise, this bit reads ‘0’. The state of the $\overline{\text{RST}}$ pin is unaffected by this reset.

## 22.7. Watchdog Timer Reset

The programmable Watchdog Timer (WDT) can be used to prevent software from running out of control during a system malfunction. The WDT function can be enabled or disabled by software as described in the watchdog timer section. If a system malfunction prevents user software from updating the WDT, a reset is generated and the WDTRSF bit is set to ‘1’. The state of the $\overline{\text{RST}}$ pin is unaffected by this reset.

## 22.8. 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.
- A flash read is attempted above user code space.
- A program read is attempted above user code space (i.e. a branch instruction to the reserved area).
- A flash read, write or erase attempt is restricted due to a flash security setting.

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

## 22.9. Software Reset

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

## 22.10. Reset Sources Control Registers

### Register 22.1. RSTSRC: Reset Source

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Reserved | FERROR | C0RSEF | SWRSF | WDTRSF | MCDRSF | PORSF | PINRSF |
| Type | R | R | RW | RW | R | RW | RW | R |
| Reset | 0 | X | X | X | X | X | X | X |
| **SFR Address: 0xEF** | | | | | | | | |

**Table 22.1. RSTSRC Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7 | Reserved | Must write reset value. |
| 6 | FERROR | **Flash Error Reset Flag.**<br>This read-only bit is set to 1 if a flash read/write/erase error caused the last reset. |
| 5 | C0RSEF | **Comparator0 Reset Enable and Flag.**<br>Read: This bit reads 1 if Comparator0 caused the last reset.<br>Write: Writing a 1 to this bit enables Comparator0 (active-low) as a reset source. |
| 4 | SWRSF | **Software Reset Force and Flag.**<br>Read: This bit reads 1 if last reset was caused by a write to SWRSF.<br>Write: Writing a 1 to this bit forces a system reset. |
| 3 | WDTRSF | **Watchdog Timer Reset Flag.**<br>This read-only bit is set to 1 if a watchdog timer overflow caused the last reset. |
| 2 | MCDRSF | **Missing Clock Detector Enable and Flag.**<br>Read: This bit reads 1 if a missing clock detector timeout caused the last reset.<br>Write: Writing a 1 to this bit enables the missing clock detector. The MCD triggers a reset if a missing clock condition is detected. |
| 1 | PORSF | **Power-On / Supply Monitor Reset Flag, and Supply Monitor Reset Enable.**<br>Read: This bit reads 1 anytime a power-on or supply monitor reset has occurred.<br>Write: Writing a 1 to this bit enables the supply monitor as a reset source. |
| 0 | PINRSF | **HW Pin Reset Flag.**<br>This read-only bit is set to 1 if the RST pin caused the last reset. |

**Notes:**
1. Reads and writes of the RSTSRC register access different logic in the device. Reading the register always returns status information to indicate the source of the most recent reset. Writing to the register activates certain options as reset sources. It is recommended to not use any kind of read-modify-write operation on this register.
2. When the PORSF bit reads back 1 all other RSTSRC flags are indeterminate.
3. Writing 1 to the PORSF bit when the supply monitor is not enabled and stabilized may cause a system reset.

## 22.11. Supply Monitor Control Registers

### Register 22.2. VDM0CN: Supply Monitor Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | VDMEN | VDDSTAT | Reserved | | | | | |
| Type | RW | R | R | | | | | |
| Reset | X | X | X | X | X | X | X | X |
| **SFR Address: 0xFF** | | | | | | | | |

**Table 22.2. VDM0CN Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7 | VDMEN | **Supply Monitor Enable.**<br>This bit turns the supply monitor circuit on/off. The supply monitor cannot generate system resets until it is also selected as a reset source in register RSTSRC. Selecting the 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 supply monitor and selecting it as a reset source.<br>0: Supply Monitor Disabled.<br>1: Supply Monitor Enabled. |
| 6 | VDDSTAT | **Supply Status.**<br>This bit indicates the current power supply status (supply monitor output).<br>0: $V_{DD}$ is at or below the supply monitor threshold.<br>1: $V_{DD}$ is above the supply monitor threshold. |
| 5:0 | Reserved | Must write reset value. |

# 23. Serial Peripheral Interface (SPI0)

The serial peripheral interface (SPI0) provides access to a flexible, full-duplex synchronous serial bus. SPI0 can operate as a master or slave device in both 3-wire or 4-wire modes, and supports multiple masters and slaves on a single SPI bus. The slave-select (NSS) signal can be configured as an input to select SPI0 in slave mode, or to disable Master Mode operation in a multi-master environment, avoiding contention on the SPI bus when more than one master attempts simultaneous data transfers. NSS can also be configured as a chip-select output in master mode, or disabled for 3-wire operation. Additional general purpose port I/O pins can be used to select multiple slave devices in master mode.

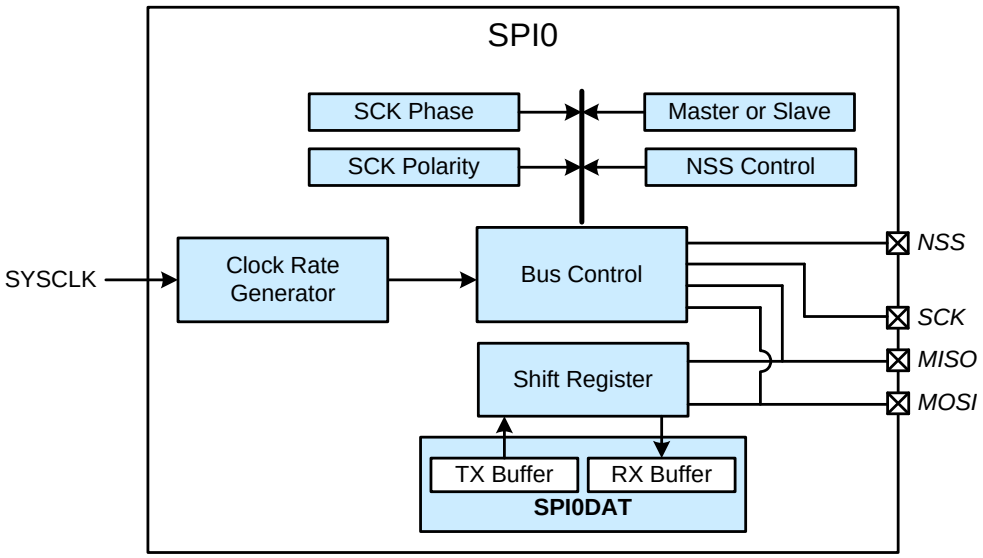



**Figure 23.1. SPI0 Block Diagram**

## 23.1. Signal Descriptions

The four signals used by SPI0 (MOSI, MISO, SCK, NSS) are described below.

### 23.1.1. Master Out, Slave In (MOSI)

The master-out, slave-in (MOSI) signal is an output from a master device and an input to slave devices. It is used to serially transfer data from the master to the slave. This signal is an output when SPI0 is operating as a master and an input when SPI0 is operating as a slave. Data is transferred most-significant bit first. When configured as a master, MOSI is driven by the MSB of the shift register in both 3- and 4-wire mode.

### 23.1.2. Master In, Slave Out (MISO)

The master-in, slave-out (MISO) signal is an output from a slave device and an input to the master device. It is used to serially transfer data from the slave to the master. This signal is an input when SPI0 is operating as a master and an output when SPI0 is operating as a slave. Data is transferred most-significant bit first. The MISO pin is placed in a high-impedance state when the SPI module is disabled and when the SPI operates in 4-wire mode as a slave that is not selected. When acting as a slave in 3-wire mode, MISO is always driven by the MSB of the shift register.

### 23.1.3. Serial Clock (SCK)

The serial clock (SCK) signal is an output from the master device and an input to slave devices. It is used to synchronize the transfer of data between the master and slave on the MOSI and MISO lines. SPI0 generates this signal when operating as a master. The SCK signal is ignored by a SPI slave when the slave is not selected (NSS = 1) in 4-wire slave mode.

### 23.1.4. Slave Select (NSS)

The function of the slave-select (NSS) signal is dependent on the setting of the NSSMD1 and NSSMD0 bits in the SPI0CN register. There are three possible modes that can be selected with these bits:

1. NSSMD[1:0] = 00: 3-Wire Master or 3-Wire Slave Mode: SPI0 operates in 3-wire mode, and NSS is disabled. When operating as a slave device, SPI0 is always selected in 3-wire mode. Since no select signal is present, SPI0 must be the only slave on the bus in 3-wire mode. This is intended for point-to-point communication between a master and one slave.
2. NSSMD[1:0] = 01: 4-Wire Slave or Multi-Master Mode: SPI0 operates in 4-wire mode, and NSS is enabled as an input. When operating as a slave, NSS selects the SPI0 device. When operating as a master, a 1-to-0 transition of the NSS signal disables the master function of SPI0 so that multiple master devices can be used on the same SPI bus.
3. NSSMD[1:0] = 1x: 4-Wire Master Mode: SPI0 operates in 4-wire mode, and NSS is enabled as an output. The setting of NSSMD0 determines what logic level the NSS pin will output. This configuration should only be used when operating SPI0 as a master device.

See Figure 23.2, Figure 23.3, and Figure 23.4 for typical connection diagrams of the various operational modes. **Note that the setting of NSSMD bits affects the pinout of the device.** When in 3-wire master or 3-wire slave mode, the NSS pin will not be mapped by the crossbar. In all other modes, the NSS signal will be mapped to a pin on the device.

## 23.2. SPI0 Master Mode Operation

A SPI master device initiates all data transfers on a SPI bus. SPI0 is placed in master mode by setting the Master Enable flag (MSTEN, SPI0CN.6). Writing a byte of data to the SPI0 data register (SPI0DAT) when in master mode writes to the transmit buffer. If the SPI shift register is empty, the byte in the transmit buffer is moved to the shift register, and a data transfer begins. The SPI0 master immediately shifts out the data serially on the MOSI line while providing the serial clock on SCK. The SPIF (SPI0CN.7) flag is set to logic 1 at the end of the transfer. If interrupts are enabled, an interrupt request is generated when the SPIF flag is set. While the SPI0 master transfers data to a slave on the MOSI line, the addressed SPI slave device simultaneously transfers 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 23.2 shows a connection diagram between two master devices and a single slave in multiple-master mode.

3-wire single-master mode is active when NSSMD1 (SPI0CN.3) = 0 and NSSMD0 (SPI0CN.2) = 0. In this mode, NSS is not used, and is not mapped to an external port pin through the crossbar. Any slave devices that must be addressed in this mode should be selected using general-purpose I/O pins. Figure 23.3 shows a connection diagram between a master device in 3-wire master mode and a slave device.

4-wire single-master mode is active when NSSMD1 (SPI0CN.3) = 1. In this mode, NSS is configured as an output pin, and can be used as a slave-select signal for a single SPI device. In this mode, the output value of NSS is controlled (in software) with the bit NSSMD0 (SPI0CN.2). Additional slave devices can be addressed using general-purpose I/O pins. Figure 23.4 shows a connection diagram for a master device and a slave device in 4-wire mode.

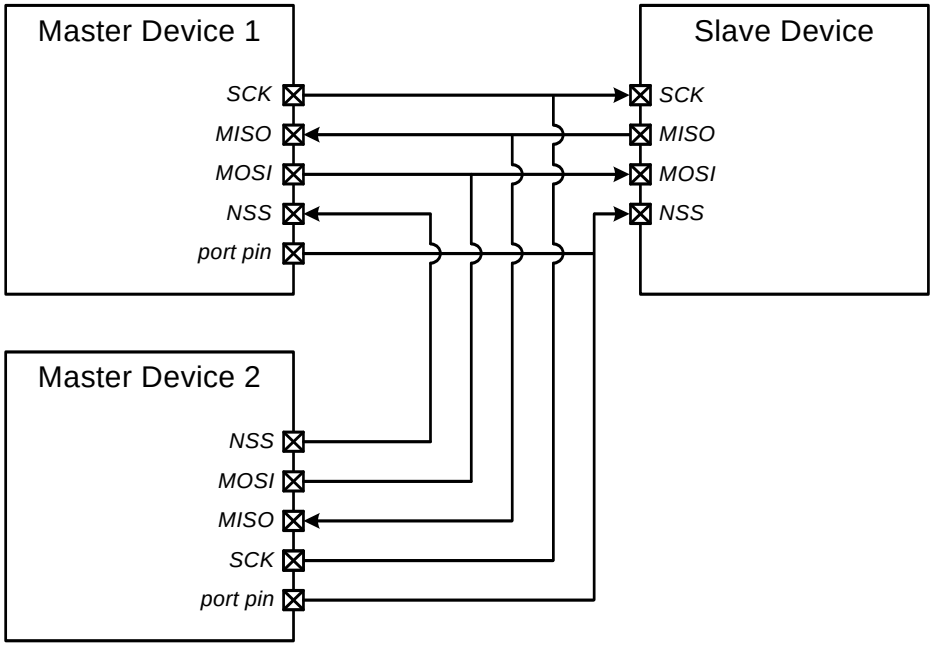



**Figure 23.2. Multiple-Master Mode Connection Diagram**

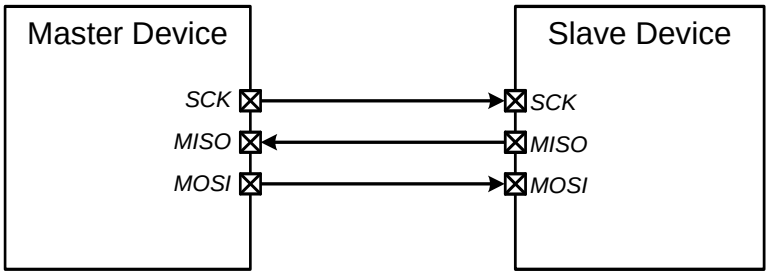



**Figure 23.3. 3-Wire Single Master and 3-Wire Single Slave Mode Connection Diagram**

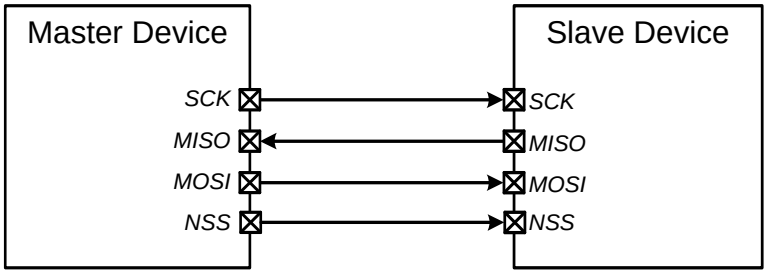



**Figure 23.4. 4-Wire Single Master Mode and 4-Wire Slave Mode Connection Diagram**

## 23.3. SPI0 Slave Mode Operation

When SPI0 is enabled and not configured as a master, it will operate as a SPI slave. As a slave, bytes are shifted in through the MOSI pin and out through the MISO pin by a master device controlling the SCK signal. A bit counter in the SPI0 logic counts SCK edges. When 8 bits have been shifted 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 23.4 shows a connection diagram between two slave devices in 4-wire slave mode and a master device.

The 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 23.3 shows a connection diagram between a slave device in 3-wire slave mode and a master device.

## 23.4. SPI0 Interrupt Sources

When SPI0 interrupts are enabled, the following four flags will generate an interrupt when they are set to logic 1:

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.

## 23.5. 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 23.5. For slave mode, the clock and data relationships are shown in Figure 23.6 and Figure 23.7. 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) controls the master mode serial clock frequency. This register is ignored when operating in slave mode. When the SPI is configured as a master, the maximum data transfer rate (bits/sec) is one-half the system clock frequency or 12.5 MHz, whichever is slower. When the SPI is configured as a slave, the maximum data transfer rate (bits/sec) for full-duplex operation is 1/10 the system clock frequency, provided that the master issues SCK, NSS (in 4-wire slave mode), and the serial input data synchronously with the slave's system clock. If the master issues SCK, NSS, and the serial input data asynchronously, the maximum data transfer rate (bits/sec) must be less than 1/10 the system clock frequency. In the special case where the master only wants to transmit data to the slave and does not need to receive data from the slave (i.e. half-duplex operation), the SPI slave can receive data at a maximum data transfer rate (bits/sec) of 1/4 the system clock frequency. This is provided that the master issues SCK, NSS, and the serial input data synchronously with the slave's system clock.

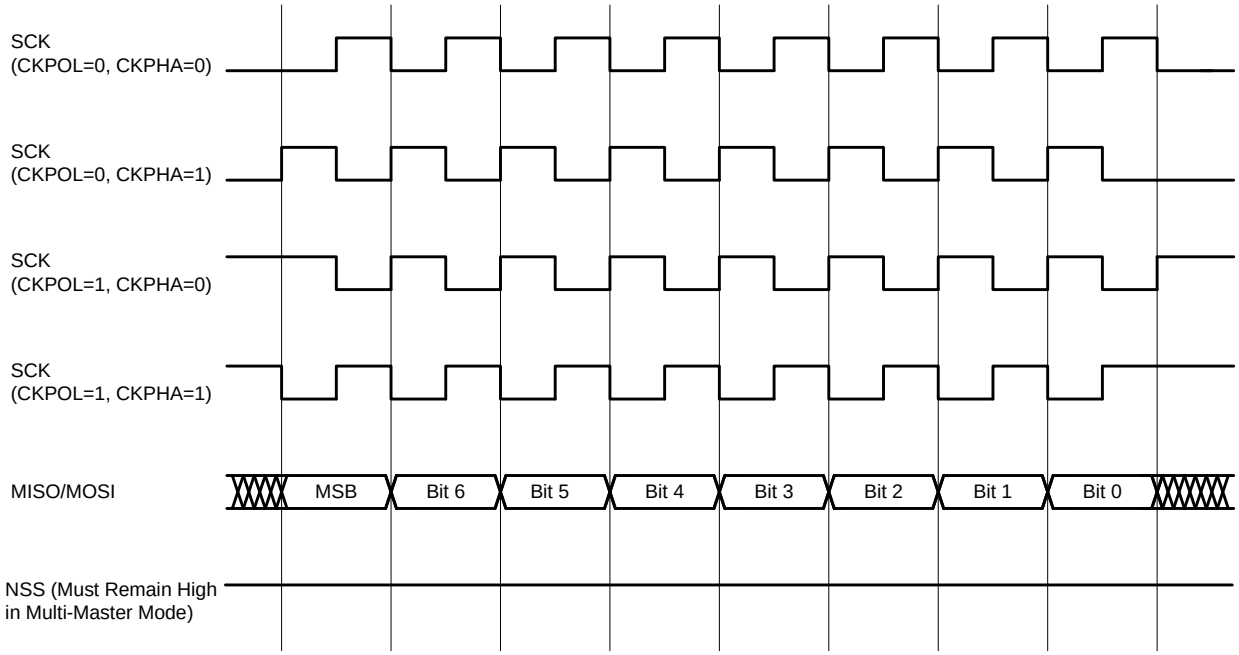



**Figure 23.5. Master Mode Data/Clock Timing**

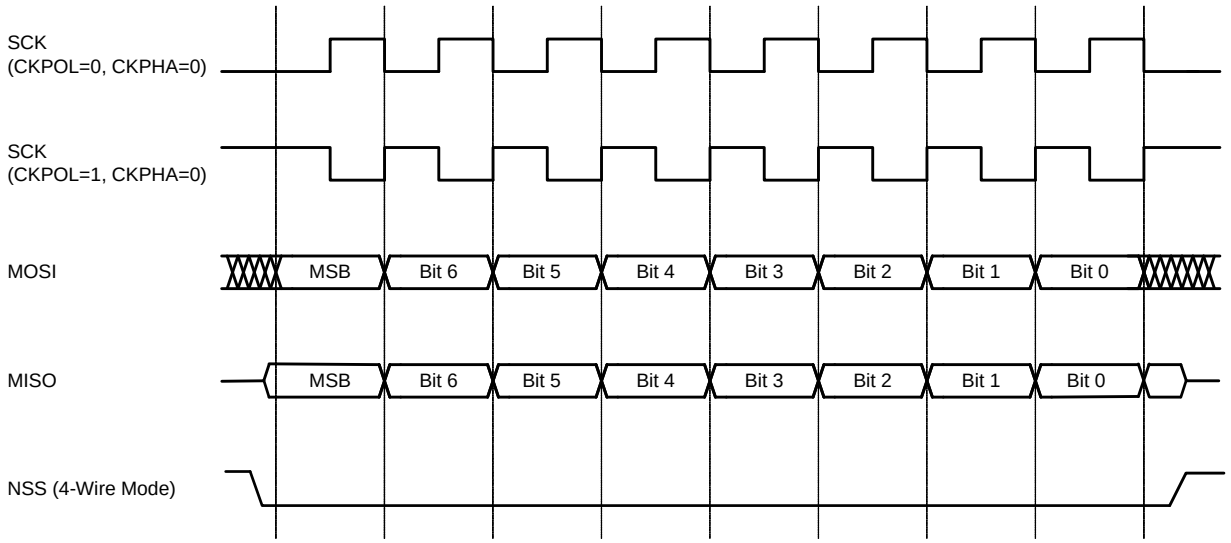



**Figure 23.6. Slave Mode Data/Clock Timing (CKPHA = 0)**

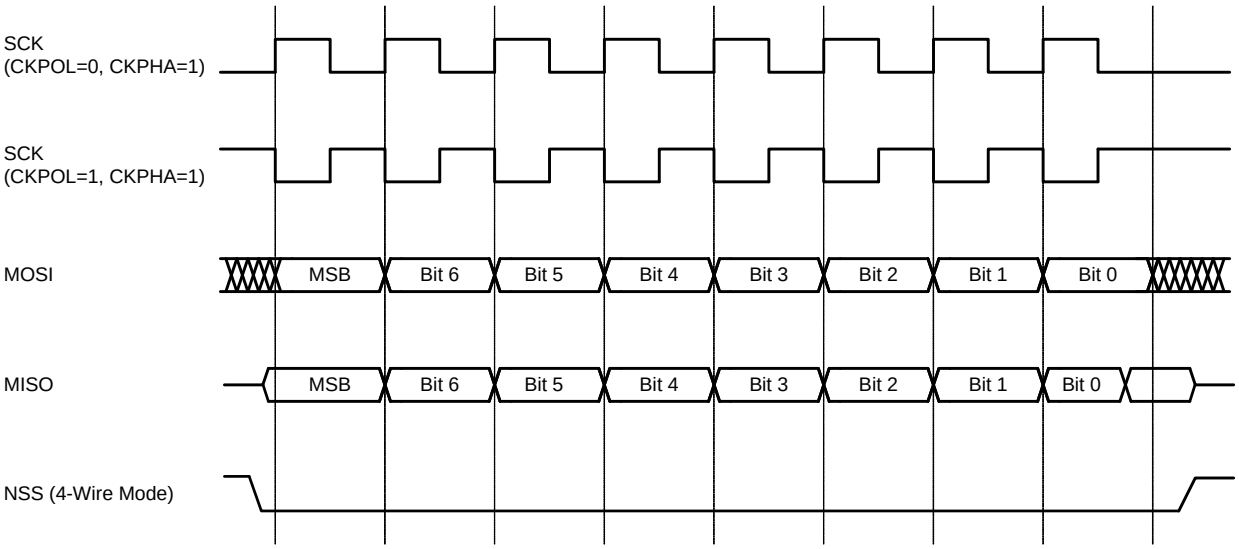
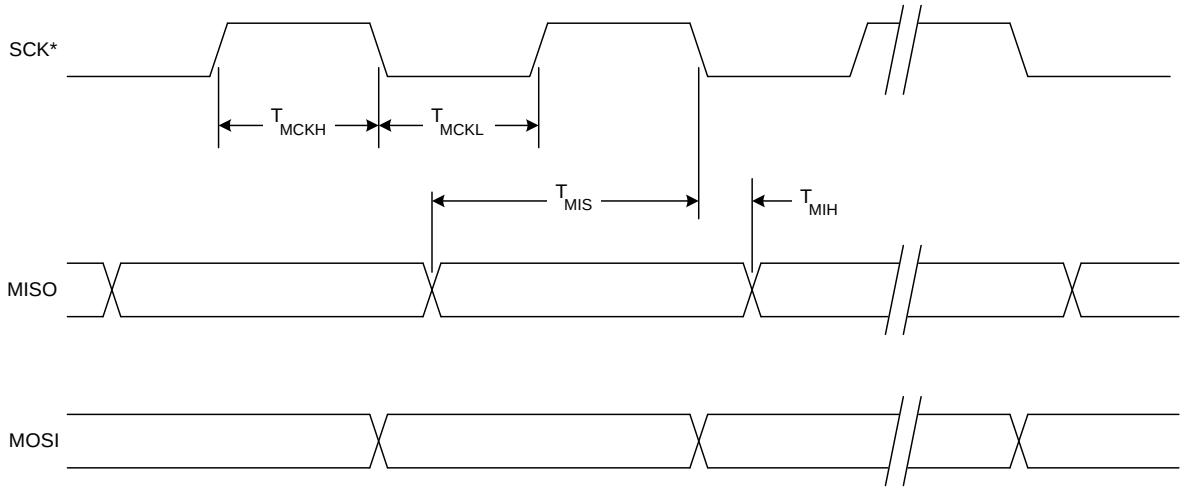



**Figure 23.7. Slave Mode Data/Clock Timing (CKPHA = 1)**

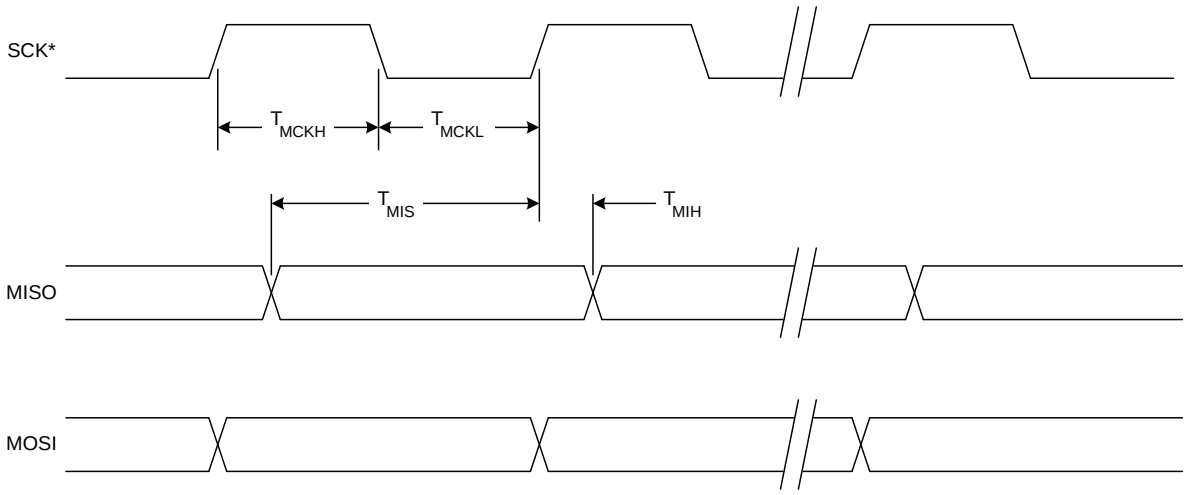
## 23.6. SPI Special Function Registers

SPI0 is accessed and controlled through four special function registers in the system controller: SPI0CN Control Register, SPI0DAT Data Register, SPI0CFG Configuration Register, and SPI0CKR Clock Rate Register. The four special function registers related to the operation of the SPI0 Bus are described in the following figures.



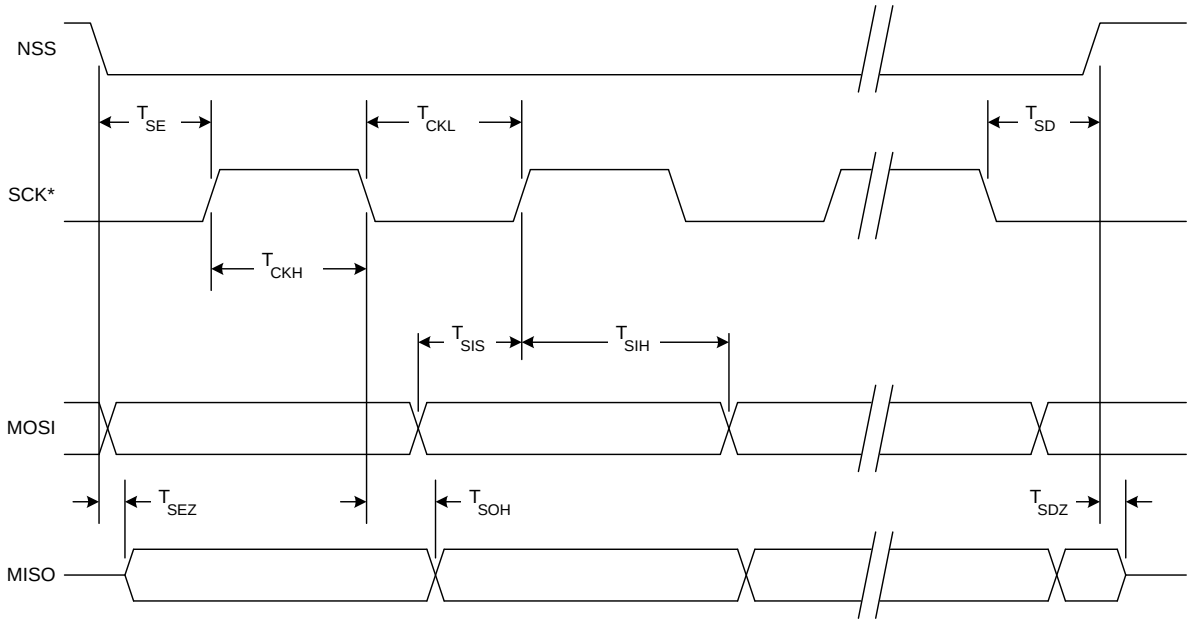

* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

**Figure 23.8. SPI Master Timing (CKPHA = 0)**



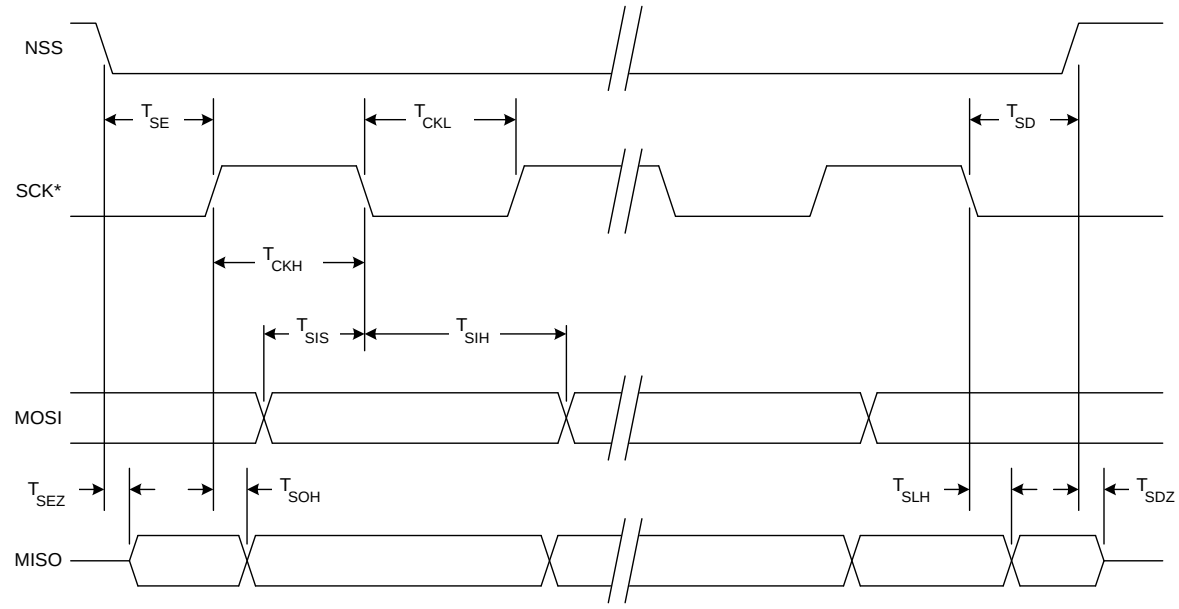

* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

**Figure 23.9. SPI Master Timing (CKPHA = 1)**




* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

**Figure 23.10. SPI Slave Timing (CKPHA = 0)**




* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

**Figure 23.11. SPI Slave Timing (CKPHA = 1)**

**Table 23.1. SPI Slave Timing Parameters**

| Parameter | Description | Min | Max | Units |
|---|---|---|---|---|
| **Master Mode Timing** (See Figure 23.8 and Figure 23.9) | | | | |
| $T_{MCKH}$ | 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 23.10 and Figure 23.11) | | | | |
| $T_{SE}$ | 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). | | | | |

## 23.7. SPI Control Registers

### Register 23.1. SPI0CFG: SPI0 Configuration

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | SPIBSY | MSTEN | CKPHA | CKPOL | SLVSEL | NSSIN | SRMT | RXBMT |
| Type | R | RW | RW | RW | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| **SFR Address: 0xA1** | | | | | | | | |

**Table 23.2. SPI0CFG Register Bit Descriptions**

| 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. |
| **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. | | |

**Table 23.2. SPI0CFG Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 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. | | |

## Register 23.2. SPI0CN: SPI0 Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | SPIF | WCOL | MODF | RXOVRN | NSSMD | | TXBMT | SPIEN |
| Type | RW | RW | RW | RW | RW | | R | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| SFR Address: 0xF8 (bit-addressable) | | | | | | | | |

**Table 23.3. SPI0CN Register Bit Descriptions**

| 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 = 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 | **Slave Select Mode.**<br>Selects between the following NSS operation modes:<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>10: 4-Wire Single-Master Mode. NSS is an output and logic low.<br>11: 4-Wire Single-Master Mode. NSS is an output and logic high. |
| 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. |

**Table 23.3. SPI0CN Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 0 | SPIEN | **SPI0 Enable.**<br>0: SPI disabled.<br>1: SPI enabled. |

## Register 23.3. SPI0CKR: SPI0 Clock Rate

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | SPI0CKR | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xA2** | | | | | | | | |

**Table 23.4. SPI0CKR Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | SPI0CKR | **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>$f_{SCK} = \frac{SYSCLK}{2 \times (SPI0CKR + 1)}$<br>for 0 <= SPI0CKR <= 255 |

## Register 23.4. SPI0DAT: SPI0 Data

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | SPI0DAT | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xA3** | | | | | | | | |

**Table 23.5. SPI0DAT Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | SPI0DAT | **SPI0 Transmit and Receive Data.**<br>The SPI0DAT register is used to transmit and receive SPI0 data. Writing data to SPI0-DAT 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. |

## 24. System Management Bus / I2C (SMBus0)

The SMBus I/O interface is a two-wire, bi-directional serial bus. The SMBus is compliant with the System Management Bus Specification, version 1.1, and compatible with the I2C serial bus.

Reads and writes to the SMBus 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 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 peripherals 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 SMBus0 peripheral is shown in Figure 24.1.

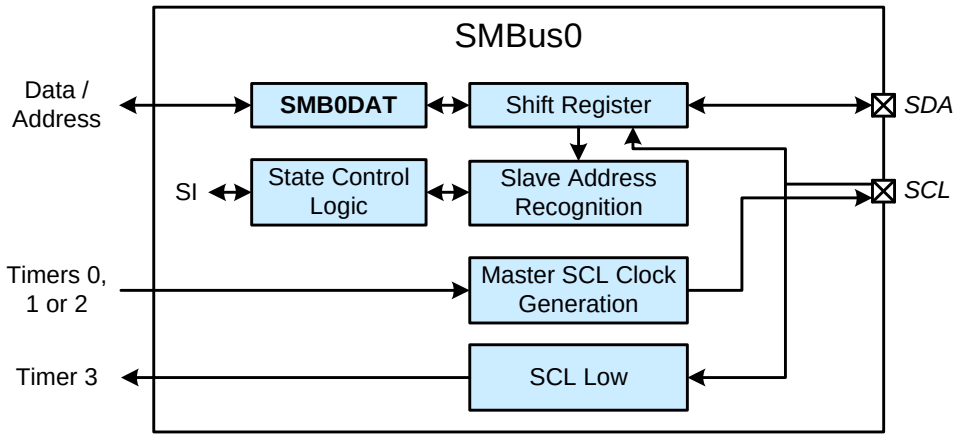



**Figure 24.1. SMBus0 Block Diagram**

## 24.1. Supporting Documents

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

1. The $I^2C$-Bus and How to Use It (including specifications), Philips Semiconductor.
2. The $I^2C$-Bus Specification—Version 2.0, Philips Semiconductor.
3. System Management Bus Specification—Version 1.1, SBS Implementers Forum.

## 24.2. SMBus Configuration

Figure 24.2 shows a typical SMBus configuration. The SMBus specification allows any recessive voltage between 3.0 V and 5.0 V; different devices on the bus may operate at different voltage levels. However, the maximum voltage on any port pin must conform to the electrical characteristics specifications. The bi-directional SCL (serial clock) and SDA (serial data) lines must be connected to a positive power supply voltage through a pullup resistor or similar circuit. Every device connected to the bus must have an open-drain or open-collector output for both the SCL and SDA lines, so that both are pulled high (recessive state) when the bus is free. The maximum number of devices on the bus is limited only by the requirement that the rise and fall times on the bus not exceed 300 ns and 1000 ns, respectively.

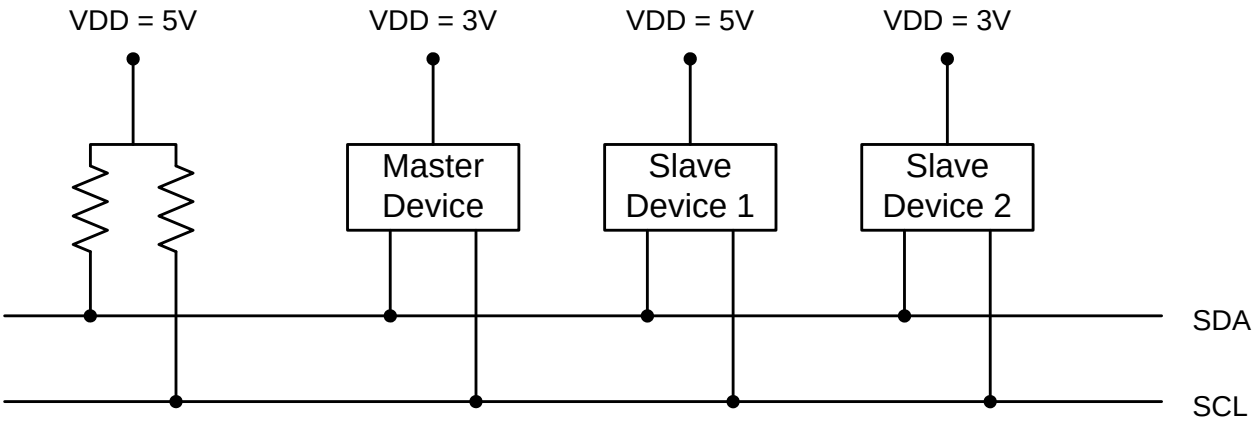



**Figure 24.2. Typical SMBus Configuration**

## 24.3. SMBus Operation

Two types of data transfers are possible: data transfers from a master transmitter to an addressed slave receiver (WRITE), and data transfers from an addressed slave transmitter to a master receiver (READ). The master device initiates both types of data transfers and provides the serial clock pulses on SCL. The SMBus interface may operate as a master or a slave, and multiple master devices on the same bus are supported. If two or more masters attempt to initiate a data transfer simultaneously, an arbitration scheme is employed with a single master always winning the arbitration. It is not necessary to specify one device as the Master in a system; any device who transmits a START and a slave address becomes the master for the duration of that transfer.

A typical SMBus transaction consists of a START condition followed by an address byte (Bits7–1: 7-bit slave address; Bit0: R/W direction bit), one or more bytes of data, and a STOP condition. Bytes that are received (by a master or slave) are acknowledged (ACK) with a low SDA during a high SCL (see Figure 24.3). If the receiving device does not ACK, the transmitting device will read a NACK (not acknowledge), which is a high SDA during a high SCL.

The direction bit (R/W) occupies the least-significant bit position of the address byte. The direction bit is set to logic 1 to indicate a "READ" operation and cleared to logic 0 to indicate a "WRITE" operation.

All transactions are initiated by a master, with one or more addressed slave devices as the target. The master generates the START condition and then transmits the slave address and direction bit. If the transaction is a WRITE operation from the master to the slave, the master transmits the data a byte at a time waiting for an ACK from the slave at the end of each byte. For READ operations, the slave transmits the data waiting for an ACK from the master at the end of each byte. At the end of the data transfer, the master generates a STOP condition to terminate the transaction and free the bus. Figure 24.3 illustrates a typical SMBus transaction.

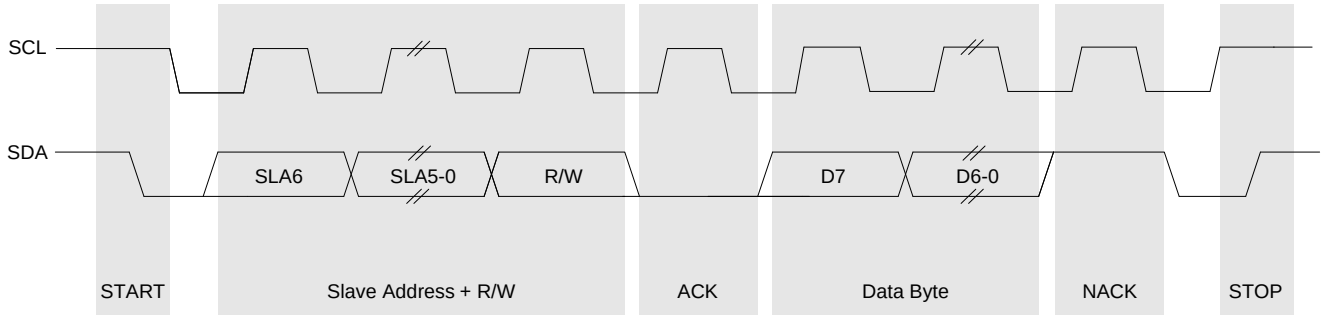



**Figure 24.3. SMBus Transaction**

### 24.3.1. Transmitter vs. Receiver

On the SMBus communications interface, a device is the “transmitter” when it is sending an address or data byte to another device on the bus. A device is a “receiver” when an address or data byte is being sent to it from another device on the bus. The transmitter controls the SDA line during the address or data byte. After each byte of address or data information is sent by the transmitter, the receiver sends an ACK or NACK bit during the ACK phase of the transfer, during which time the receiver controls the SDA line.

### 24.3.2. Arbitration

A master may start a transfer only if the bus is free. The bus is free after a STOP condition or after the SCL and SDA lines remain high for a specified time (see Section “24.3.5. SCL High (SMBus Free) Timeout” on page 236). In the event that two or more devices attempt to begin a transfer at the same time, an arbitration scheme is employed to force one master to give up the bus. The master devices continue transmitting until one attempts a HIGH while the other transmits a LOW. Since the bus is open-drain, the bus will be pulled LOW. The master attempting the HIGH will detect a LOW SDA and lose the arbitration. The winning master continues its transmission without interruption; the losing master becomes a slave and receives the rest of the transfer if addressed. This arbitration scheme is non-destructive: one device always wins, and no data is lost.

### 24.3.3. Clock Low Extension

SMBus provides a clock synchronization mechanism, similar to I2C, which allows devices with different speed capabilities to coexist on the bus. A clock-low extension is used during a transfer in order to allow slower slave devices to communicate with faster masters. The slave may temporarily hold the SCL line LOW to extend the clock low period, effectively decreasing the serial clock frequency.

### 24.3.4. SCL Low Timeout

If the SCL line is held low by a slave device on the bus, no further communication is possible. Furthermore, the master cannot force the SCL line high to correct the error condition. To solve this problem, the SMBus protocol specifies that devices participating in a transfer must detect any clock cycle held low longer than 25 ms as a “timeout” condition. Devices that have detected the timeout condition must reset the communication no later than 10 ms after detecting the timeout condition.

For the SMBus0 interface, Timer 3 is used to implement SCL low timeouts. The SCL low timeout feature is enabled by setting the SMB0TOE bit in SMB0CF. The associated timer is forced to reload when SCL is high, and allowed to count when SCL is low. With the associated timer enabled and configured to overflow after 25 ms (and SMB0TOE set), the timer interrupt service routine can be used to reset (disable and re-enable) the SMBus in the event of an SCL low timeout.

#### 24.3.5. SCL High (SMBus Free) Timeout

The SMBus specification stipulates that if the SCL and SDA lines remain high for more that 50 µs, the bus is designated as free. When the SMB0FTE 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. A clock source is required for free timeout detection, even in a slave-only implementation.

## 24.4. Using the SMBus

The SMBus can operate in both Master and Slave modes. The interface provides timing and shifting control for serial transfers; higher level protocol is determined by user software. The SMBus interface provides the following application-independent features:

- Byte-wise serial data transfers
- Clock signal generation on SCL (Master Mode only) and SDA data synchronization
- Timeout/bus error recognition, as defined by the SMB0CF configuration register
- START/STOP timing, detection, and generation
- Bus arbitration
- Interrupt generation
- Status information
- Optional hardware recognition of slave address and automatic acknowledgement of address/data

SMBus interrupts are generated for each data byte or slave address that is transferred. When hardware acknowledgement is disabled, the point at which the interrupt is generated depends on whether the hardware is acting as a data transmitter or receiver. When a transmitter (i.e., sending address/data, receiving an ACK), this interrupt is generated after the ACK cycle so that software may read the received ACK value; when receiving data (i.e., receiving address/data, sending an ACK), this interrupt is generated before the ACK cycle so that software may define the outgoing ACK value. If hardware acknowledgement is enabled, these interrupts are always generated after the ACK cycle. See Section 24.5 for more details on transmission sequences.

Interrupts are also generated to indicate the beginning of a transfer when a master (START generated), or the end of a transfer when a slave (STOP detected). Software should read the SMB0CN (SMBus Control register) to find the cause of the SMBus interrupt. Table 24.5 provides a quick SMB0CN decoding reference.

#### 24.4.1. SMBus Configuration Register

The SMBus Configuration register (SMB0CF) is used to enable the SMBus Master and/or Slave modes, select the SMBus clock source, and select the SMBus timing and timeout options. When the ENSMB bit is set, the SMBus is enabled for all master and slave events. Slave events may be disabled by setting the INH bit. With slave events inhibited, the SMBus interface will still monitor the SCL and SDA pins; however, the interface will NACK all received addresses and will not generate any slave interrupts. When the INH bit is set, all slave events will be inhibited following the next START (interrupts will continue for the duration of the current transfer).

**Table 24.1. SMBus Clock Source Selection**

| SMBCS | SMBus0 Clock Source |
|---|---|
| 00 | Timer 0 Overflow |
| 01 | Timer 1 Overflow |
| 10 | Timer 2 High Byte Overflow |
| 11 | Timer 2 Low Byte Overflow |

The SMBCS bit field selects the SMBus clock source, which is used only when operating as a master or when the Free Timeout detection is enabled. When operating as a master, overflows from the selected source determine the absolute minimum SCL low and high times as defined in Equation 24.1.The selected clock source may be shared by other peripherals so long as the timer is left running at all times.

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

**Equation 24.1. Minimum SCL High and Low Times**

The selected clock source should be configured to establish the minimum SCL High and Low times as per Equation 24.1. When the interface is operating as a master (and SCL is not driven or extended by any other devices on the bus), the typical SMBus bit rate is approximated by Equation 24.2.

$$BitRate = \frac{f_{ClockSourceOverflow}}{3}$$

**Equation 24.2. Typical SMBus Bit Rate**

Figure 24.4 shows the typical SCL generation described by Equation 24.2. Notice that $T_{HIGH}$ is typically twice as large as $T_{LOW}$. The actual SCL output may vary due to other devices on the bus (SCL may be extended low by slower slave devices, or driven low by contending master devices). The bit rate when operating as a master will never exceed the limits defined by equation Equation 24.1.

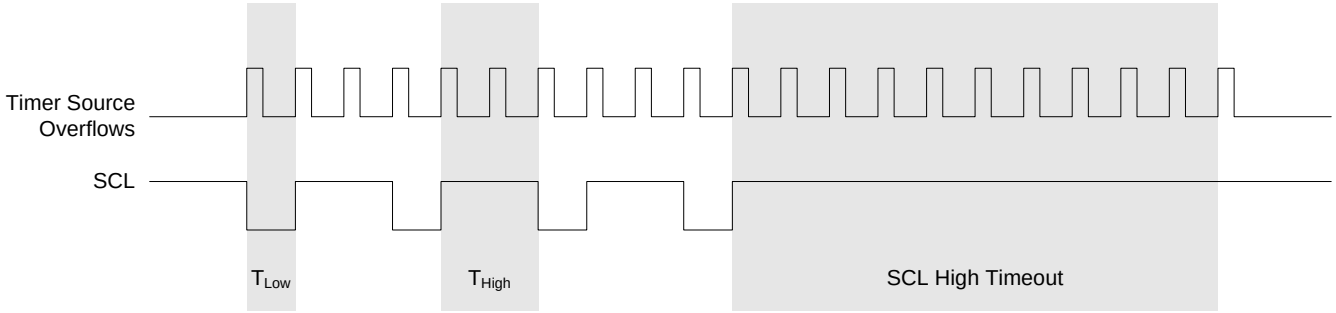



**Figure 24.4. Typical SMBus SCL Generation**

Setting the EXTHOLD bit extends the minimum setup and hold times for the SDA line. The minimum SDA setup time defines the absolute minimum time that SDA is stable before SCL transitions from low-to-high. The minimum SDA hold time defines the absolute minimum time that the current SDA value remains stable after SCL transitions from high-to-low. EXTHOLD should be set so that the minimum setup and hold times meet the SMBus Specification requirements of 250 ns and 300 ns, respectively. Table 24.2 shows the minimum setup and hold times for the two EXTHOLD settings. Setup and hold time extensions are typically necessary for SMBus compliance when SYSCLK is above 10 MHz.

**Table 24.2. Minimum SDA Setup and Hold Times**

| EXTHOLD | Minimum SDA Setup Time | Minimum SDA Hold Time |
|---|---|---|
| 0 | $T_{low}$ – 4 system clocks<br>or<br>1 system clock + s/w delay* | 3 system clocks |
| 1 | 11 system clocks | 12 system clocks |

**Note:** Setup Time for ACK bit transmissions and the MSB of all data transfers. When using software acknowledgment, the s/w delay occurs between the time SMB0DAT or ACK is written and when SI0 is cleared. Note that if SI is cleared in the same write that defines the outgoing ACK value, s/w delay is zero.

With the SMBTOE bit set, Timer 3 should be configured to overflow after 25 ms in order to detect SCL low timeouts (see Section “24.3.4. SCL Low Timeout” on page 235). The SMBus interface will force the associated timer to reload while SCL is high, and allow the timer to count when SCL is low. The timer interrupt service routine should be used to reset SMBus communication by disabling and re-enabling the SMBus.

SMBus Free Timeout detection can be enabled by setting the SMBFTE bit. When this bit is set, the bus will be considered free if SDA and SCL remain high for more than 10 SMBus clock source periods (see Figure 24.4).

### 24.4.2. SMBus Pin Swap

The SMBus peripheral is assigned to pins using the priority crossbar decoder. By default, the SMBus signals are assigned to port pins starting with SDA on the lower-numbered pin, and SCL on the next available pin. The SWAP bit in the SMBTC register can be set to 1 to reverse the order in which the SMBus signals are assigned.

### 24.4.3. SMBus Timing Control

The SDD field in the SMBTC register is used to restrict the detection of a START condition under certain circumstances. In some systems where there is significant mismatch between the impedance or the capacitance on the SDA and SCL lines, it may be possible for SCL to fall after SDA during an address or data transfer. Such an event can cause a false START detection on the bus. These kind of events are not expected in a standard SMBus or I2C-compliant system. **In most systems this parameter should not be adjusted, and it is recommended that it be left at its default value.**

By default, if the SCL falling edge is detected after the falling edge of SDA (i.e. one SYSCLK cycle or more), the device will detect this as a START condition. The SDD field is used to increase the amount of hold time that is required between SDA and SCL falling before a START is recognized. An additional 2, 4, or 8 SYSCLKs can be added to prevent false START detection in systems where the bus conditions warrant this.

### 24.4.4. SMB0CN Control Register

SMB0CN is used to control the interface and to provide status information. The higher four bits of SMB0CN (MASTER, TXMODE, STA, and STO) form a status vector that can be used to jump to service routines. MASTER indicates whether a device is the master or slave during the current transfer. TXMODE indicates whether the device is transmitting or receiving data for the current byte.

STA and STO indicate that a START and/or STOP has been detected or generated since the last SMBus interrupt. STA and STO are also used to generate START and STOP conditions when operating as a master. Writing a 1 to STA will cause the SMBus interface to enter Master Mode and generate a START when the bus becomes free (STA is not cleared by hardware after the START is generated). Writing a 1 to STO while in Master Mode will cause the interface to generate a STOP and end the current transfer after the next ACK cycle. If STO and STA are both set (while in Master Mode), a STOP followed by a START will be generated.

The ARBLOST bit indicates that the interface has lost an arbitration. This may occur anytime the interface is transmitting (master or slave). A lost arbitration while operating as a slave indicates a bus error condition. ARBLOST is cleared by hardware each time SI is cleared.

The SI bit (SMBus Interrupt Flag) is set at the beginning and end of each transfer, after each byte frame, or when an arbitration is lost; see Table 24.3 for more details.

**Important Note About the SI Bit:** The SMBus interface is stalled while SI is set; thus SCL is held low, and the bus is stalled until software clears SI.

#### 24.4.4.1. Software ACK Generation

When the EHACK bit in register SMB0ADM is cleared to 0, the firmware on the device must detect incoming slave addresses and ACK or NACK the slave address and incoming data bytes. As a receiver, writing the ACK bit defines the outgoing ACK value; as a transmitter, reading the ACK bit indicates the value received during the last ACK cycle. ACKRQ is set each time a byte is received, indicating that an outgoing ACK value is needed. When ACKRQ is set, software should write the desired outgoing value to the ACK bit before clearing SI. A NACK will be generated if software does not write the ACK bit before clearing SI. SDA will reflect the defined ACK value immediately following a write to the ACK bit; however SCL will remain low until SI is cleared. If a received slave address is not acknowledged, further slave events will be ignored until the next START is detected.

#### 24.4.4.2. Hardware ACK Generation

When the EHACK bit in register SMB0ADM is set to 1, automatic slave address recognition and ACK generation is enabled. More detail about automatic slave address recognition can be found in Section 24.4.5. As a receiver, the value currently specified by the ACK bit will be automatically sent on the bus during the ACK cycle of an incoming data byte. As a transmitter, reading the ACK bit indicates the value received on the last ACK cycle. The ACKRQ bit is not used when hardware ACK generation is enabled. If a received slave address is NACKed by hardware, further slave events will be ignored until the next START is detected, and no interrupt will be generated.

Table 24.3 lists all sources for hardware changes to the SMB0CN bits. Refer to Table 24.5 for SMBus status decoding using the SMB0CN register.

**Table 24.3. Sources for Hardware Changes to SMB0CN**

| Bit | Set by Hardware When: | Cleared by Hardware When: |
|---|---|---|
| MASTER | ■ A START is generated. | ■ A STOP is generated.<br>■ Arbitration is lost. |
| TXMODE | ■ START is generated.<br>■ SMB0DAT is written before the start of an SMBus frame. | ■ A START is detected.<br>■ Arbitration is lost.<br>■ SMB0DAT is not written before the start of an SMBus frame. |

**Table 24.3. Sources for Hardware Changes to SMB0CN (Continued)**

| Bit | Set by Hardware When: | Cleared by Hardware When: |
|---|---|---|
| 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 SIn is cleared. |
| ACK | ■ The incoming ACK value is low (ACKNOWLEDGE). | ■ The incoming ACK value is high (NOT ACKNOWLEDGE). |
| SI | ■ A START has been generated.<br>■ Lost arbitration.<br>■ A byte has been transmitted and an ACK/NACK received.<br>■ A byte has been received.<br>■ A START or repeated START followed by a slave address + R/W has been received.<br>■ A STOP has been received. | ■ Must be cleared by software. |

### 24.4.5. Hardware Slave Address Recognition

The SMBus hardware has the capability to automatically recognize incoming slave addresses and send an ACK without software intervention. Automatic slave address recognition is enabled by setting the EHACK bit in register SMB0ADM to 1. This will enable both automatic slave address recognition and automatic hardware ACK generation for received bytes (as a master or slave). More detail on automatic hardware ACK generation can be found in Section 24.4.4.2.

The registers used to define which address(es) are recognized by the hardware are the SMBus Slave Address register and the SMBus Slave Address Mask register. 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 a bit of the slave address mask SLVM enables a comparison between the received slave address and the hardware's slave address SLV for that bit. A 0 in a bit of the slave address mask means that bit will be treated as a "don't care" for comparison purposes. In this case, either a 1 or a 0 value are acceptable on the incoming slave address. Additionally, if the GC bit in register SMB0ADR is set to 1, hardware will recognize the General Call Address (0x00). Table 24.4 shows some example parameter settings and the slave addresses that will be recognized by hardware under those conditions.

**Table 24.4. Hardware Address Recognition Examples (EHACK = 1)**

| Hardware Slave Address SLV | Slave Address Mask SLVM | 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 |

### 24.4.6. 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.

## 24.5. SMBus Transfer Modes

The SMBus interface may be configured to operate as master and/or slave. At any particular time, it will be operating in one of the following four modes: Master Transmitter, Master Receiver, Slave Transmitter, or Slave Receiver. The SMBus interface enters Master Mode any time a START is generated, and remains in Master Mode until it loses an arbitration or generates a STOP. An SMBus interrupt is generated at the end of all SMBus byte frames. The position of the ACK interrupt when operating as a receiver depends on whether hardware ACK generation is enabled. As a receiver, the interrupt for an ACK occurs **before** the ACK with hardware ACK generation disabled, and **after** the ACK when hardware ACK generation is enabled. As a transmitter, interrupts occur **after** the ACK, regardless of whether hardware ACK generation is enabled or not.

### 24.5.1. Write Sequence (Master)

During a write sequence, an SMBus master writes data to a slave device. The master in this transfer will be a transmitter during the address byte, and a transmitter during all data bytes. The SMBus interface generates the START condition and transmits the first byte containing the address of the target slave and the data direction bit. In this case the data direction bit (R/W) will be logic 0 (WRITE). The master then transmits one or more bytes of serial data. After each byte is transmitted, an acknowledge bit is generated by the slave. The transfer is ended when the STO bit is set and a STOP is generated. The interface will switch to Master Receiver Mode if SMB0DAT is not written following a Master Transmitter interrupt. Figure 24.5 shows a typical master write sequence. Two transmit data bytes are shown, though any number of bytes may be transmitted. Notice that all of the “data byte transferred” interrupts occur **after** the ACK cycle in this mode, regardless of whether hardware ACK generation is enabled.

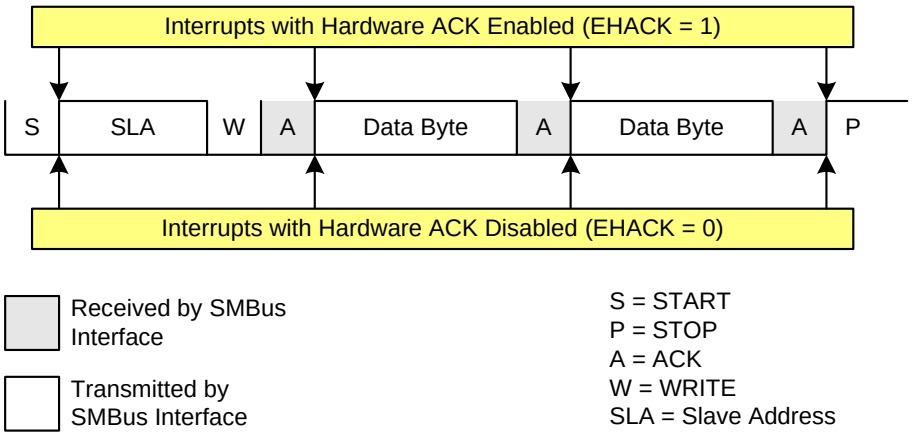



**Figure 24.5. Typical Master Write Sequence**

### 24.5.2. Read Sequence (Master)

During a read sequence, an SMBus master reads data from a slave device. The master in this transfer will be a transmitter during the address byte, and a receiver during all data bytes. The SMBus interface generates the START condition and transmits the first byte containing the address of the target slave and the data direction bit. In this case the data direction bit (R/W) will be logic 1 (READ). Serial data is then received from the slave on SDA while the SMBus outputs the serial clock. The slave transmits one or more bytes of serial data.

If hardware ACK generation is disabled, the ACKRQ is set to 1 and an interrupt is generated after each received byte. Software must write the ACK bit at that time to ACK or NACK the received byte.

With hardware ACK generation enabled, the SMBus hardware will automatically generate the ACK/NACK, and then post the interrupt. **It is important to note that the appropriate ACK or NACK value should be set up by the software prior to receiving the byte when hardware ACK generation is enabled.**

Writing a 1 to the ACK bit generates an ACK; writing a 0 generates a NACK. Software should write a 0 to the ACK bit for the last data transfer, to transmit a NACK. The interface exits Master Receiver Mode after the STO bit is set and a STOP is generated. The interface will switch to Master Transmitter Mode if SMB0DAT is written while an active Master Receiver. Figure 24.6 shows a typical master read sequence. Two received data bytes are shown, though any number of bytes may be received. Notice that the ‘data byte transferred’ interrupts occur at different places in the sequence, depending on whether hardware ACK generation is enabled. The interrupt occurs **before** the ACK with hardware ACK generation disabled, and **after** the ACK when hardware ACK generation is enabled.

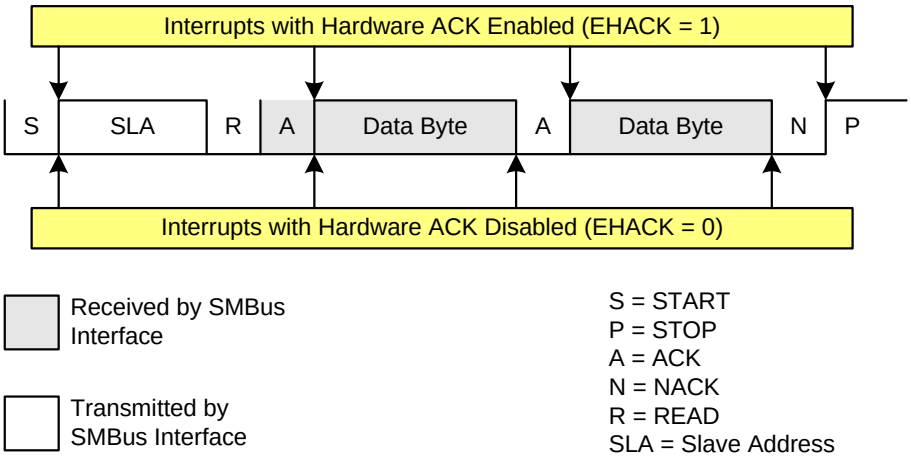



**Figure 24.6. Typical Master Read Sequence**

### 24.5.3. Write Sequence (Slave)

During a write sequence, an SMBus master writes data to a slave device. The slave in this transfer will be a receiver during the address byte, and a receiver during all data bytes. When slave events are enabled (INH = 0), the interface enters Slave Receiver Mode when a START followed by a slave address and direction bit (WRITE in this case) is received. If hardware ACK generation is disabled, upon entering Slave Receiver Mode, an interrupt is generated and the ACKRQ bit is set. The software must respond to the received slave address with an ACK, or ignore the received slave address with a NACK. If hardware ACK generation is enabled, the hardware will apply the ACK for a slave address which matches the criteria set up by SMB0ADR and SMB0ADM. The interrupt will occur after the ACK cycle.

If the received slave address is ignored (by software or hardware), slave interrupts will be inhibited until the next START is detected. If the received slave address is acknowledged, zero or more data bytes are received.

If hardware ACK generation is disabled, the ACKRQ is set to 1 and an interrupt is generated after each received byte. Software must write the ACK bit at that time to ACK or NACK the received byte.

With hardware ACK generation enabled, the SMBus hardware will automatically generate the ACK/NACK, and then post the interrupt. **It is important to note that the appropriate ACK or NACK value should be set up by the software prior to receiving the byte when hardware ACK generation is enabled.**

The interface exits Slave Receiver Mode after receiving a STOP. The interface will switch to Slave Transmitter Mode if SMB0DAT is written while an active Slave Receiver. Figure 24.7 shows a typical slave write sequence. Two received data bytes are shown, though any number of bytes may be received. Notice that the ‘data byte transferred’ interrupts occur at different places in the sequence, depending on whether hardware ACK generation is enabled. The interrupt occurs **before** the ACK with hardware ACK generation disabled, and **after** the ACK when hardware ACK generation is enabled.

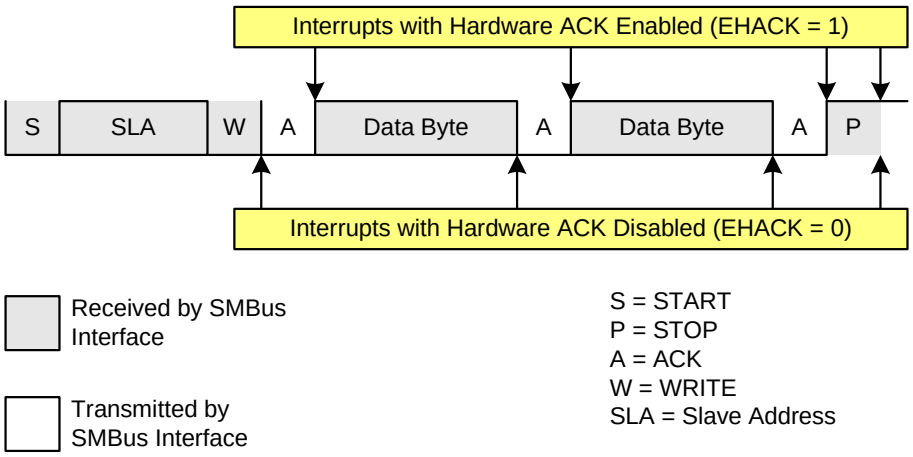



**Figure 24.7. Typical Slave Write Sequence**

### 24.5.4. Read Sequence (Slave)

During a read sequence, an SMBus master reads data from a slave device. The slave in this transfer will be a receiver during the address byte, and a transmitter during all data bytes. When slave events are enabled (INH = 0), the interface enters Slave Receiver Mode (to receive the slave address) when a START followed by a slave address and direction bit (READ in this case) is received. If hardware ACK generation is disabled, upon entering Slave Receiver Mode, an interrupt is generated and the ACKRQ bit is set. The software must respond to the received slave address with an ACK, or ignore the received slave address with a NACK. If hardware ACK generation is enabled, the hardware will apply the ACK for a slave address which matches the criteria set up by SMB0ADR and SMB0ADM. The interrupt will occur after the ACK cycle.

If the received slave address is ignored (by software or hardware), slave interrupts will be inhibited until the next START is detected. If the received slave address is acknowledged, zero or more data bytes are transmitted. If the received slave address is acknowledged, data should be written to SMB0DAT to be transmitted. The interface enters slave transmitter mode, and transmits one or more bytes of data. After each byte is transmitted, the master sends an acknowledge bit; if the acknowledge bit is an ACK, SMB0DAT should be written with the next data byte. If the acknowledge bit is a NACK, SMB0DAT should not be written to before SI is cleared (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. The interface will switch to slave receiver mode if SMB0DAT is not written following a Slave Transmitter interrupt. Figure 24.8 shows a typical slave read sequence. Two transmitted data bytes are shown, though any number of bytes may be transmitted. Notice that all of the “data byte transferred” interrupts occur **after** the ACK cycle in this mode, regardless of whether hardware ACK generation is enabled.

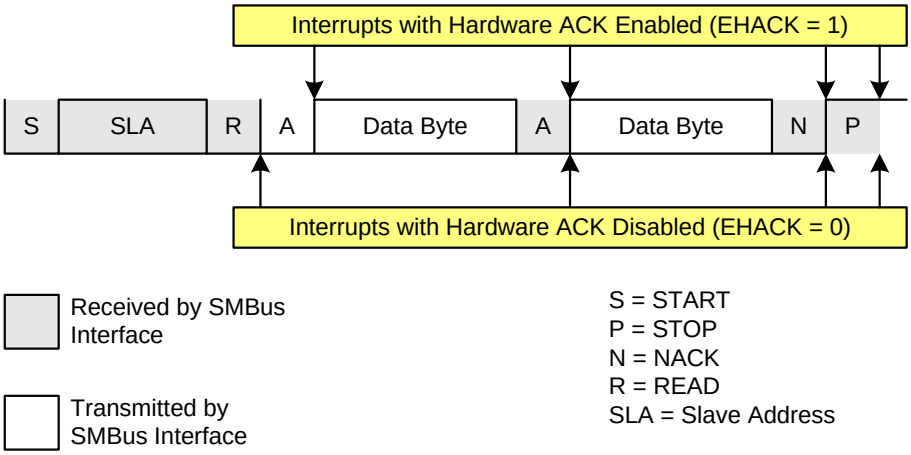



**Figure 24.8. Typical Slave Read Sequence**

## 24.6. SMBus Status Decoding

The current SMBus status can be easily decoded using the SMB0CN register. The appropriate actions to take in response to an SMBus event depend on whether hardware slave address recognition and ACK generation is enabled or disabled. Table 24.5 describes the typical actions when hardware slave address recognition and ACK generation is disabled. Table 24.6 describes the typical actions when hardware slave address recognition and ACK generation is enabled. In the tables, STATUS VECTOR refers to the four upper bits of SMB0CN: MASTER, TXMODE, STA, and STO. The shown response options are only the typical responses; application-specific procedures are allowed as long as they conform to the SMBus specification. Highlighted responses are allowed by hardware but do not conform to the SMBus specification.

**Table 24.5. SMBus Status Decoding: Hardware ACK 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 SMB0-DAT. | 0 | 0 | X | 1100 |
| | | | | | | End transfer with STOP. | 0 | 1 | X | — |
| | | | | | | End transfer with STOP and start another transfer. | 1 | 1 | X | — |
| | | | | | | Send repeated START. | 1 | 0 | X | 1110 |
| | | | | | | Switch to Master Receiver Mode (clear SI without writing new data to SMB0DAT). | 0 | 0 | X | 1000 |
| Master Receiver | 1000 | 1 | 0 | X | A master data byte was received; ACK requested. | Acknowledge received byte; Read SMB0DAT. | 0 | 0 | 1 | 1000 |
| | | | | | | Send NACK to indicate last byte, and send STOP. | 0 | 1 | 0 | — |
| | | | | | | Send NACK to indicate last byte, and send STOP followed by START. | 1 | 1 | 0 | 1110 |
| | | | | | | Send ACK followed by repeated START. | 1 | 0 | 1 | 1110 |
| | | | | | | Send NACK to indicate last byte, and send repeated START. | 1 | 0 | 0 | 1110 |
| | | | | | | Send ACK and switch to Master Transmitter Mode (write to SMB0DAT before clearing SI). | 0 | 0 | 1 | 1100 |
| | | | | | | Send NACK and switch to Master Transmitter Mode (write to SMB0DAT before clearing SI). | 0 | 0 | 0 | 1100 |

**Table 24.5. SMBus Status Decoding: Hardware ACK 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 | — |

**Table 24.5. SMBus Status Decoding: Hardware ACK 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 | |
| Bus Error Condition | 0010 | 0 | 1 | X | Lost arbitration while attempting a repeated START. | Abort failed transfer. | 0 | 0 | X | — |
| | | | | | | Reschedule failed transfer. | 1 | 0 | X | 1110 |
| | 0001 | 0 | 1 | X | Lost arbitration due to a detected STOP. | Abort failed transfer. | 0 | 0 | X | — |
| | | | | | | Reschedule failed transfer. | 1 | 0 | X | 1110 |
| | 0000 | 1 | 1 | X | Lost arbitration while transmitting a data byte as master. | Abort failed transfer. | 0 | 0 | 0 | — |
| | | | | | | Reschedule failed transfer. | 1 | 0 | 0 | 1110 |

**Table 24.6. SMBus Status Decoding: Hardware ACK 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 SMB0-DAT. | 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 |

**Table 24.6. SMBus Status Decoding: Hardware ACK 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 | |
| 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 |
| 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 | — |

**Table 24.6. SMBus Status Decoding: Hardware ACK 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 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 |

## 24.7. I2C / SMBus Control Registers

### Register 24.1. SMB0CF: SMBus0 Configuration

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ENSMB | INH | BUSY | EXTHOLD | SMBTOE | SMBFTE | SMBCS | |
| Type | RW | RW | R | RW | RW | RW | RW | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xC1** | | | | | | | | |

**Table 24.7. SMB0CF Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7 | ENSMB | **SMBus0 Enable.**<br>This bit enables the SMBus0 interface when set to 1. When enabled, the interface constantly monitors the SDA and SCL pins. |
| 6 | INH | **SMBus0 Slave Inhibit.**<br>When this bit is set to logic 1, the SMBus0 does not generate an interrupt when slave events occur. This effectively removes the SMBus0 slave from the bus. Master Mode interrupts are not affected. |
| 5 | BUSY | **SMBus0 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 | **SMBus0 Setup and Hold Time Extension Enable.**<br>This bit controls the SDA setup and hold times.<br>0: SDA Extended Setup and Hold Times disabled.<br>1: SDA Extended Setup and Hold Times enabled. |
| 3 | SMBTOE | **SMBus0 SCL Timeout Detection Enable.**<br>This bit enables SCL low timeout detection. If set to logic 1, the SMBus0 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 SMBus0 communication. |
| 2 | SMBFTE | **SMBus0 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. |

**Table 24.7. SMB0CF Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 1:0 | SMBCS | **SMBus0 Clock Source Selection.**<br>These two bits select the SMBus0 clock source, which is used to generate the SMBus0 bit rate. See the SMBus clock timing section for additional details.<br>00: Timer 0 Overflow<br>01: Timer 1 Overflow<br>10: Timer 2 High Byte Overflow<br>11: Timer 2 Low Byte Overflow |

## Register 24.2. SMB0TC: SMBus0 Timing and Pin Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | SWAP | Reserved | | | | | SDD | |
| Type | RW | R | | | | | RW | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xAC** | | | | | | | | |

**Table 24.8. SMB0TC Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7 | SWAP | **SMBus0 Swap Pins.**<br>This bit swaps the order of the SMBus0 pins on the crossbar.<br>0: SDA is mapped to the lower-numbered port pin, and SCL is mapped to the higher-numbered port pin.<br>1: SCL is mapped to the lower-numbered port pin, and SDA is mapped to the higher-numbered port pin. |
| 6:2 | Reserved | Must write reset value. |
| 1:0 | SDD | **SMBus0 Start Detection Window.**<br>These bits increase the hold time requirement between SDA falling and SCL falling for START detection.<br>00: No additional hold time window (0-1 SYSCLK).<br>01: Increase hold time window to 2-3 SYSCLKs.<br>10: Increase hold time window to 4-5 SYSCLKs.<br>11: Increase hold time window to 8-9 SYSCLKs. |

## Register 24.3. SMB0CN: SMBus0 Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | MASTER | TXMODE | STA | STO | ACKRQ | ARBLOST | ACK | SI |
| Type | R | R | RW | RW | R | R | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xC0 (bit-addressable)** | | | | | | | | |

**Table 24.9. SMB0CN Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7 | MASTER | **SMBus0 Master/Slave Indicator.**<br>This read-only bit indicates when the SMBus0 is operating as a master.<br>0: SMBus0 operating in slave mode.<br>1: SMBus0 operating in master mode. |
| 6 | TXMODE | **SMBus0 Transmit Mode Indicator.**<br>This read-only bit indicates when the SMBus0 is operating as a transmitter.<br>0: SMBus0 in Receiver Mode.<br>1: SMBus0 in Transmitter Mode. |
| 5 | STA | **SMBus0 Start Flag.**<br>When reading STA, a 1 indicates that a start or repeated start condition was detected on the bus.<br>Writing a 1 to the STA bit initiates a start or repeated start on the bus. |
| 4 | STO | **SMBus0 Stop Flag.**<br>When reading STO, a 1 indicates that a stop condition was detected on the bus (in slave mode) or is pending (in master mode).<br>When acting as a master, writing a 1 to the STO bit initiates a stop condition on the bus. This bit is cleared by hardware. |
| 3 | ACKRQ | **SMBus0 Acknowledge Request.**<br>0: No ACK requested.<br>1: ACK requested. |
| 2 | ARBLOST | **SMBus0 Arbitration Lost Indicator.**<br>0: No arbitration error.<br>1: Arbitration error occurred. |
| 1 | ACK | **SMBus0 Acknowledge.**<br>When read as a master, the ACK bit indicates whether an ACK (1) or NACK (0) is received during the most recent byte transfer.<br>As a slave, this bit should be written to send an ACK (1) or NACK (0) to a master request. Note that the logic level of the ACK bit on the SMBus interface is inverted from the logic of the register ACK bit. |

**Table 24.9. SMB0CN Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 0 | SI | **SMBus0 Interrupt Flag.**<br>This bit is set by hardware to indicate that the current SMBus0 state machine operation (such as writing a data or address byte) is complete. While SI is set, SCL0 is held low and SMBus0 is stalled. SI0 must be cleared by software. Clearing SI0 initiates the next SMBus0 state machine operation. |

## Register 24.4. SMB0ADR: SMBus0 Slave Address

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | SLV | | | | | | | GC |
| Type | RW | | | | | | | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xD7** | | | | | | | | |

**Table 24.10. SMB0ADR Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:1 | SLV | **SMBus Hardware Slave Address.**<br>Defines the SMBus0 Slave Address(es) for automatic hardware acknowledgement. Only address bits which have a 1 in the corresponding bit position in SLVM 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. |

## Register 24.5. SMB0ADM: SMBus0 Slave Address Mask

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | SLVM | | | | | | | EHACK |
| Type | RW | | | | | | | RW |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| **SFR Address: 0xD6** | | | | | | | | |

**Table 24.11. SMB0ADM Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:1 | SLVM | **SMBus0 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 enables comparisons with the corresponding bit in SLV. 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. |

## Register 24.6. SMB0DAT: SMBus0 Data

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | SMB0DAT | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xC2** | | | | | | | | |

**Table 24.12. SMB0DAT Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | SMB0DAT | **SMBus0 Data.**<br>The SMB0DAT register contains a byte of data to be transmitted on the SMBus0 serial interface or a byte that has just been received on the SMBus0 serial interface. The CPU can safely read from or write to this register whenever the SI serial interrupt flag 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. |

# 25. Timers (Timer0, Timer1, Timer2 and Timer3)

Each MCU in the C8051F85x/86x family 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 timers for timing peripherals or for general purpose use. These timers can be used to measure time intervals, count external events and generate periodic interrupt requests. Timer 0 and Timer 1 are nearly identical and have four primary modes of operation. Timer 2 and Timer 3 are also identical and offer both 16-bit and split 8-bit timer functionality with auto-reload capabilities. Timer 2 and Timer 3 both offer a capture function, but are different in their system-level connections. Timer 2 is capable of performing a capture function on an external signal input routed through the crossbar, while the Timer 3 capture is dedicated to the low-frequency oscillator output. Table 25.1 summarizes the modes available to each timer.

**Table 25.1. Timer Modes**

| 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 | Two 8-bit timers with auto-reload | Two 8-bit timers with auto-reload |
| 8-bit counter/timer with auto-reload | Input pin capture | Low-frequency oscillator capture |
| 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.

Timer 0/1 may then be configured to use this pre-scaled clock signal or the system clock. Timer 2 and Timer 3 may be clocked by the system clock, the system clock divided by 12, or the external oscillator clock source divided by 8.

Timer 0 and Timer 1 may also be operated as counters. When functioning as a counter, a counter/timer register is incremented on each high-to-low transition at the selected input pin (T0 or T1). Events with a frequency of up to one-fourth the system clock frequency can be counted. The input signal need not be periodic, but it must be held at a given level for at least two full system clock cycles to ensure the level is properly sampled.

All four timers are capable of clocking other peripherals and triggering events in the system. The individual peripherals select which timer to use for their respective functions. Table 25.2 summarizes the peripheral connections for each timer. Note that the Timer 2 and Timer 3 high overflows apply to the full timer when operating in 16-bit mode or the high-byte timer when operating in 8-bit split mode.

**Table 25.2. Timer Peripheral Clocking / Event Triggering**

| Function | T0 Overflow | T1 Overflow | T2 High Overflow | T2 Low Overflow | T3 High Overflow | T3 Low Overflow |
|---|---|---|---|---|---|---|
| UART0 Baud Rate | | X | | | | |
| SMBus0 Clock Rate | X | X | X | X | | |
| SMBus0 SCL Low Timeout | | | | | X | |
| PCA0 Clock | X | | | | | |

**Table 25.2. Timer Peripheral Clocking / Event Triggering**

| Function | T0 Overflow | T1 Overflow | T2 High Overflow | T2 Low Overflow | T3 High Overflow | T3 Low Overflow |
|---|---|---|---|---|---|---|
| ADC0 Conversion Start | X | | X* | X* | X* | X* |
| ***Note:** The high-side overflow is used when the timer is in16-bit mode. The low-side overflow is used in 8-bit mode. | | | | | | |

## 25.1. Timer 0 and Timer 1

Timer 0 and Timer 1 are each implemented as a16-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. Timer 1 interrupts can be enabled by setting the ET1 bit in the IE register. 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 for the operating modes 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 in TCON is set and an interrupt will occur if Timer 0 interrupts are enabled.

The CT0 bit in the TMOD register selects the counter/timer's clock source. When CT0 is set to logic 1, high-to-low transitions at the selected Timer 0 input pin (T0) increment the timer register. Clearing CT selects the clock defined by the T0M bit in register CKCON. 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.

Setting the TR0 bit enables the timer when either GATE0 in the TMOD register is logic 0 or the input signal $\overline{INT0}$ is active as defined by bit IN0PL in register IT01CF. Setting GATE0 to 1 allows the timer to be controlled by the external input signal $\overline{INT0}$, facilitating pulse width measurements.

| TR0 | GATE0 | $\overline{INT0}$ | Counter/Timer |
|---|---|---|---|
| 0 | X | X | Disabled |
| 1 | 0 | X | Enabled |
| 1 | 1 | 0 | Disabled |
| 1 | 1 | 1 | Enabled |
| **Note:** X = Don't Care | | | |

Setting TR0 does not force the timer to reset. The timer registers should be loaded with the desired initial value before the timer is enabled.

TL1 and TH1 form the 13-bit register for Timer 1 in the same manner as described above for TL0 and TH0. Timer 1 is configured and controlled using the relevant TCON and TMOD bits just as with Timer 0. The input signal $\overline{INT1}$ is used with Timer 1; the /INT1 polarity is defined by bit IN1PL in register IT01CF.

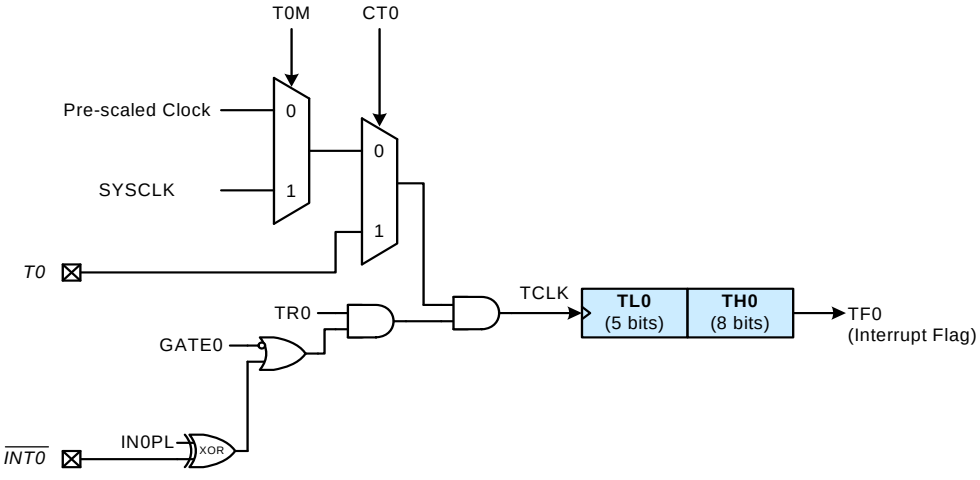



**Figure 25.1. T0 Mode 0 Block Diagram**

### 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 in the TCON register 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 enables the timer when either GATE0 in the TMOD register is logic 0 or when the input signal $\overline{INT0}$ is active as defined by bit IN0PL in register IT01CF.

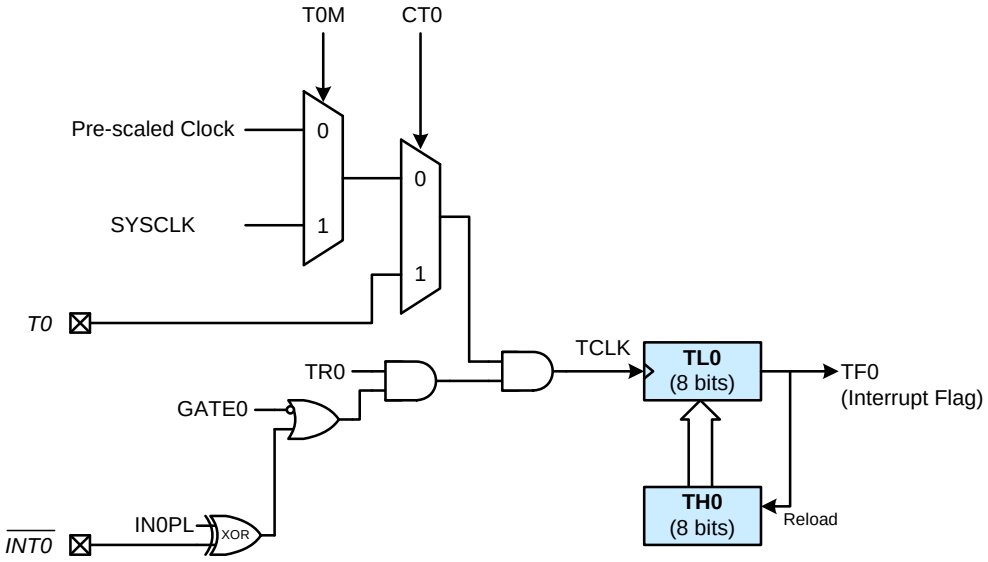



**Figure 25.2. T0 Mode 2 Block Diagram**

### 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, CT0, 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.

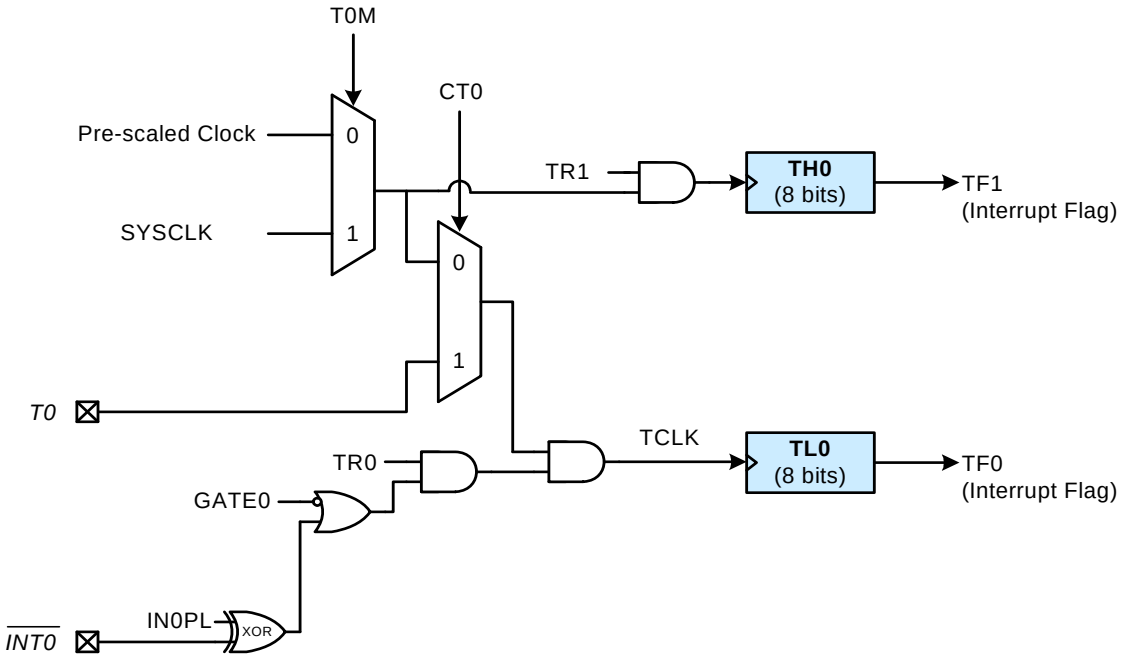



**Figure 25.3. T0 Mode 3 Block Diagram**

## 25.2. Timer 2 and Timer 3

Timer 2 and Timer 3 are functionally equivalent, with the only differences being the top-level connections to other parts of the system, as detailed in Table 25.1 and Table 25.2.

The timers are 16 bits wide, formed by two 8-bit SFRs: TMRnL (low byte) and TMRnH (high byte). Each timer may operate in 16-bit auto-reload mode or (split) 8-bit auto-reload mode. The TnSPLIT bit in TMRnCN defines the timer operation mode.

The timers may be clocked by the system clock, the system clock divided by 12, or the external oscillator source divided by 8. Note that the external oscillator source divided by 8 is synchronized with the system clock.

### 25.2.1. 16-bit Timer with Auto-Reload

When TnSPLIT is zero, the timer operates as a 16-bit timer with auto-reload. In this mode, the timer may be configured to clock from SYSCLK, SYSCLK divided by 12, or the external oscillator clock source divided by 8. As the 16-bit timer register increments and overflows from 0xFFFF to 0x0000, the 16-bit value in the timer reload registers (TMRnRLH and TMRnRLL) is loaded into the main timer count register as shown in Figure 25.4, and the High Byte Overflow Flag (TFnH) is set. If the timer interrupts are enabled, an interrupt will be generated on each timer overflow. Additionally, if the timer interrupts are enabled and the TFnLEN bit is set, an interrupt will be generated each time the lower 8 bits (TMRnL) overflow from 0xFF to 0x00.

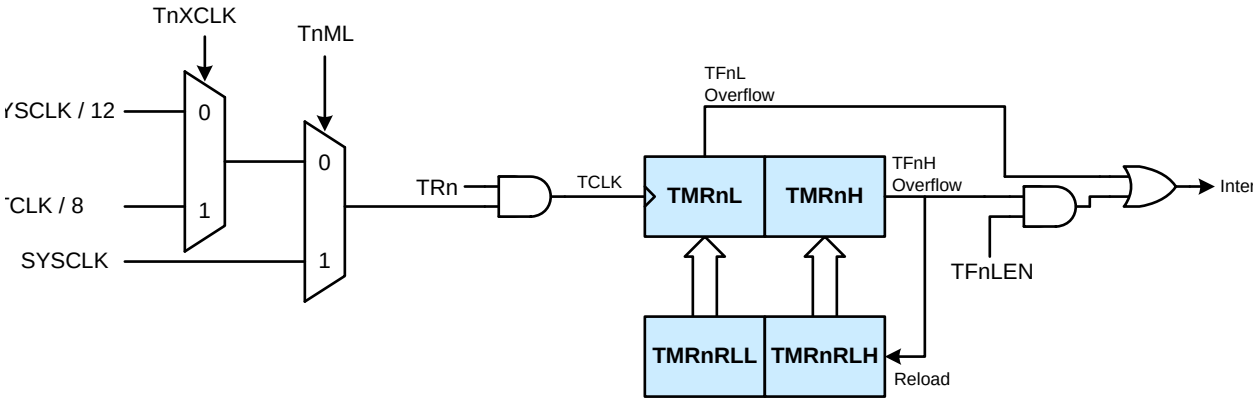



**Figure 25.4. 16-Bit Mode Block Diagram**

### 25.2.2. 8-bit Timers with Auto-Reload

When TnSPLIT is set, the timer operates as two 8-bit timers (TMRnH and TMRnL). Both 8-bit timers operate in auto-reload mode as shown in Figure 25.5. TMRnRLL holds the reload value for TMRnL; TMRnRLH holds the reload value for TMRnH. The TRn bit in TMRnCN handles the run control for TMRnH. TMRnL is always running when configured for 8-bit auto-reload mode.

Each 8-bit timer may be configured to clock from SYSCLK, SYSCLK divided by 12, or the external oscillator clock source divided by 8. The Clock Select bits (TnMH and TnML in CKCON) select either SYSCLK or the clock defined by the External Clock Select bit (TnXCLK in TMRnCN), as follows:

| TnMH | TnXCLK | TMRnH Clock Source |
|---|---|---|
| 0 | 0 | SYSCLK / 12 |
| 0 | 1 | External Clock / 8 |
| 1 | X | SYSCLK |

| TnML | TnXCLK | TMRnL Clock Source |
|---|---|---|
| 0 | 0 | SYSCLK / 12 |
| 0 | 1 | External Clock / 8 |
| 1 | X | SYSCLK |

The TFnH bit is set when TMRnH overflows from 0xFF to 0x00; the TFnL bit is set when TMRnL overflows from 0xFF to 0x00. When timer interrupts are enabled, an interrupt is generated each time TMRnH overflows. If timer interrupts are enabled and TFnLEN is set, an interrupt is generated each time either TMRnL or TMRnH overflows. When TFnLEN is enabled, software must check the TFnH and TFnL flags to determine the source of the timer interrupt. The TFnH and TFnL interrupt flags are not cleared by hardware and must be manually cleared by software.

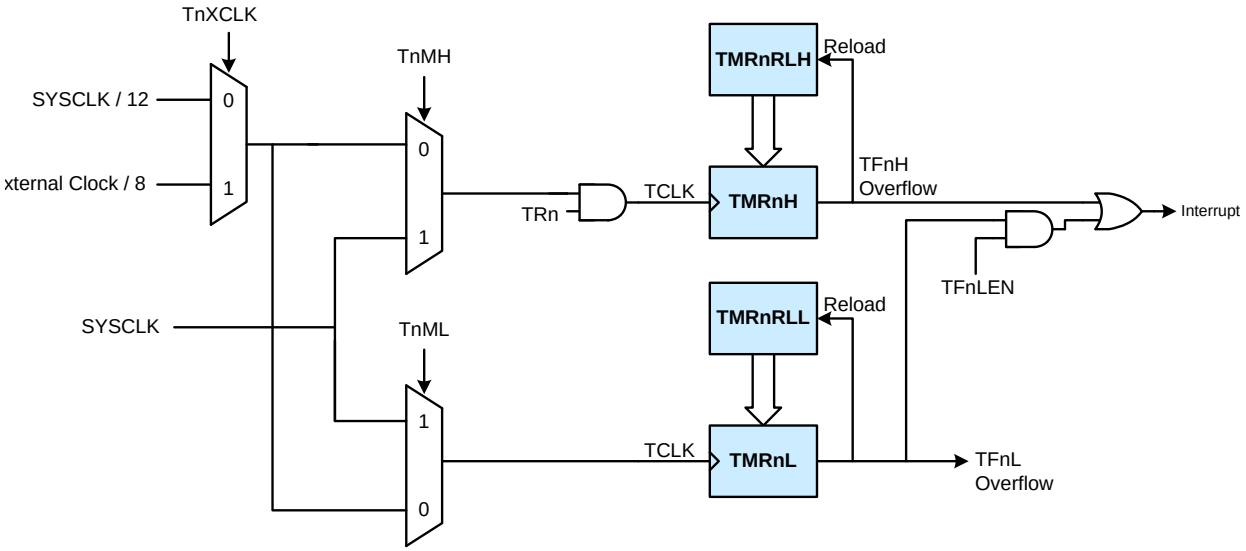



**Figure 25.5. 8-Bit Mode Block Diagram**

### 25.2.3. Capture Mode

Capture mode allows an external input (Timer 2) or the low-frequency oscillator clock (Timer 3) to be measured against the system clock or an external oscillator source. The timer can be clocked from the system clock, the system clock divided by 12, or the external oscillator divided by 8, depending on the TnML, and TnXCLK settings.

Setting TFnCEN to 1 enables Capture Mode. In this mode, TnSPLIT should be set to 0, as the full 16-bit timer is used. Upon a falling edge of the input capture signal, the contents of the timer register (TMRnH:TMRnL) are loaded into the reload registers (TMRnRLH:TMRnRLL) and the TFnH flag is set. By recording the difference between two successive timer capture values, the period of the captured signal can be determined with respect to the selected timer clock.

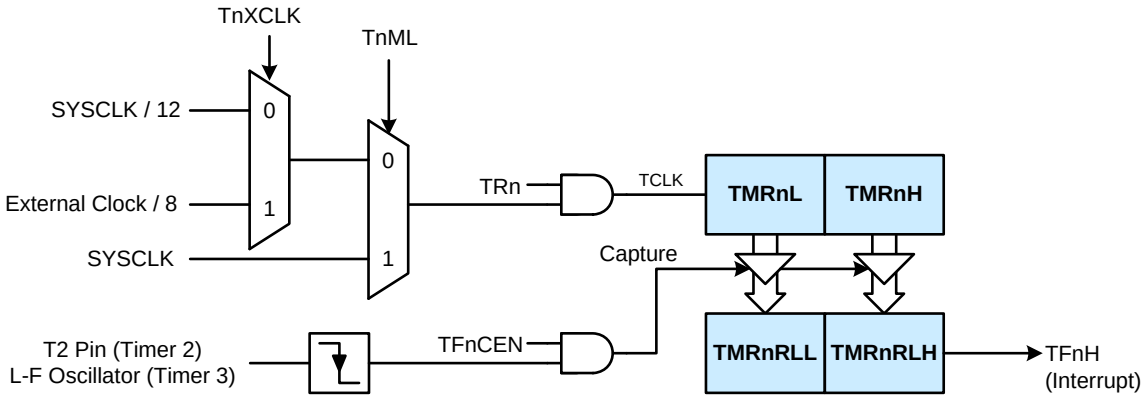



**Figure 25.6. Capture Mode Block Diagram**

## 25.3. Timer Control Registers

### Register 25.1. CKCON: Clock Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | T3MH | T3ML | T2MH | T2ML | T1M | T0M | SCA | |
| Type | RW | RW | RW | RW | RW | RW | RW | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0x8E** | | | | | | | | |

**Table 25.3. CKCON Register Bit Descriptions**

| 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 field, SCA.<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 field, SCA.<br>1: Counter/Timer 0 uses the system clock. |

**Table 25.3. CKCON Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 1:0 | SCA | **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) |

## Register 25.2. TCON: Timer 0/1 Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TF1 | TR1 | TF0 | TR0 | IE1 | IT1 | IE0 | IT0 |
| Type | RW | RW | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0x88 (bit-addressable)** | | | | | | | | |

**Table 25.4. TCON Register Bit Descriptions**

| 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 INT1 interrupt will be edge or level sensitive. INT1 is configured active low or high by the IN1PL bit in register IT01CF.<br>0: INT1 is level triggered.<br>1: INT1 is edge triggered. |
| 1 | IE0 | **External Interrupt 0.**<br>This flag is set by hardware when an edge/level of type defined by IT0 is detected. It can be cleared by software but is automatically cleared when the CPU vectors to the External Interrupt 0 service routine in edge-triggered mode. |
| 0 | IT0 | **Interrupt 0 Type Select.**<br>This bit selects whether the configured INT0 interrupt will be edge or level sensitive. INT0 is configured active low or high by the IN0PL bit in register IT01CF.<br>0: INT0 is level triggered.<br>1: INT0 is edge triggered. |

## Register 25.3. TMOD: Timer 0/1 Mode

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | GATE1 | CT1 | T1M | | GATE0 | CT0 | T0M | |
| Type | RW | RW | RW | | RW | RW | RW | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0x89** | | | | | | | | |

**Table 25.5. TMOD Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7 | GATE1 | **Timer 1 Gate Control.**<br>0: Timer 1 enabled when TR1 = 1 irrespective of INT1 logic level.<br>1: Timer 1 enabled only when TR1 = 1 and INT1 is active as defined by bit IN1PL in register IT01CF. |
| 6 | CT1 | **Counter/Timer 1 Select.**<br>0: Timer Mode. Timer 1 increments on the clock defined by T1M in the CKCON register.<br>1: Counter Mode. Timer 1 increments on high-to-low transitions of an external pin (T1). |
| 5:4 | T1M | **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 INT0 logic level.<br>1: Timer 0 enabled only when TR0 = 1 and INT0 is active as defined by bit IN0PL in register IT01CF. |
| 2 | CT0 | **Counter/Timer 0 Select.**<br>0: Timer Mode. Timer 0 increments on the clock defined by T0M in the CKCON register.<br>1: Counter Mode. Timer 0 increments on high-to-low transitions of an external pin (T0). |
| 1:0 | T0M | **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 |

**Register 25.4. TL0: Timer 0 Low Byte**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TL0 | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0x8A** | | | | | | | | |

**Table 25.6. TL0 Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | TL0 | **Timer 0 Low Byte.**<br>The TL0 register is the low byte of the 16-bit Timer 0. |

## Register 25.5. TL1: Timer 1 Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TL1 | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0x8B** | | | | | | | | |

**Table 25.7. TL1 Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | TL1 | **Timer 1 Low Byte.**<br>The TL1 register is the low byte of the 16-bit Timer 1. |

**Register 25.6. TH0: Timer 0 High Byte**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TH0 | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0x8C** | | | | | | | | |

**Table 25.8. TH0 Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | TH0 | **Timer 0 High Byte.**<br>The TH0 register is the high byte of the 16-bit Timer 0. |

## Register 25.7. TH1: Timer 1 High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TH1 | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0x8D** | | | | | | | | |

**Table 25.9. TH1 Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | TH1 | **Timer 1 High Byte.**<br>The TH1 register is the high byte of the 16-bit Timer 1. |

## Register 25.8. TMR2CN: Timer 2 Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TF2H | TF2L | TF2LEN | TF2CEN | T2SPLIT | TR2 | Reserved | T2XCLK |
| Type | RW | RW | RW | RW | RW | RW | R | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xC8 (bit-addressable)** | | | | | | | | |

**Table 25.10. TMR2CN Register Bit Descriptions**

| 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. If TF2CEN is set and Timer 2 interrupts are enabled, an interrupt will be generated on a falling edge of the selected T2 input pin, and the current 16-bit timer value in TMR2H:TMR2L will be copied to TMR2RLH:TMR2RLL. |
| 3 | T2SPLIT | **Timer 2 Split Mode Enable.**<br>When this bit is set, Timer 2 operates as two 8-bit timers with auto-reload.<br>0: Timer 2 operates in 16-bit auto-reload mode.<br>1: Timer 2 operates as two 8-bit auto-reload timers. |
| 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 | Reserved | Must write reset value. |

**Table 25.10. TMR2CN Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 0 | T2XCLK | **Timer 2 External Clock Select.**<br>This bit selects the external clock source for Timer 2. If Timer 2 is in 8-bit mode, this bit selects the external oscillator clock source for both timer bytes. However, the Timer 2 Clock Select bits (T2MH and T2ML in register CKCON) may still be used to select between the external clock and the system clock for either timer.<br>0: Timer 2 clock is the system clock divided by 12.<br>1: Timer 2 clock is the external clock divided by 8 (synchronized with SYSCLK). |

## Register 25.9. TMR2RLL: Timer 2 Reload Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TMR2RLL | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xCA** | | | | | | | | |

**Table 25.11. TMR2RLL Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | TMR2RLL | **Timer 2 Reload Low Byte.**<br>When operating in one of the auto-reload modes, TMR2RLL holds the reload value for the low byte of Timer 2 (TMR2L). When operating in capture mode, TMR2RLL is the captured value of TMR2L. |

## Register 25.10. TMR2RLH: Timer 2 Reload High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TMR2RLH | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xCB** | | | | | | | | |

**Table 25.12. TMR2RLH Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | TMR2RLH | **Timer 2 Reload High Byte.**<br>When operating in one of the auto-reload modes, TMR2RLH holds the reload value for the high byte of Timer 2 (TMR2H). When oeprating in capture mode, TMR2RLH is the captured value of TMR2H. |

## Register 25.11. TMR2L: Timer 2 Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TMR2L | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xCC** | | | | | | | | |

**Table 25.13. TMR2L Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | TMR2L | **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. |

## Register 25.12. TMR2H: Timer 2 High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TMR2H | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0xCD** | | | | | | | | |

**Table 25.14. TMR2H Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | TMR2H | **Timer 2 High 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. |

## Register 25.13. TMR3CN: Timer 3 Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TF3H | TF3L | TF3LEN | TF3CEN | T3SPLIT | TR3 | Reserved | T3XCLK |
| Type | RW | RW | RW | RW | RW | RW | R | RW |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0x91** | | | | | | | | |

**Table 25.15. TMR3CN Register Bit Descriptions**

| 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 Capture Enable.**<br>When set to 1, this bit enables Timer 3 Capture Mode. If TF3CEN is set and Timer 3 interrupts are enabled, an interrupt will be generated on a falling edge of the low-frequency oscillator output, and the current 16-bit timer value in TMR3H:TMR3L will be copied to TMR3RLH:TMR3RLL. |
| 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 | Reserved | Must write reset value. |

**Table 25.15. TMR3CN Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 0 | T3XCLK | **Timer 3 External Clock Select.**<br>This bit selects the external clock source for Timer 3. If Timer 3 is in 8-bit mode, this bit selects the external oscillator clock source for both timer bytes. However, the Timer 3 Clock Select bits (T3MH and T3ML in register CKCON) may still be used to select between the external clock and the system clock for either timer.<br>0: Timer 3 clock is the system clock divided by 12.<br>1: Timer 3 clock is the external clock divided by 8 (synchronized with SYSCLK). |

## Register 25.14. TMR3RLL: Timer 3 Reload Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TMR3RLL | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0x92** | | | | | | | | |

**Table 25.16. TMR3RLL Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | TMR3RLL | **Timer 3 Reload Low Byte.**<br>When operating in one of the auto-reload modes, TMR3RLL holds the reload value for the low byte of Timer 3 (TMR3L). When operating in capture mode, TMR3RLL is the captured value of TMR3L. |

## Register 25.15. TMR3RLH: Timer 3 Reload High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TMR3RLH | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0x93** | | | | | | | | |

**Table 25.17. TMR3RLH Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | TMR3RLH | **Timer 3 Reload High Byte.**<br>When operating in one of the auto-reload modes, TMR3RLH holds the reload value for the high byte of Timer 3 (TMR3H). When oeprating in capture mode, TMR3RLH is the captured value of TMR3H. |

## Register 25.16. TMR3L: Timer 3 Low Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TMR3L | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0x94** | | | | | | | | |

**Table 25.18. TMR3L Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | TMR3L | **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. |

## Register 25.17. TMR3H: Timer 3 High Byte

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TMR3H | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0x95** | | | | | | | | |

**Table 25.19. TMR3H Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | TMR3H | **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. Universal Asynchronous Receiver/Transmitter (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 “26.1. Enhanced Baud Rate Generation” on page 289). 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 (TI is set in SCON0), or a data byte has been received (RI 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).

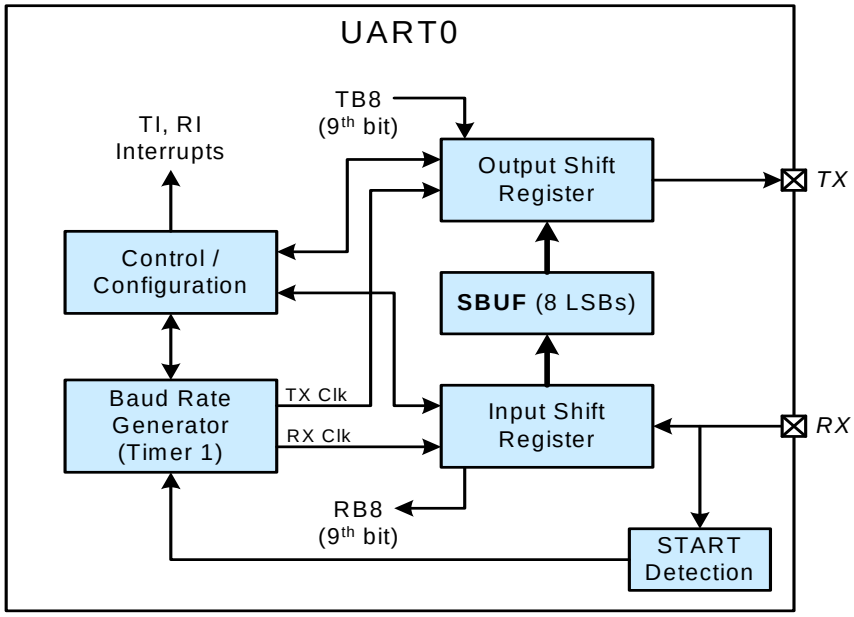



**Figure 26.1. UART0 Block Diagram**

## 26.1. 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 26.2), 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.

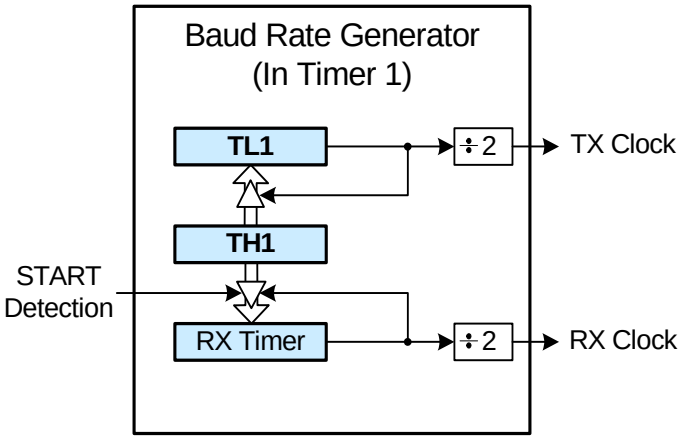



**Figure 26.2. UART0 Baud Rate Logic**

Timer 1 should be configured for Mode 2, 8-bit auto-reload. 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 overflow rate, the UART0 baud rate is determined by Equation 26.1.

$$\text{UartBaudRate} = \frac{1}{2} \times \text{T1_Overflow_Rate}$$

**Equation 26.1. UART0 Baud Rate**

Timer 1 overflow rate is selected as described in the Timer section. A quick reference for typical baud rates and system clock frequencies is given in Table 26.1.

## 26.2. Operational Modes

UART0 provides standard asynchronous, full duplex communication. The UART mode (8-bit or 9-bit) is selected by the S0MODE bit in register SCON.

### 26.2.1. 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 TX pin and received at the RX pin. On receive, the eight data bits are stored in SBUF0 and the stop bit goes into RB8 in the SCON register.

Data transmission begins when software writes a data byte to the SBUF0 register. The TI Transmit Interrupt Flag is set at the end of the transmission (the beginning of the stop-bit time). Data reception can begin any time after the REN Receive Enable bit 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: RI must be logic 0, and if MCE 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 RB8 and the RI flag is set. If these conditions are not met, SBUF0 and RB8 will not be loaded and the RI flag will not be set. An interrupt will occur if enabled when either TI or RI is set.

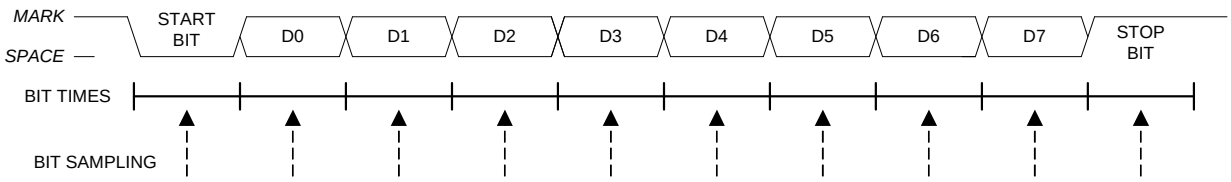



**Figure 26.3. 8-Bit UART Timing Diagram**

### 26.2.2. 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 TB8, 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 RB8 and the stop bit is ignored.

Data transmission begins when an instruction writes a data byte to the SBUF0 register. The TI Transmit Interrupt Flag is set at the end of the transmission (the beginning of the stop-bit time). Data reception can begin any time after the REN Receive Enable bit 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) RI must be logic 0, and (2) if MCE is logic 1, the 9th bit must be logic 1 (when MCE 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 RB8, and the RI flag is set to 1. If the above conditions are not met, SBUF0 and RB8 will not be loaded and the RI flag will not be set to 1. A UART0 interrupt will occur if enabled when either TI or RI is set to 1.

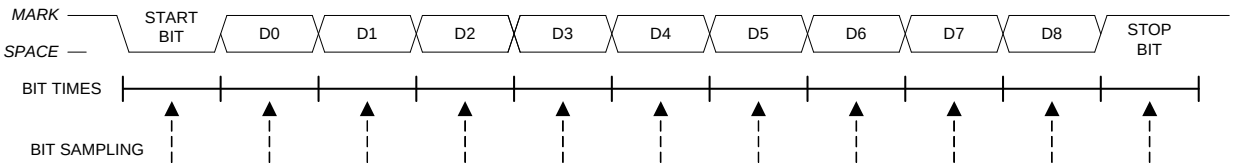



**Figure 26.4. 9-Bit UART Timing Diagram**

## 26.3. 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 MCE bit 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 (RB8 = 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 MCE bit to enable interrupts on the reception of the following data byte(s). Slaves that weren't addressed leave their MCE 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 MCE 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).

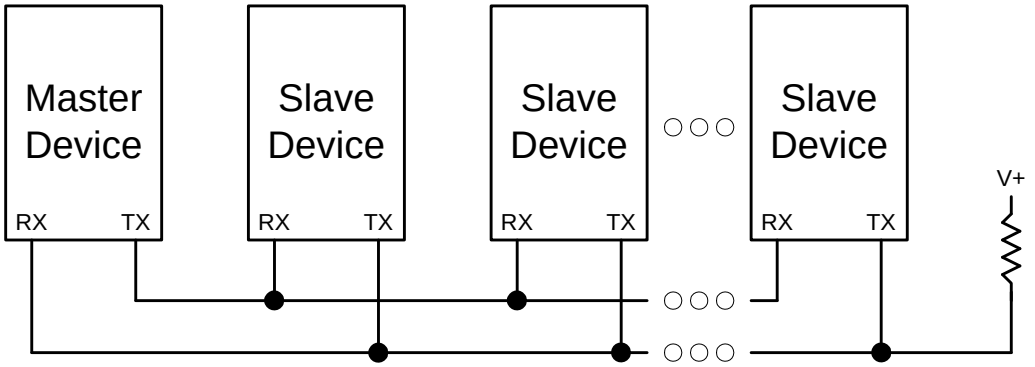



**Figure 26.5. UART Multi-Processor Mode Interconnect Diagram**

**Table 26.1. Timer Settings for Standard Baud Rates Using the Internal 24.5 MHz Oscillator**

| | Frequency: 49 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 Timer1 chapter.
2. X = Don't care.

## 26.4. UART Control Registers

### Register 26.1. SCON0: UART0 Serial Port Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | SMODE | Reserved | MCE | REN | TB8 | RB8 | TI | RI |
| Type | RW | R | RW | RW | RW | RW | RW | RW |
| Reset | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0x98 (bit-addressable)** | | | | | | | | |

**Table 26.2. SCON0 Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7 | SMODE | **Serial Port 0 Operation Mode.**<br>Selects the UART0 Operation Mode.<br>0: 8-bit UART with Variable Baud Rate (Mode 0).<br>1: 9-bit UART with Variable Baud Rate (Mode 1). |
| 6 | Reserved | Must write reset value. |
| 5 | MCE | **Multiprocessor Communication Enable.**<br>This bit enables checking of the stop bit or the 9th bit in multi-drop communication buses. The function of this bit is dependent on the UART0 operation mode selected by the SMODE bit. In Mode 0 (8-bits), the peripheral will check that the stop bit is logic 1. In Mode 1 (9-bits) the peripheral will check for a logic 1 on the 9th bit.<br>0: Ignore level of 9th bit / Stop bit.<br>1: RI is set and an interrupt is generated only when the stop bit is logic 1 (Mode 0) or when the 9th bit is logic 1 (Mode 1). |
| 4 | REN | **Receive Enable.**<br>0: UART0 reception disabled.<br>1: UART0 reception enabled. |
| 3 | TB8 | **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 | RB8 | **Ninth Receive Bit.**<br>RB8 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 | TI | **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. |

**Table 26.2. SCON0 Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 0 | RI | **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. |

## Register 26.2. SBUF0: UART0 Serial Port Data Buffer

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | SBUF0 | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **SFR Address: 0x99** | | | | | | | | |

**Table 26.3. SBUF0 Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | SBUF0 | **Serial Data Buffer Bits.**<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. |

# 27. Watchdog Timer (WDT0)

The C8051F85x/86x family includes a programmable Watchdog Timer (WDT) running off the low-frequency oscillator. A WDT overflow will force the MCU into the reset state. To prevent the reset, the WDT must be restarted by application software before overflow. If the system experiences a software or hardware malfunction preventing the software from restarting the WDT, the WDT will overflow and cause a reset.

Following a reset the WDT is automatically enabled and running with the default maximum time interval. If desired the WDT can be disabled by system software or locked on to prevent accidental disabling. Once locked, the WDT cannot be disabled until the next system reset. The state of the $\overline{\text{RST}}$ pin is unaffected by this reset.

The WDT consists of an internal timer running from the low-frequency oscillator. The timer measures the period between specific writes to its control register. If this period exceeds the programmed limit, a WDT reset is generated. The WDT can be enabled and disabled as needed in software, or can be permanently enabled if desired. When the WDT is active, the low-frequency oscillator is forced on. All watchdog features are controlled via the Watchdog Timer Control Register (WDTCN).

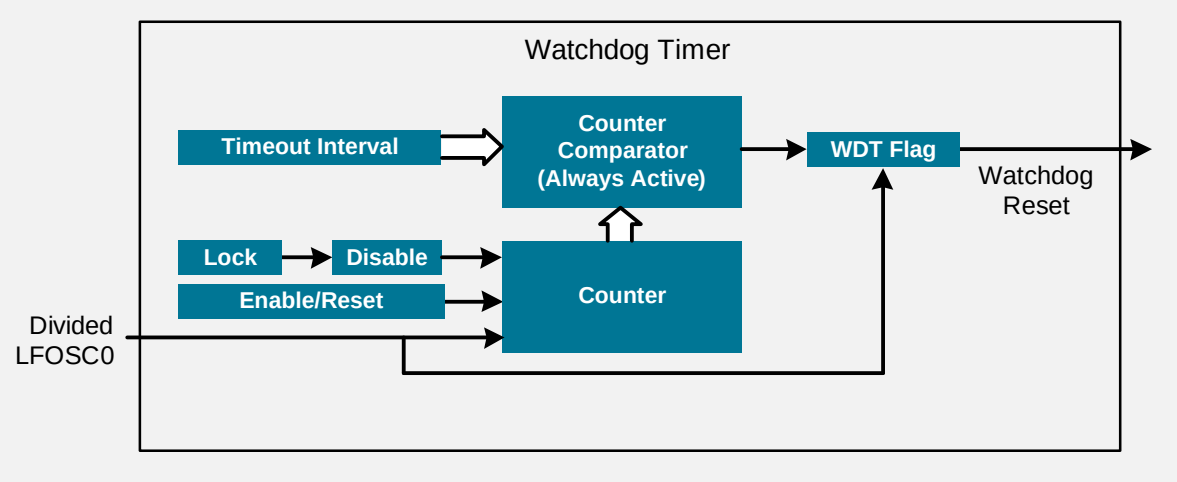



**Figure 27.1. Watchdog Timer Block Diagram**

## 27.1. Features

The watchdog timer includes a 16-bit timer with a programmable reset period. The registers are protected from inadvertent access by an independent lock and key interface.

The Watchdog Timer has the following features:

- Programmable timeout interval
- Runs from the low-frequency oscillator
- Lock-out feature to prevent any modification until a system reset

## 27.2. Enabling / Resetting the WDT

The watchdog timer is both enabled and reset when writing 0xA5 to the WDTCN register. The user's application software should include periodic writes of 0xA5 to WDTCN as needed to prevent a watchdog timer overflow. The counter is incremented on every divided LFOSC0 when the WDT is enabled. The WDT is enabled and reset as a result of any system reset.

## 27.3. Disabling the WDT

Writing 0xDE followed by 0xAD to the WDTCN register disables the WDT. The following code segment illustrates disabling the WDT:

```
CLR EA           ; disable all interrupts
MOV WDTCN,#0DEh  ; disable software watchdog timer
MOV WDTCN,#0ADh
; insert wait for 3 divided LFOSCO0 clock periods
SETB EA          ; re-enable interrupts
```

**Note:** Code that implements the wait must be inserted. Code that implements the wait is not explicitly implemented in the above sequence because it depends on the divided LFOSC0 clock and the SYSCLK clock selected.

The writes of 0xDE and 0xAD must occur within 4 clock cycles of each other, or the disable operation is ignored. Interrupts should be disabled during this procedure to avoid delay between the two writes.

The counter retains its value when the WDT is disabled. The counter comparator is always active and can generate a watchdog timer reset even if the watchdog timer is disabled. For example, a watchdog timer reset can be generated when changing from a higher to a lower interval as the counter is not cleared when the WDT is disabled. To avoid this always clear the counter by resetting the WDT before disabling and changing the timeout interval to a lower interval i.e., follow the code sequence shown below in “27.5. Setting the WDT Interval” .

## 27.4. Disabling the WDT Lockout

Writing 0xFF to WDTCN locks out the disable feature. Once locked out, the disable operation is ignored until the next system reset. Writing 0xFF does not enable or reset the watchdog timer. Applications always intending to use the watchdog should write 0xFF to WDTCN in the initialization code.

## 27.5. Setting the WDT Interval

WDTCN.[2:0] controls the watchdog timeout interval. The interval is given by the following equation, where $T_{lfosc}$ is the low-frequency oscillator clock period:

$$T_{LFOSC} \times 4^{(WDTCN[2:0] + 3)}$$

This provides a nominal interval range of 0.8 ms to 13.1 s. WDTCN.7 must be logic 0 when setting this interval. Reading WDTCN returns the programmed interval. WDTCN.[2:0] reads 111b after a system reset.

The following code segment illustrates changing the WDT interval to a lower interval:

```
MOV WDTCN,#0A5h  ; reset watchdog timer
; insert wait for 2 divided LFOSC0 clock periods
CLR EA           ; disable all interrupts
MOV WDTCN,#0DEh  ; disable software watchdog timer
MOV WDTCN,#0ADh
; insert wait for 3 divided LFOSC0 clock periods
SETB EA          ; re-enable interrupts
MOV WDTCN,WDT_interval  ; change the current WDT interval to a lower interval with
                          MSB cleared to 0
; insert wait for 1 SYSCLK periods
```

**Note:** Code that implements the wait must be inserted. Code that implements the wait is not explicitly implemented in the above sequence because it depends on the divided LFOSC0 clock and the SYSCLK clock selected.

## 27.6. Synchronization

The watchdog timer is controlled via the WDTCN control register using commands. Commands require synchronization between the system clock and WDT clock source, the divided LFOSC0 clock. The table below lists each WDT command and the number of clock periods from the specified clock source for the command to take effect.

**Table 27.1. Synchronization Delay**

| Command | Clock Delay Required to Apply |
|---|---|
| Enabling/Resetting the WDT | 2 divided LFOSC0 clock periods |
| Setting the WDT Interval | 1 SYSCLK clock period |
| Disabling the WDT | 3 divided LFOSC0 clock periods |

Due to the WDT command synchronization delay, observe the following guidelines while operating the WDT:

- Only issue one command to WDTCN register within one divided LFOSC0 clock period.
- Change the LFOSC0 divider or disable the LFOSC0 only while the WDT is disabled.
- Change the WDT interval only while the WDT is disabled.

## 27.7. Watchdog Timer Control Registers

### Register 27.1. WDTCN: Watchdog Timer Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | WDTCN | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 |
| **SFR Address: 0x97** | | | | | | | | |

**Table 27.2. WDTCN Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | WDTCN | **WDT Control.**<br>The WDT control field has different behavior for reads and writes.<br>Read:<br>When reading the WDTCN register, the lower three bits (WDTCN[2:0]) indicate the current timeout interval. Bit WDTCN.4 indicates whether the WDT is active (logic 1) or inactive (logic 0).<br>Write:<br>Writing the WDTCN register can set the timeout interval, enable the WDT, disable the WDT, reset the WDT, or lock the WDT to prevent disabling.<br>Writing to WDTCN with the MSB (WDTCN.7) cleared to 0 will set the timeout interval to the value in bits WDTCN[2:0].<br>Writing 0xA5 both enables and reloads the WDT.<br>Writing 0xDE followed within 4 system clocks by 0xAD disables the WDT.<br>Writing 0xFF locks out the disable feature until the next device reset. |

# 28. Revision-Specific Behavior

C8051F85x/86x Revision B devices have differences from Revision C devices:

- Temperature Sensor offset and slope
- Flash endurance
- Latch-up performance
- Unique Identifier

## 28.1. Revision Identification

The Lot ID Code on the top side of the device package can be used for decoding device revision information. Figure 28.1, Figure 28.2, and Figure 28.3 show how to find the Lot ID Code on the top side of the device package.

Firmware can distinguish between a Revision B and Revision C device using the value of the REVID register described in “Device Identification and Unique Identifier” on page 68.

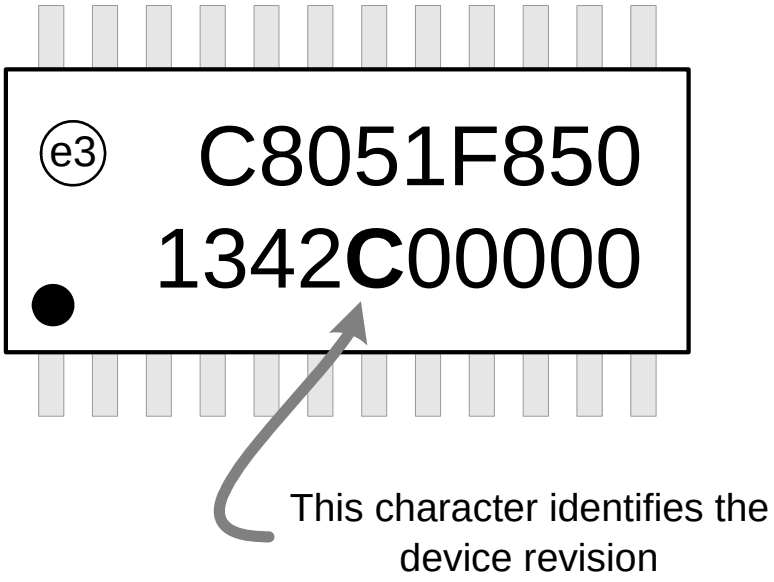



**Figure 28.1. QSOP-24 Package Revision Marking**

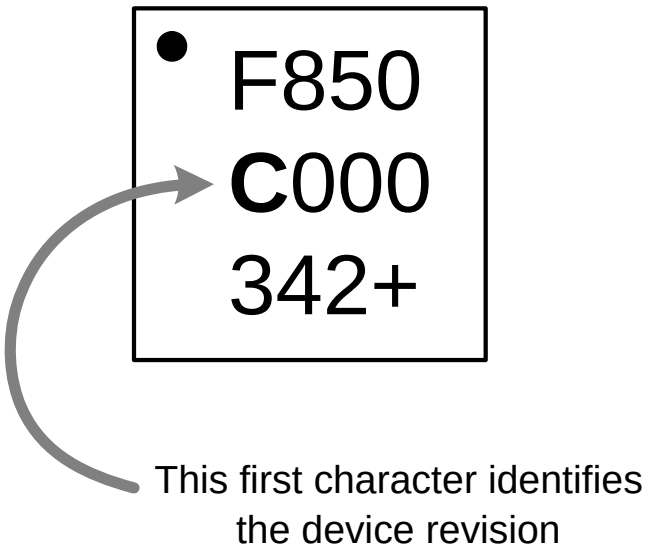



**Figure 28.2. QFN-20 Package Revision Marking**

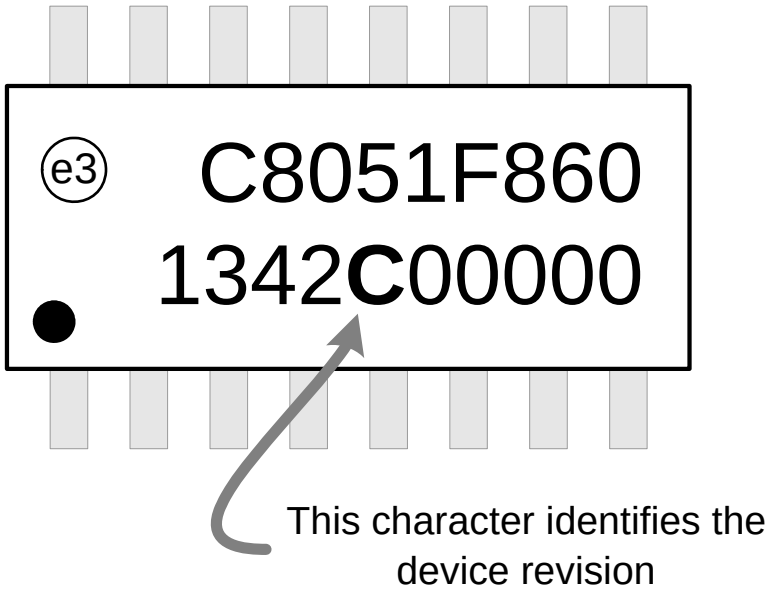



**Figure 28.3. SOIC-16 Package Revision Marking**

## 28.2. Temperature Sensor Offset and Slope

The temperature sensor slope and offset characteristics of Revision B devices are different than the slope and offset characteristics of Revision C devices. The differences are:

**Table 28.1. Temperature Sensor Revision Differences**

| Parameter | Symbol | Test Condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| **Revision B** | | | | | | |
| Offset | $V_{OFF}$ | $T_A$ = 0 °C | — | 713 | — | mV |
| Slope | M | | — | 2.67 | — | mV/°C |
| **Revision C** | | | | | | |
| Offset | $V_{OFF}$ | $T_A$ = 0 °C | — | 757 | — | mV |
| Slope | M | | — | 2.85 | — | mV/°C |

Firmware that uses the slope and offset of the temperature sensor to calculate the temperature from the sensor ADC reading can detect the revision of the device by reading the REVID register and adjust the slope and offset calculations based on the result. A REVID value of 0x01 indicates a Revision B device, and a REVID value of 0x02 indicates a Revision C device.

## 28.3. Flash Endurance

The flash endurance, or number of times the flash may be written and erased, on some Revision B devices may be lower than expected. Table 1.4 specifies a minimum Endurance (Write/Erase Cycles) as 20000, but some Revision B devices may support a minimum of ~5000 cycles.

## 28.4. Latch-Up Performance

Pulling the device pins below ground and drawing significant current (~3.5 mA) can cause a Power-On Reset event with Revision B devices. Some pins, like P0.0 and P0.1, are more susceptible to this behavior than others. This behavior is outside normal operating parameters and would typically be seen during latch-up or ESD performance testing.

## 28.5. Unique Identifier

Revision B devices do not implement the unique identifier described in “Device Identification and Unique Identifier” on page 68.

# 29. C2 Interface

C8051F85x/86x 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. Details on the C2 protocol can be found in the C2 Interface Specification.

## 29.1. 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. C2CK is shared with the RST pin, while the C2D signal is shared with a port I/O pin. 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 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 29.1.

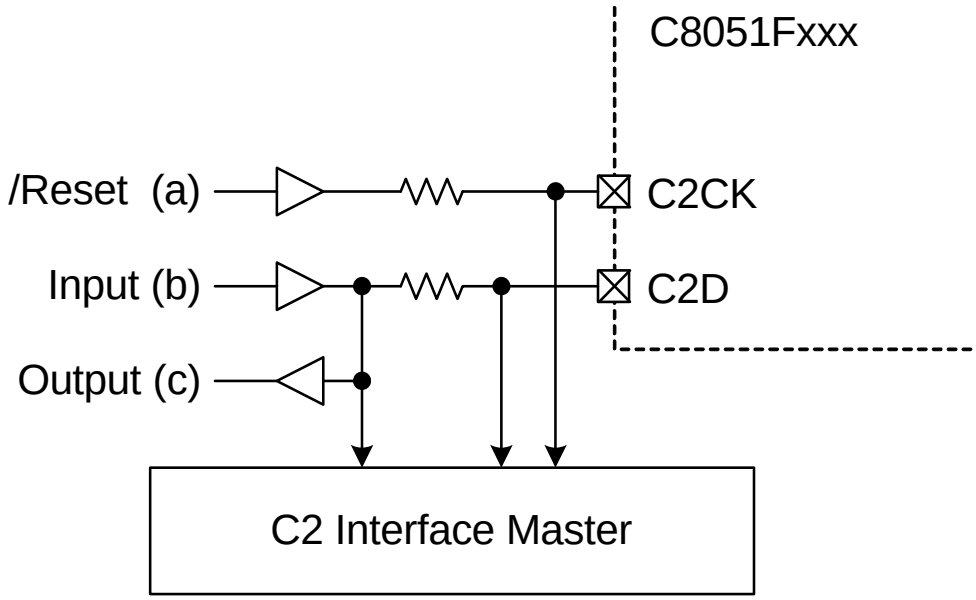



**Figure 29.1. Typical C2 Pin Sharing**

The configuration in Figure 29.1 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.

## 29.2. 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, and are not available in the SFR map for firmware access.

### Register 29.1. C2ADD: C2 Address

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | C2ADD | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **This register is part of the C2 protocol.** | | | | | | | | |

**Table 29.1. C2ADD Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | C2ADD | **C2 Address.**<br>The C2ADD register is accessed via the C2 interface. The value written to C2ADD selects the target data register for C2 Data Read and Data Write commands.<br>0x00: C2DEVID<br>0x01: C2REVID<br>0x02: C2FPCTL<br>0xB4: C2FPDAT |

## Register 29.2. C2DEVID: C2 Device ID

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | C2DEVID | | | | | | | |
| Type | R | | | | | | | |
| Reset | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| **C2 Address: 0x00** | | | | | | | | |

**Table 29.2. C2DEVID Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | C2DEVID | **Device ID.**<br>This read-only register returns the 8-bit device ID: 0x30 (C8051F85x/86x). |

## Register 29.3. C2REVID: C2 Revision ID

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | C2REVID | | | | | | | |
| Type | R | | | | | | | |
| Reset | X | X | X | X | X | X | X | X |
| **C2 Address: 0x01** | | | | | | | | |

**Table 29.3. C2REVID Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | C2REVID | **Revision ID.**<br>This read-only register returns the 8-bit revision ID. For example: 0x00 = Revision A, 0x01 = Revision B and 0x02 = Revision C. |

## Register 29.4. C2FPCTL: C2 Flash Programming Control

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | C2FPCTL | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **C2 Address: 0x02** | | | | | | | | |

**Table 29.4. C2FPCTL Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | C2FPCTL | **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. |

## Register 29.5. C2FPDAT: C2 Flash Programming Data

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | C2FPDAT | | | | | | | |
| Type | RW | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **C2 Address: 0xB4** | | | | | | | | |

**Table 29.5. C2FPDAT Register Bit Descriptions**

| Bit | Name | Function |
|---|---|---|
| 7:0 | C2FPDAT | **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.<br>0x03: Device Erase<br>0x06: Flash Block Read<br>0x07: Flash Block Write<br>0x08: Flash Page Erase |

# Document Change List

## Revision 0.5 to Revision 0.6

- Updated front page block diagram.
- Updated ADC supply current parameters in Table 1.2, “Power Consumption,” on page 8.
- Corrected flash programming voltage range in "Table 1.4. Flash Memory" on page 11.
- Added ADC Power-On Time specification in Table 1.7, “ADC,” on page 13.
- Added section "1.2. Typical Performance Curves" on page 19.
- Corrected DERIVID Information in Table 11.3, “DERIVID Register Bit Descriptions,” on page 70.
- Updated ADC chapter ("14. Analog-to-Digital Converter (ADC0)" on page 85) and expanded section "14.5. Power Considerations" on page 85 with recommended power configuration settings.
- Updated Figure 21.1, “Port I/O Functional Block Diagram,” on page 184.
- Corrected reset value in Register 24.5, “SMB0ADM: SMBus0 Slave Address Mask,” on page 257.
- Corrected description of IE0 in "Table 25.4. TCON Register Bit Descriptions" on page 259.

## Revision 0.6 to Revision 0.7

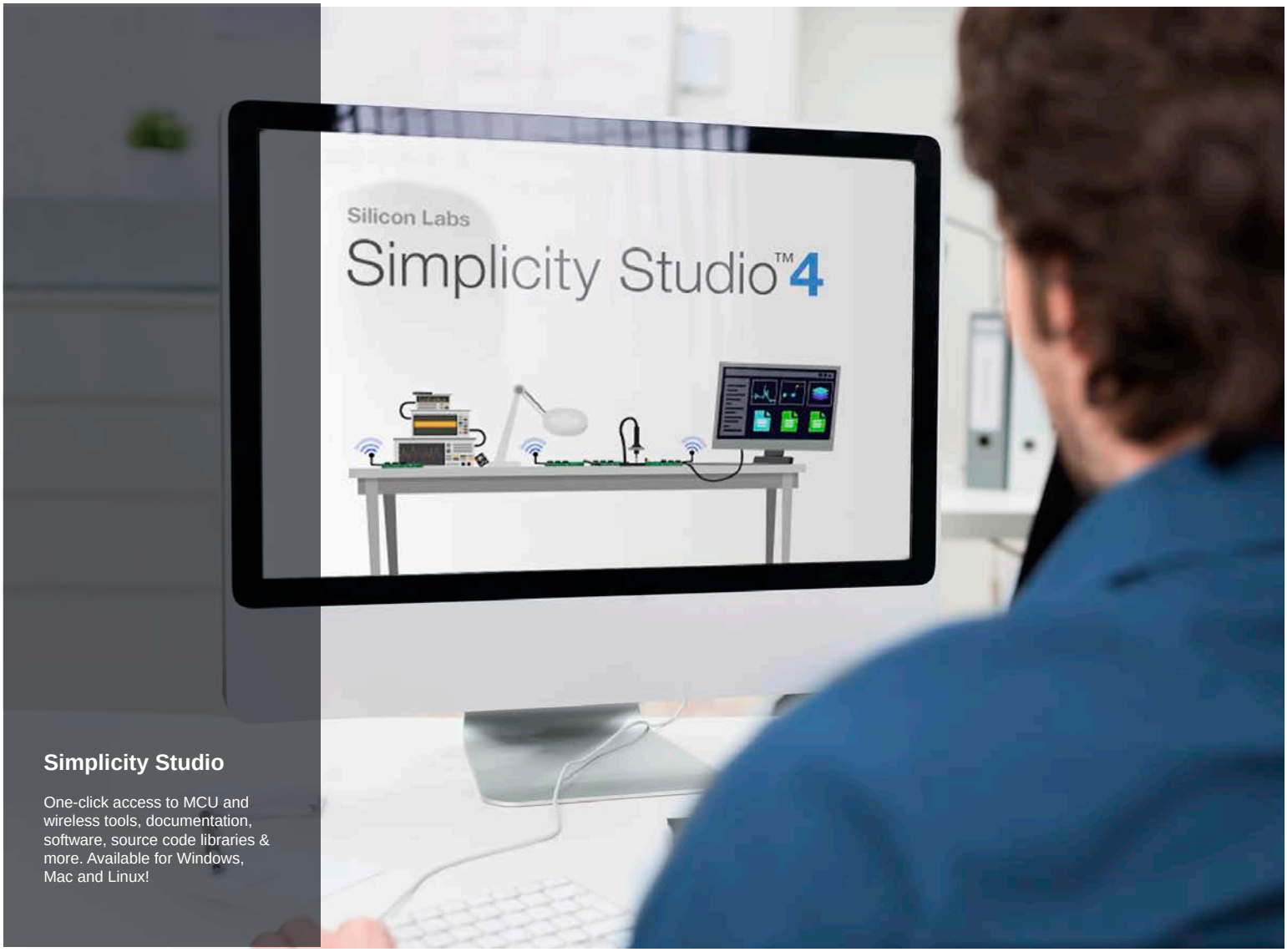
- Added mention of the UID to the front page.
- Updated some TBD values in the "1. Electrical Specifications" on page 8 section.
- Updated Power-On Reset (POR) Threshold maximum Falling Voltage on $V_{DD}$ specification in Table 1.3.
- Updated Reset Delay from non-POR source typical specification in Table 1.3.
- Removed $V_{DD}$ Ramp Time maximum specification in Table 1.3.
- Updated Flash Memory Erase Time specification and added Note 2 to Table 1.4.
- Updated maximum ADC DC performance specifications in Table 1.7.
- Updated minimum and maximum ADC offset error and slope error specifications in Table 1.7.
- Updated conditions on Internal Fast Settling Reference Output Voltage (Full Temperature and Supply Range) in Table 1.8.
- Added a new section "1.2.3. Port I/O Output Drive" on page 21.
- Updated pinout Figure 3.1, Figure 3.2, Figure 3.3, Table 3.1, Table 3.2, and Table 3.3 titles to the correct part numbers.
- Updated the Ordering Information ("4. Ordering Information" on page 42.) for Revision C devices.
- Added mention of the unique identifier to "8. Memory Organization" on page 52.
- Added unique identifier information to "11. Device Identification and Unique Identifier" on page 68.
- Updated device part numbers listed in Table 11.3, “DERIVID Register Bit Descriptions,” on page 70 to include the revision.
- Added "28. Revision-Specific Behavior" on page 303.

## Revision 0.7 to Revision 1.0

- Updated Digital Core, ADC, and Temperature Sensor electrical specifications information for -I devices.
- Updated -I part number information in "4. Ordering Information" on page 42.
- Replaced reference to AMX0P and AMX0N with ADC0MX in Table 21.1, “Port I/O Assignment for Analog Functions,” on page 186.
- Added a note to Table 1.13, “Absolute Maximum Ratings,” on page 22 and added a link to the Quality and Reliability Monitor Report.
- Added Operating Junction Temperature to Table 1.13, “Absolute Maximum Ratings,” on page 22.
- Updated all TBDs in "1. Electrical Specifications" on page 8.

## Revision 1.0 to Revision 1.1

- Updated Figure 27.1, “Watchdog Timer Block Diagram,” on page 298.
- Added section “27.1. Features” to the Watchdog Timer (WDT0) on page 298.
- Updated WDT behavior for all WDT commands.
- Added section “27.6. Synchronization” to the Watchdog Timer (WDT0) on page 300.




## Simplicity Studio

One-click access to MCU and wireless tools, documentation, software, source code libraries & more. Available for Windows, Mac and Linux!

**IoT Portfolio**
*www.silabs.com/IoT*

**SW/HW**
*www.silabs.com/simplicity*

**Quality**
*www.silabs.com/quality*

**Support and Community**
*community.silabs.com*

**Disclaimer**
Silicon Labs intends to provide customers with the latest, accurate, and in-depth documentation of all peripherals and modules available for system and software implementers using or intending to use the Silicon Labs products. Characterization data, available modules and peripherals, memory sizes and memory addresses refer to each specific device, and "Typical" parameters provided can and do vary in different applications. Application examples described herein are for illustrative purposes only. Silicon Labs reserves the right to make changes without further notice and limitation to product information, specifications, and descriptions herein, and does not give warranties as to the accuracy or completeness of the included information. Silicon Labs shall have no liability for the consequences of use of the information supplied herein. This document does not imply or express copyright licenses granted hereunder to design or fabricate any integrated circuits. The products are not designed or authorized to be used within any Life Support System without the specific written consent of Silicon Labs. A "Life Support System" is any product or system intended to support or sustain life and/or health, which, if it fails, can be reasonably expected to result in significant personal injury or death. Silicon Labs products are not designed or authorized for military applications. Silicon Labs products shall under no circumstances be used in weapons of mass destruction including (but not limited to) nuclear, biological or chemical weapons, or missiles capable of delivering such weapons.

**Trademark Information**
Silicon Laboratories Inc.® , Silicon Laboratories®, Silicon Labs®, SiLabs® and the Silicon Labs logo®, Bluegiga®, Bluegiga Logo®, Clockbuilder®, CMEMS®, DSPLL®, EFM®, EFM32®, EFR, Ember®, Energy Micro, Energy Micro logo and combinations thereof, "the world's most energy friendly microcontrollers", Ember®, EZLink®, EZRadio®, EZRadioPRO®, Gecko®, ISOmodem®, Micrium, Precision32®, ProSLIC®, Simplicity Studio®, SiPHY®, Telegesis, the Telegesis Logo®, USBXpress®, Zentri, Z-Wave, and others are trademarks or registered trademarks of Silicon Labs. ARM, CORTEX, Cortex-M3 and THUMB are trademarks or registered trademarks of ARM Holdings. Keil is a registered trademark of ARM Limited. All other products or brand names mentioned herein are trademarks of their respective holders.

SILICON LABS

**Silicon Laboratories Inc.**
**400 West Cesar Chavez**
**Austin, TX 78701**
**USA**

**http://www.silabs.com**