C8051F70x/71x

Mixed Signal ISP Flash MCU Family

Analog Peripherals
- **10-Bit ADC**
  - Up to 500 ksps
  - Up to 16 external single-ended inputs
  - VREF from on-chip VREF, external pin or V_{DD}
  - Internal or external start of conversion source
  - Built-in temperature sensor
- **Comparator**
  - Programmable hysteresis and response time
  - Configurable as interrupt or reset source

Capacitive Sense Interface
- 16-bit precision measurement
- Up to 32 channels
- Auto-scan and compare
- Auto-accumulate 4x, 8x, and 16x samples
- High conversion speed (40 µs per input)

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
- Superior performance to emulation systems using ICE-chips, target pods, and sockets
- Low cost, **complete** development kit

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

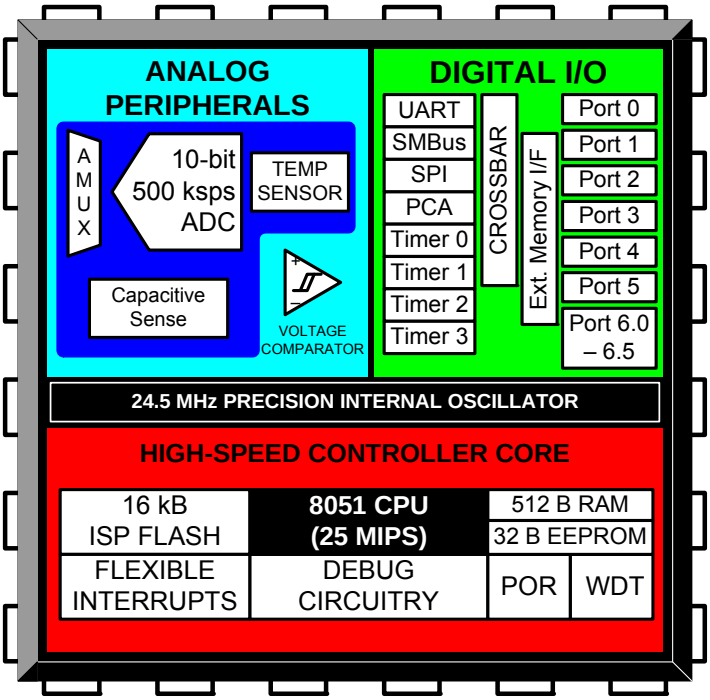
Memory
- 512 bytes internal data RAM (256 + 256)
- Up to 16 kB Flash; In-system programmable in 512-byte Sectors
- Up to 32-byte data EEPROM

Digital Peripherals
- Up to 54 Port I/O with high sink current
- Hardware enhanced UART, SMBus™ (I^2C compatible), and enhanced SPI™ serial ports
- Four general purpose 16-bit counter/timers
- 16-Bit programmable counter array (PCA) with 3 capture/compare modules and enhanced PWM functionality
- Real time clock mode using timer and crystal

Clock Sources
- 24.5 MHz ±2% Oscillator
  - Supports crystal-less UART operation
- External oscillator: Crystal, RC, C, or clock (1 or 2 pin modes)
- Can switch between clock sources on-the-fly; useful in power saving modes

Supply Voltage 1.8 to 3.6 V
- Built-in voltage supply monitor

64-Pin TQFP, 48-Pin TQFP, 48-Pin QFN
Temperature Range: –40 to +85 °C



Copyright © 2009 by Silicon Laboratories

This information applies to a product under development. Its characteristics and specifications are subject to change without notice.

Table of Contents

1. System Overview ... 17
2. Ordering Information ... 24
3. Pin Definitions ... 26
4. TQFP-64 Package Specifications ... 33
5. TQFP-48 Package Specifications ... 35
6. QFN-48 Package Specifications ... 37
7. Electrical Characteristics ... 39
7.1. Absolute Maximum Specifications ... 39
7.2. Electrical Characteristics ... 40
8. 10-Bit ADC (ADC0, C8051F700/2/4/6/8 and C8051F710/2/4 only) ... 46
8.1. Output Code Formatting ... 47
8.2. 8-Bit Mode ... 47
8.3. Modes of Operation ... 47
8.3.1. Starting a Conversion ... 47
8.3.2. Tracking Modes ... 48
8.3.3. Settling Time Requirements ... 49
8.4. Programmable Window Detector ... 53
8.4.1. Window Detector Example ... 55
8.5. ADC0 Analog Multiplexer ... 56
9. Temperature Sensor (C8051F700/2/4/6/8 and C8051F710/2/4 only) ... 58
9.1. Calibration ... 58
10. Voltage and Ground Reference Options ... 60
10.1. External Voltage References ... 61
10.2. Internal Voltage Reference Options ... 61
10.3. Analog Ground Reference ... 61
10.4. Temperature Sensor Enable ... 61
11. Voltage Regulator (REG0) ... 63
12. Comparator0 ... 65
12.1. Comparator Multiplexer ... 70
13. Capacitive Sense (CS0) ... 72
13.1. Configuring Port Pins as Capacitive Sense Inputs ... 73
13.2. Capacitive Sense Start-Of-Conversion Sources ... 73
13.3. Automatic Scanning ... 73
13.4. CS0 Comparator ... 74
13.5. CS0 Conversion Accumulator ... 75
13.6. Capacitive Sense Multiplexer ... 81
14. CIP-51 Microcontroller ... 83
14.1. Instruction Set ... 84
14.1.1. Instruction and CPU Timing ... 84
14.2. CIP-51 Register Descriptions ... 89
15. Memory Organization ... 93
15.1. Program Memory ... 94
15.1.1. MOVX Instruction and Program Memory ... 94

15.2. EEPROM Memory ... 94
15.3. Data Memory ... 94
15.3.1. Internal RAM ... 94
15.3.1.1. General Purpose Registers ... 95
15.3.1.2. Bit Addressable Locations ... 95
15.3.1.3. Stack ... 95
16. External Data Memory Interface and On-Chip XRAM ... 96
16.1. Accessing XRAM ... 96
16.1.1. 16-Bit MOVX Example ... 96
16.1.2. 8-Bit MOVX Example ... 96
16.2. Configuring the External Memory Interface ... 97
16.3. Port Configuration ... 97
16.4. Multiplexed and Non-multiplexed Selection ... 100
16.4.1. Multiplexed Configuration ... 100
16.4.2. Non-multiplexed Configuration ... 101
16.5. Memory Mode Selection ... 102
16.5.1. Internal XRAM Only ... 102
16.5.2. Split Mode without Bank Select ... 102
16.5.3. Split Mode with Bank Select ... 103
16.5.4. External Only ... 103
16.6. Timing ... 103
16.6.1. Non-Multiplexed Mode ... 105
16.6.1.1. 16-bit MOVX: EMI0CF[4:2] = 101, 110, or 111 ... 105
16.6.1.2. 8-bit MOVX without Bank Select: EMI0CF[4:2] = 101 or 111 ... 106
16.6.1.3. 8-bit MOVX with Bank Select: EMI0CF[4:2] = 110 ... 107
16.6.2. Multiplexed Mode ... 108
16.6.2.1. 16-bit MOVX: EMI0CF[4:2] = 001, 010, or 011 ... 108
16.6.2.2. 8-bit MOVX without Bank Select: EMI0CF[4:2] = 001 or 011 ... 109
16.6.2.3. 8-bit MOVX with Bank Select: EMI0CF[4:2] = 010 ... 110
17. In-System Device Identification ... 113
18. Special Function Registers ... 115
19. Interrupts ... 122
19.1. MCU Interrupt Sources and Vectors ... 123
19.1.1. Interrupt Priorities ... 123
19.1.2. Interrupt Latency ... 123
19.2. Interrupt Register Descriptions ... 124
19.3. INT0 and INT1 External Interrupts ... 131
20. Flash Memory ... 133
20.1. Programming The Flash Memory ... 133
20.1.1. Flash Lock and Key Functions ... 133
20.1.2. Flash Erase Procedure ... 133
20.1.3. Flash Write Procedure ... 134
20.2. Non-volatile Data Storage ... 134
20.3. Security Options ... 134
20.4. Flash Write and Erase Guidelines ... 135

20.4.1. VDD Maintenance and the VDD Monitor 136
20.4.2. PSWE Maintenance 136
20.4.3. System Clock 137
21. EEPROM 140
21.1. RAM Reads and Writes 140
21.2. Auto Increment 140
21.3. Interfacing with the EEPROM 140
21.4. EEPROM Security 141
22. Power Management Modes 145
22.1. Idle Mode 145
22.2. Stop Mode 146
22.3. Suspend Mode 146
23. Reset Sources 148
23.1. Power-On Reset 149
23.2. Power-Fail Reset / VDD Monitor 150
23.3. External Reset 151
23.4. Missing Clock Detector Reset 151
23.5. Comparator0 Reset 152
23.6. Watchdog Timer Reset 152
23.7. Flash Error Reset 152
23.8. Software Reset 152
24. Watchdog Timer 154
24.1. Enable/Reset WDT 154
24.2. Disable WDT 154
24.3. Disable WDT Lockout 154
24.4. Setting WDT Interval 154
25. Oscillators and Clock Selection 156
25.1. System Clock Selection 156
25.2. Programmable Internal High-Frequency (H-F) Oscillator 158
25.3. External Oscillator Drive Circuit 160
25.3.1. External Crystal Example 162
25.3.2. External RC Example 163
25.3.3. External Capacitor Example 164
26. Port Input/Output 165
26.1. Port I/O Modes of Operation 166
26.1.1. Port Pins Configured for Analog I/O 166
26.1.2. Port Pins Configured For Digital I/O 166
26.1.3. Interfacing Port I/O to 5 V Logic 167
26.1.4. Increasing Port I/O Drive Strength 168
26.2. Assigning Port I/O Pins to Analog and Digital Functions 168
26.2.1. Assigning Port I/O Pins to Analog Functions 168
26.2.2. Assigning Port I/O Pins to Digital Functions 169
26.2.3. Assigning Port I/O Pins to External Event Trigger Functions 169
26.3. Priority Crossbar Decoder 170
26.4. Port I/O Initialization 172

26.5. Port Match ... 175
26.6. Special Function Registers for Accessing and Configuring Port I/O ... 177
27. Cyclic Redundancy Check Unit (CRC0) ... 193
27.1. 16-bit CRC Algorithm ... 194
27.2. 32-bit CRC Algorithm ... 195
27.3. Preparing for a CRC Calculation ... 196
27.4. Performing a CRC Calculation ... 196
27.5. Accessing the CRC0 Result ... 196
27.6. CRC0 Bit Reverse Feature ... 199
28. SMBus ... 201
28.1. Supporting Documents ... 202
28.2. SMBus Configuration ... 202
28.3. SMBus Operation ... 202
28.3.1. Transmitter Vs. Receiver ... 203
28.3.2. Arbitration ... 203
28.3.3. Clock Low Extension ... 203
28.3.4. SCL Low Timeout ... 203
28.3.5. SCL High (SMBus Free) Timeout ... 204
28.4. Using the SMBus ... 204
28.4.1. SMBus Configuration Register ... 204
28.4.2. SMB0CN Control Register ... 208
28.4.2.1. Software ACK Generation ... 208
28.4.2.2. Hardware ACK Generation ... 208
28.4.3. Hardware Slave Address Recognition ... 210
28.4.4. Data Register ... 213
28.5. SMBus Transfer Modes ... 214
28.5.1. Write Sequence (Master) ... 214
28.5.2. Read Sequence (Master) ... 215
28.5.3. Write Sequence (Slave) ... 216
28.5.4. Read Sequence (Slave) ... 217
28.6. SMBus Status Decoding ... 217
29. Enhanced Serial Peripheral Interface (SPI0) ... 223
29.1. Signal Descriptions ... 224
29.1.1. Master Out, Slave In (MOSI) ... 224
29.1.2. Master In, Slave Out (MISO) ... 224
29.1.3. Serial Clock (SCK) ... 224
29.1.4. Slave Select (NSS) ... 224
29.2. SPI0 Master Mode Operation ... 225
29.3. SPI0 Slave Mode Operation ... 226
29.4. SPI0 Interrupt Sources ... 227
29.5. Serial Clock Phase and Polarity ... 227
29.6. SPI Special Function Registers ... 229
30. UART0 ... 236
30.1. Enhanced Baud Rate Generation ... 237
30.2. Operational Modes ... 238

30.2.1. 8-Bit UART 238
30.2.2. 9-Bit UART 239
30.3. Multiprocessor Communications 240
31. Timers 244
31.1. Timer 0 and Timer 1 246
31.1.1. Mode 0: 13-bit Counter/Timer 246
31.1.2. Mode 1: 16-bit Counter/Timer 247
31.1.3. Mode 2: 8-bit Counter/Timer with Auto-Reload 247
31.1.4. Mode 3: Two 8-bit Counter/Timers (Timer 0 Only) 248
31.2. Timer 2 254
31.2.1. 16-bit Timer with Auto-Reload 254
31.2.2. 8-bit Timers with Auto-Reload 255
31.3. Timer 3 260
31.3.1. 16-bit Timer with Auto-Reload 260
31.3.2. 8-bit Timers with Auto-Reload 261
32. Programmable Counter Array 266
32.1. PCA Counter/Timer 267
32.2. PCA0 Interrupt Sources 268
32.3. Capture/Compare Modules 268
32.3.1. Edge-triggered Capture Mode 270
32.3.2. Software Timer (Compare) Mode 271
32.3.3. High-Speed Output Mode 272
32.3.4. Frequency Output Mode 273
32.3.5. 8-bit, 9-bit, 10-bit and 11-bit Pulse Width Modulator Modes 274
32.3.5.1. 8-bit Pulse Width Modulator Mode 274
32.3.5.2. 9/10/11-bit Pulse Width Modulator Mode 275
32.3.6. 16-Bit Pulse Width Modulator Mode 277
32.4. Register Descriptions for PCA0 278
33. C2 Interface 284
33.1. C2 Interface Registers 284
33.2. C2CK Pin Sharing 287
Document Change List 288
Contact Information 290

List of Figures

1. System Overview
Figure 1.1. C8051F700/1 Block Diagram 18
Figure 1.2. C8051F702/3 Block Diagram 19
Figure 1.3. C8051F704/5 Block Diagram 20
Figure 1.4. C8051F706/07 Block Diagram 21
Figure 1.5. C8051F708/09/10/11 Block Diagram 22
Figure 1.6. C8051F712/13/14/15 Block Diagram 23
2. Ordering Information
3. Pin Definitions
Figure 3.1. C8051F7xx-GM TQFP64 Pinout Diagram (Top View) 30
Figure 3.2. C8051F7xx-GQ QFP48 Pinout Diagram (Top View) 31
Figure 3.3. C8051F7xx-GQ QFN48 Pinout Diagram (Top View) 32
4. TQFP-64 Package Specifications
Figure 4.1. TQFP-64 Package Drawing 33
Figure 4.2. TQFP-64 PCB Land Pattern 34
5. TQFP-48 Package Specifications
Figure 5.1. TQFP-48 Package Drawing 35
Figure 5.2. TQFP-48 PCB Land Pattern 36
6. QFN-48 Package Specifications
Figure 6.1. QFN-48 Package Drawing 37
Figure 6.2. QFN-48 PCB Land Pattern 38
7. Electrical Characteristics
8. 10-Bit ADC (ADC0, C8051F700/2/4/6/8 and C8051F710/2/4 only)
Figure 8.1. ADC0 Functional Block Diagram 46
Figure 8.2. 10-Bit ADC Track and Conversion Example Timing 48
Figure 8.3. ADC0 Equivalent Input Circuits 49
Figure 8.4. ADC Window Compare Example: Right-Justified Data 55
Figure 8.5. ADC Window Compare Example: Left-Justified Data 55
Figure 8.6. ADC0 Multiplexer Block Diagram 56
9. Temperature Sensor (C8051F700/2/4/6/8 and C8051F710/2/4 only)
Figure 9.1. Temperature Sensor Transfer Function 58
Figure 9.2. Temperature Sensor Error with 1-Point Calibration at 0 Celsius 59
10. Voltage and Ground Reference Options
Figure 10.1. Voltage Reference Functional Block Diagram 60
11. Voltage Regulator (REG0)
12. Comparator0
Figure 12.1. Comparator0 Functional Block Diagram 65
Figure 12.2. Comparator Hysteresis Plot 66
Figure 12.3. Comparator Input Multiplexer Block Diagram 70
13. Capacitive Sense (CS0)
Figure 13.1. CS0 Block Diagram 72
Figure 13.2. Auto-Scan Example 74
Figure 13.3. CS0 Multiplexer Block Diagram 81

14. CIP-51 Microcontroller
Figure 14.1. CIP-51 Block Diagram 83
15. Memory Organization
Figure 15.1. C8051F70x/71x Memory Map 93
Figure 15.2. Flash Program Memory Map 94
16. External Data Memory Interface and On-Chip XRAM
Figure 16.1. Multiplexed Configuration Example 100
Figure 16.2. Non-multiplexed Configuration Example 101
Figure 16.3. EMIF Operating Modes 102
Figure 16.4. Non-multiplexed 16-bit MOVX Timing 105
Figure 16.5. Non-multiplexed 8-bit MOVX without Bank Select Timing 106
Figure 16.6. Non-multiplexed 8-bit MOVX with Bank Select Timing 107
Figure 16.7. Multiplexed 16-bit MOVX Timing 108
Figure 16.8. Multiplexed 8-bit MOVX without Bank Select Timing 109
Figure 16.9. Multiplexed 8-bit MOVX with Bank Select Timing 110
17. In-System Device Identification
18. Special Function Registers
19. Interrupts
20. Flash Memory
21. EEPROM
Figure 21.1. EEPROM Block Diagram 140
22. Power Management Modes
23. Reset Sources
Figure 23.1. Reset Sources 148
Figure 23.2. Power-On and VDD Monitor Reset Timing 149
24. Watchdog Timer
25. Oscillators and Clock Selection
Figure 25.1. Oscillator Options 156
Figure 25.2. External 32.768 kHz Quartz Crystal Oscillator Connection Diagram 163
26. Port Input/Output
Figure 26.1. Port I/O Functional Block Diagram 165
Figure 26.2. Port I/O Cell Block Diagram 167
Figure 26.3. Port I/O Overdrive Current 167
Figure 26.4. Crossbar Priority Decoder—Possible Pin Assignments 170
Figure 26.5. Crossbar Priority Decoder Example 171
27. Cyclic Redundancy Check Unit (CRC0)
Figure 27.1. CRC0 Block Diagram 193
28. SMBus
Figure 28.1. SMBus Block Diagram 201
Figure 28.2. Typical SMBus Configuration 202
Figure 28.3. SMBus Transaction 203
Figure 28.4. Typical SMBus SCL Generation 205
Figure 28.5. Typical Master Write Sequence 214
Figure 28.6. Typical Master Read Sequence 215
Figure 28.7. Typical Slave Write Sequence 216

Figure 28.8. Typical Slave Read Sequence 217
29. Enhanced Serial Peripheral Interface (SPI0)
Figure 29.1. SPI Block Diagram 223
Figure 29.2. Multiple-Master Mode Connection Diagram 225
Figure 29.3. 3-Wire Single Master and 3-Wire Single Slave Mode Connection Diagram 226
Figure 29.4. 4-Wire Single Master Mode and 4-Wire Slave Mode Connection Diagram 226
Figure 29.5. Master Mode Data/Clock Timing 228
Figure 29.6. Slave Mode Data/Clock Timing (CKPHA = 0) 228
Figure 29.7. Slave Mode Data/Clock Timing (CKPHA = 1) 229
Figure 29.8. SPI Master Timing (CKPHA = 0) 233
Figure 29.9. SPI Master Timing (CKPHA = 1) 233
Figure 29.10. SPI Slave Timing (CKPHA = 0) 234
Figure 29.11. SPI Slave Timing (CKPHA = 1) 234
30. UART0
Figure 30.1. UART0 Block Diagram 236
Figure 30.2. UART0 Baud Rate Logic 237
Figure 30.3. UART Interconnect Diagram 238
Figure 30.4. 8-Bit UART Timing Diagram 238
Figure 30.5. 9-Bit UART Timing Diagram 239
Figure 30.6. UART Multi-Processor Mode Interconnect Diagram 240
31. Timers
Figure 31.1. T0 Mode 0 Block Diagram 247
Figure 31.2. T0 Mode 2 Block Diagram 248
Figure 31.3. T0 Mode 3 Block Diagram 249
Figure 31.4. Timer 2 16-Bit Mode Block Diagram 254
Figure 31.5. Timer 2 8-Bit Mode Block Diagram 255
Figure 31.7. Timer 3 16-Bit Mode Block Diagram 260
Figure 31.8. Timer 3 8-Bit Mode Block Diagram 261
Figure 31.9. Timer 3 Capture Mode Block Diagram 262
32. Programmable Counter Array
Figure 32.1. PCA Block Diagram 266
Figure 32.2. PCA Counter/Timer Block Diagram 267
Figure 32.3. PCA Interrupt Block Diagram 268
Figure 32.4. PCA Capture Mode Diagram 270
Figure 32.5. PCA Software Timer Mode Diagram 271
Figure 32.6. PCA High-Speed Output Mode Diagram 272
Figure 32.7. PCA Frequency Output Mode 273
Figure 32.8. PCA 8-Bit PWM Mode Diagram 275
Figure 32.9. PCA 9, 10 and 11-Bit PWM Mode Diagram 276
Figure 32.10. PCA 16-Bit PWM Mode 277
33. C2 Interface
Figure 33.1. Typical C2CK Pin Sharing 287

List of Tables

1. System Overview
2. Ordering Information
Table 2.1. Product Selection Guide 25
3. Pin Definitions
Table 3.1. Pin Definitions for the C8051F70x/71x 26
4. TQFP-64 Package Specifications
Table 4.1. TQFP-64 Package Dimensions 33
Table 4.2. TQFP-64 PCB Land Pattern Dimensions 34
5. TQFP-48 Package Specifications
Table 5.1. TQFP-48 Package Dimensions 35
Table 5.2. TQFP-48 PCB Land Pattern Dimensions 36
6. QFN-48 Package Specifications
Table 6.1. QFN-48 Package Dimensions 37
Table 6.2. QFN-48 PCB Land Pattern Dimensions 38
7. Electrical Characteristics
Table 7.1. Absolute Maximum Ratings 39
Table 7.2. Global Electrical Characteristics 40
Table 7.3. Port I/O DC Electrical Characteristics 41
Table 7.4. Reset Electrical Characteristics 41
Table 7.5. Internal Voltage Regulator Electrical Characteristics 42
Table 7.6. Flash Electrical Characteristics 42
Table 7.7. Internal High-Frequency Oscillator Electrical Characteristics 42
Table 7.8. Capacitive Sense Electrical Characteristics 42
Table 7.9. EEPROM Electrical Characteristics 43
Table 7.10. ADC0 Electrical Characteristics 43
Table 7.11. Power Management Electrical Characteristics 44
Table 7.12. Temperature Sensor Electrical Characteristics 44
Table 7.13. Voltage Reference Electrical Characteristics 44
Table 7.14. Comparator Electrical Characteristics 45
8. 10-Bit ADC (ADC0, C8051F700/2/4/6/8 and C8051F710/2/4 only)
9. Temperature Sensor (C8051F700/2/4/6/8 and C8051F710/2/4 only)
10. Voltage and Ground Reference Options
11. Voltage Regulator (REG0)
12. Comparator0
13. Capacitive Sense (CS0)
Table 13.1. Operation with Auto-scan and Accumulate 75
14. CIP-51 Microcontroller
Table 14.1. CIP-51 Instruction Set Summary 85
15. Memory Organization
16. External Data Memory Interface and On-Chip XRAM
Table 16.1. AC Parameters for External Memory Interface 111
Table 16.2. EMIF Pinout (C8051F70x/71x) 112
17. In-System Device Identification

18. Special Function Registers
Table 18.1. Special Function Register (SFR) Memory Map 116
Table 18.2. Special Function Registers 117
19. Interrupts
Table 19.1. Interrupt Summary 124
20. Flash Memory
Table 20.1. Flash Security Summary 135
21. EEPROM
22. Power Management Modes
23. Reset Sources
24. Watchdog Timer
25. Oscillators and Clock Selection
26. Port Input/Output
Table 26.1. Port I/O Assignment for Analog Functions 168
Table 26.2. Port I/O Assignment for Digital Functions 169
Table 26.3. Port I/O Assignment for External Event Trigger Functions 169
27. Cyclic Redundancy Check Unit (CRC0)
Table 27.1. Example 16-bit CRC Outputs 194
Table 27.2. Example 32-bit CRC Outputs 195
28. SMBus
Table 28.1. SMBus Clock Source Selection 205
Table 28.2. Minimum SDA Setup and Hold Times 206
Table 28.3. Sources for Hardware Changes to SMB0CN 210
Table 28.4. Hardware Address Recognition Examples (EHACK = 1) 211
Table 28.5. SMBus Status Decoding With Hardware ACK Generation Disabled (EHACK = 0) 218
Table 28.6. SMBus Status Decoding With Hardware ACK Generation Enabled (EHACK = 1) 220
29. Enhanced Serial Peripheral Interface (SPI0)
Table 29.1. SPI Slave Timing Parameters 235
30. UART0
Table 30.1. Timer Settings for Standard Baud Rates Using The Internal 24.5 MHz Oscillator 243
Table 30.2. Timer Settings for Standard Baud Rates Using an External 22.1184 MHz Oscillator 243
31. Timers
32. Programmable Counter Array
Table 32.1. PCA Timebase Input Options 267
Table 32.2. PCA0CPM and PCA0PWM Bit Settings for PCA Capture/Compare Modules 269
33. C2 Interface

List of Registers

SFR Definition 8.1. ADC0CF: ADC0 Configuration 50
SFR Definition 8.2. ADC0H: ADC0 Data Word MSB 51
SFR Definition 8.3. ADC0L: ADC0 Data Word LSB 51
SFR Definition 8.4. ADC0CN: ADC0 Control 52
SFR Definition 8.5. ADC0GTH: ADC0 Greater-Than Data High Byte 53
SFR Definition 8.6. ADC0GTL: ADC0 Greater-Than Data Low Byte 53
SFR Definition 8.7. ADC0LTH: ADC0 Less-Than Data High Byte 54
SFR Definition 8.8. ADC0LTL: ADC0 Less-Than Data Low Byte 54
SFR Definition 8.9. ADC0MX: AMUX0 Channel Select 57
SFR Definition 10.1. REF0CN: Voltage Reference Control 62
SFR Definition 11.1. REG0CN: Voltage Regulator Control 64
SFR Definition 12.1. CPT0CN: Comparator0 Control 68
SFR Definition 12.2. CPT0MD: Comparator0 Mode Selection 69
SFR Definition 12.3. CPT0MX: Comparator0 MUX Selection 71
SFR Definition 13.1. CS0CN: Capacitive Sense Control 76
SFR Definition 13.2. CS0CF: Capacitive Sense Configuration 77
SFR Definition 13.3. CS0DH: Capacitive Sense Data High Byte 78
SFR Definition 13.4. CS0DL: Capacitive Sense Data Low Byte 78
SFR Definition 13.5. CS0SS: Capacitive Sense Auto-Scan Start Channel 79
SFR Definition 13.6. CS0SE: Capacitive Sense Auto-Scan End Channel 79
SFR Definition 13.7. CS0THH: Capacitive Sense Comparator Threshold High Byte ... 80
SFR Definition 13.8. CS0THL: Capacitive Sense Comparator Threshold Low Byte 80
SFR Definition 13.9. CS0MX: Capacitive Sense Mux Channel Select 82
SFR Definition 14.1. DPL: Data Pointer Low Byte 89
SFR Definition 14.2. DPH: Data Pointer High Byte 89
SFR Definition 14.3. SP: Stack Pointer 90
SFR Definition 14.4. ACC: Accumulator 90
SFR Definition 14.5. B: B Register 91
SFR Definition 14.6. PSW: Program Status Word 92
SFR Definition 16.1. EMI0CN: External Memory Interface Control 98
SFR Definition 16.2. EMI0CF: External Memory Configuration 99
SFR Definition 16.3. EMI0TC: External Memory Timing Control 104
SFR Definition 17.1. HWID: Hardware Identification Byte 113
SFR Definition 17.2. DERIVID: Derivative Identification Byte 113
SFR Definition 17.3. REVID: Hardware Revision Identification Byte 114
SFR Definition 18.1. SFRPAGE: SFR Page 117
SFR Definition 19.1. IE: Interrupt Enable 125
SFR Definition 19.2. IP: Interrupt Priority 126
SFR Definition 19.3. EIE1: Extended Interrupt Enable 1 127
SFR Definition 19.4. EIE2: Extended Interrupt Enable 2 128
SFR Definition 19.5. EIP1: Extended Interrupt Priority 1 129
SFR Definition 19.6. EIP2: Extended Interrupt Priority 2 130
SFR Definition 19.7. IT01CF: INT0/INT1 Configuration 132

SFR Definition 20.1. PSCTL: Program Store R/W Control 138
SFR Definition 20.2. FLKEY: Flash Lock and Key 139
SFR Definition 21.1. EEADDR: EEPROM Byte Address 141
SFR Definition 21.2. EEDATA: EEPROM Byte Data 142
SFR Definition 21.3. EECNTL: EEPROM Control 143
SFR Definition 21.4. EEKEY: EEPROM Protect Key 144
SFR Definition 22.1. PCON: Power Control 147
SFR Definition 23.1. VDM0CN: VDD Monitor Control 151
SFR Definition 23.2. RSTSRC: Reset Source 153
SFR Definition 24.1. WDTCN: Watchdog Timer Control 155
SFR Definition 25.1. CLKSEL: Clock Select 157
SFR Definition 25.2. OSCICL: Internal H-F Oscillator Calibration 158
SFR Definition 25.3. OSCICN: Internal H-F Oscillator Control 159
SFR Definition 25.4. OSCXCN: External Oscillator Control 161
SFR Definition 26.1. XBR0: Port I/O Crossbar Register 0 173
SFR Definition 26.2. XBR1: Port I/O Crossbar Register 1 174
SFR Definition 26.3. P0MASK: Port 0 Mask Register 175
SFR Definition 26.4. P0MAT: Port 0 Match Register 176
SFR Definition 26.5. P1MASK: Port 1 Mask Register 176
SFR Definition 26.6. P1MAT: Port 1 Match Register 177
SFR Definition 26.7. P0: Port 0 178
SFR Definition 26.8. P0MDIN: Port 0 Input Mode 178
SFR Definition 26.9. P0MDOUT: Port 0 Output Mode 179
SFR Definition 26.10. P0SKIP: Port 0 Skip 179
SFR Definition 26.11. P0DRV: Port 0 Drive Strength 180
SFR Definition 26.12. P1: Port 1 180
SFR Definition 26.13. P1MDIN: Port 1 Input Mode 181
SFR Definition 26.14. P1MDOUT: Port 1 Output Mode 181
SFR Definition 26.15. P1SKIP: Port 1 Skip 182
SFR Definition 26.16. P1DRV: Port 1 Drive Strength 182
SFR Definition 26.17. P2: Port 2 183
SFR Definition 26.18. P2MDIN: Port 2 Input Mode 183
SFR Definition 26.19. P2MDOUT: Port 2 Output Mode 184
SFR Definition 26.20. P2SKIP: Port 2 Skip 184
SFR Definition 26.21. P2DRV: Port 2 Drive Strength 185
SFR Definition 26.22. P3: Port 3 185
SFR Definition 26.23. P3MDIN: Port 3 Input Mode 186
SFR Definition 26.24. P3MDOUT: Port 3 Output Mode 186
SFR Definition 26.25. P3DRV: Port 3 Drive Strength 187
SFR Definition 26.26. P4: Port 4 187
SFR Definition 26.27. P4MDIN: Port 4 Input Mode 188
SFR Definition 26.28. P4MDOUT: Port 4 Output Mode 188
SFR Definition 26.29. P4DRV: Port 4 Drive Strength 189
SFR Definition 26.30. P5: Port 5 189
SFR Definition 26.31. P5MDIN: Port 5 Input Mode 190

SFR Definition 26.32. P5MDOUT: Port 5 Output Mode 190
SFR Definition 26.33. P5DRV: Port 5 Drive Strength 191
SFR Definition 26.34. P6: Port 6 191
SFR Definition 26.35. P6MDOUT: Port 6 Output Mode 192
SFR Definition 26.36. P6DRV: Port 6 Drive Strength 192
SFR Definition 27.1. CRC0CN: CRC0 Control 197
SFR Definition 27.2. CRC0IN: CRC Data Input 198
SFR Definition 27.3. CRC0DATA: CRC Data Output 198
SFR Definition 27.4. CRC0AUTO: CRC Automatic Control 199
SFR Definition 27.5. CRC0CNT: CRC Automatic Flash Sector Count 199
SFR Definition 27.6. CRC0FLIP: CRC Bit Flip 200
SFR Definition 28.1. SMB0CF: SMBus Clock/Configuration 207
SFR Definition 28.2. SMB0CN: SMBus Control 209
SFR Definition 28.3. SMB0ADR: SMBus Slave Address 211
SFR Definition 28.4. SMB0ADM: SMBus Slave Address Mask 212
SFR Definition 28.5. SMB0DAT: SMBus Data 213
SFR Definition 29.1. SPI0CFG: SPI0 Configuration 230
SFR Definition 29.2. SPI0CN: SPI0 Control 231
SFR Definition 29.3. SPI0CKR: SPI0 Clock Rate 232
SFR Definition 29.4. SPI0DAT: SPI0 Data 232
SFR Definition 30.1. SCON0: Serial Port 0 Control 241
SFR Definition 30.2. SBUF0: Serial (UART0) Port Data Buffer 242
SFR Definition 31.1. CKCON: Clock Control 245
SFR Definition 31.2. TCON: Timer Control 250
SFR Definition 31.3. TMOD: Timer Mode 251
SFR Definition 31.4. TL0: Timer 0 Low Byte 252
SFR Definition 31.5. TL1: Timer 1 Low Byte 252
SFR Definition 31.6. TH0: Timer 0 High Byte 253
SFR Definition 31.7. TH1: Timer 1 High Byte 253
SFR Definition 31.8. TMR2CN: Timer 2 Control 257
SFR Definition 31.9. TMR2RLL: Timer 2 Reload Register Low Byte 258
SFR Definition 31.10. TMR2RLH: Timer 2 Reload Register High Byte 258
SFR Definition 31.11. TMR2L: Timer 2 Low Byte 259
SFR Definition 31.12. TMR2H Timer 2 High Byte 259
SFR Definition 31.13. TMR3CN: Timer 3 Control 263
SFR Definition 31.14. TMR3RLL: Timer 3 Reload Register Low Byte 264
SFR Definition 31.15. TMR3RLH: Timer 3 Reload Register High Byte 264
SFR Definition 31.16. TMR3L: Timer 3 Low Byte 265
SFR Definition 31.17. TMR3H Timer 3 High Byte 265
SFR Definition 32.1. PCA0CN: PCA Control 278
SFR Definition 32.2. PCA0MD: PCA Mode 279
SFR Definition 32.3. PCA0PWM: PCA PWM Configuration 280
SFR Definition 32.4. PCA0CPMn: PCA Capture/Compare Mode 281
SFR Definition 32.5. PCA0L: PCA Counter/Timer Low Byte 282
SFR Definition 32.6. PCA0H: PCA Counter/Timer High Byte 282

SFR Definition 32.7. PCA0CPLn: PCA Capture Module Low Byte 283
SFR Definition 32.8. PCA0CPHn: PCA Capture Module High Byte 283
C2 Register Definition 33.1. C2ADD: C2 Address 284
C2 Register Definition 33.2. DEVICEID: C2 Device ID 285
C2 Register Definition 33.3. REVID: C2 Revision ID 285
C2 Register Definition 33.4. FPCTL: C2 Flash Programming Control 286
C2 Register Definition 33.5. FPDAT: C2 Flash Programming Data 286

1. System Overview

C8051F70x/71x devices are fully integrated, mixed-signal, system-on-a-chip MCUs. Highlighted features are listed below. Refer to Table 2.1 for specific product feature selection and part ordering numbers.

- High-speed pipelined 8051-compatible microcontroller core (up to 25 MIPS)
- In-system, full-speed, non-intrusive debug interface (on-chip)
- 10-bit 500 ksps Single-ended ADC with 16-channel analog multiplexer and integrated temperature sensor
- Precision calibrated 24.5 MHz internal oscillator
- 16Kb of on-chip Flash memory
- 512 bytes of on-chip RAM
- SMBus/I^2C, Enhanced UART, and Enhanced SPI serial interfaces implemented in hardware
- Four general-purpose 16-bit timers
- Capacitive Sense interface with 32 input channels
- Programmable Counter/Timer Array (PCA) with three capture/compare modules
- On-chip internal voltage reference
- On-chip Watchdog timer
- On-chip Power-On Reset and Supply Monitor
- On-chip Voltage Comparator
- 54 general purpose I/O

With on-chip power-on reset, V_{DD} monitor, watchdog timer, and clock oscillator, the C8051F70x/71x devices are truly stand-alone, system-on-a-chip solutions. The Flash memory can be reprogrammed even in-circuit, providing non-volatile data storage, and also allowing field upgrades of the 8051 firmware. User software has complete control of all peripherals, and may individually shut down any or all peripherals for power savings.

The C8051F70x/71x processors include Silicon Laboratories' 2-Wire C2 Debug and Programming interface, which 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 of memory, viewing and modification of special function registers, setting breakpoints, single stepping, and run and halt commands. All analog and digital peripherals are fully functional while debugging using C2. The two C2 interface pins can be shared with user functions, allowing in-system debugging without occupying package pins.

Each device is specified for 1.8–3.6 V operation over the industrial temperature range (–45 to +85 °C). An internal LDO is used to supply the processor core voltage at 1.8 V. The Port I/O and $\overline{RST}$ pins are tolerant of input signals up to 2 V above the V_{DD} supply. See Table 2.1 for ordering information. Block diagrams of the devices in the C8051F70x/71x family are shown in Figure 1.1.

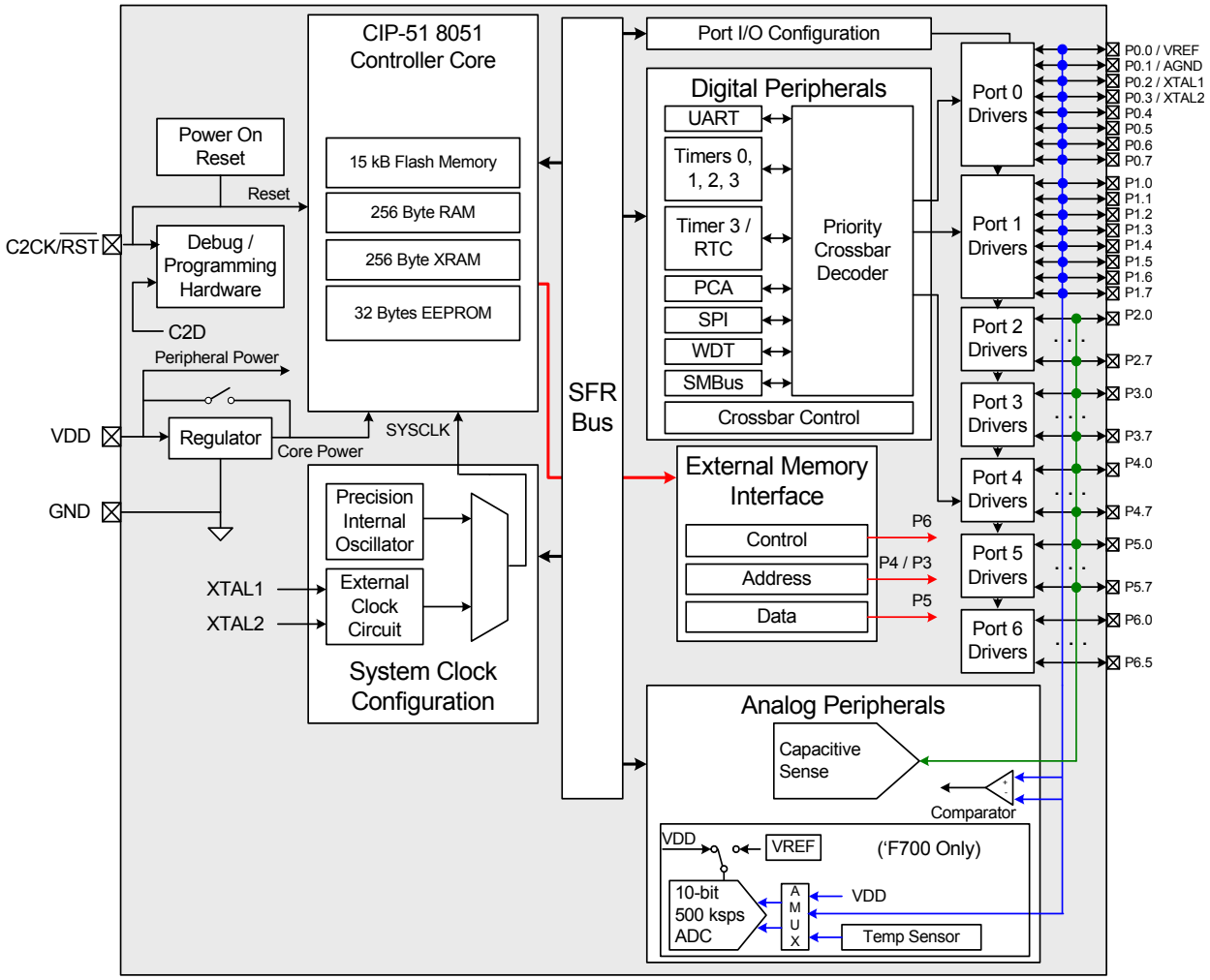


Figure 1.1. C8051F700/1 Block Diagram

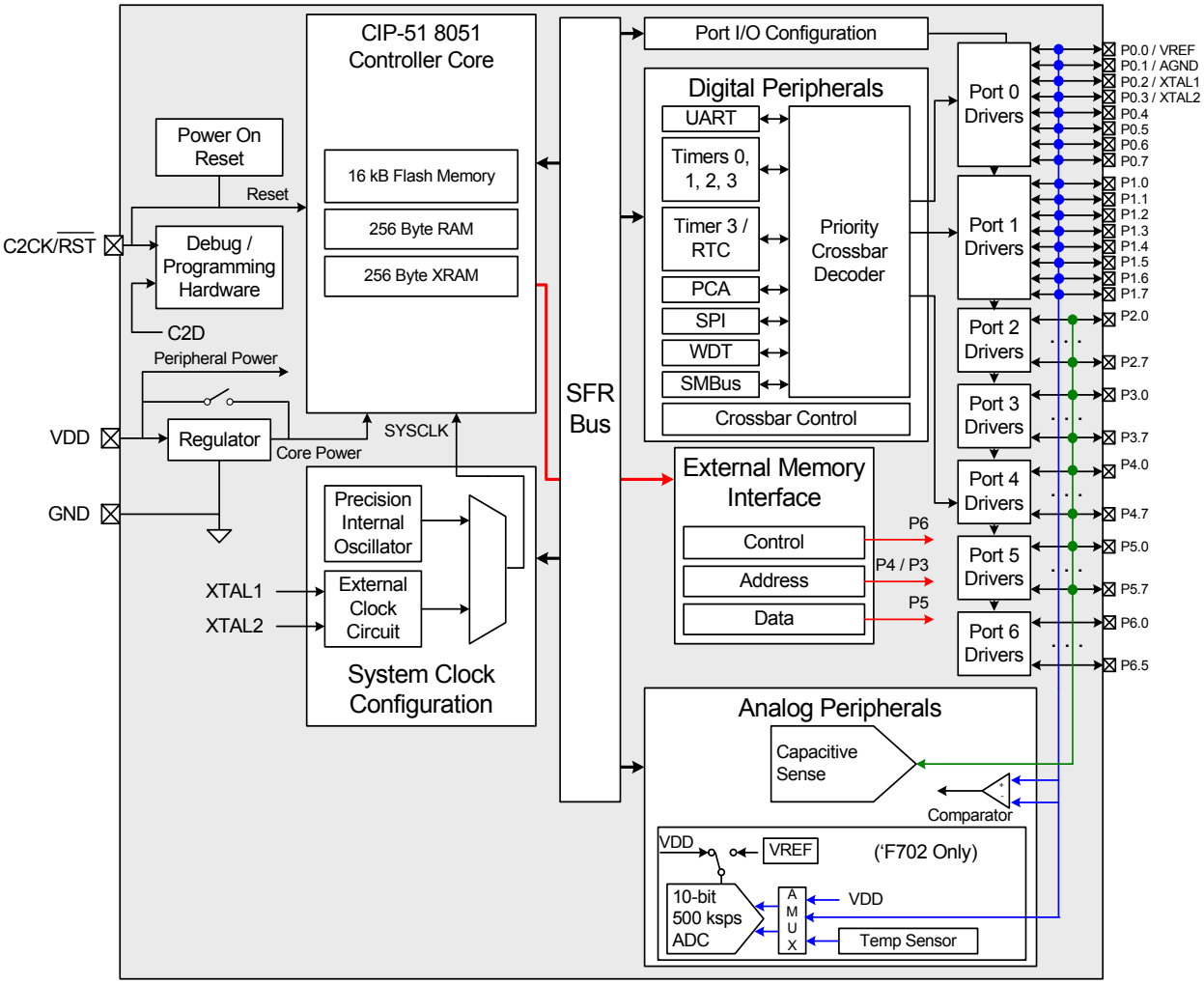


Figure 1.2. C8051F702/3 Block Diagram

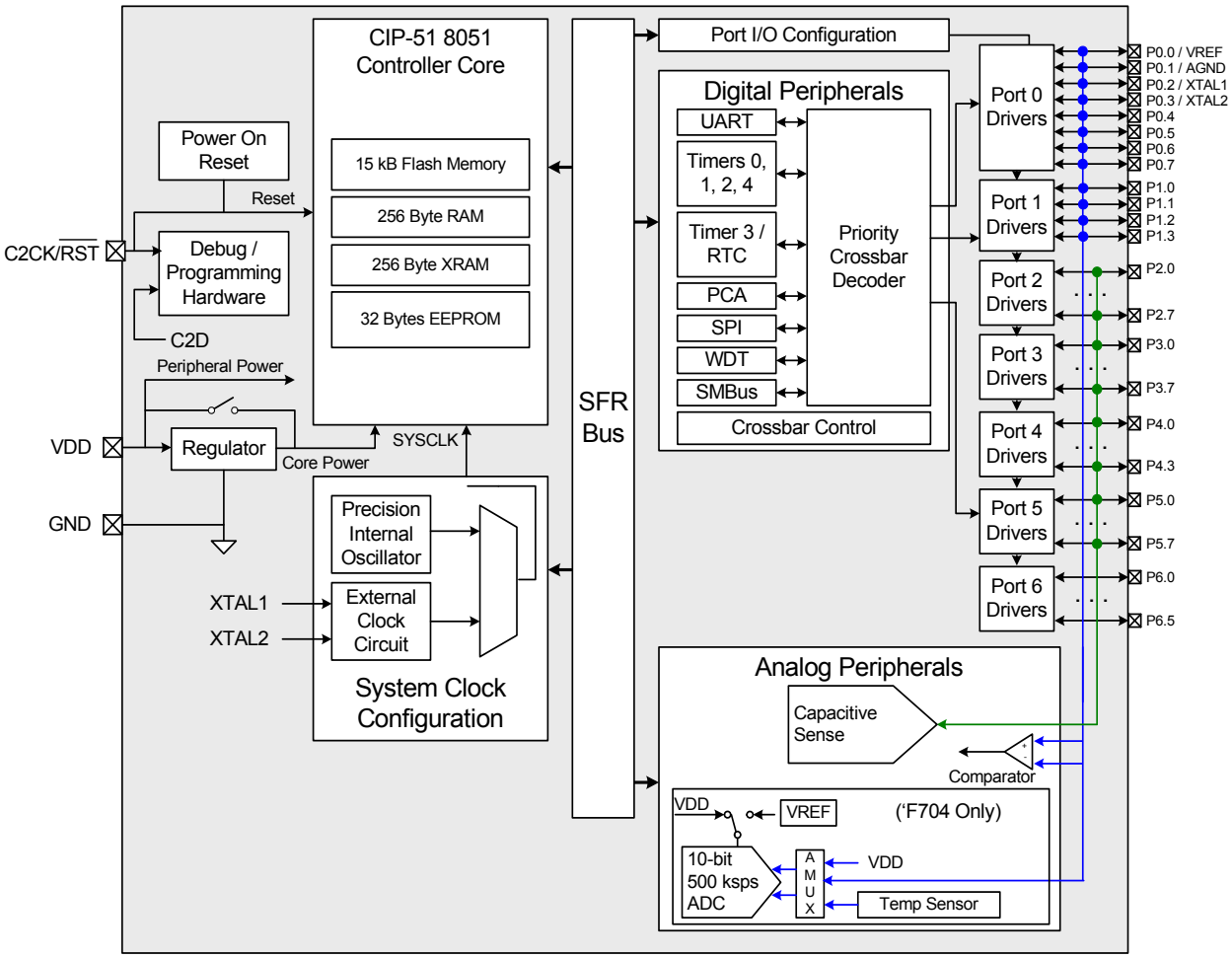


Figure 1.3. C8051F704/5 Block Diagram

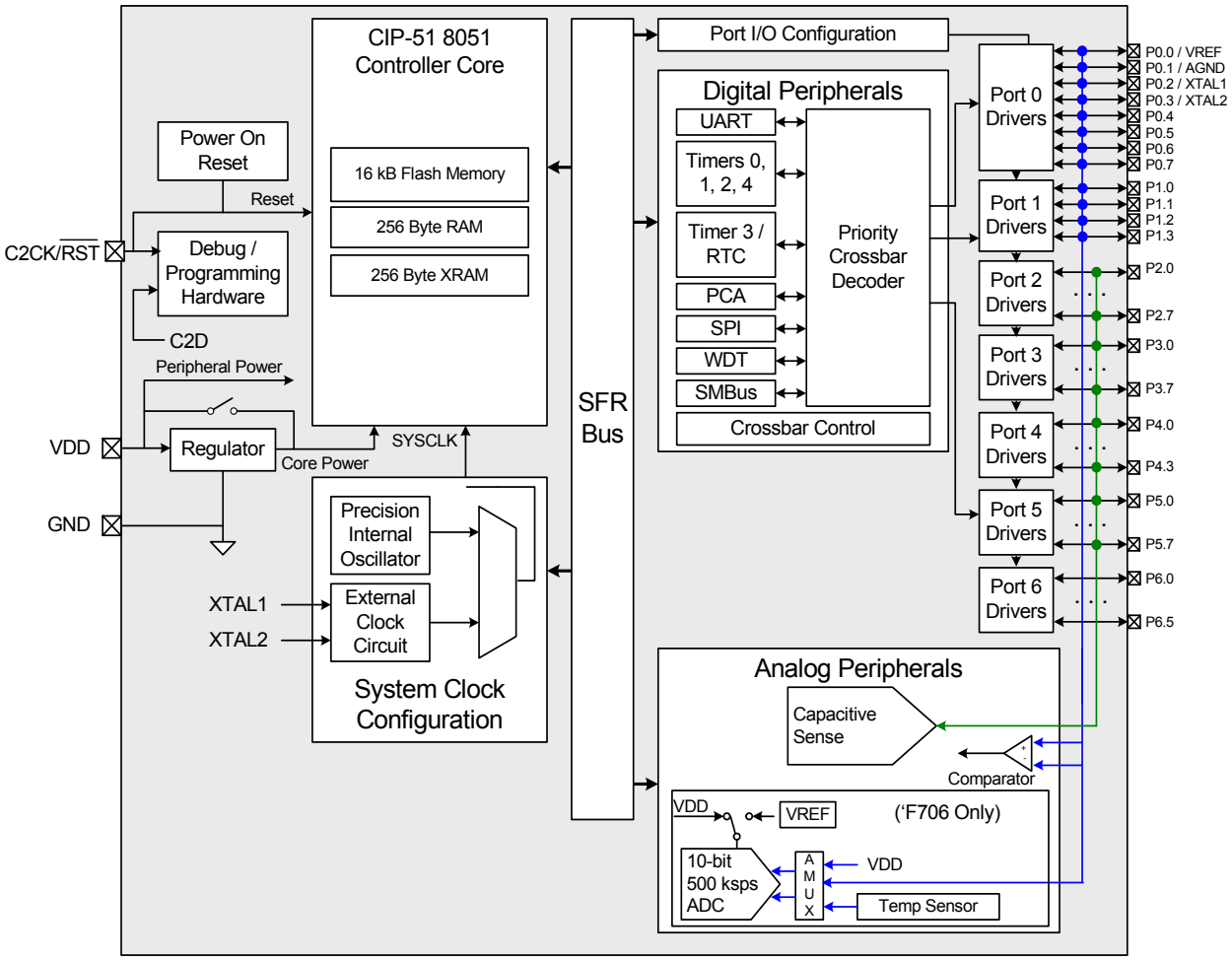


Figure 1.4. C8051F706/07 Block Diagram

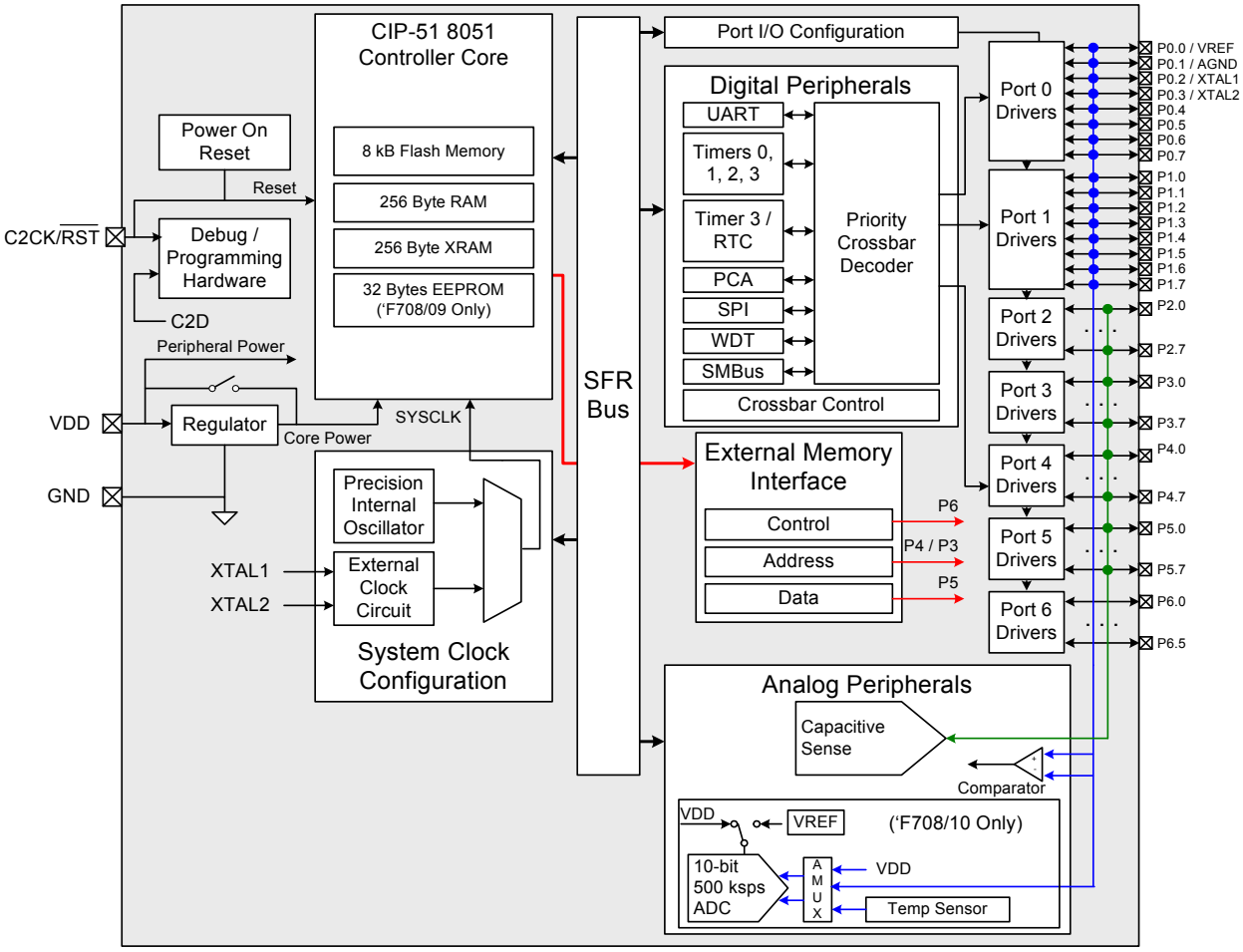


Figure 1.5. C8051F708/09/10/11 Block Diagram

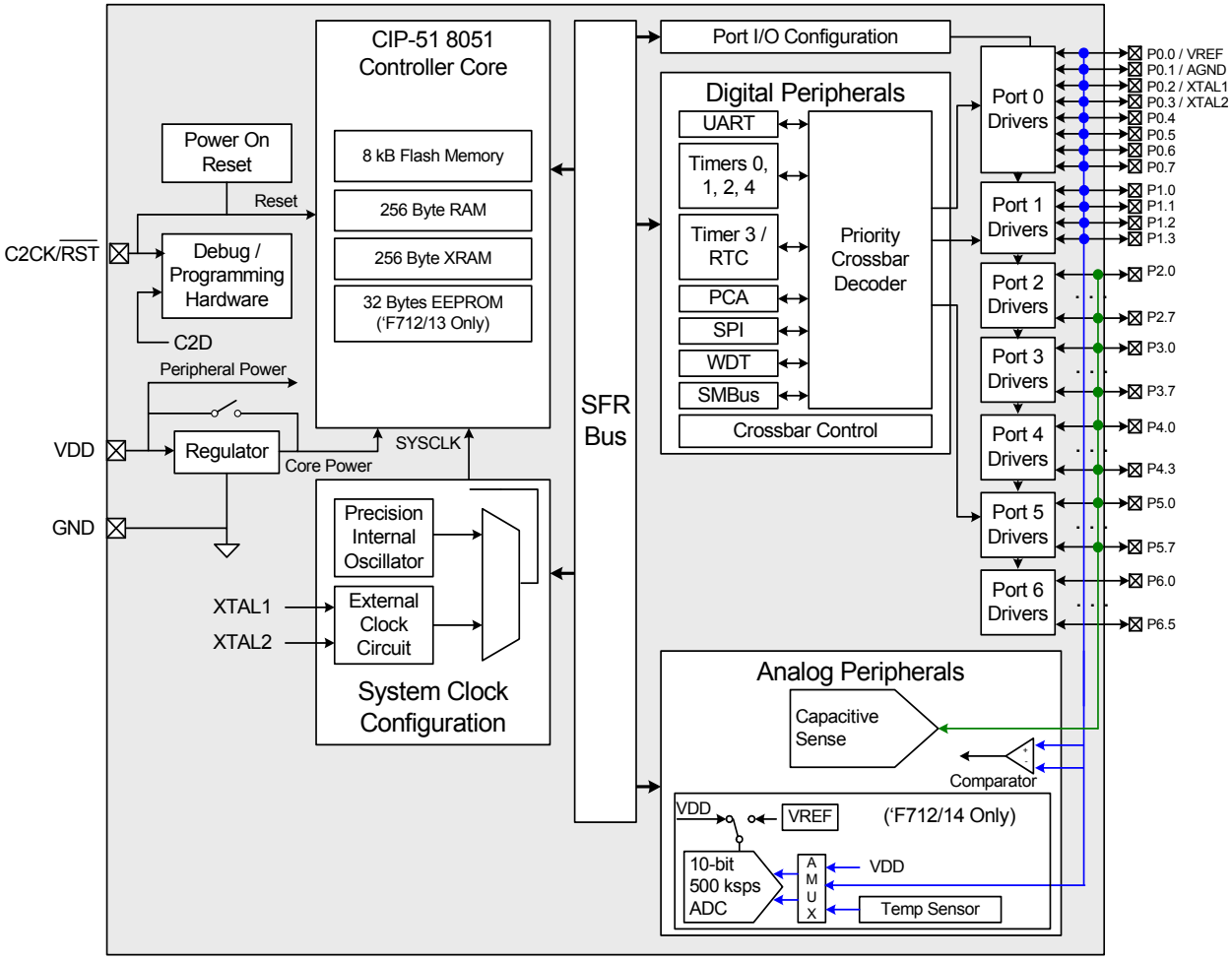


Figure 1.6. C8051F712/13/14/15 Block Diagram

2. Ordering Information

All C8051F70x/71x devices have the following features:

- 25 MIPS (Peak)
- Calibrated Internal Oscillator
- SMBus/I2C
- UART
- Programmable counter array (3 channels)
- 4 Timers (16-bit)
- 1 Comparator
- Lead-Free (RoHS compliant) package
- 512 bytes RAM

In addition to the features listed above, each device in the C8051F70x/71x family has a set of features that vary across the product line. See Table 2.1 for a complete list of the unique feature sets for each device in the family.

Table 2.1. Product Selection Guide

Part Number	Digital Port I/Os	Capacitive Sense Channels	Flash Memory (kB)	EEPROM (Bytes)	External Memory Interface	10-bit 500 ksps ADC	ADC Channels	Temperature Sensor	Package (RoHS)
C8051F700-GQ	54	32	15	32	Y	Y	16	Y	TQFP-64
C8051F701-GQ	54	32	15	32	Y	N	-	-	TQFP-64
C8051F702-GQ	54	32	16	-	Y	Y	16	Y	TQFP-64
C8051F703-GQ	54	32	16	-	Y	N	-	-	TQFP-64
C8051F704-GQ	39	24	15	32	N	Y	16	Y	TQFP-48
C8051F704-GM	39	24	15	32	N	Y	16	Y	QFN-48
C8051F705-GQ	39	24	15	32	N	N	-	-	TQFP-48
C8051F705-GM	39	24	15	32	N	N	-	-	QFN-48
C8051F706-GQ	39	24	16	-	N	Y	16	Y	TQFP-48
C8051F706-GM	39	24	16	-	N	Y	16	Y	QFN-48
C8051F707-GQ	39	24	16	-	N	N	-	-	TQFP-48
C8051F707-GM	39	24	16	-	N	N	-	-	QFN-48
C8051F708-GQ	54	32	8	32	Y	Y	16	Y	TQFP-64
C8051F709-GQ	54	32	8	32	Y	N	-	-	TQFP-64
C8051F710-GQ	54	32	8	-	Y	Y	16	Y	TQFP-64
C8051F711-GQ	54	32	8	-	Y	N	-	-	TQFP-64
C8051F712-GQ	39	24	8	32	N	Y	16	Y	TQFP-48
C8051F712-GM	39	24	8	32	N	Y	16	Y	QFN-48
C8051F713-GQ	39	24	8	32	N	N	-	-	TQFP-48
C8051F713-GM	39	24	8	32	N	N	-	-	QFN-48
C8051F714-GQ	39	24	8	-	N	Y	16	Y	TQFP-48
C8051F714-GM	39	24	8	-	N	Y	16	Y	QFN-48
C8051F715-GQ	39	24	8	-	N	N	-	-	TQFP-48
C8051F715-GM	39	24	8	-	N	N	-	-	QFN-48
Lead finish material on all devices is 100% matte tin (Sn).									

3. Pin Definitions

Table 3.1. Pin Definitions for the C8051F70x/71x

Name	64-Pin Packages	48-Pin Packages	Type	Description
V_{DD}	8, 24, 41, 57	8, 20, 44		Power Supply Voltage.
GND	9, 25, 40, 56	9, 21, 30, 43		Ground.
$\overline{RST}$ / C2CK	58	45	D I/O D I/O	Device Reset. Open-drain output of internal POR or V_{DD} monitor. Clock signal for the C2 Debug Interface.
C2D	59	46	D I/O or A In D I/O	Bi-directional data signal for the C2 Debug Interface.
P0.0 / VREF	55	42	D I/O or A In A In	Port 0.0. External VREF input.
P0.1/ AGND	54	41	D I/O or A In	Port 0.1. External AGND input.
P0.2 / XTAL1	53	40	D I/O or A In A In	Port 0.2. External Clock Pin. This pin can be used for crystal clock mode.
P0.3 / XTAL2	52	39	D I/O or A In A I/O or D In	Port 0.3. External Clock Pin. This pin can be used for RC, crystal, and CMOS clock modes.
P0.4	51	38	D I/O or A In	Port 0.4.
P0.5	50	37	D I/O or A In	Port 0.5.
P0.6	49	36	D I/O or A I	Port 0.6.

Table 3.1. Pin Definitions for the C8051F70x/71x (Continued)

Name	64-Pin Packages	48-Pin Packages	Type	Description
P0.7	48	35	D I/O or A In	Port 0.7.
P1.0	47	34	D I/O or A In	Port 1.0.
P1.1	46	33	D I/O or A In	Port 1.1.
P1.2	45	32	D I/O or A In	Port 1.2.
P1.3	44	31	D I/O or A In	Port 1.3.
P1.4	43		D I/O or A In	Port 1.4.
P1.5	42		D I/O or A In	Port 1.5.
P1.6	39		D I/O or A In	Port 1.6.
P1.7	38		D I/O or A In	Port 1.7.
P2.0	37	29	D I/O or A In	Port 2.0. Cap Sense input pin 1.
P2.1	36	28	D I/O or A In	Port 2.1. Cap Sense input pin 2.
P2.2	35	27	D I/O or A In	Port 2.2. Cap Sense input pin 3.
P2.3	34	26	D I/O or A In	Port 2.3. Cap Sense input pin 4.
P2.4	33	25	D I/O or A In	Port 2.4. Cap Sense input pin 5.
P2.5	32	24	D I/O or A In	Port 2.5. Cap Sense input pin 6.
P2.6	31	23	D I/O or A In	Port 2.6. Cap Sense input pin 7.
P2.7	30	22	D I/O or A In	Port 2.7. Cap Sense input pin 8.

Table 3.1. Pin Definitions for the C8051F70x/71x (Continued)

Name	64-Pin Packages	48-Pin Packages	Type	Description
P3.0	29		D I/O or A In	Port 3.0. Cap Sense input pin 9.
P3.1	28		D I/O or A In	Port 3.1. Cap Sense input pin 10.
P3.2	27		D I/O or A In	Port 3.2. Cap Sense input pin 11.
P3.3	26		D I/O or A In	Port 3.3. Cap Sense input pin 12.
P3.4	23	19	D I/O or A In	Port 3.4. Cap Sense input pin 13.
P3.5	22	18	D I/O or A In	Port 3.5. Cap Sense input pin 14.
P3.6	21	17	D I/O or A In	Port 3.6. Cap Sense input pin 15.
P3.7	20	16	D I/O or A In	Port 3.7. Cap Sense input pin 16.
P4.0	19	15	D I/O or A In	Port 4.0. Cap Sense input pin 17.
P4.1	18	14	D I/O or A In	Port 4.1. Cap Sense input pin 18.
P4.2	17	13	D I/O or A In	Port 4.2. Cap Sense input pin 19.
P4.3	16	12	D I/O or A In	Port 4.3. Cap Sense input pin 20.
P4.4	15		D I/O or A In	Port 4.4. Cap Sense input pin 21.
P4.5	14		D I/O or A In	Port 4.5. Cap Sense input pin 22.
P4.6	13		D I/O or A In	Port 4.6. Cap Sense input pin 23.
P4.7	12		D I/O or A In	Port 4.7. Cap Sense input pin 24.
P5.0	11	11	D I/O or A In	Port 5.0. Cap Sense input pin 25.

Table 3.1. Pin Definitions for the C8051F70x/71x (Continued)

Name	64-Pin Packages	48-Pin Packages	Type	Description
P5.1	10	10	D I/O or A In	Port 5.0. Cap Sense input pin 26.
P5.2	7	7	D I/O or A In	Port 5.2. Cap Sense input pin 27
P5.3	6	6	D I/O or A In	Port 5.3. Cap Sense input pin 28.
P5.4	5	5	D I/O or A In	Port 5.4. Cap Sense input pin 29.
P5.5	4	4	D I/O or A In	Port 5.5. Cap Sense input pin 30.
P5.6	3	3	D I/O or A In	Port 5.6. Cap Sense input pin 31.
P5.7	2	2	D I/O or A In	Port 5.7. Cap Sense input pin 32.
P6.0	1		D I/O	Port 6.0.
P6.1	64		D I/O	Port 6.1.
P6.2	63		D I/O	Port 6.2.
P6.3	62	1	D I/O	Port 6.3.
P6.4	61	48	D I/O	Port 6.4.
P6.5	60	47	D I/O	Port 6.5.

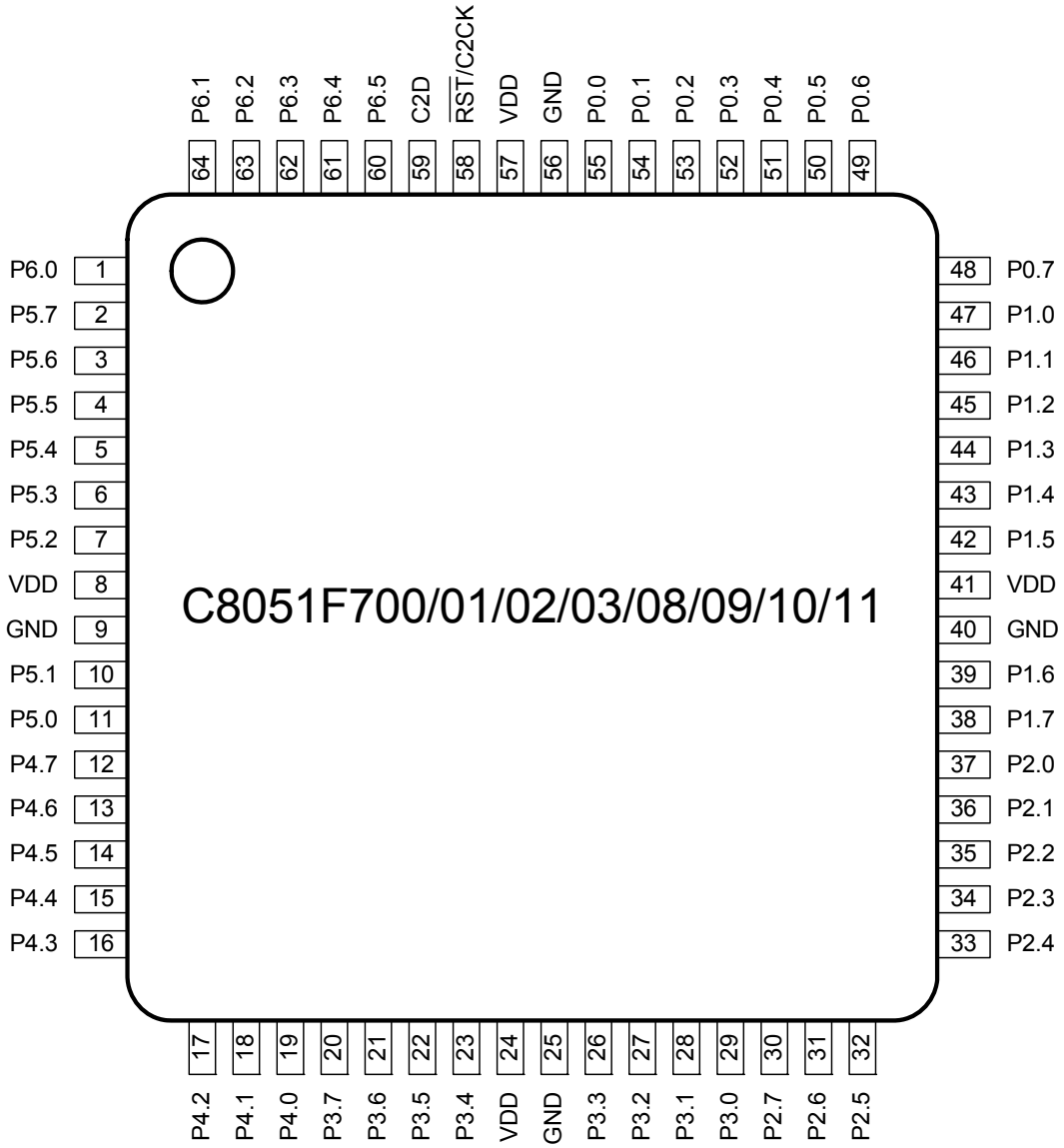


Figure 3.1. C8051F7xx-GM TQFP64 Pinout Diagram (Top View)

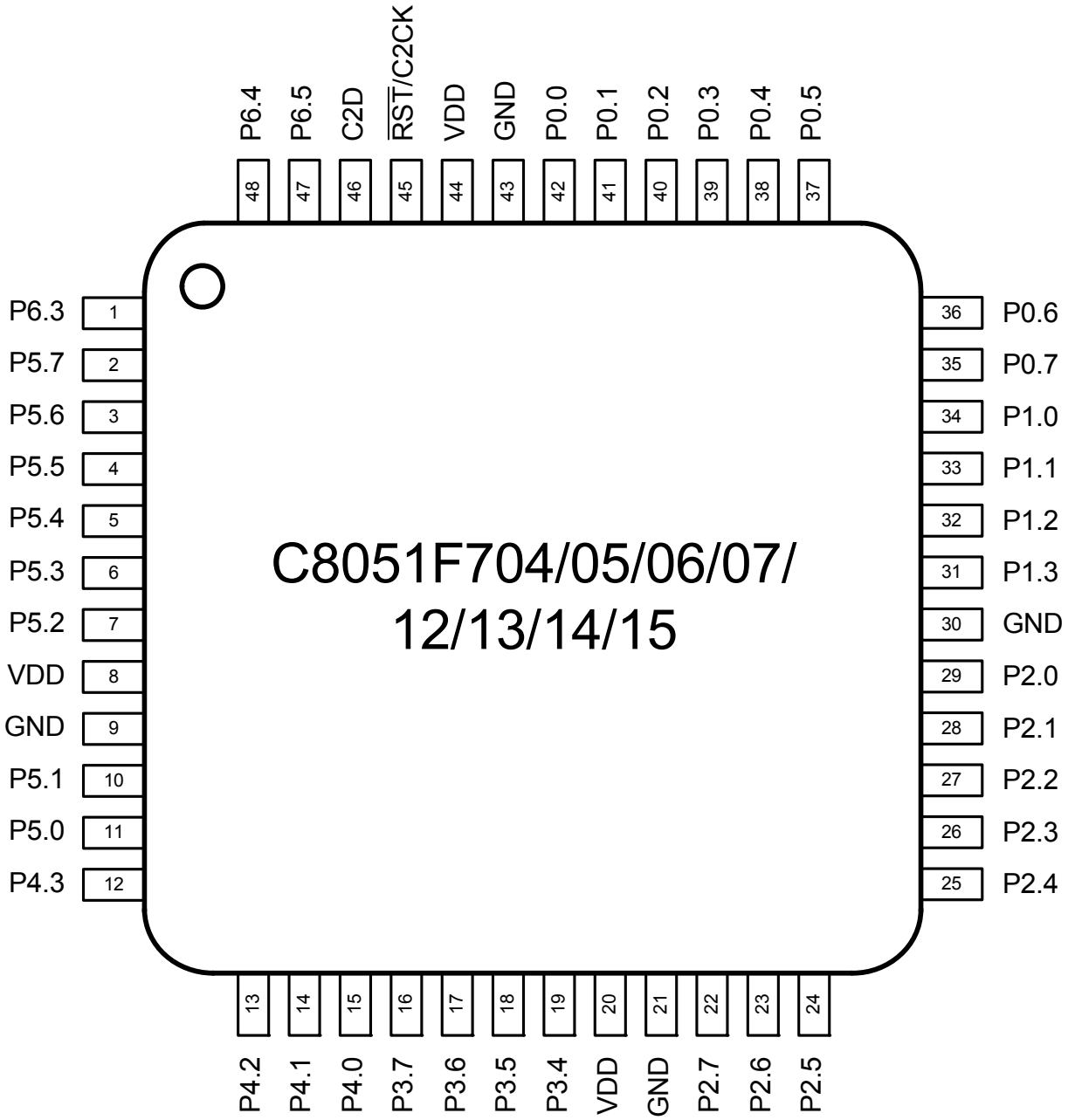


Figure 3.2. C8051F7xx-GQ QFP48 Pinout Diagram (Top View)

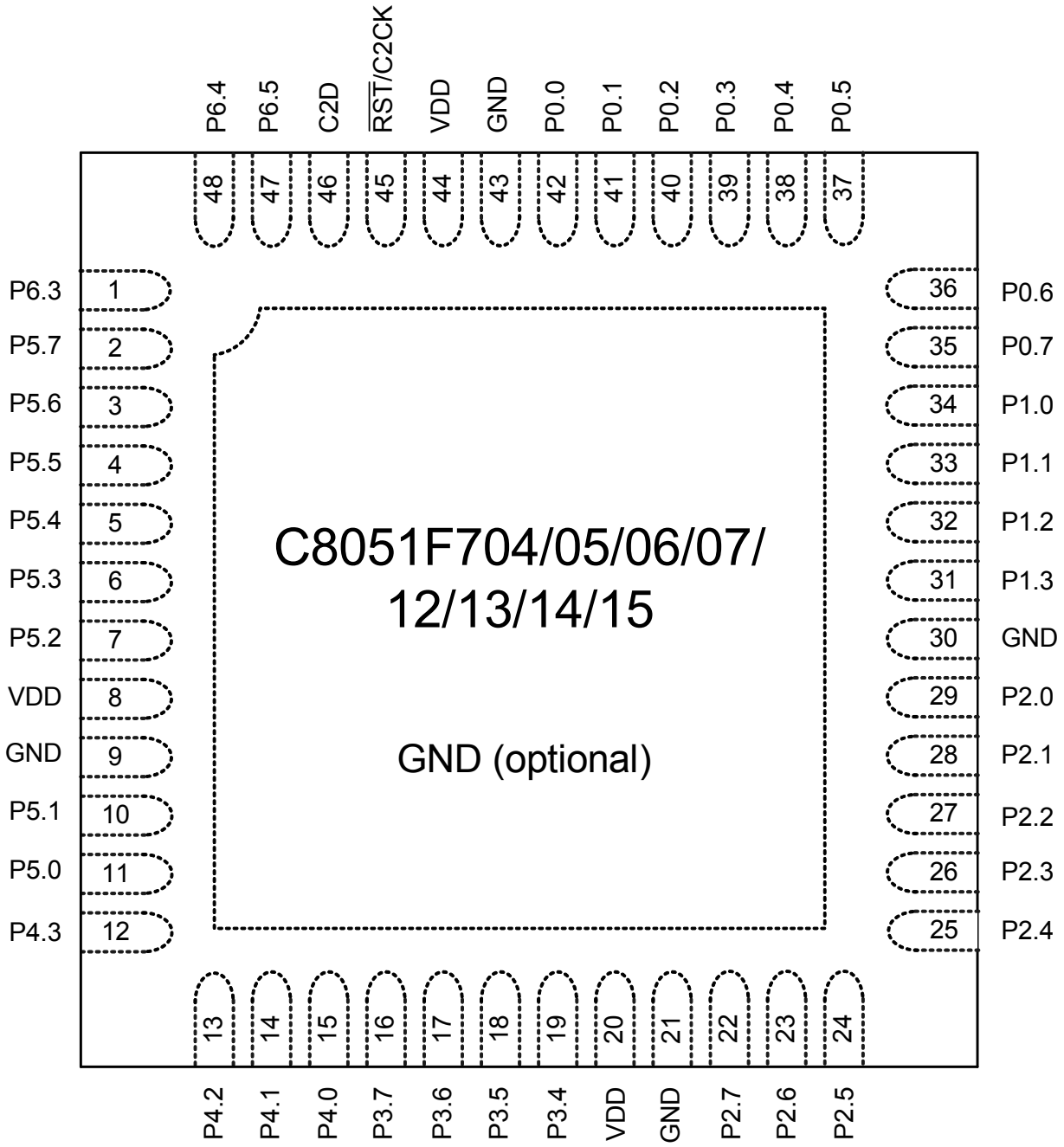


Figure 3.3. C8051F7xx-GQ QFN48 Pinout Diagram (Top View)

4. TQFP-64 Package Specifications

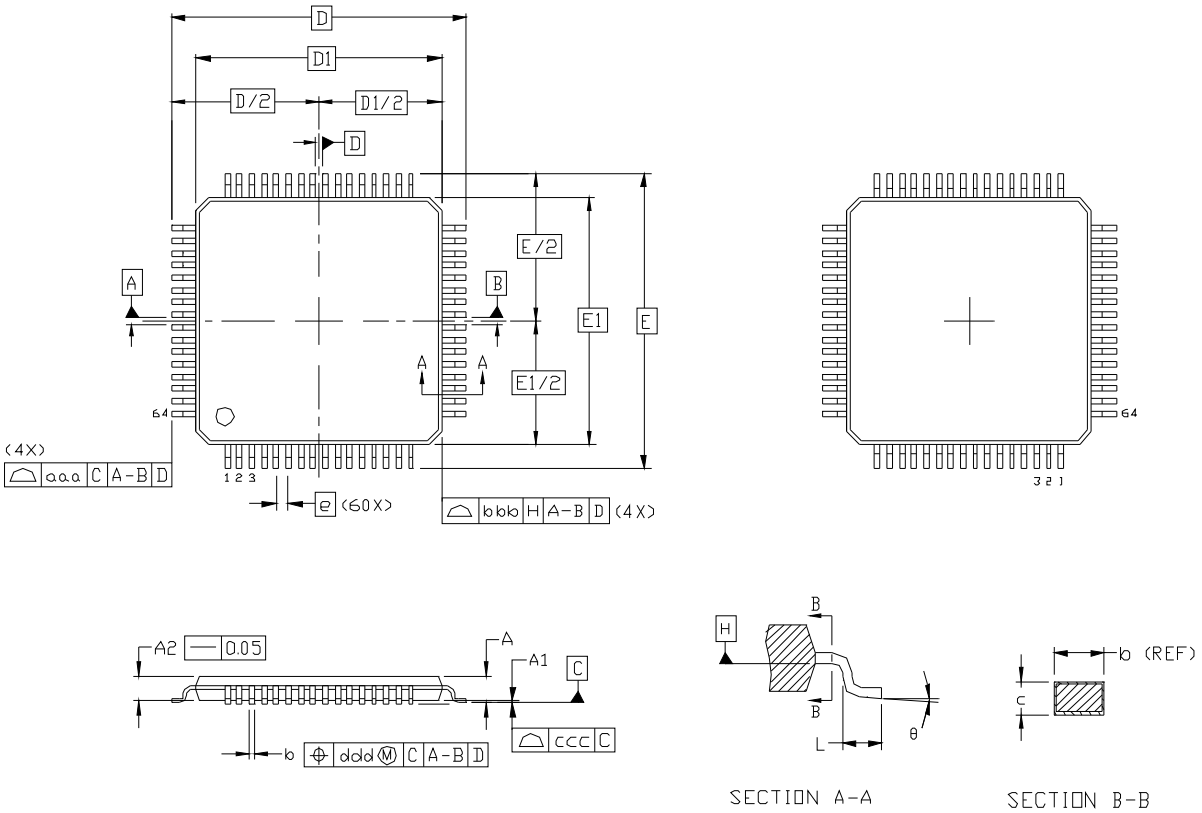


Figure 4.1. TQFP-64 Package Drawing

Table 4.1. TQFP-64 Package Dimensions

Dimension	Min	Nom	Max
A	--	--	1.20
A1	0.05	--	0.15
A2	0.95	1.00	1.05
b	0.17	0.22	0.27
c	0.09	--	0.20
D	12.00 BSC.		
D1	10.00 BSC.		
e	0.50 BSC.		

Dimension	Min	Nom	Max
E	12.00 BSC.		
E1	10.00 BSC.		
L	0.45	0.60	0.75
aaa	--	--	0.20
bbb	--	--	0.20
ccc	--	--	0.08
ddd	--	--	0.08
Θ	0°	3.5°	7°

Notes:

1. All dimensions shown are in millimeters (mm) unless otherwise noted.
2. Dimensioning and Tolerancing per ANSI Y14.5M-1994.
3. This package outline conforms to JEDEC MS-026, variant ACD.
4. Recommended card reflow profile is per the JEDEC/IPC J-STD-020 specification for Small Body Components.

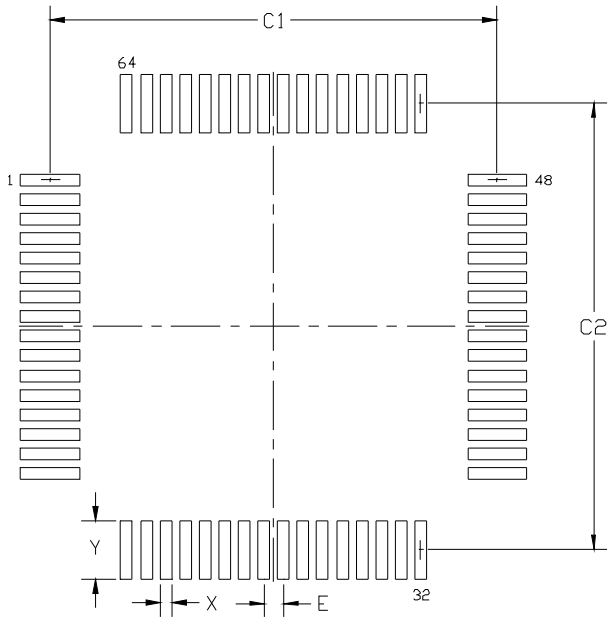


Figure 4.2. TQFP-64 PCB Land Pattern

Table 4.2. TQFP-64 PCB Land Pattern Dimensions

Dimension	Min	Max
C1	11.30	11.40
C2	11.30	11.40
E	0.50 BSC	
X	0.20	0.30
Y	1.40	1.50

Notes:

General

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

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 60μm 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 0.125mm (5 mils).
6. The ratio of stencil aperture to land pad size should be 1:1 for all perimeter pins.

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.

5. TQFP-48 Package Specifications

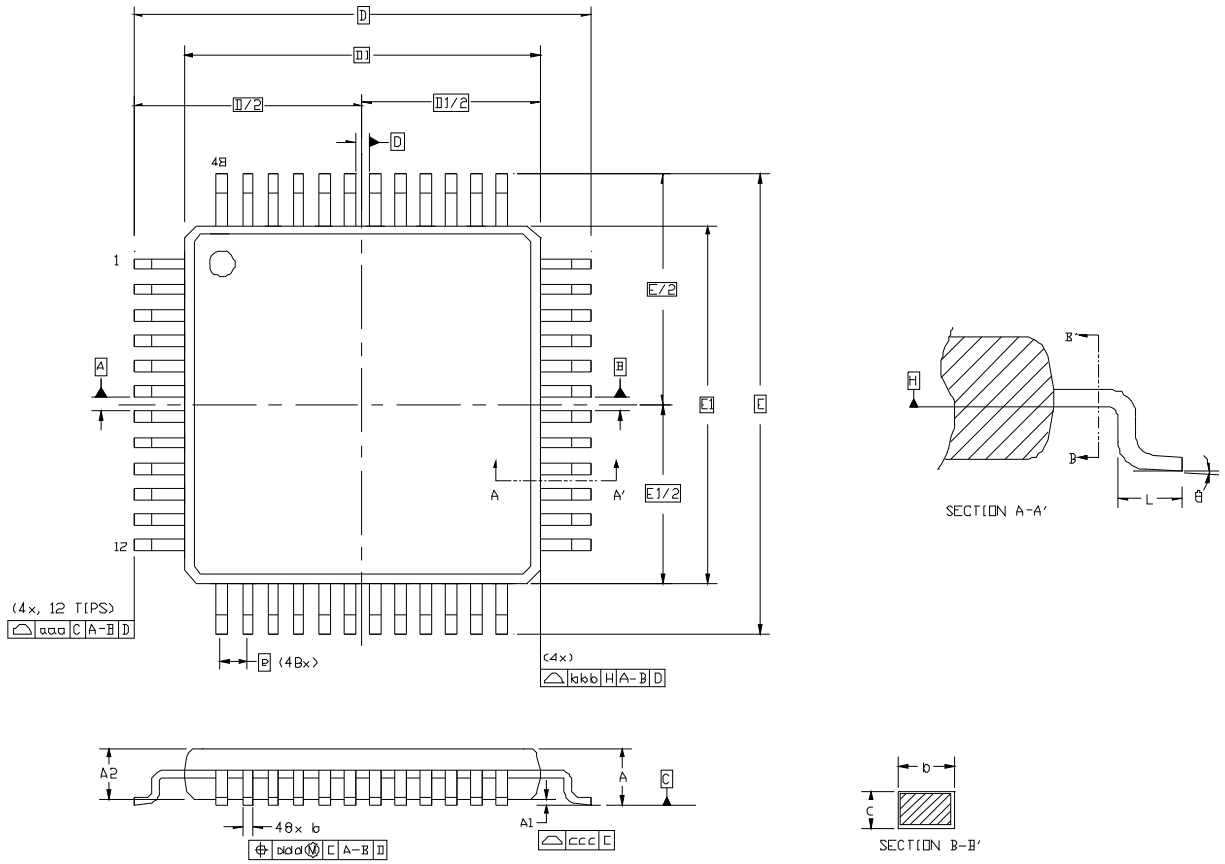


Figure 5.1. TQFP-48 Package Drawing

Table 5.1. TQFP-48 Package Dimensions

Dimension	Min	Nom	Max	Dimension	Min	Nom	Max
A	—	—	1.20	E	9.00 BSC.		
A1	0.05	—	0.15	E1	7.00 BSC.		
A2	0.95	1.00	1.05	L	0.45	0.60	0.75
b	0.17	0.22	0.27	aaa	0.20		
c	0.09	—	0.20	bbb	0.20		
D	9.00 BSC.			ccc	0.08		
D1	7.00 BSC.			ddd	0.08		
e	0.50 BSC.			Θ	0°	3.5°	7°

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 MS-026, variation ABC.
4. Recommended card reflow profile is per the JEDEC/IPC J-STD-020 specification for Small Body Components.

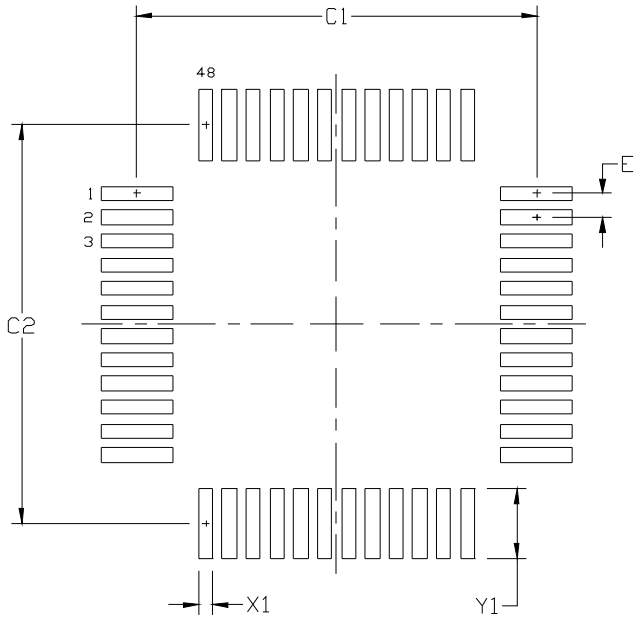


Figure 5.2. TQFP-48 PCB Land Pattern

Table 5.2. TQFP-48 PCB Land Pattern Dimensions

Dimension	Min	Max
C1	8.30	8.40
C2	8.30	8.40
E	0.50 BSC	
X1	0.20	0.30
Y1	1.40	1.50

Notes:

General

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

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 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-48 Package Specifications

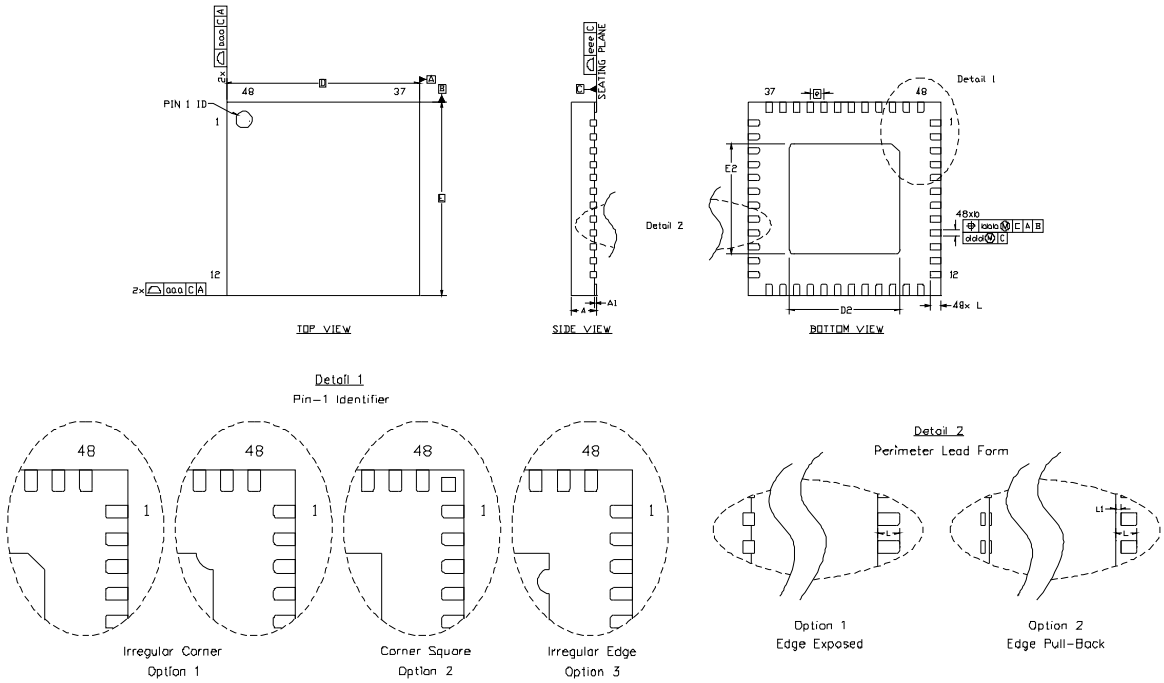


Figure 6.1. QFN-48 Package Drawing

Table 6.1. QFN-48 Package Dimensions

Dimension	Min	Nom	Max	Dimension	Min	Nom	Max
A	0.80	0.90	1.00	E2	3.90	4.00	4.10
A1	0.00	—	0.05	L	0.30	0.40	0.50
b	0.18	0.23	0.30	L1	0.00	—	0.10
D	7.00 BSC.			aaa	—	—	0.10
D2	3.90	4.00	4.10	bbb	—	—	0.10
e	0.50 BSC.			ccc	—	—	0.05
E	7.00 BSC.			ddd	—	—	0.08

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

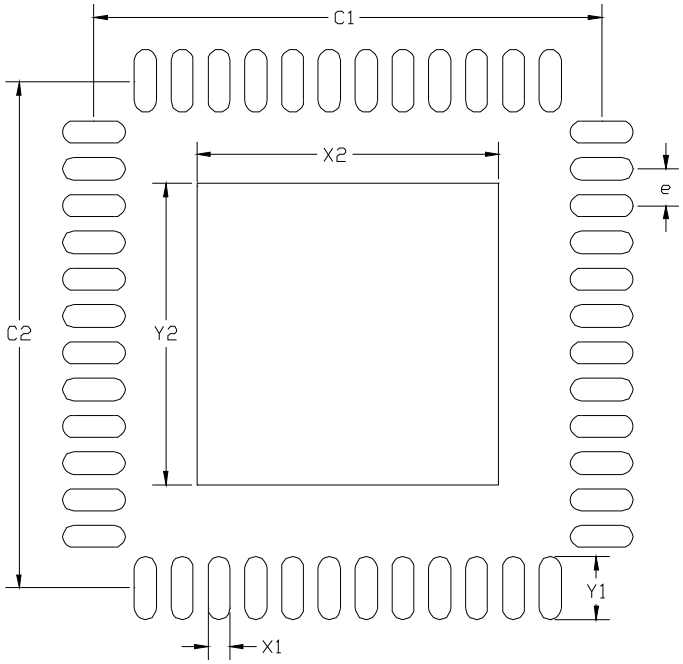


Figure 6.2. QFN-48 PCB Land Pattern

Table 6.2. QFN-48 PCB Land Pattern Dimensions

Dimension	Min	Max
e	0.50 BSC	
C1	6.80	6.90
C2	6.80	6.90
X1	0.20	0.30
X2	4.00	4.10
Y1	0.75	0.85
Y2	4.00	4.10

Notes:

General

1. All dimensions shown are in millimeters (mm) unless otherwise noted.
2. Dimensioning and Tolerancing is per the ANSI Y14.5M-1994 specification.
3. This Land Pattern Design is based on IPC-SM-782 guidelines.
4. All dimensions shown are at Maximum Material Condition (MMC). Least Material Condition (LMC) is calculated based on a Fabrication Allowance of 0.05 mm.

Solder Mask Design

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

Stencil Design

6. A stainless steel, laser-cut and electro-polished stencil with trapezoidal walls should be used to assure good solder paste release.
7. The stencil thickness should be 0.125 mm (5 mils).
8. The ratio of stencil aperture to land pad size should be 1:1 for all perimeter pads.
9. A 3x3 array of 1.2 mm square openings on 1.4 mm pitch should be used for the center ground pad.

Card Assembly

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

7. Electrical Characteristics

7.1. Absolute Maximum Specifications

Table 7.1. Absolute Maximum Ratings

Parameter	Conditions	Min	Typ	Max	Units
Ambient temperature under bias		–55	—	125	°C
Storage Temperature		–65	—	150	°C
Voltage on $\overline{RST}$ or any Port I/O Pin (except V_{DD} during programming) with respect to GND		–0.3	—	V_{DD} + 2.0	V
Voltage on V_{DD} with respect to GND during a programming operation	V_{DD} > 2.4 V	–0.3	—	7.0	V
Voltage on V_{DD} with respect to GND	Regulator in Normal Mode Regulator in Bypass Mode	–0.3 –0.3	— —	4.2 1.98	V V
Maximum Total current through V_{DD} and GND		—	—	500	mA
Maximum output current sunk by $\overline{RST}$ or any Port pin		—	—	100	mA

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

7.2. Electrical Characteristics

Table 7.2. Global Electrical Characteristics

–40 to +85 °C, 25 MHz system clock unless otherwise specified.

Parameter	Conditions	Min	Typ	Max	Units
Supply Voltage (Note 1)	Regulator in Normal Mode Regulator in Bypass Mode	1.8 1.7	3.0 1.8	3.6 1.9	V V
Digital Supply Current with CPU Active (Normal Mode, Note 2)	V_{DD} = 1.8 V, Clock = 25 MHz V_{DD} = 1.8 V, Clock = 1 MHz V_{DD} = 1.8 V, Clock = 32 kHz V_{DD} = 3.0 V, Clock = 25 MHz V_{DD} = 3.0 V, Clock = 1 MHz V_{DD} = 3.0 V, Clock = 32 kHz	— — — — — —	5.5 1.2 175 6.5 1.3 190	TBD — — TBD — —	mA mA µA mA mA µA
Digital Supply Current with CPU Inactive (Idle Mode, Note 2)	V_{DD} = 1.8 V, Clock = 25 MHz V_{DD} = 1.8 V, Clock = 1 MHz V_{DD} = 1.8 V, Clock = 32 kHz V_{DD} = 3.0 V, Clock = 25 MHz V_{DD} = 3.0 V, Clock = 1 MHz V_{DD} = 3.0 V, Clock = 32 kHz	— — — — — —	2.5 180 90 3.2 200 110	TBD — — TBD — —	mA µA µA mA µA µA
Digital Supply Current (shutdown)	Oscillator not running (stop mode), Internal Regulator Off	—	TBD	—	µA
	Oscillator not running (stop or suspend mode), Internal Regulator On	—	85	TBD	µA
	Oscillator not running (stop or suspend mode), Internal Regulator in Bypass	—	55	TBD	µA
Digital Supply RAM Data Retention Voltage		—	TBD	—	V
Specified Operating Temperature Range		–40	—	+85	°C
SYSCLK (system clock frequency)	(Note 3)	0	—	25	MHz
Tsysl (SYSCLK low time)		18	—	—	ns
Tsysh (SYSCLK high time)		18	—	—	ns

Notes:
1. Analog performance is not guaranteed when V_{DD} is below 1.8 V.
2. Includes bias current for internal voltage regulator.
3. SYSCLK must be at least 32 kHz to enable debugging.
4. Supply current parameters specified with Memory Power Controller enabled.

Table 7.3. Port I/O DC Electrical Characteristics

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

Parameters	Conditions	Min	Typ	Max	Units
Output High Voltage	High Drive Strength				
	I_{OH} = –3 mA, Port I/O push-pull	V_{DD} - 0.7	—	—	V
	I_{OH} = –10 µA, Port I/O push-pull	V_{DD} - 0.1	—	—	V
	I_{OH} = –10 mA, Port I/O push-pull	—	V_{DD} - 0.8	—	V
	Low Drive Strength				
	I_{OH} = –1 mA, Port I/O push-pull	V_{DD} - 0.7	—	—	V
	I_{OH} = –10 µA, Port I/O push-pull	V_{DD} - 0.1	—	—	V
	I_{OH} = –3 mA, Port I/O push-pull	—	V_{DD} - 0.8	—	V
Output Low Voltage	High Drive Strength				
	I_{OL} = 8.5 mA	—	—	0.6	V
	I_{OL} = 10 µA	—	—	0.1	V
	I_{OL} = 25 mA	—	1.0	—	V
	Low Drive Strength				
	I_{OL} = 1.4 mA	—	—	0.6	V
	I_{OL} = 10 µA	—	—	0.1	V
	I_{OL} = 4 mA	—	1.0	—	V
Input High Voltage		0.75 x V_{DD}	—	—	V
Input Low Voltage		—	—	0.6	V
Input Leakage Current	Weak Pullup Off	-1	—	1	µA
	Weak Pullup On, V_{IN} = 0 V	—	25	50	µA

Table 7.4. Reset Electrical Characteristics

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

Parameter	Conditions	Min	Typ	Max	Units
$\overline{RST}$ Output Low Voltage	I_{OL} = 8.5 mA, V_{DD} = 1.8 V to 3.6 V	—	—	0.6	V
$\overline{RST}$ Input High Voltage		0.75 x V_{DD}	—	—	V
$\overline{RST}$ Input Low Voltage		—	—	0.3 x V_{DD}	V_{DD}
$\overline{RST}$ Input Pullup Current	$\overline{RST}$ = 0.0 V	—	25	50	µA
V_{DD} POR Ramp Time		—	—	1	ms
V_{DD} Monitor Threshold (V_{RST})		1.7	1.75	1.8	V
Missing Clock Detector Timeout	Time from last system clock rising edge to reset initiation	TBD	500	TBD	µs
Reset Time Delay	Delay between release of any reset source and code execution at location 0x0000	—	TBD	TBD	µs
Minimum $\overline{RST}$ Low Time to Generate a System Reset		15	—	—	µs
V_{DD} Monitor Turn-on Time	V_{DD} = V_{RST} - 0.1 V	—	TBD	—	µs
V_{DD} Monitor Supply Current		—	25	TBD	µA

Table 7.5. Internal Voltage Regulator Electrical Characteristics

V_{DD} = 3.0 V, –40 to +85 °C unless otherwise specified.

Parameter	Conditions	Min	Typ	Max	Units
Input Voltage Range		1.8	—	3.6	V
Bias Current	Normal Mode Bypass Mode	— —	80 50	TBD TBD	µA µA

Table 7.6. Flash Electrical Characteristics

Parameter	Conditions	Min	Typ	Max	Units
Flash Size (Note 1)	C8051F702/3/6/7 C8051F700/1/4/5 C8051F708/9, C8051F710/1/2/3/4/5		16384 15360 8192		bytes bytes bytes
Endurance (Erase/Write)		TBD	TBD	—	cycles
Erase Cycle Time	25 MHz Clock	TBD	20	TBD	ms
Write Cycle Time	25 MHz Clock	TBD	40	TBD	µs

Notes:
1. Includes Security Lock Byte

Table 7.7. Internal High-Frequency Oscillator Electrical Characteristics

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

Parameter	Conditions	Min	Typ	Max	Units
Oscillator Frequency	IFCN = 11b	24	24.5	25	MHz
Oscillator Supply Current	25 °C, V_{DD} = 3.0 V, OSCICN.7 = 1, OCSICN.5 = 0	—	350	TBD	µA

Table 7.8. Capacitive Sense Electrical Characteristics

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

Parameter	Conditions	Min	Typ	Max	Units
Conversion Time	Single Conversion	35	40	50	µs
Number of Channels			32		Channels
Capacitance per Code		—	1	—	fF
External Capacitive Load		—	—	TBD	pF
Quantization Noise (Note 1)	RMS Peak-to-Peak	— —	3 20	TBD TBD	fF fF
Supply Current	CS module alone, No Cap CS module alone, Maximum Cap Wake-on-CS Threshold (Note 2)	— — —	40 75 130	TBD TBD TBD	µA µA µA
Power Supply Rejection		—	TBD	—	fF/V

Notes:
1. RMS Noise is equivalent to one standard deviation. Peak-to-peak noise encompasses ±3.3 standard deviations.
2. Includes only current from regulator, CS module, and MCU in suspend mode.

Table 7.9. EEPROM Electrical Characteristics

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

Parameter	Conditions	Min	Typ	Max	Units
Write to EEPROM from RAM			3.0	TBD	ms
Read of EEPROM to RAM		TBD	50 x T_{SYSCLK}	TBD	µs
Endurance (Writes)		TBD	TBD	—	cycles
Note: T_{SYSCLK} is equal to one period of the device system clock (SYSCLK).					

Table 7.10. ADC0 Electrical Characteristics

V_{DD} = 3.0 V, VREF = 2.40 V (REFSL=0), –40 to +85 °C unless otherwise specified.

Parameter	Conditions	Min	Typ	Max	Units
DC Accuracy					
Resolution			10		bits
Integral Nonlinearity		—	±0.5	±1	LSB
Differential Nonlinearity	Guaranteed Monotonic	—	±0.5	±1	LSB
Offset Error		–2	0	2	LSB
Full Scale Error		–2	0	2	LSB
Offset Temperature Coefficient		—	45	—	ppm/°C
Dynamic performance (10 kHz sine-wave single-ended input, 1 dB below Full Scale, 200 ksps)					
Signal-to-Noise Plus Distortion		56	60	—	dB
Total Harmonic Distortion	Up to the 5th harmonic	—	72	—	dB
Spurious-Free Dynamic Range		—	–75	—	dB
Conversion Rate					
SAR Conversion Clock		—	—	8.33	MHz
Conversion Time in SAR Clocks	10-bit Mode 8-bit Mode	13 11	— —	— —	clocks clocks
Track/Hold Acquisition Time	V_{DD} >= 2.0 V V_{DD} < 2.0 V	300 2.0	— —	— —	ns µs
Throughput Rate		—	—	500	ksps
Analog Inputs					
ADC Input Voltage Range		0	—	VREF	V
Sampling Capacitance	1x Gain 0.5x Gain	— —	5 3	— —	pF pF
Input Multiplexer Impedance		—	5	—	kΩ
Power Specifications					
Power Supply Current	Operating Mode, 500 ksps	—	600	TBD	µA
Power Supply Rejection		—	–70	—	dB

Table 7.11. Power Management Electrical Characteristics

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

Parameter	Conditions	Min	Typ	Max	Units
Idle Mode Wake-up time		2	—	3	SYSCLKs
Suspend Mode Wake-up Time		—	TBD	—	ns

Table 7.12. Temperature Sensor Electrical Characteristics

V_{DD} = 3.0 V, –40 to +85 °C unless otherwise specified.

Parameter	Conditions	Min	Typ	Max	Units
Linearity		—	TBD	—	°C
Slope		—	TBD	—	mV/°C
Slope Error*		—	TBD	—	µV/°C
Offset	Temp = 0 °C	—	TBD	—	mV
Offset Error*	Temp = 0 °C	—	TBD	—	mV
Note: Represents one standard deviation from the mean.					

Table 7.13. Voltage Reference Electrical Characteristics

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

Parameter	Conditions	Min	Typ	Max	Units
Internal High Speed Reference (REFSL[1:0] = 11)					
Output Voltage	25 °C ambient	TBD	1.65	TBD	V
Temperature Coefficient		—	—	TBD	ppm/°C
Turn-on Time		—	—	TBD	µs
PSRR	Includes PSRR of ADC0	—	TBD	—	ppm/V
Supply Current		—	200	—	µA
External Reference (REF0E = 0)					
Input Voltage Range		0	—	V_{DD}	
Input Current	Sample Rate = 500 ksps; VREF = 3.0 V	—	TBD	—	µA

Table 7.14. Comparator Electrical Characteristics

V_{DD} = 3.0 V, –40 to +85 °C unless otherwise noted.

Parameter	Conditions	Min	Typ	Max	Units
Response Time: Mode 0, Vcm* = 1.5 V	CP0+ – CP0– = 100 mV	—	120	—	ns
	CP0+ – CP0– = –100 mV	—	140	—	ns
Response Time: Mode 1, Vcm* = 1.5 V	CP0+ – CP0– = 100 mV	—	190	—	ns
	CP0+ – CP0– = –100 mV	—	290	—	ns
Response Time: Mode 2, Vcm* = 1.5 V	CP0+ – CP0– = 100 mV	—	380	—	ns
	CP0+ – CP0– = –100 mV	—	780	—	ns
Response Time: Mode 3, Vcm* = 1.5 V	CP0+ – CP0– = 100 mV	—	1400	—	ns
	CP0+ – CP0– = –100 mV	—	4400	—	ns
Common-Mode Rejection Ratio		—	1	4	mV/V
Positive Hysteresis 1	CP0HYP1–0 = 00	—	0	1	mV
Positive Hysteresis 2	CP0HYP1–0 = 01	2	5	10	mV
Positive Hysteresis 3	CP0HYP1–0 = 10	7	10	20	mV
Positive Hysteresis 4	CP0HYP1–0 = 11	10	20	30	mV
Negative Hysteresis 1	CP0HYN1–0 = 00	—	0	1	mV
Negative Hysteresis 2	CP0HYN1–0 = 01	2	5	10	mV
Negative Hysteresis 3	CP0HYN1–0 = 10	7	10	20	mV
Negative Hysteresis 4	CP0HYN1–0 = 11	10	20	30	mV
Inverting or Non-Inverting Input Voltage Range		–0.25	—	V_{DD} + 0.25	V
Input Offset Voltage		–7.5	—	7.5	mV
Power Specifications					
Power Supply Rejection		—	TBD	—	mV/V
Powerup Time		—	TBD	—	µs
Supply Current at DC	Mode 0	—	25	TBD	µA
	Mode 1	—	10	TBD	µA
	Mode 2	—	3	TBD	µA
	Mode 3	—	0.5	TBD	µA
Note: Vcm is the common-mode voltage on CP0+ and CP0–.					

8. 10-Bit ADC (ADC0, C8051F700/2/4/6/8 and C8051F710/2/4 only)

ADC0 on the C8051F700/2/4/6/8 and C8051F710/2/4 is a 500 ksps, 10-bit successive-approximation-register (SAR) ADC with integrated track-and-hold, a gain stage programmable to 1x or 0.5x, and a programmable window detector. The ADC is fully configurable under software control via Special Function Registers. The ADC may be configured to measure various different signals using the analog multiplexer described in Section "8.5. ADC0 Analog Multiplexer" on page 56. The voltage reference for the ADC is selected as described in Section "9. Temperature Sensor (C8051F700/2/4/6/8 and C8051F710/2/4 only)" on page 58. The ADC0 subsystem is enabled only when the AD0EN bit in the ADC0 Control register (ADC0CN) is set to logic 1. The ADC0 subsystem is in low power shutdown when this bit is logic 0.

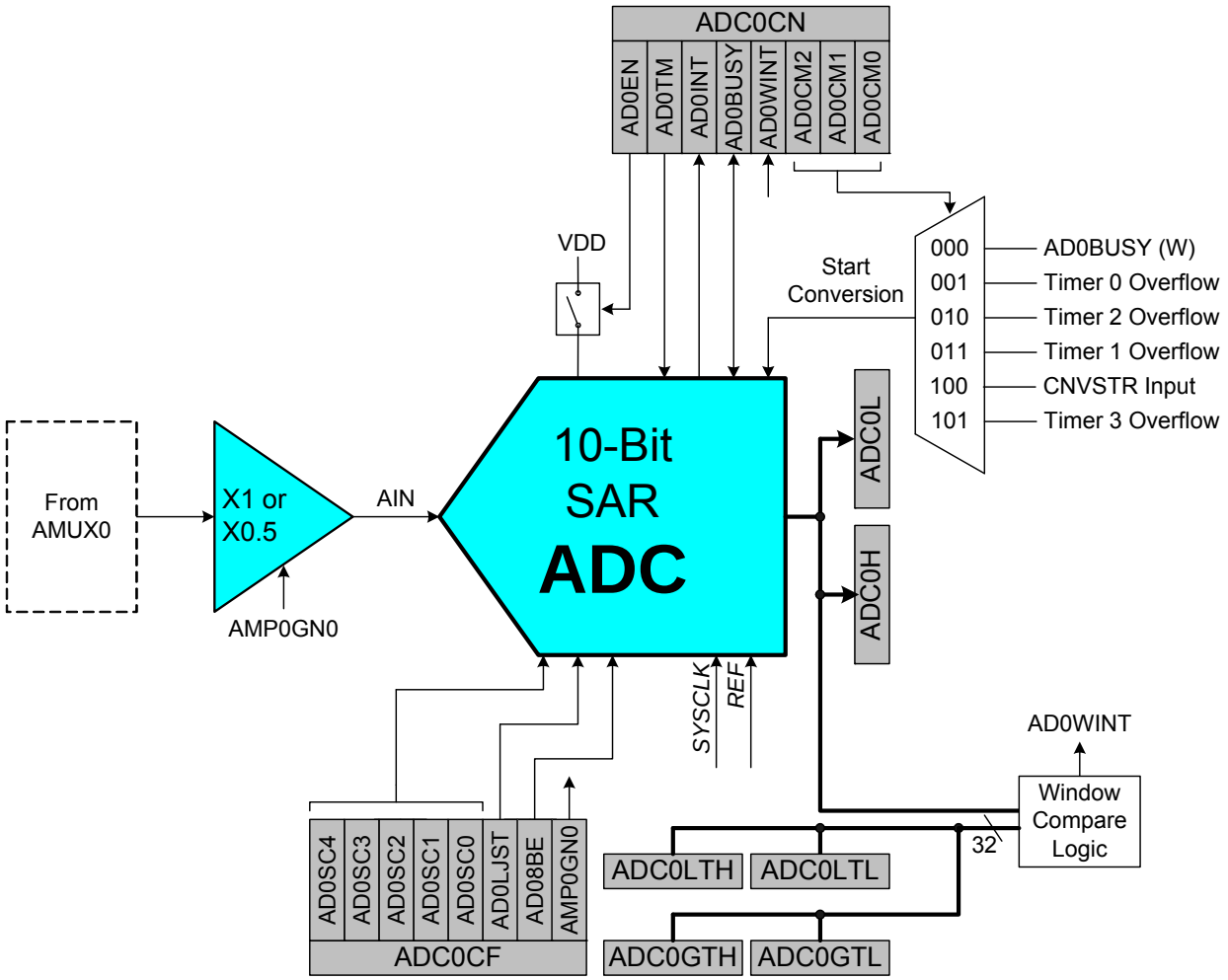


Figure 8.1. ADC0 Functional Block Diagram

8.1. Output Code Formatting

The ADC measures the input voltage with reference to GND. 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 AD0LJST bit. Conversion codes are represented as 10-bit unsigned integers. Inputs are measured from 0 to VREF x 1023/1024. Example codes are shown below for both right-justified and left-justified data. Unused bits in the ADC0H and ADC0L registers are set to 0.

Input Voltage	Right-Justified ADC0H:ADC0L (AD0LJST = 0)	Left-Justified ADC0H:ADC0L (AD0LJST = 1)
VREF x 1023/1024	0x03FF	0xFFC0
VREF x 512/1024	0x0200	0x8000
VREF x 256/1024	0x0100	0x4000
0	0x0000	0x0000

8.2. 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, and the ADC0H register holds the results. The AD0LJST bit is ignored for 8-bit mode. 8-bit conversions take two fewer SAR clock cycles than 10-bit conversions, so the conversion is completed faster, and a 500 ksps sampling rate can be achieved with a slower SAR clock.

8.3. Modes of Operation

ADC0 has a maximum conversion speed of 500 ksps. The ADC0 conversion clock is a divided version of the system clock, determined by the AD0SC bits in the ADC0CF register.

8.3.1. Starting a Conversion

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

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

Writing a 1 to AD0BUSY provides software control of ADC0 whereby conversions are performed "on-demand". During conversion, the AD0BUSY bit is set to logic 1 and reset to logic 0 when the conversion is complete. The falling edge of AD0BUSY triggers an interrupt (when enabled) and sets the ADC0 interrupt flag (AD0INT). When polling for ADC conversion completions, the ADC0 interrupt flag (AD0INT) should be used. Converted data is available in the ADC0 data registers, ADC0H:ADC0L, when bit AD0INT is logic 1. When Timer 2 or Timer 3 overflows are used as the conversion source, Low Byte overflows are used if Timer 2/3 is in 8-bit mode; High byte overflows are used if Timer 2/3 is in 16-bit mode. See Section “31. Timers” on page 244 for timer configuration.

Important Note About Using CNVSTR: The CNVSTR input pin also functions as a Port I/O pin. When the CNVSTR input is used as the ADC0 conversion source, the associated pin should be skipped by the Digital Crossbar. See Section “26. Port Input/Output” on page 165 for details on Port I/O configuration.

8.3.2. Tracking Modes

The AD0TM bit in register ADC0CN enables "delayed conversions", and will delay the actual conversion start by three SAR clock cycles, during which time the ADC will continue to track the input. If AD0TM is left at logic 0, a conversion will begin immediately, without the extra tracking time. For internal start-of-conversion sources, the ADC will track anytime it is not performing a conversion. When the CNVSTR signal is used to initiate conversions, ADC0 will track either when AD0TM is logic 1, or when AD0TM is logic 0 and CNVSTR is held low. See Figure 8.2 for track and convert timing details. Delayed conversion mode is useful when AMUX settings are frequently changed, due to the settling time requirements described in Section “8.3.3. Settling Time Requirements” on page 49.

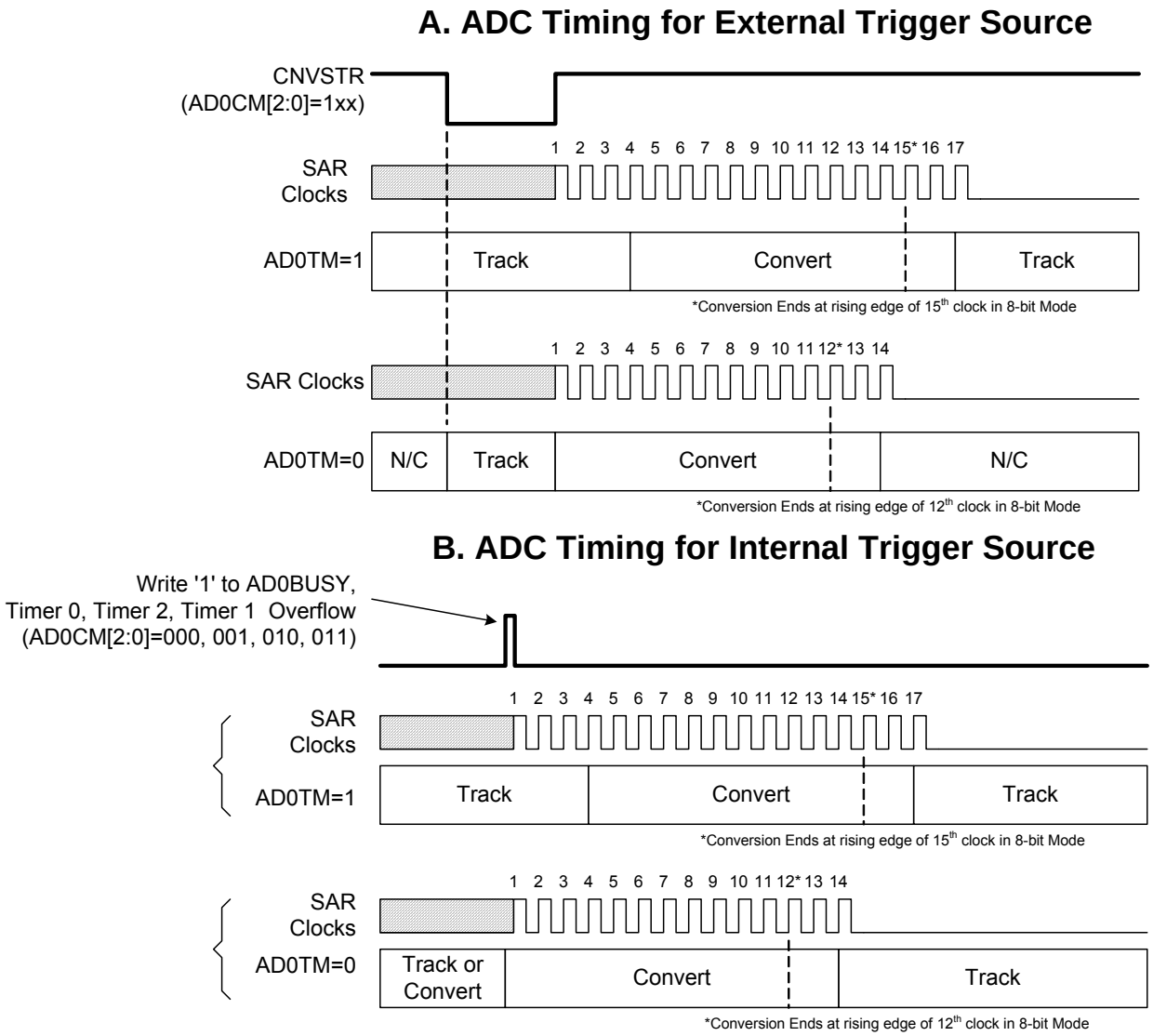


Figure 8.2. 10-Bit ADC Track and Conversion Example Timing

8.3.3. Settling Time Requirements

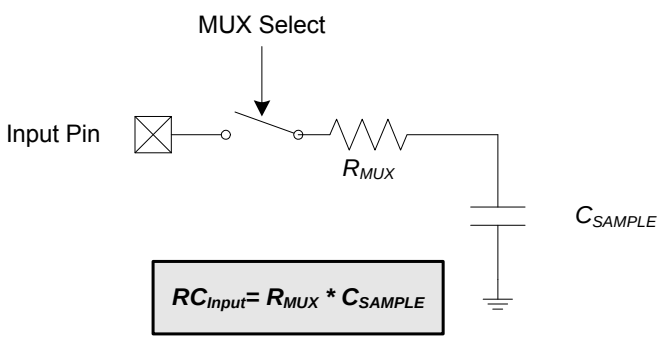
A minimum tracking time is required before each conversion to ensure that an accurate conversion is performed. This tracking time is determined by any series impedance, including the AMUX0 resistance, the the ADC0 sampling capacitance, and the accuracy required for the conversion. In delayed tracking mode, three SAR clocks are used for tracking at the start of every conversion. For many applications, these three SAR clocks will meet the minimum tracking time requirements.

Figure 8.3 shows the equivalent ADC0 input circuit. The required ADC0 settling time for a given settling accuracy (SA) may be approximated by Equation 8.1. See Table 7.10 for ADC0 minimum settling time requirements as well as the mux impedance and sampling capacitor values.

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

Equation 8.1. ADC0 Settling Time Requirements

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



Note: See electrical specification tables for R_{MUX} and C_{SAMPLE} parameters.

Figure 8.3. ADC0 Equivalent Input Circuits

SFR Definition 8.1. ADC0CF: ADC0 Configuration

Bit	7	6	5	4	3	2	1	0
Name	AD0SC[4:0]					AD0LJST	AD08BE	AMP0GN0
Type	R/W					R/W	R/W	R/W
Reset	1	1	1	1	1	0	0	1

SFR Address = 0xBC; SFR Page = F

Bit	Name	Function
7:3	AD0SC[4:0]	**ADC0 SAR Conversion Clock Period Bits.** SAR Conversion clock is derived from system clock by the following equation, where *AD0SC* refers to the 5-bit value held in bits AD0SC4–0. SAR Conversion clock requirements are given in the ADC specification table. $\mathrm{AD0SC} = \frac{\mathrm{SYSCLK}}{\mathrm{CLK_{SAR}}} - 1$
2	AD0LJST	**ADC0 Left Justify Select.** 0: Data in ADC0H:ADC0L registers are right-justified. 1: Data in ADC0H:ADC0L registers are left-justified. **Note:** The AD0LJST bit is only valid for 10-bit mode (AD08BE = 0).
1	AD08BE	**8-Bit Mode Enable.** 0: ADC operates in 10-bit mode (normal). 1: ADC operates in 8-bit mode. **Note:** When AD08BE is set to 1, the AD0LJST bit is ignored.
0	AMP0GN0	**ADC Gain Control Bit.** 0: Gain = 0.5 1: Gain = 1

SFR Definition 8.2. ADC0H: ADC0 Data Word MSB

Bit	7	6	5	4	3	2	1	0
Name	ADC0H[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xBE; SFR Page = 0

Bit	Name	Function
7:0	ADC0H[7:0]	**ADC0 Data Word High-Order Bits.** For AD0LJST = 0: Bits 7–2 will read 000000b. Bits 1–0 are the upper 2 bits of the 10-bit ADC0 Data Word. For AD0LJST = 1: Bits 7–0 are the most-significant bits of the 10-bit ADC0 Data Word. **Note:** In 8-bit mode AD0LJST is ignored, and ADC0H holds the 8-bit data word.

SFR Definition 8.3. ADC0L: ADC0 Data Word LSB

Bit	7	6	5	4	3	2	1	0
Name	ADC0L[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xBD; SFR Page = 0

Bit	Name	Function
7:0	ADC0L[7:0]	**ADC0 Data Word Low-Order Bits.** For AD0LJST = 0: Bits 7–0 are the lower 8 bits of the 10-bit Data Word. For AD0LJST = 1: Bits 7–6 are the lower 2 bits of the 10-bit Data Word. Bits 5–0 will always read 0. **Note:** In 8-bit mode AD0LJST is ignored, and ADC0L will read back 00000000b.

SFR Definition 8.4. ADC0CN: ADC0 Control

Bit	7	6	5	4	3	2	1	0
Name	AD0EN	AD0TM	AD0INT	AD0BUSY	AD0WINT	AD0CM[2:0]		
Type	R/W	R/W	R/W	R/W	R/W	R/W		
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xE8; SFR Page = All Pages; Bit-Addressable

Bit	Name	Function		
7	AD0EN	**ADC0 Enable Bit.** 0: ADC0 Disabled. ADC0 is in low-power shutdown. 1: ADC0 Enabled. ADC0 is active and ready for data conversions.		
6	AD0TM	**ADC0 Track Mode Bit.** 0: Normal Track Mode: When ADC0 is enabled, tracking is continuous unless a conversion is in progress. Conversion begins immediately on start-of-conversion event, as defined by AD0CM[2:0]. 1: Delayed Track Mode: When ADC0 is enabled, input is tracked when a conversion is not in progress. A start-of-conversion signal initiates three SAR clocks of additional tracking, and then begins the conversion.		
5	AD0INT	**ADC0 Conversion Complete Interrupt Flag.** 0: ADC0 has not completed a data conversion since AD0INT was last cleared. 1: ADC0 has completed a data conversion.		
4	AD0BUSY	**ADC0 Busy Bit.**	**Read:** 0: ADC0 conversion is not in progress. 1: ADC0 conversion is in progress.	**Write:** 0: No Effect. 1: Initiates ADC0 Conversion if AD0CM[2:0] = 000b
3	AD0WINT	**ADC0 Window Compare Interrupt Flag.** 0: ADC0 Window Comparison Data match has not occurred since this flag was last cleared. 1: ADC0 Window Comparison Data match has occurred.		
2:0	AD0CM[2:0]	**ADC0 Start of Conversion Mode Select.** 000: ADC0 start-of-conversion source is write of 1 to AD0BUSY. 001: ADC0 start-of-conversion source is overflow of Timer 0. 010: ADC0 start-of-conversion source is overflow of Timer 2. 011: ADC0 start-of-conversion source is overflow of Timer 1. 100: ADC0 start-of-conversion source is rising edge of external CNVSTR. 101: ADC0 start-of-conversion source is overflow of Timer 3. 11x: Reserved.		

8.4. Programmable Window Detector

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

SFR Definition 8.5. ADC0GTH: ADC0 Greater-Than Data High Byte

Bit	7	6	5	4	3	2	1	0
Name	ADC0GTH[7:0]							
Type	R/W							
Reset	1	1	1	1	1	1	1	1

SFR Address = 0xC4; SFR Page = 0

Bit	Name	Function
7:0	ADC0GTH[7:0]	**ADC0 Greater-Than Data Word High-Order Bits.**

SFR Definition 8.6. ADC0GTL: ADC0 Greater-Than Data Low Byte

Bit	7	6	5	4	3	2	1	0
Name	ADC0GTL[7:0]							
Type	R/W							
Reset	1	1	1	1	1	1	1	1

SFR Address = 0xC3; SFR Page = 0

Bit	Name	Function
7:0	ADC0GTL[7:0]	**ADC0 Greater-Than Data Word Low-Order Bits.**

SFR Definition 8.7. ADC0LTH: ADC0 Less-Than Data High Byte

Bit	7	6	5	4	3	2	1	0
Name	ADC0LTH[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xC6; SFR Page = 0

Bit	Name	Function
7:0	ADC0LTH[7:0]	**ADC0 Less-Than Data Word High-Order Bits.**

SFR Definition 8.8. ADC0LTL: ADC0 Less-Than Data Low Byte

Bit	7	6	5	4	3	2	1	0
Name	ADC0LTL[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xC5; SFR Page = 0

Bit	Name	Function
7:0	ADC0LTL[7:0]	**ADC0 Less-Than Data Word Low-Order Bits.**

8.4.1. Window Detector Example

Figure 8.4 shows two example window comparisons for right-justified data, with ADC0LTH:ADC0LTL = 0x0080 (128d) and ADC0GTH:ADC0GTL = 0x0040 (64d). The input voltage can range from 0 to VREF x (1023/1024) with respect to GND, and is represented by a 10-bit unsigned integer value. In the left example, an AD0WINT interrupt will be generated if the ADC0 conversion word (ADC0H:ADC0L) is within the range defined by ADC0GTH:ADC0GTL and ADC0LTH:ADC0LTL (if 0x0040 < ADC0H:ADC0L < 0x0080). In the right example, and AD0WINT interrupt will be generated if the ADC0 conversion word is outside of the range defined by the ADC0GT and ADC0LT registers (if ADC0H:ADC0L < 0x0040 or ADC0H:ADC0L > 0x0080). Figure 8.5 shows an example using left-justified data with the same comparison values.

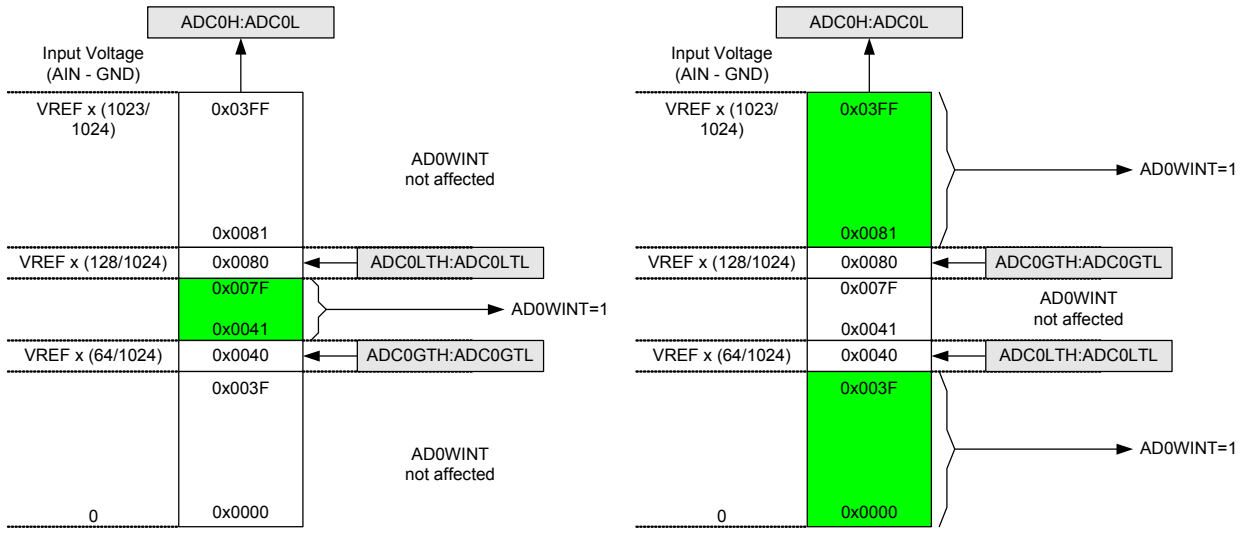


Figure 8.4. ADC Window Compare Example: Right-Justified Data

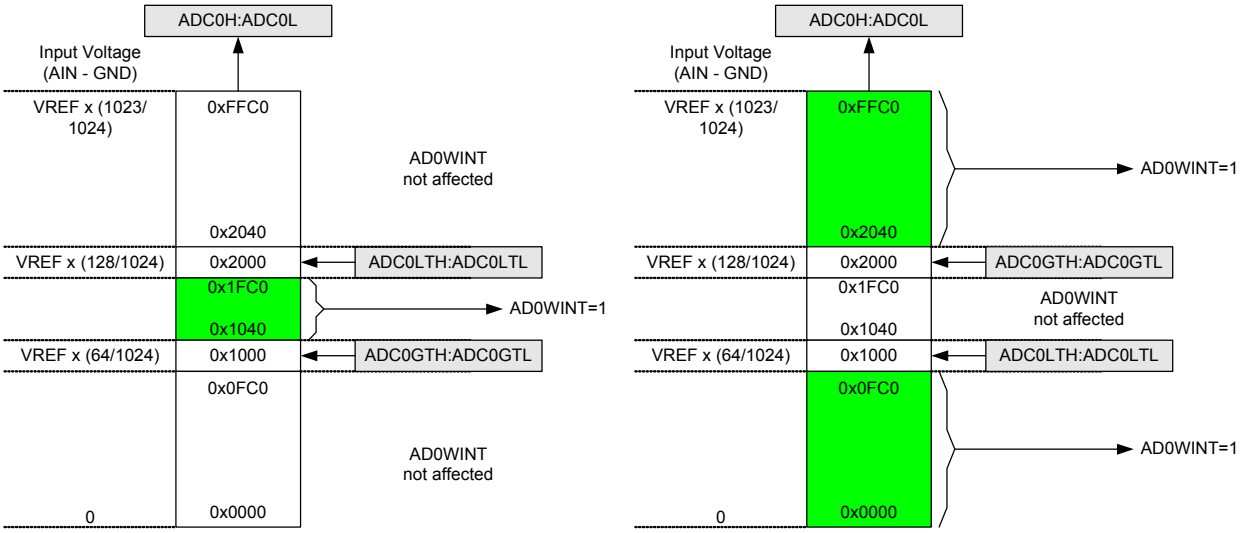


Figure 8.5. ADC Window Compare Example: Left-Justified Data

8.5. ADC0 Analog Multiplexer

ADC0 on the C8051F700/2/4/6/8 and C8051F710/2/4 uses an analog input multiplexer to select the positive input to the ADC. Any of the following may be selected as the positive input: Port 0 or Port 1 I/O pins, the on-chip temperature sensor, or the positive power supply (V_{DD}). The ADC0 input channel is selected in the ADC0MX register described in SFR Definition 8.9.

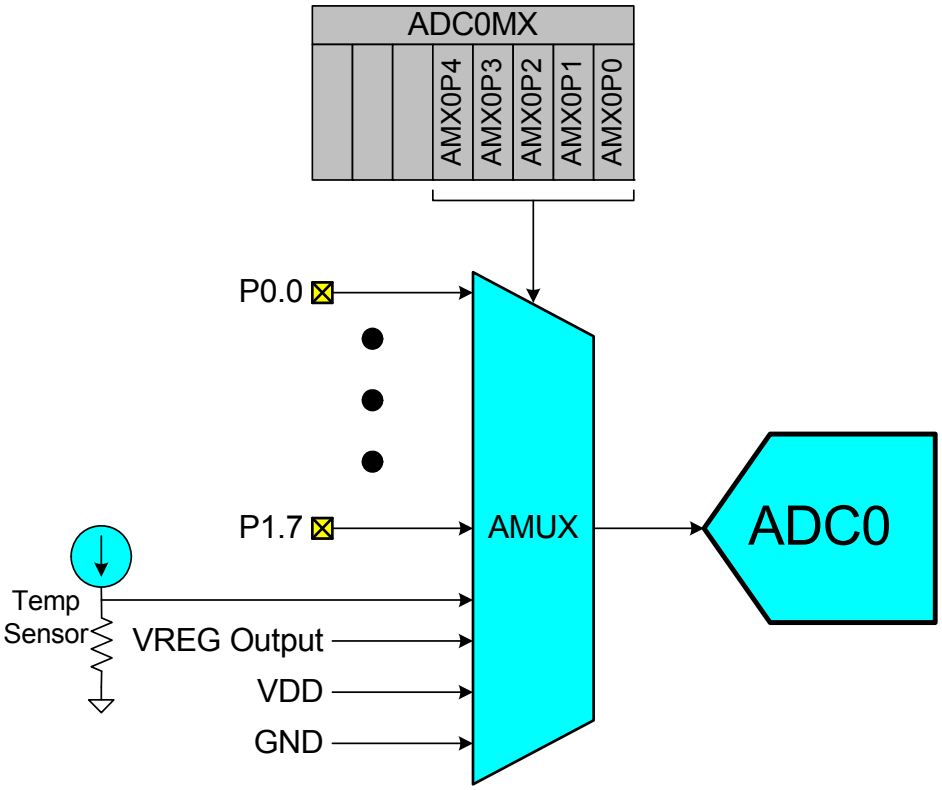


Figure 8.6. ADC0 Multiplexer Block Diagram

Important Note About ADC0 Input Configuration: Port pins selected as ADC0 inputs should be configured as analog inputs, and should be skipped by the Digital Crossbar. To configure a Port pin for analog input, set the corresponding bit in register PnMDIN to 0. To force the Crossbar to skip a Port pin, set the corresponding bit in register PnSKIP to 1. See Section “26. Port Input/Output” on page 165 for more Port I/O configuration details.

SFR Definition 8.9. ADC0MX: AMUX0 Channel Select

Bit	7	6	5	4	3	2	1	0
Name				AMX0P[3:0]				
Type	R	R	R	R/W				
Reset	0	0	0	1	1	1	1	1

SFR Address = 0xBB; SFR Page = 0

Bit	Name	Function		
7:5	Unused	Read = 0b; Write = Don't Care.		
4:0	AMX0P[4:0]	**AMUX0 Positive Input Selection.**		
			64-Pin Devices	**48-Pin Devices**
		00000:	P0.0	P0.0
		00001:	P0.1	P0.1
		00010:	P0.2	P0.2
		00011:	P0.3	P0.3
		00100:	P0.4	P0.4
		00101:	P0.5	P0.5
		00110:	P0.6	P0.6
		00111:	P0.7	P0.7
		01000	P1.0	P1.0
		01001	P1.1	P1.1
		01010	P1.2	P1.2
		01011	P1.3	P1.3
		01100	P1.4	Reserved.
		01101	P1.5	Reserved.
		01110	P1.6	Reserved.
		01111	P1.7	Reserved.
		10000:	Temp Sensor	Temp Sensor
		10001:	VREG Output	VREG Output
		10010:	VDD	VDD
		10011:	GND	GND
		10100 – 11111:	no input selected	

9. Temperature Sensor (C8051F700/2/4/6/8 and C8051F710/2/4 only)

An on-chip temperature sensor is included on the C8051F700/2/4/6/8 and C8051F710/2/4 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 be configured to connect to the temperature sensor. The temperature sensor transfer function is shown in Figure 9.1. The output voltage (V_{TEMP}) is the positive ADC input when the ADC multiplexer is set correctly. The TEMPE bit in register REF0CN enables/disables the temperature sensor, as described in SFR Definition 10.1. While disabled, the temperature sensor defaults to a high impedance state and any ADC measurements performed on the sensor will result in meaningless data. Refer to Table 7.12 for the slope and offset parameters of the temperature sensor.

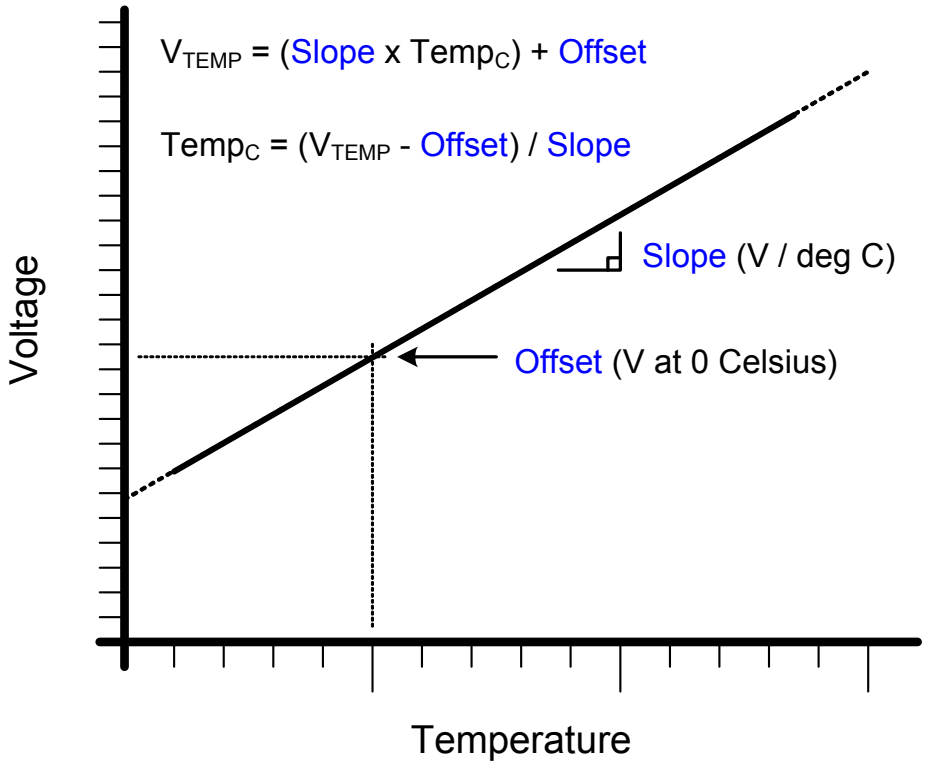


Figure 9.1. Temperature Sensor Transfer Function

9.1. Calibration

The uncalibrated temperature sensor output is extremely linear and suitable for relative temperature measurements (see Table 5.1 for linearity specifications). For absolute temperature measurements, offset and/or gain calibration is recommended. Typically a 1-point (offset) calibration includes the following steps:

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

Figure 5.3 shows the typical temperature sensor error assuming a 1-point calibration at 0 °C.

Parameters that affect ADC measurement, in particular the voltage reference value, will also affect temperature measurement.

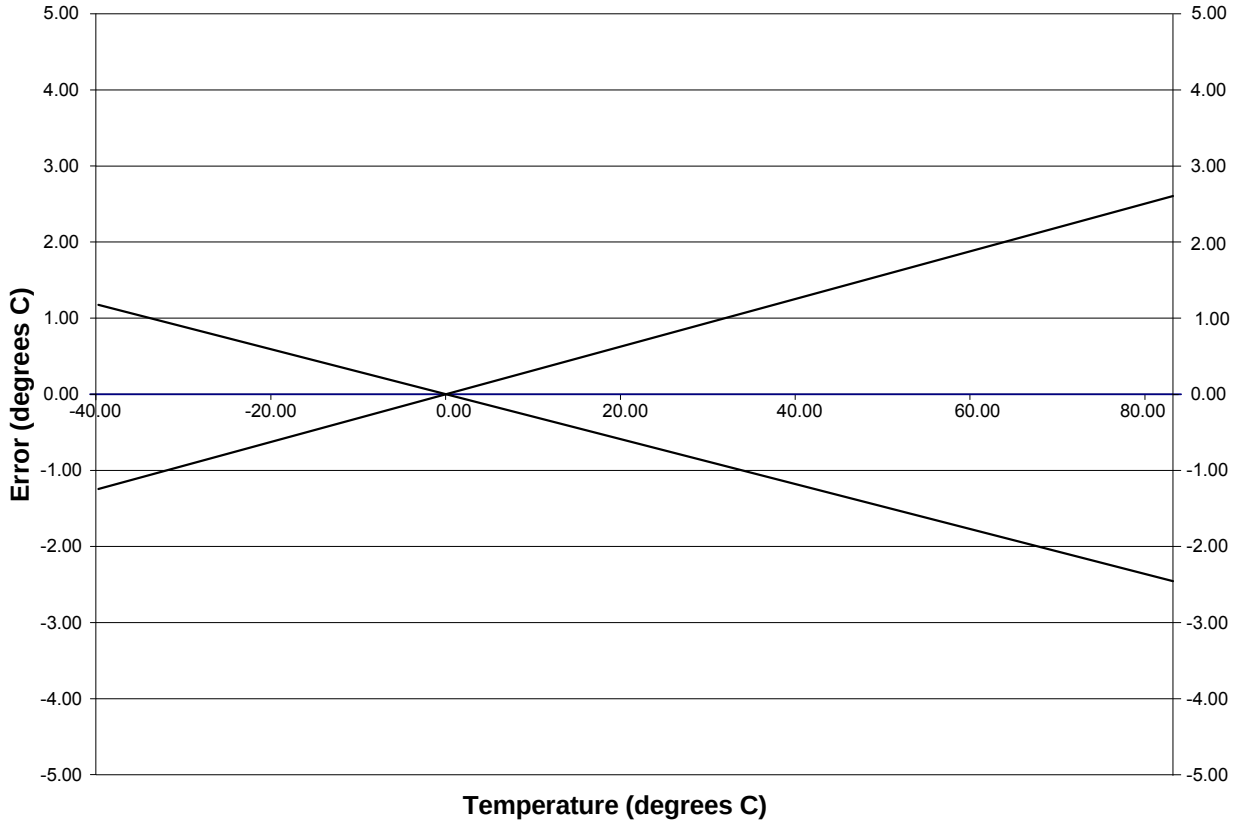


Figure 9.2. Temperature Sensor Error with 1-Point Calibration at 0 Celsius

10. Voltage and Ground Reference Options

The voltage reference MUX is configurable to use an externally connected voltage reference, the on-chip voltage reference, or one of two power supply voltages (see Figure 10.1). The ground reference MUX allows the ground reference for ADC0 to be selected between the ground pin (GND) or a port pin dedicated to analog ground (P0.1/AGND).

The voltage and ground reference options are configured using the REF0CN SFR described on page 62. Electrical specifications are can be found in the Electrical Specifications Chapter.

Important Note About the V_{REF} and AGND Inputs: Port pins are used as the external V_{REF} and AGND inputs. When using an external voltage reference, P0.0/VREF should be configured as an analog input and skipped by the Digital Crossbar. When using AGND as the ground reference to ADC0, P0.1/AGND should be configured as an analog input and skipped by the Digital Crossbar. Refer to Section “26. Port Input/Output” on page 165 for complete Port I/O configuration details. The external reference voltage must be within the range $0 \le V_{REF} \le V_{DD}$ and the external ground reference must be at the same DC voltage potential as GND.

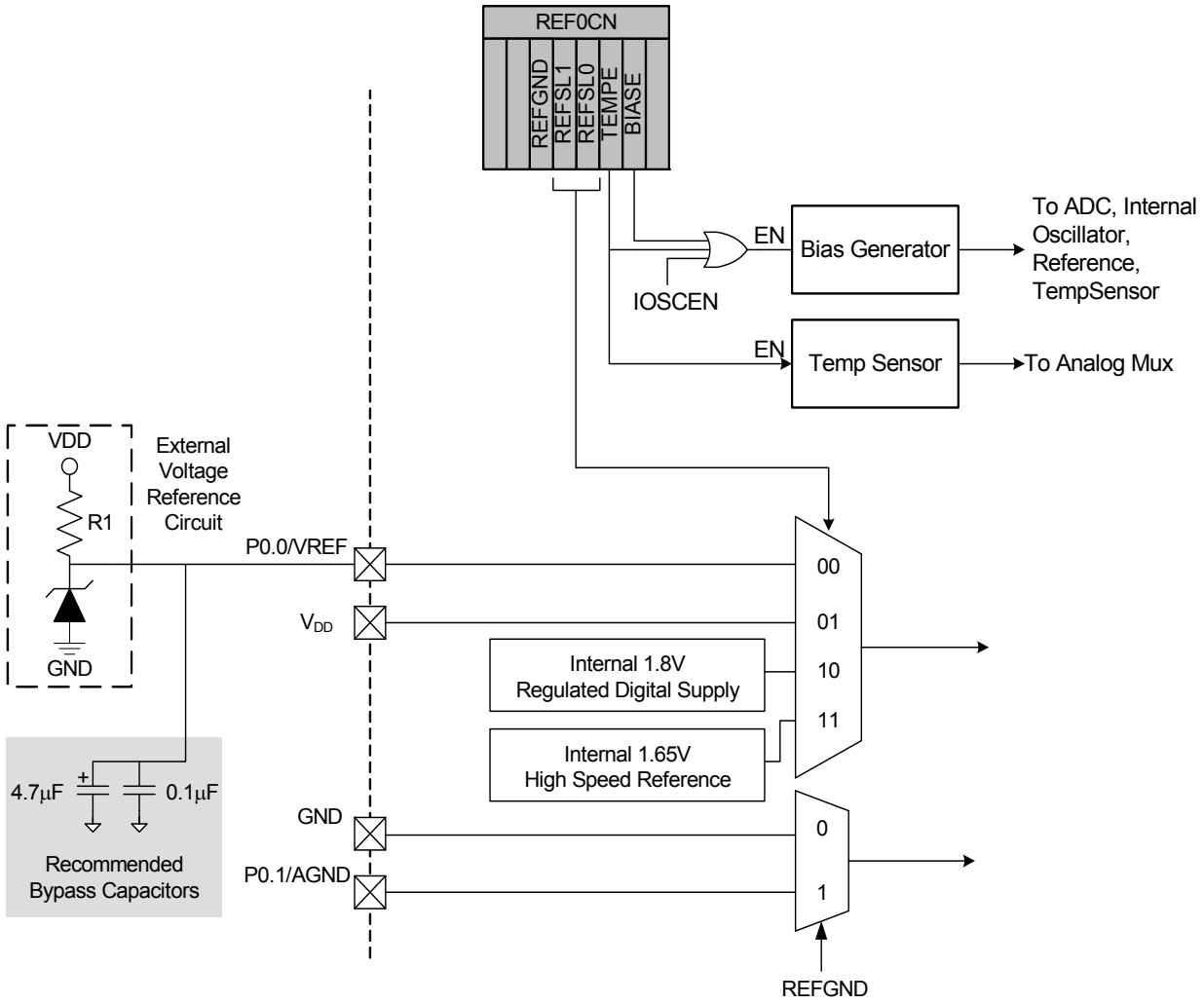


Figure 10.1. Voltage Reference Functional Block Diagram

10.1. External Voltage References

To use an external voltage reference, REFSL[1:0] should be set to 00. Bypass capacitors should be added as recommended by the manufacturer of the external voltage reference.

10.2. Internal Voltage Reference Options

A 1.65 V high-speed reference is included on-chip. The high speed internal reference is selected by setting REFSL[1:0] to 11. When selected, the high speed internal reference will be automatically enabled on an as-needed basis by ADC0.

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

10.3. 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 ADC0 is taken from the P0.1/AGND pin. Any external sensors sampled by ADC0 should be referenced to the P0.1/AGND pin. The separate analog ground reference option is enabled by setting REFGND to 1. Note that when using this option, P0.1/AGND must be connected to the same potential as GND.

10.4. Temperature Sensor Enable

The TEMPE bit in register REF0CN enables the temperature sensor. While disabled, the temperature sensor defaults to a high impedance state and any ADC0 measurements performed on the sensor result in meaningless data.

SFR Definition 10.1. REF0CN: Voltage Reference Control

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

SFR Address = 0xD2; SFR Page = F

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

11. Voltage Regulator (REG0)

C8051F70x/71x devices include an internal voltage regulator (REG0) to regulate the internal core supply to 1.8 V from a V_{DD} supply of 1.8 to 3.6 V. Two power-saving modes are built into the regulator to help reduce current consumption in low-power applications. These modes are accessed through the REG0CN register (SFR Definition 11.1). Electrical characteristics for the on-chip regulator are specified in Table 7.5 on page 42

If an external regulator is used to power the device, the internal regulator may be put into bypass mode using the BYPASS bit. **The internal regulator should never be placed in bypass mode unless an external 1.8 V regulator is used to supply V_{DD}. Doing so could cause permanent damage to the device.**

Under default conditions, when the device enters STOP mode the internal regulator will remain on. 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 $\overline{RST}$ pin or a full power cycle of the device are the only methods of generating a reset.

SFR Definition 11.1. REG0CN: Voltage Regulator Control

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

SFR Address = 0xB9; SFR Page = F

Bit	Name	Function
7	STOPCF	**Stop Mode Configuration.** This bit configures the regulator's behavior when the device enters STOP mode. 0: Regulator is still active in STOP mode. Any enabled reset source will reset the device. 1: Regulator is shut down in STOP mode. Only the $\overline{RST}$ pin or power cycle can reset the device.
6	BYPASS	**Bypass Internal Regulator.** This bit places the regulator in bypass mode, allowing the core to run directly from the V_{DD} supply pin. 0: Normal Mode—Regulator is on and regulates V_{DD} down to the core voltage. 1: Bypass Mode—Regulator is in bypass mode, and the microcontroller core operates directly from the V_{DD} supply voltage. **IMPORTANT: Bypass mode is for use with an external regulator as the supply voltage only. Never place the regulator in bypass mode when the V_{DD} supply voltage is greater than the specifications given in Table 7.1 on page 39. Doing so may cause permanent damage to the device.**
5:0	Reserved	Reserved. Must Write 000000b

12. Comparator0

C8051F70x/71x devices include an on-chip programmable voltage comparator, Comparator0, shown in Figure 12.1.

The Comparator offers programmable response time and hysteresis, an analog input multiplexer, and two outputs that are optionally available at the Port pins: a synchronous “latched” output (CP0), or an asynchronous “raw” output (CP0A). The asynchronous CP0A signal is available even when the system clock is not active. This allows the Comparator to operate and generate an output with the device in STOP mode. When assigned to a Port pin, the Comparator output may be configured as open drain or push-pull (see Section “26.4. Port I/O Initialization” on page 172). Comparator0 may also be used as a reset source (see Section “23.5. Comparator0 Reset” on page 152).

The Comparator0 inputs are selected by the comparator input multiplexer, as detailed in Section “12.1. Comparator Multiplexer” on page 70.

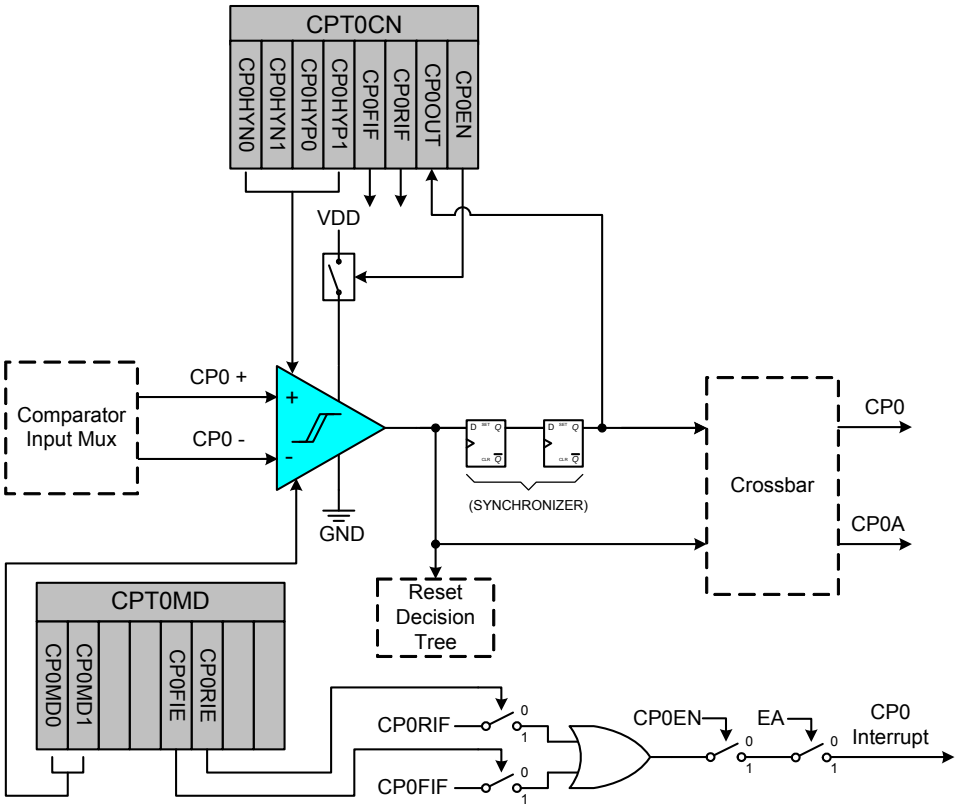


Figure 12.1. Comparator0 Functional Block Diagram

The Comparator output can be polled in software, used as an interrupt source, and/or routed to a Port pin. When routed to a Port pin, the Comparator output is available asynchronous or synchronous to the system clock; the asynchronous output is available even in STOP mode (with no system clock active). When disabled, the Comparator output (if assigned to a Port I/O pin via the Crossbar) defaults to the logic low state, and the power supply to the comparator is turned off. See Section “26.3. Priority Crossbar Decoder” on page 170 for details on configuring Comparator outputs via the digital Crossbar. Comparator inputs can be

externally driven from –0.25 V to (V_{DD}) + 0.25 V without damage or upset. The complete Comparator electrical specifications are given in Section “7. Electrical Characteristics” on page 39.

The Comparator response time may be configured in software via the CPT0MD register (see SFR Definition 12.2). Selecting a longer response time reduces the Comparator supply current.

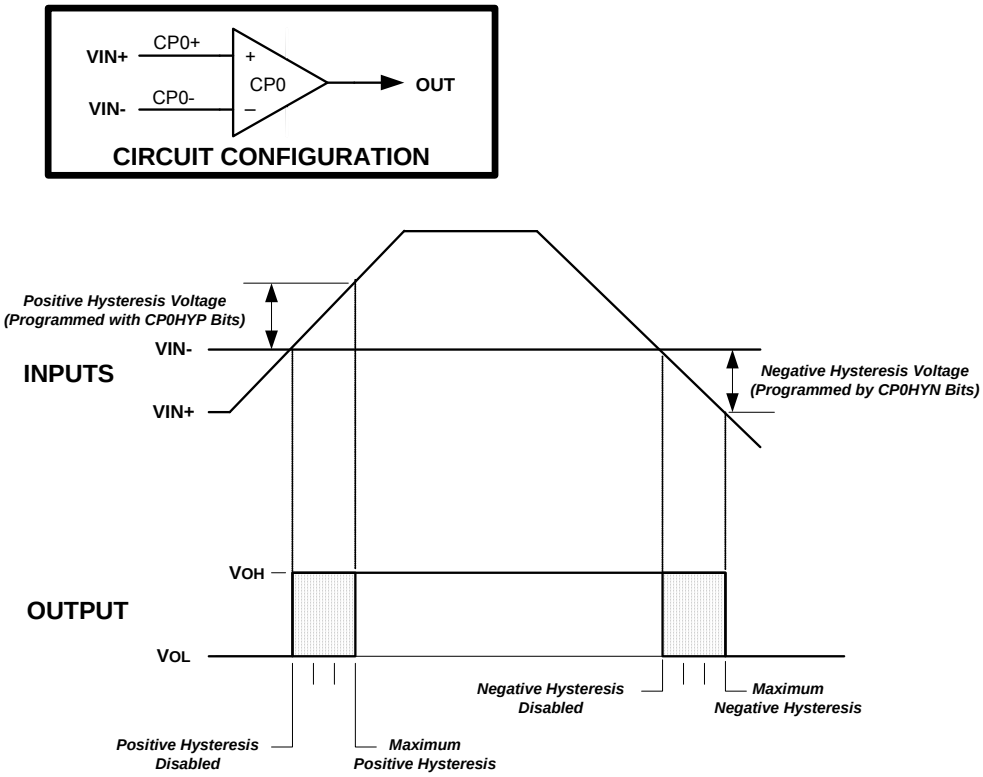


Figure 12.2. Comparator Hysteresis Plot

The Comparator hysteresis is software-programmable via its Comparator Control register CPT0CN. The user can program both the amount of hysteresis voltage (referred to the input voltage) and the positive and negative-going symmetry of this hysteresis around the threshold voltage.

The Comparator hysteresis is programmed using Bits3–0 in the Comparator Control Register CPT0CN (shown in SFR Definition 12.1). The amount of negative hysteresis voltage is determined by the settings of the CP0HYN bits. As shown in Figure 12.2, settings of 20, 10 or 5 mV of negative hysteresis can be programmed, or negative hysteresis can be disabled. In a similar way, the amount of positive hysteresis is determined by the setting the CP0HYP bits.

Comparator interrupts can be generated on both rising-edge and falling-edge output transitions. (For Interrupt enable and priority control, see Section “19.1. MCU Interrupt Sources and Vectors” on page 123). The CP0FIF flag is set to logic 1 upon a Comparator falling-edge occurrence, and the CP0RIF 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 CP0RIE to a logic 1. The Comparator0 falling-edge interrupt mask is enabled by setting CP0FIE to a logic 1.

The output state of the Comparator can be obtained at any time by reading the CP0OUT bit. The Comparator is enabled by setting the CP0EN 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.

SFR Definition 12.1. CPT0CN: Comparator0 Control

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

SFR Address = 0x9B; SFR Page = 0

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

SFR Definition 12.2. CPT0MD: Comparator0 Mode Selection

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

SFR Address = 0x9D; SFR Page = 0

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

12.1. Comparator Multiplexer

C8051F70x/71x devices include an analog input multiplexer to connect Port I/O pins to the comparator inputs. The Comparator0 inputs are selected in the CPT0MX register (SFR Definition 12.3). The CMX0P2–CMX0P0 bits select the Comparator0 positive input; the CMX0N2–CMX0N0 bits select the Comparator0 negative input. **Important Note About Comparator Inputs:** The Port pins selected as comparator inputs should be configured as analog inputs in their associated Port configuration register, and configured to be skipped by the Crossbar (for details on Port configuration, see Section “26.6. Special Function Registers for Accessing and Configuring Port I/O” on page 177).

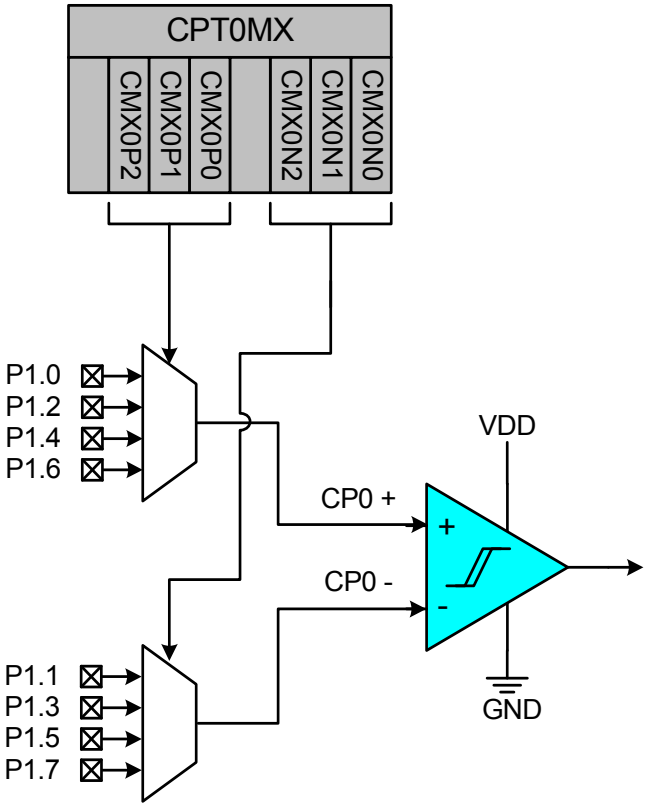


Figure 12.3. Comparator Input Multiplexer Block Diagram

SFR Definition 12.3. CPT0MX: Comparator0 MUX Selection

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

SFR Address = 0x9F; SFR Page = 0

Bit	Name	Function		
7	Unused	Read = 0b; Write = don't care.		
6:4	CMX0N[2:0]	**Comparator0 Negative Input MUX Selection.**		
			64-Pin Devices	**48-Pin Devices**
		000	P1.1	P1.1
		001	P1.3	P1.3
		010	P1.5	Reserved.
		011	P1.7	Reserved.
		100-111	No input selected.	No input selected.
3	Unused	Read = 0b; Write = don't care.		
2:0	CMX0P[2:0]	**Comparator0 Positive Input MUX Selection.**		
			64-Pin Devices	**48-Pin Devices**
		000	P1.0	P1.0
		001	P1.2	P1.2
		010	P1.4	Reserved.
		011	P1.6	Reserved.
		100-111	No input selected.	No input selected.

13. Capacitive Sense (CS0)

The Capacitive Sense subsystem uses a capacitance-to-digital circuit to determine the capacitance on a port pin. The module can take measurements from different port pins using the module's 32-channel analog multiplexer. The module is enabled only when the CS0EN bit (CS0CN) is set to 1. Otherwise the module is in a low-power shutdown state. The module can be configured to take measurements on one port pin or a group of port pins, using auto-scan. An accumulator can be configured to accumulate multiple conversions on an input channel. Interrupts can be generated when CS0 completes a conversion or when the measured value crosses a threshold defined in CS0THH:L.

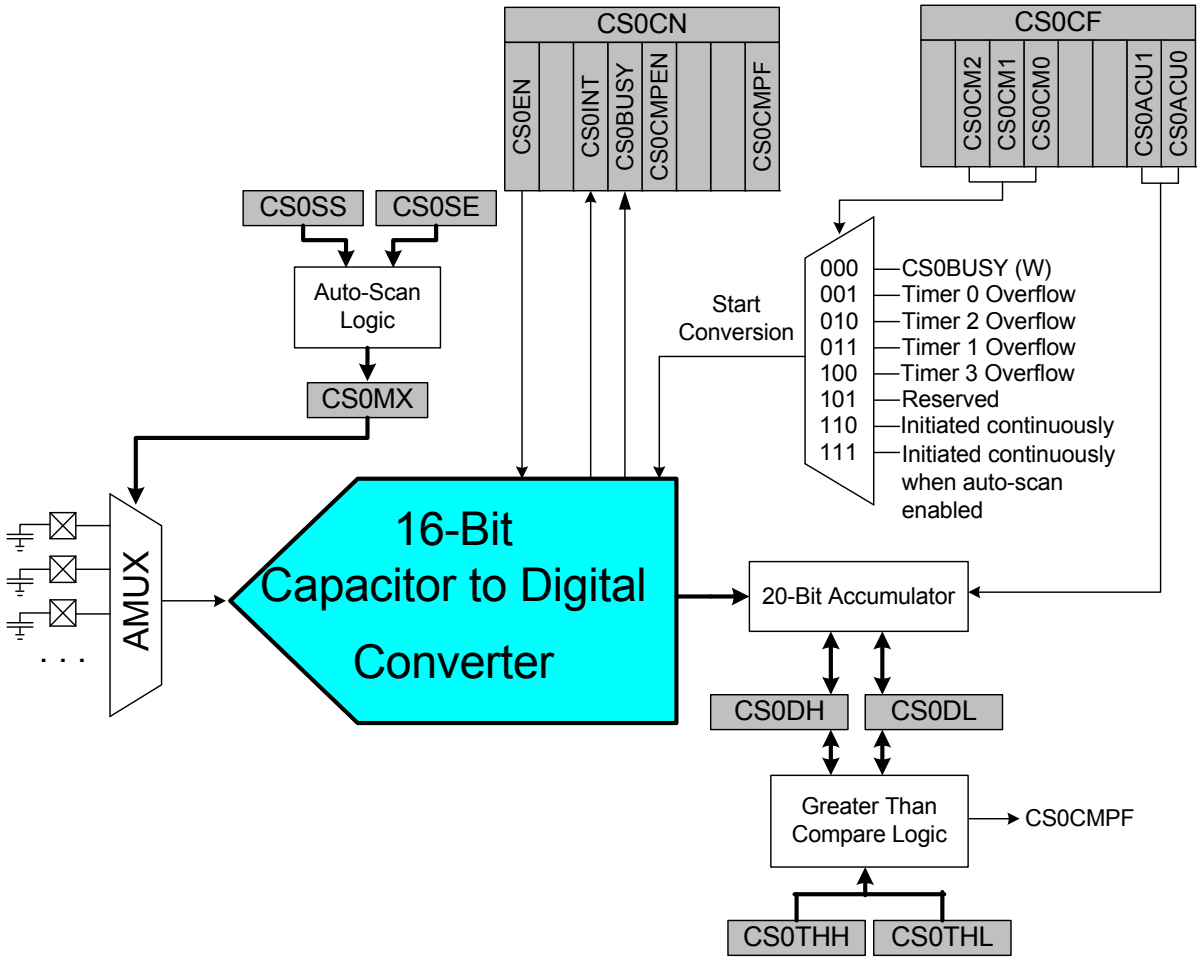


Figure 13.1. CS0 Block Diagram

13.1. Configuring Port Pins as Capacitive Sense Inputs

In order for a port pin to be measured by CS0, that port pin must be configured as an analog input (see “26. Port Input/Output”). Configuring the input multiplexer to a port pin not configured as an analog input will cause the capacitive sense comparator to output incorrect measurements.

13.2. Capacitive Sense Start-Of-Conversion Sources

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

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

Conversions can be configured to be initiated continuously through one of two methods. CS0 can be configured to convert at a single channel continuously or it can be configured to convert continuously with auto-scan enabled. When configured to convert continuously, conversions will begin after the CS0BUSY bit in CS0CF has been set.

An interrupt will be generated if CS0 conversion complete interrupts are enabled by setting the ECSCPT bit (EIE2.0).

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

13.3. Automatic Scanning

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

Auto-scan is enabled by setting the CS0 start-of-conversion bits (CS0CF6:4) to 111b. After enabling auto-scan, the starting and ending channels should be set to appropriate values in CS0SS and CS0SE, respectively. Writing to CS0SS when auto-scan is enabled will cause the value written to CS0SS to be copied into CS0MX. After being enabled, writing a 1 to CS0BUSY will start auto-scan conversions. When auto-scan completes the number of conversions defined in the CS0 accumulator bits (CS0CF1:0) (see “13.5. CS0 Conversion Accumulator”), auto-scan configures CS0MX to the next highest port pin configured as an analog input and begins a conversion on that channel. This scan sequence continues until CS0MX reaches the ending input channel value defined in CS0SE. After one or more conversions have been taken at this channel, auto-scan configures CS0MX back to the starting input channel. For an example system configured to use auto-scan, please see Figure “13.2 Auto-Scan Example” on page 74.

Note: Auto-scan attempts one conversion on a CS0MX channel regardless of whether that channel’s port pin has been configured as an analog input.

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

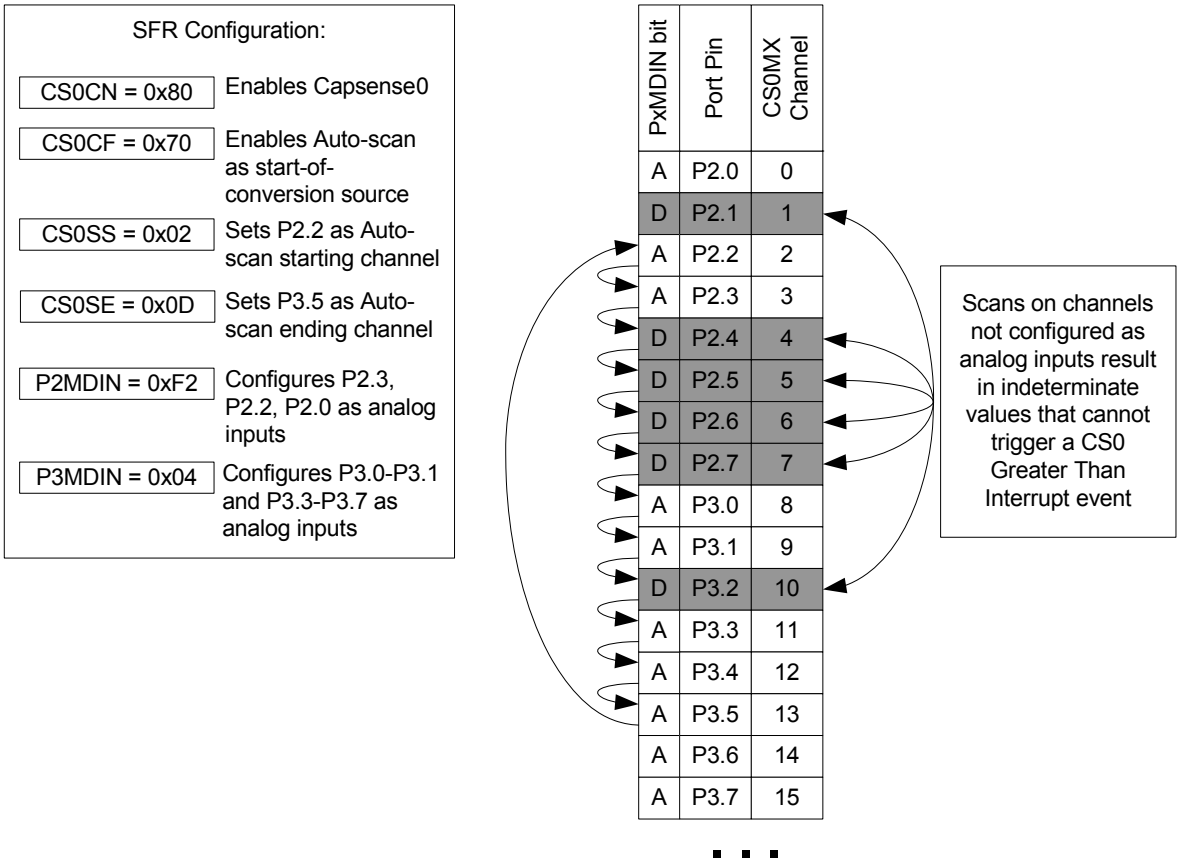


Figure 13.2. Auto-Scan Example

13.4. CS0 Comparator

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

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

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

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

A CS0 greater-than comparator event can wake a device from suspend mode. This feature is useful in systems configured to continuously sample one or more capacitive sense channels. The device will remain in the low-power suspend state until the captured value of one of the scanned channels causes a CS0

greater-than comparator event to occur. It is not necessary to have CS0 comparator interrupts enabled in order to wake a device from suspend with a greater-than event.

For a summary of behavior with different CS0 comparator, auto-scan, and auto accumulator settings, please see Table 13.1.

13.5. CS0 Conversion Accumulator

CS0 can be configured to accumulate multiple conversions on an input channel. The number of samples to be accumulated is configured using the CS0ACU1:0 bits (CS0CF1:0). The accumulator can accumulate 1 sample, 4 samples, 8 samples, or 16 samples. After the defined number of samples have been accumulated, the result is converted to a 16-bit value by dividing the 20-bit accumulator by either 1, 4, 8, or 16 (depending on the CS0ACU[1:0] setting) and copied to the CS0DH:CS0DL SFRs.

Table 13.1. Operation with Auto-scan and Accumulate

Auto-Scan Enabled	Accumulator Enabled	CS0 Conversion Complete Interrupt Behavior	CS0 Greater Than Interrupt Behavior	CS0MX Behavior
N	**N**	CS0INT Interrupt serviced after 1 conversion completes	Interrupt serviced after 1 conversion completes if value in CS0DH:CS0DL is greater than CS0THH:CS0THL	CS0MX unchanged.
N	**Y**	CS0INT Interrupt serviced after *M* conversions complete	Interrupt serviced after *M* conversions complete if value in 16-bit accumulator is greater than CS0THH:CS0THL	CS0MX unchanged.
Y	**N**	CS0INT Interrupt serviced after 1 conversion completes	Interrupt serviced after conversion completes if value in CS0DH:CS0DL is greater than CS0THH:CS0THL; Auto-Scan stopped	If greater-than comparator detects conversion value is greater than CS0THH:CS0THL, CMUX0 is left unchanged; otherwise, CMUX0 updates to the next channel (CS0MX + 1) and wraps back to CS0SS after passing CS0SE
Y	**Y**	CS0INT Interrupt serviced after *M* conversions complete	Interrupt serviced after *M* conversions complete if value in 16-bit accumulator is greater than CS0THH:CS0THL; Auto-Scan stopped	If greater-than comparator detects conversion value is greater than CS0THH:CS0THL, CS0MX is left unchanged; otherwise, CS0MX updates to the next channel (CS0MX + 1) and wraps back to CS0SS after passing CS0SE
M = Accumulator setting (1x, 4x, 8x, 16x)				

SFR Definition 13.1. CS0CN: Capacitive Sense Control

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

SFR Address = 0x9A; SFR Page = 0

Bit	Name	Description
7	CS0EN	**CS0 Enable.** 0: CS0 disabled and in low-power mode. 1: CS0 enabled and ready to convert.
6	Unused	Read = 0b; Write = Don't care
5	CS0INT	**CS0 Interrupt Flag.** 0: CS0 has not completed a data conversion since the last time CS0INT was cleared. 1: CS0 has completed a data conversion. This bit is not automatically cleared by hardware.
4	CS0BUSY	**CS0 Busy.** Read: 0: CS0 conversion is complete or a conversion is not currently in progress. 1: CS0 conversion is in progress. Write: 0: No effect. 1: Initiates CS0 conversion if CS0CM[2:0] = 000b, 110b, or 111b.
3	CS0CMPEN	**CS0 Digital Comparator Enable Bit.** Enables the digital comparator, which compares accumulated CS0 conversion output to the value stored in CS0THH:CS0THL. 0: CS0 digital comparator disabled. 1: CS0 digital comparator enabled.
2:1	Unused	Read = 00b; Write = Don't care
0	CS0CMPF	**CS0 Digital Comparator Interrupt Flag.** 0: CS0 result is smaller than the value set by CS0THH and CS0THL since the last time CS0CMPF was cleared. 1: CS0 result is greater than the value set by CS0THH and CS0THL since the last time CS0CMPF was cleared.

SFR Definition 13.2. CS0CF: Capacitive Sense Configuration

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

SFR Address = 0x9E; SFR Page = 0

Bit	Name	Description
7	Unused	Read = 0b; Write = Don't care
6:4	CS0CM[2:0]	**CS0 Start of Conversion Mode Select.** 000: Conversion initiated on every write of 1 to CS0BUSY. 001: Conversion initiated on overflow of Timer 0. 010: Conversion initiated on overflow of Timer 2. 011: Conversion initiated on overflow of Timer 1. 100: Conversion initiated on overflow of Timer 3. 101: Reserved. 110: Conversion initiated continuously after writing 1 to CS0BUSY. 111: Auto-scan enabled, conversions initiated continuously after writing 1 to CS0BUSY.
3:2	Unused	Read = 00b; Write = Don't care
1:0	CS0ACU[1:0]	**CS0 Accumulator Mode Select.** 00: Accumulate 1 sample. 01: Accumulate 4 samples. 10: Accumulate 8 samples. 11: Accumulate 16 samples.

SFR Definition 13.3. CS0DH: Capacitive Sense Data High Byte

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

SFR Address = 0xAA; SFR Page = 0

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

SFR Definition 13.4. CS0DL: Capacitive Sense Data Low Byte

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

SFR Address = 0xA9; SFR Page = 0

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

SFR Definition 13.5. CS0SS: Capacitive Sense Auto-Scan Start Channel

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

SFR Address = 0x92; SFR Page = F

Bit	Name	Description
7:5	Unused	Read = 000b; Write = Don't care
	CS0SS[4:0]	**Starting Channel for Auto-Scan.** Sets the first CS0 channel to be selected by the mux for Capacitive Sense conversion when auto-scan is enabled and active. When auto-scan is enabled, a write to CS0SS will also update CS0MX.

SFR Definition 13.6. CS0SE: Capacitive Sense Auto-Scan End Channel

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

SFR Address = 0x93; SFR Page = F

Bit	Name	Description
7:5	Unused	Read = 000b; Write = Don't care
	CS0SE[4:0]	**Ending Channel for Auto-Scan.** Sets the last CS0 channel to be selected by the mux for Capacitive Sense conversion when auto-scan is enabled and active.

SFR Definition 13.7. CS0THH: Capacitive Sense Comparator Threshold High Byte

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

SFR Address = 0x97; SFR Page = 0

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

SFR Definition 13.8. CS0THL: Capacitive Sense Comparator Threshold Low Byte

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

SFR Address = 0x96; SFR Page = 0

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

13.6. Capacitive Sense Multiplexer

The input multiplexer can be controlled through two methods. The CS0MX register can be written to through firmware, or the register can be configured automatically using the modules auto-scan functionality (see “13.3. Automatic Scanning”).

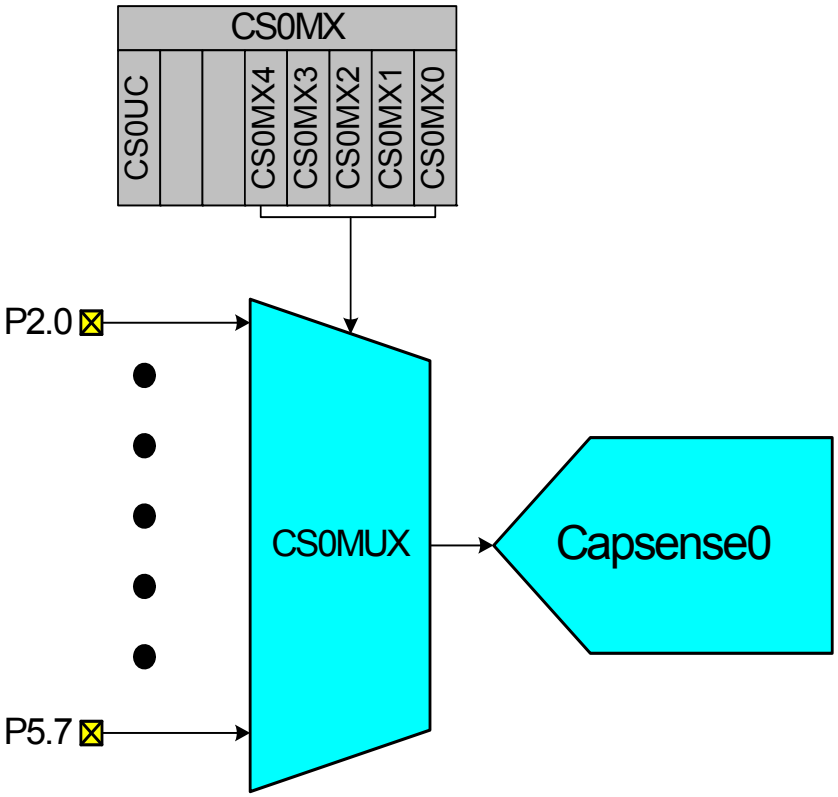


Figure 13.3. CS0 Multiplexer Block Diagram

SFR Definition 13.9. CS0MX: Capacitive Sense Mux Channel Select

Bit	7	6	5	4	3	2	1	0
Name	CS0UC			CS0MX[4:0]				
Type	W/R	R	R	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x9C; SFR Page = 0

Bit	Name	Description
7	CS0UC	**CS0 Unconnected.** Disconnects CS0 from all port pins, regardless of the selected channel. 0: CS0 connected to port pins 1: CS0 disconnected from port pins
6:5	Reserved	Read = 00b; Write = 00b
4:0	CS0MX[4:0]	**CS0 Mux Channel Select.** Selects one of the 32 input channels for Capacitive Sense conversion.

Value	64-pin packages	48-pin package	Value	64-pin packages	48-pin package
00000	P2.0	P2.0	**10000**	P4.0	P4.0
00001	P2.1	P2.1	**10001**	P4.1	P4.1
00010	P2.2	P2.2	**10010**	P4.2	P4.2
00011	P2.3	P2.3	**10011**	P4.3	P4.3
00100	P2.4	P2.4	**10100**	P4.4	Reserved.
00101	P2.5	P2.5	**10101**	P4.5	Reserved.
00110	P2.6	P2.6	**10110**	P4.6	Reserved.
00111	P2.7	P2.7	**10111**	P4.7	Reserved.
01000	P3.0	Reserved.	**11000**	P5.0	P5.0
01001	P3.1	Reserved.	**11001**	P5.1	P5.1
01010	P3.2	Reserved.	**11010**	P5.2	P5.2
01011	P3.3	Reserved	**11011**	P5.3	P5.3
01100	P3.4	P3.4	**11100**	P5.4	P5.4
01101	P3.5	P3.5	**11101**	P5.5	P5.5
01110	P3.6	P3.6	**11110**	P5.6	P5.6
01111	P3.7	P3.7	**11111**	P5.7	P5.7

14. CIP-51 Microcontroller

The MCU system controller core is the CIP-51 microcontroller. The CIP-51 is fully compatible with the MCS-51™ instruction set; standard 803x/805x assemblers and compilers can be used to develop software. The MCU family has a superset of all the peripherals included with a standard 8051. The CIP-51 also includes on-chip debug hardware (see description in Section 33), 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 14.1 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

Performance

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

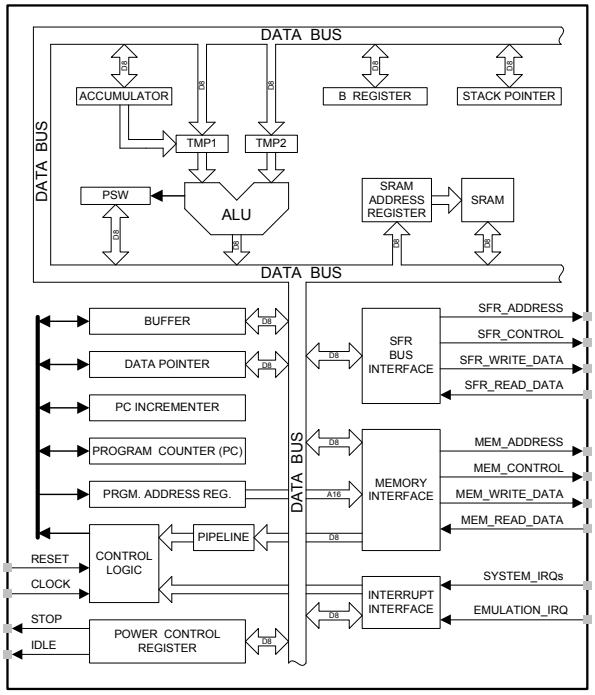


Figure 14.1. CIP-51 Block Diagram

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

14.1. Instruction Set

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

14.1.1. Instruction and CPU Timing

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

Due to the pipelined architecture of the CIP-51, most instructions execute in the same number of clock cycles as there are program bytes in the instruction. Conditional branch instructions take one less clock cycle to complete when the branch is not taken as opposed to when the branch is taken. Table 14.1 is the CIP-51 Instruction Set Summary, which includes the mnemonic, number of bytes, and number of clock cycles for each instruction.

Table 14.1. CIP-51 Instruction Set Summary

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

Table 14.1. CIP-51 Instruction Set Summary (Continued)

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

Table 14.1. CIP-51 Instruction Set Summary (Continued)

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

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

14.2. CIP-51 Register Descriptions

Following are descriptions of SFRs related to the operation of the CIP-51 System Controller. Reserved bits should always be written to the value indicated in the SFR description. Future product versions may use these bits to implement new features in which case the reset value of the bit will be the indicated value, selecting the feature's default state. Detailed descriptions of the remaining SFRs are included in the sections of the data sheet associated with their corresponding system function.

SFR Definition 14.1. DPL: Data Pointer Low Byte

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

SFR Address = 0x82; SFR Page = All Pages

Bit	Name	Function
7:0	DPL[7:0]	**Data Pointer Low.** The DPL register is the low byte of the 16-bit DPTR.

SFR Definition 14.2. DPH: Data Pointer High Byte

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

SFR Address = 0x83; SFR Page = All Pages

Bit	Name	Function
7:0	DPH[7:0]	**Data Pointer High.** The DPH register is the high byte of the 16-bit DPTR.

SFR Definition 14.3. SP: Stack Pointer

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

SFR Address = 0x81; SFR Page = All Pages

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

SFR Definition 14.4. ACC: Accumulator

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

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

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

SFR Definition 14.5. B: B Register

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

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

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

SFR Definition 14.6. PSW: Program Status Word

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

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

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

15. 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 C8051F70x/71x device family is shown in Figure 15.1

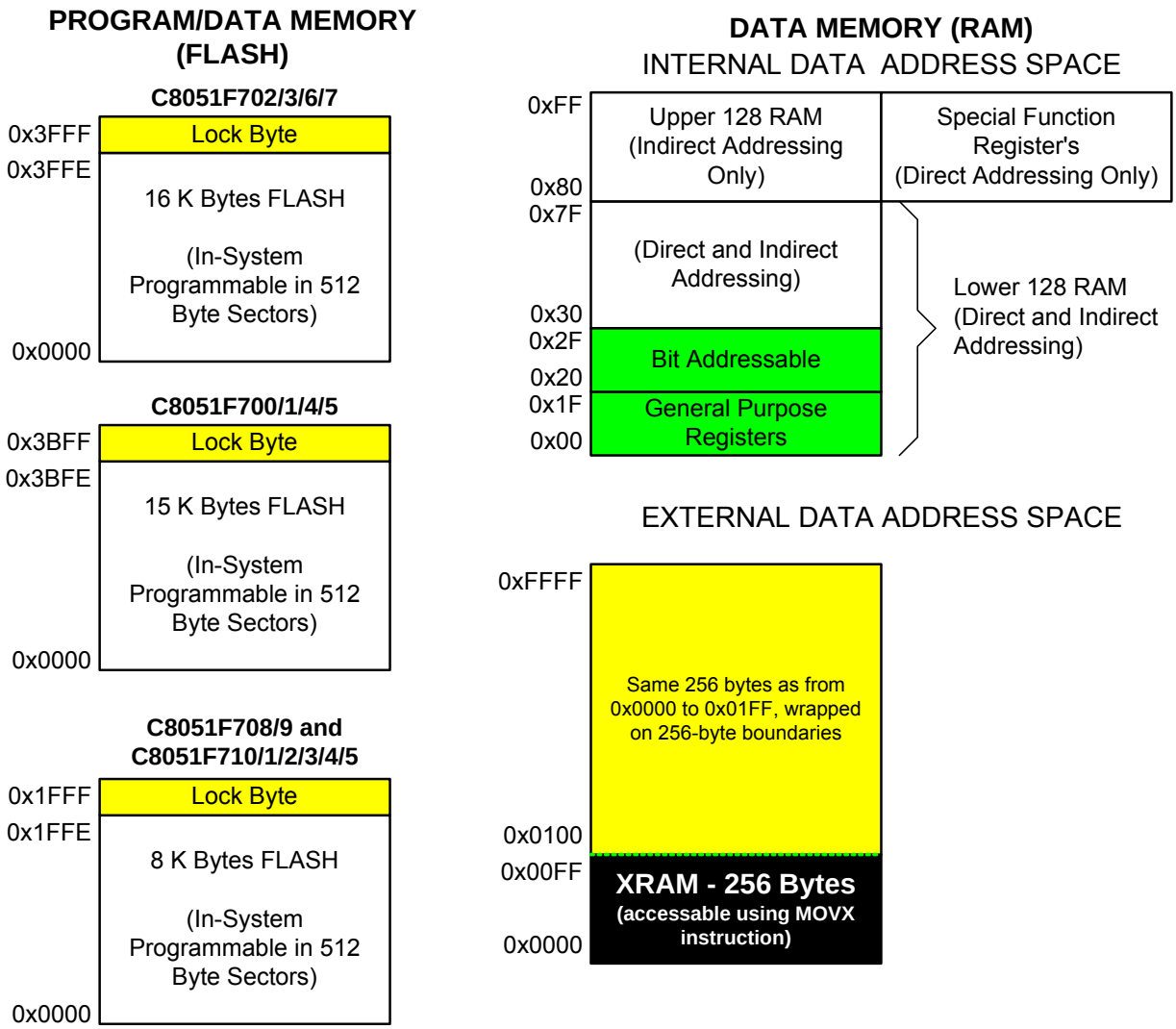


Figure 15.1. C8051F70x/71x Memory Map

15.1. Program Memory

The members of the C8051F70x/71x device family contain 16 kB (C8051F702/3/6/7), 15 kB (C8051F700/1/4/5), or 8 kB (C8051F708/9 and C8051F710/1/2/3/4/5) of re-programmable Flash memory that can be used as non-volatile program or data storage. The last byte of user code space is used as the security lock byte (0x3FFF on 16 kB devices, 0x3BFF on 15 kB devices and 0x1FFF on 8 kB devices).

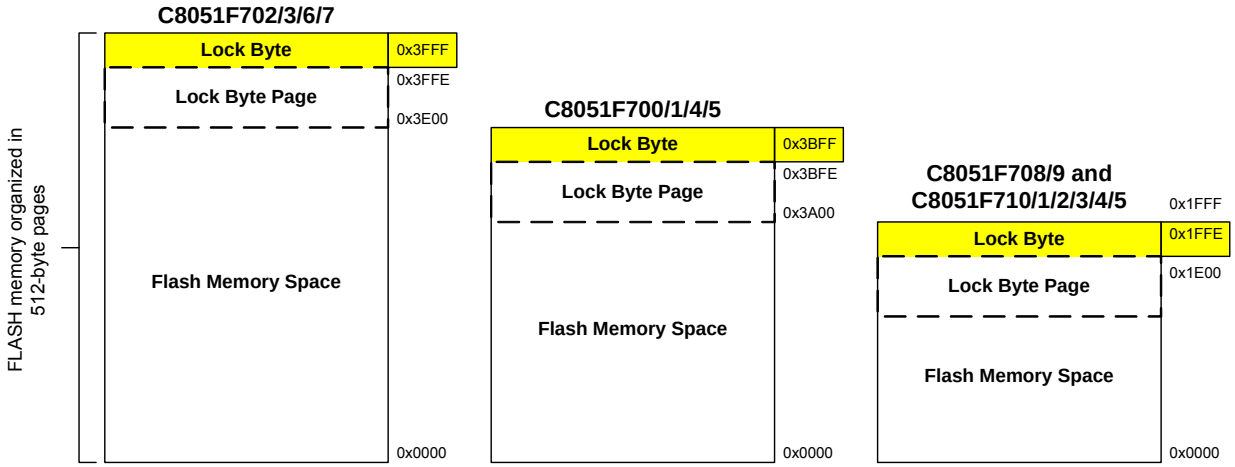


Figure 15.2. Flash Program Memory Map

15.1.1. MOVX Instruction and Program Memory

The MOVX instruction in an 8051 device is typically used to access external data memory. On the C8051F70x/71x 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 C8051F70x/71x to update program code and use the program memory space for non-volatile data storage. Refer to Section “20. Flash Memory” on page 133 for further details.

15.2. EEPROM Memory

The C8051F700/1/4/5/8/9 and C8051F712/3 contain EEPROM emulation hardware, which uses Flash memory to emulate a 32-byte EEPROM memory space for non-volatile data storage. The EEPROM data is accessed through a RAM buffer for increased speed. More details about the EEPROM can be found in Section “21. EEPROM” on page 140.

15.3. Data Memory

The C8051F70x/71x device family includes 512 bytes of RAM data memory. 256 bytes of this memory is mapped into the internal RAM space of the 8051. 256 bytes of this memory is on-chip “external” memory. The data memory map is shown in Figure 15.1 for reference.

15.3.1. 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 15.1 illustrates the data memory organization of the C8051F70x/71x.

15.3.1.1. General Purpose Registers

The lower 32 bytes of data memory, locations 0x00 through 0x1F, may be addressed as four banks of general-purpose registers. Each bank consists of eight byte-wide registers designated R0 through R7. Only one of these banks may be enabled at a time. Two bits in the program status word, RS0 (PSW.3) and RS1 (PSW.4), select the active register bank (see description of the PSW in SFR Definition 14.6). This allows fast context switching when entering subroutines and interrupt service routines. Indirect addressing modes use registers R0 and R1 as index registers.

15.3.1.2. 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.

15.3.1.3. 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.

16. External Data Memory Interface and On-Chip XRAM

For C8051F70x/71x devices, 256 B of RAM are included on-chip and mapped into the external data memory space (XRAM). Additionally, an External Memory Interface (EMIF) is available on the C8051F700/1/2/3/8/9 and C8051F710/1 devices, which can be used to access off-chip data memories and memory-mapped devices connected to the GPIO ports. The external memory space may be accessed using the external move instruction (MOVX) and the data pointer (DPTR), or using the MOVX indirect addressing mode using R0 or R1. If the MOVX instruction is used with an 8-bit address operand (such as @R1), then the high byte of the 16-bit address is provided by the External Memory Interface Control Register (EMI0CN, shown in SFR Definition 16.1).

Note: The MOVX instruction can also be used for writing to the Flash memory. See Section "20. Flash Memory" on page 133 for details. The MOVX instruction accesses XRAM by default.

16.1. Accessing XRAM

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

16.1.1. 16-Bit MOVX Example

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

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

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

16.1.2. 8-Bit MOVX Example

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

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

16.2. Configuring the External Memory Interface

Configuring the External Memory Interface consists of five steps:

1. Configure the Output Modes of the associated port pins as either push-pull or open-drain (push-pull is most common).
2. Configure Port latches to “park” the EMIF pins in a dormant state (usually by setting them to logic 1).
3. Select Multiplexed mode or Non-multiplexed mode.
4. Select the memory mode (on-chip only, split mode without bank select, split mode with bank select, or off-chip only).
5. Set up timing to interface with off-chip memory or peripherals.

Each of these five steps is explained in detail in the following sections. The Port selection, Multiplexed mode selection, and Mode bits are located in the EMI0CF register shown in SFR Definition .

16.3. Port Configuration

The EMIF pinout is shown in Figure 16.2 on Page 112

The External Memory Interface claims the associated Port pins for memory operations ONLY during the execution of an off-chip MOVX instruction. Once the MOVX instruction has completed, control of the Port pins reverts to the Port latches for those pins. See Section “26. Port Input/Output” on page 165 for more information about Port operation and configuration. **The Port latches should be explicitly configured to “park” the External Memory Interface pins in a dormant state, most commonly by setting them to a logic 1**.

During the execution of the MOVX instruction, the External Memory Interface will explicitly disable the drivers on all Port pins that are acting as Inputs (Data[7:0] during a READ operation, for example). The Output mode of the Port pins (whether the pin is configured as Open-Drain or Push-Pull) is unaffected by the External Memory Interface operation, and remains controlled by the PnMDOUT registers. In most cases, the output modes of all EMIF pins should be configured for push-pull mode.

SFR Definition 16.1. EMI0CN: External Memory Interface Control

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

SFR Address = 0xAA; SFR Page = F

Bit	Name	Function
7:0	PGSEL[7:0]	**XRAM Page Select Bits.** The XRAM Page Select Bits provide the high byte of the 16-bit external data memory address when using an 8-bit MOVX command, effectively selecting a 256-byte page of RAM. 0x00: 0x0000 to 0x00FF 0x01: 0x0100 to 0x01FF ... 0xFE: 0xFE00 to 0xFEFF 0xFF: 0xFF00 to 0xFFFF

SFR Definition 16.2. EMI0CF: External Memory Configuration

Bit	7	6	5	4	3	2	1	0
Name				EMD2	EMD[1:0]		EALE[1:0]	
Type	R			R/W				
Reset	0	0	0	0	0	0	1	1

SFR Address = 0xC7; SFR Page = F

Bit	Name	Function
7:5	Unused	Read = 000b; Write = Don't Care.
4	EMD2	**EMIF Multiplex Mode Select Bit.** 0: EMIF operates in multiplexed address/data mode 1: EMIF operates in non-multiplexed mode (separate address and data pins)
3:2	EMD[1:0]	**EMIF Operating Mode Select Bits.** 00: Internal Only: MOVX accesses on-chip XRAM only. All effective addresses alias to on-chip memory space 01: Split Mode without Bank Select: Accesses below the 256 B boundary are directed on-chip. Accesses above the 256 B boundary are directed off-chip. 8-bit off-chip MOVX operations use current contents of the Address high port latches to resolve the upper address byte. To access off chip space, EMI0CN must be set to a page that is not contained in the on-chip address space. 10: Split Mode with Bank Select: Accesses below the 256 B boundary are directed on-chip. Accesses above the 256 B boundary are directed off-chip. 8-bit off-chip MOVX operations uses the contents of EMI0CN to determine the high-byte of the address. 11: External Only: MOVX accesses off-chip XRAM only. On-chip XRAM is not visible to the CPU.
1:0	EALE[1:0]	**ALE Pulse-Width Select Bits.** These bits only have an effect when EMD2 = 0. 00: ALE high and ALE low pulse width = 1 SYSCLK cycle. 01: ALE high and ALE low pulse width = 2 SYSCLK cycles. 10: ALE high and ALE low pulse width = 3 SYSCLK cycles. 11: ALE high and ALE low pulse width = 4 SYSCLK cycles.

16.4. Multiplexed and Non-multiplexed Selection

The External Memory Interface is capable of acting in a Multiplexed mode or a Non-multiplexed mode, depending on the state of the EMD2 (EMI0CF.4) bit.

16.4.1. Multiplexed Configuration

In Multiplexed mode, the Data Bus and the lower 8-bits of the Address Bus share the same Port pins: AD[7:0]. In this mode, an external latch (74HC373 or equivalent logic gate) is used to hold the lower 8-bits of the RAM address. The external latch is controlled by the ALE (Address Latch Enable) signal, which is driven by the External Memory Interface logic. An example of a Multiplexed Configuration is shown in Figure 16.1.

In Multiplexed mode, the external MOVX operation can be broken into two phases delineated by the state of the ALE signal. During the first phase, ALE is high and the lower 8-bits of the Address Bus are presented to AD[7:0]. During this phase, the address latch is configured such that the Q outputs reflect the states of the 'D' inputs. When ALE falls, signaling the beginning of the second phase, the address latch outputs remain fixed and are no longer dependent on the latch inputs. Later in the second phase, the Data Bus controls the state of the AD[7:0] port at the time $\overline{RD}$ or $\overline{WR}$ is asserted.

See Section "16.6.2. Multiplexed Mode" on page 108 for more information.

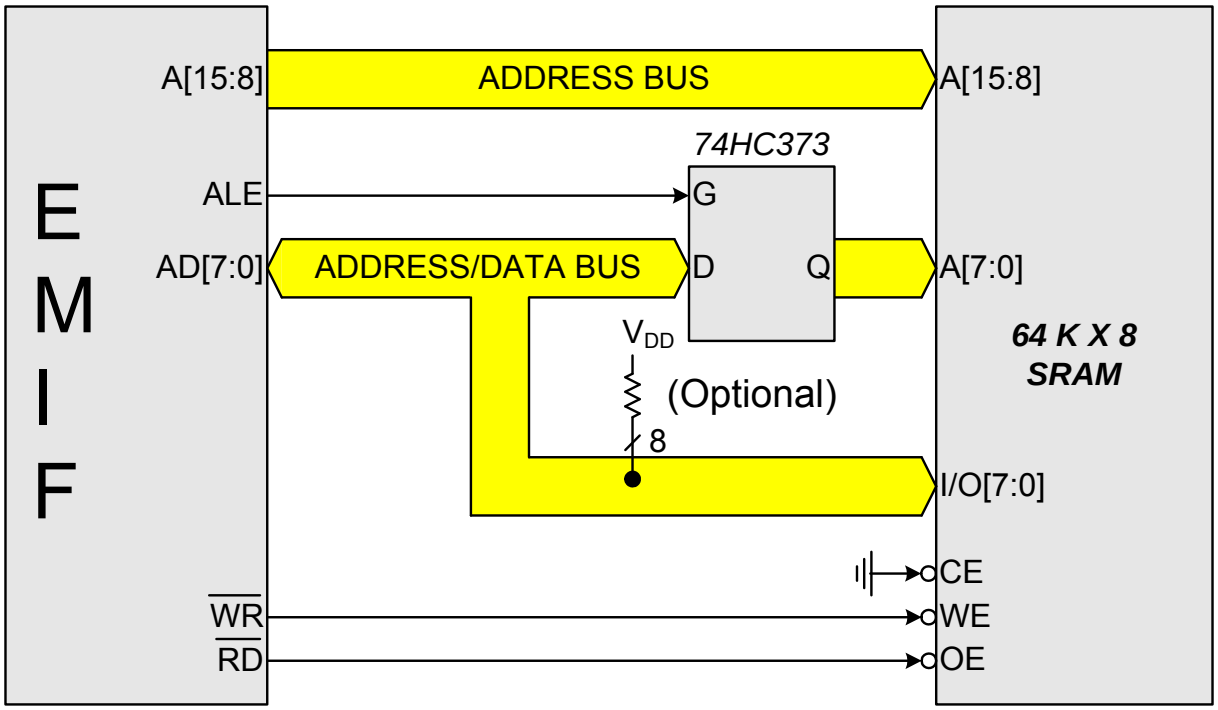


Figure 16.1. Multiplexed Configuration Example

16.4.2. Non-multiplexed Configuration

In Non-multiplexed mode, the Data Bus and the Address Bus pins are not shared. An example of a Non-multiplexed Configuration is shown in Figure 16.2. See Section “16.6.1. Non-Multiplexed Mode” on page 105 for more information about Non-multiplexed operation.

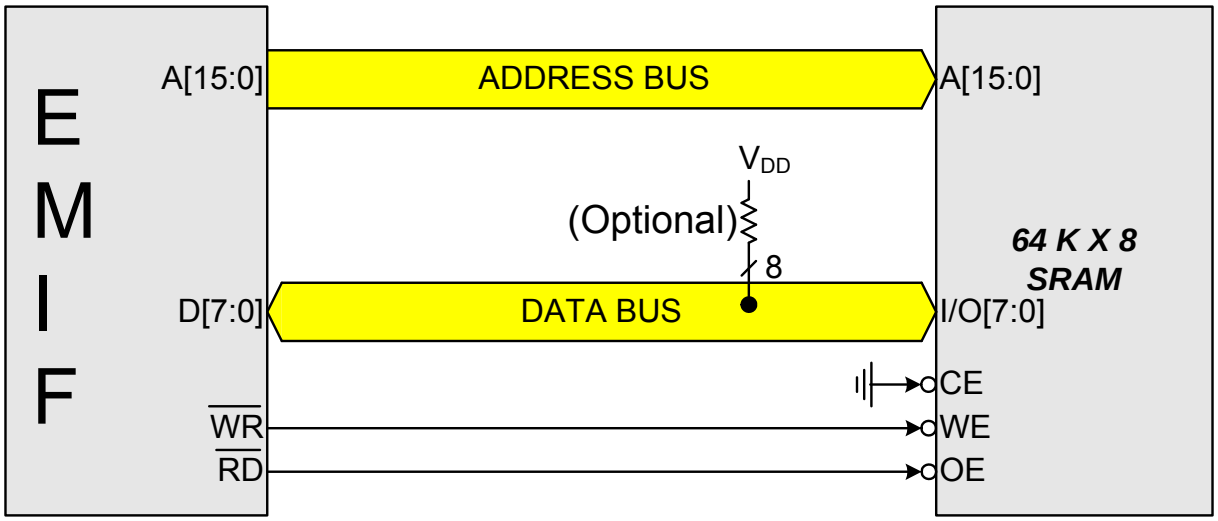


Figure 16.2. Non-multiplexed Configuration Example

16.5. Memory Mode Selection

The external data memory space can be configured in one of four modes, shown in Figure 16.3, based on the EMIF Mode bits in the EMI0CF register (SFR Definition 16.2). These modes are summarized below. More information about the different modes can be found in Section “16.6. Timing” on page 103.

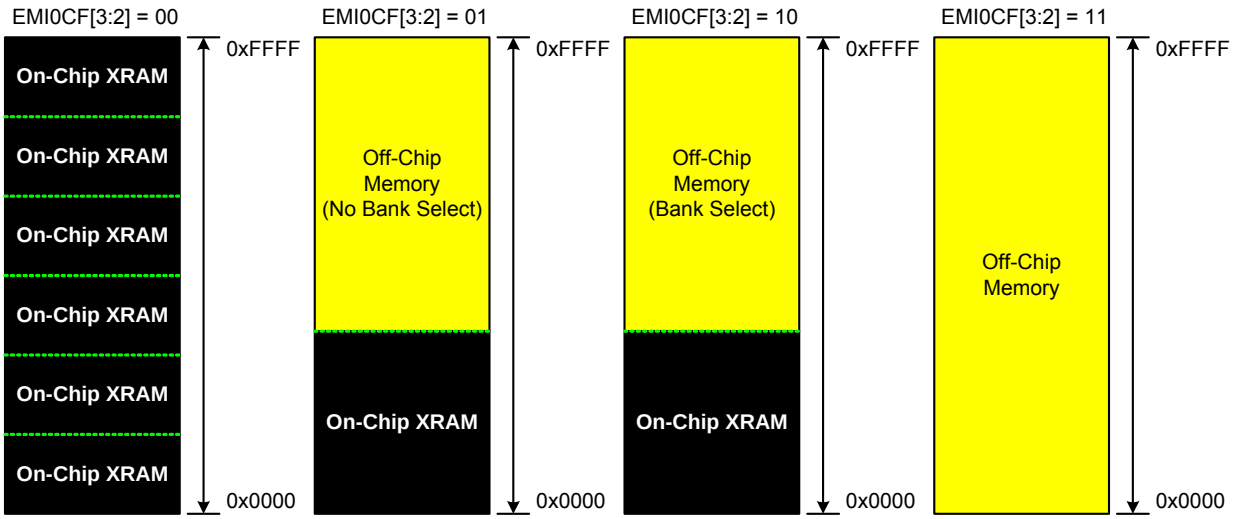


Figure 16.3. EMIF Operating Modes

16.5.1. Internal XRAM Only

When bits EMI0CF[3:2] are set to 00, all MOVX instructions will target the internal XRAM space on the device. Memory accesses to addresses beyond the populated space will wrap on 4 kB boundaries. As an example, the addresses 0x1000 and 0x2000 both evaluate to address 0x0000 in on-chip XRAM space.

- 8-bit MOVX operations use the contents of EMI0CN to determine the high-byte of the effective address and R0 or R1 to determine the low-byte of the effective address.
- 16-bit MOVX operations use the contents of the 16-bit DPTR to determine the effective address.

16.5.2. Split Mode without Bank Select

When bit EMI0CF.[3:2] are set to 01, the XRAM memory map is split into two areas, on-chip space and off-chip space.

- Effective addresses below the internal XRAM size boundary will access on-chip XRAM space.
- Effective addresses above the internal XRAM size boundary will access off-chip space.
- 8-bit MOVX operations use the contents of EMI0CN to determine whether the memory access is on-chip or off-chip. However, in the “No Bank Select” mode, an 8-bit MOVX operation will not drive the upper 8-bits A[15:8] of the Address Bus during an off-chip access. This allows the user to manipulate the upper address bits at will by setting the Port state directly via the port latches. This behavior is in contrast with “Split Mode with Bank Select” described below. The lower 8-bits of the Address Bus A[7:0] are driven, determined by R0 or R1.
- 16-bit MOVX operations use the contents of DPTR to determine whether the memory access is on-chip or off-chip, and unlike 8-bit MOVX operations, the full 16-bits of the Address Bus A[15:0] are driven during the off-chip transaction.

16.5.3. Split Mode with Bank Select

When EMI0CF[3:2] are set to 10, the XRAM memory map is split into two areas, on-chip space and off-chip space.

- Effective addresses below the internal XRAM size boundary will access on-chip XRAM space.
- Effective addresses above the internal XRAM size boundary will access off-chip space.
- 8-bit MOVX operations use the contents of EMI0CN to determine whether the memory access is on-chip or off-chip. The upper 8-bits of the Address Bus A[15:8] are determined by EMI0CN, and the lower 8-bits of the Address Bus A[7:0] are determined by R0 or R1. All 16-bits of the Address Bus A[15:0] are driven in “Bank Select” mode.
- 16-bit MOVX operations use the contents of DPTR to determine whether the memory access is on-chip or off-chip, and the full 16-bits of the Address Bus A[15:0] are driven during the off-chip transaction.

16.5.4. External Only

When EMI0CF[3:2] are set to 11, all MOVX operations are directed to off-chip space. On-chip XRAM is not visible to the CPU. This mode is useful for accessing off-chip memory located between 0x0000 and the internal XRAM size boundary.

- 8-bit MOVX operations ignore the contents of EMI0CN. The upper Address bits A[15:8] are not driven (identical behavior to an off-chip access in “Split Mode without Bank Select” described above). This allows the user to manipulate the upper address bits at will by setting the Port state directly. The lower 8-bits of the effective address A[7:0] are determined by the contents of R0 or R1.
- 16-bit MOVX operations use the contents of DPTR to determine the effective address A[15:0]. The full 16-bits of the Address Bus A[15:0] are driven during the off-chip transaction.

16.6. Timing

The timing parameters of the External Memory Interface can be configured to enable connection to devices having different setup and hold time requirements. The Address Setup time, Address Hold time, $\overline{RD}$ and $\overline{WR}$ strobe widths, and in multiplexed mode, the width of the ALE pulse are all programmable in units of SYSCLK periods through EMI0TC, shown in SFR Definition 16.3, and EMI0CF[1:0].

The timing for an off-chip MOVX instruction can be calculated by adding 4 SYSCLK cycles to the timing parameters defined by the EMI0TC register. Assuming non-multiplexed operation, the minimum execution time for an off-chip XRAM operation is 5 SYSCLK cycles (1 SYSCLK for $\overline{RD}$ or $\overline{WR}$ pulse + 4 SYSCLKs). For multiplexed operations, the Address Latch Enable signal will require a minimum of 2 additional SYSCLK cycles. Therefore, the minimum execution time for an off-chip XRAM operation in multiplexed mode is 7 SYSCLK cycles (2 for /ALE + 1 for $\overline{RD}$ or $\overline{WR}$ + 4). The programmable setup and hold times default to the maximum delay settings after a reset. Table 16.1 lists the ac parameters for the External Memory Interface, and Figure 16.4 through Figure 16.9 show the timing diagrams for the different External Memory Interface modes and MOVX operations.

SFR Definition 16.3. EMI0TC: External Memory Timing Control

Bit	7	6	5	4	3	2	1	0
Name	EAS[1:0]		EWR[3:0]				EAH[1:0]	
Type	R/W		R/W				R/W	
Reset	1	1	1	1	1	1	1	1

SFR Address = 0xEE; SFR Page = F

Bit	Name	Function
7:6	EAS[1:0]	**EMIF Address Setup Time Bits.** 00: Address setup time = 0 SYSCLK cycles. 01: Address setup time = 1 SYSCLK cycle. 10: Address setup time = 2 SYSCLK cycles. 11: Address setup time = 3 SYSCLK cycles.
5:2	EWR[3:0]	**EMIF $\overline{WR}$ and $\overline{RD}$ Pulse-Width Control Bits.** 0000: $\overline{WR}$ and $\overline{RD}$ pulse width = 1 SYSCLK cycle. 0001: $\overline{WR}$ and $\overline{RD}$ pulse width = 2 SYSCLK cycles. 0010: $\overline{WR}$ and $\overline{RD}$ pulse width = 3 SYSCLK cycles. 0011: $\overline{WR}$ and $\overline{RD}$ pulse width = 4 SYSCLK cycles. 0100: $\overline{WR}$ and $\overline{RD}$ pulse width = 5 SYSCLK cycles. 0101: $\overline{WR}$ and $\overline{RD}$ pulse width = 6 SYSCLK cycles. 0110: $\overline{WR}$ and $\overline{RD}$ pulse width = 7 SYSCLK cycles. 0111: $\overline{WR}$ and $\overline{RD}$ pulse width = 8 SYSCLK cycles. 1000: $\overline{WR}$ and $\overline{RD}$ pulse width = 9 SYSCLK cycles. 1001: $\overline{WR}$ and $\overline{RD}$ pulse width = 10 SYSCLK cycles. 1010: $\overline{WR}$ and $\overline{RD}$ pulse width = 11 SYSCLK cycles. 1011: $\overline{WR}$ and $\overline{RD}$ pulse width = 12 SYSCLK cycles. 1100: $\overline{WR}$ and $\overline{RD}$ pulse width = 13 SYSCLK cycles. 1101: $\overline{WR}$ and $\overline{RD}$ pulse width = 14 SYSCLK cycles. 1110: $\overline{WR}$ and $\overline{RD}$ pulse width = 15 SYSCLK cycles. 1111: $\overline{WR}$ and $\overline{RD}$ pulse width = 16 SYSCLK cycles.
1:0	EAH[1:0]	**EMIF Address Hold Time Bits.** 00: Address hold time = 0 SYSCLK cycles. 01: Address hold time = 1 SYSCLK cycle. 10: Address hold time = 2 SYSCLK cycles. 11: Address hold time = 3 SYSCLK cycles.

16.6.1. Non-Multiplexed Mode

16.6.1.1. 16-bit MOVX: EMI0CF[4:2] = 101, 110, or 111

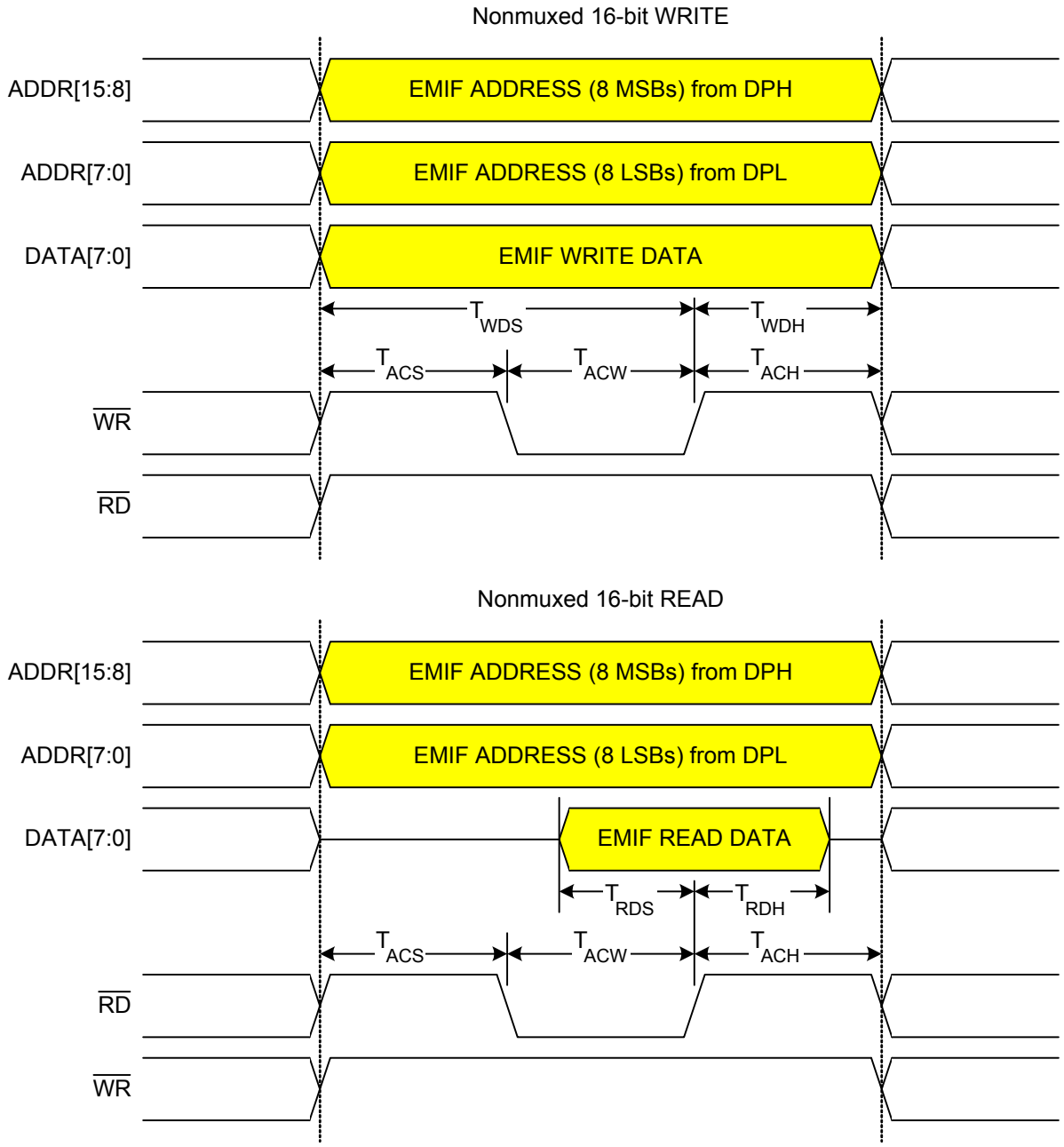


Figure 16.4. Non-multiplexed 16-bit MOVX Timing

16.6.1.2. 8-bit MOVX without Bank Select: EMI0CF[4:2] = 101 or 111

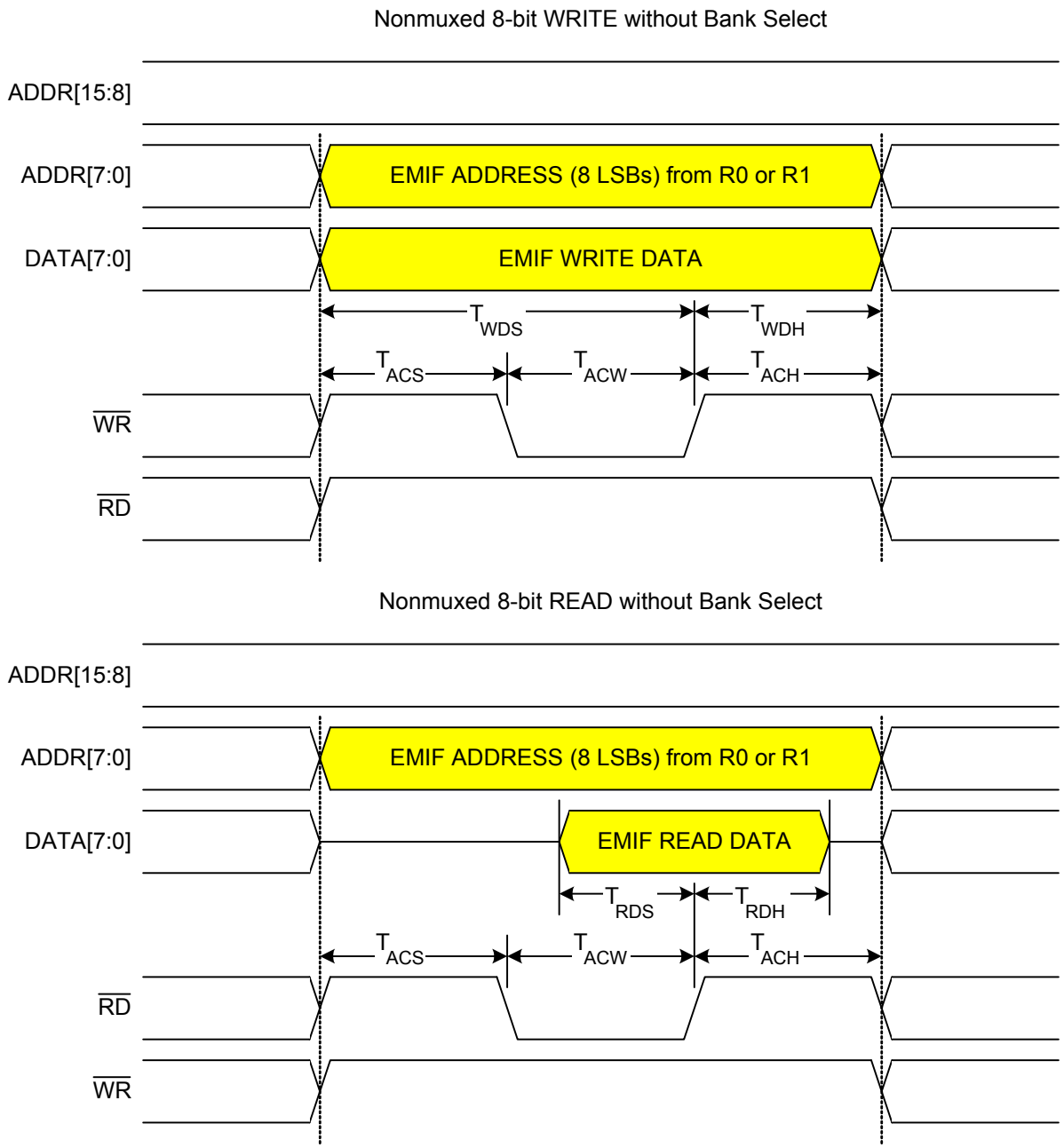


Figure 16.5. Non-multiplexed 8-bit MOVX without Bank Select Timing

16.6.1.3. 8-bit MOVX with Bank Select: EMI0CF[4:2] = 110

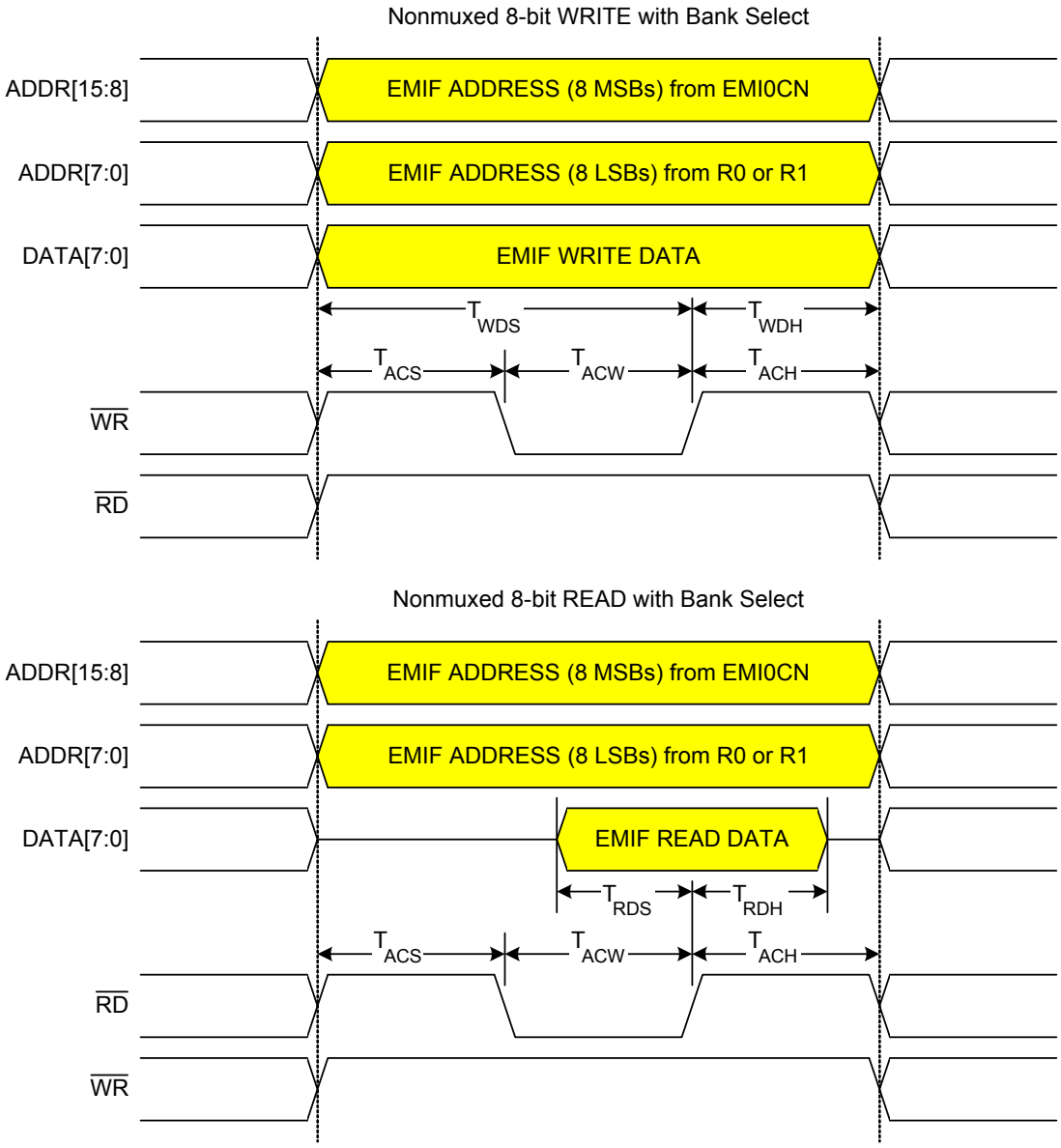


Figure 16.6. Non-multiplexed 8-bit MOVX with Bank Select Timing

16.6.2. Multiplexed Mode

16.6.2.1. 16-bit MOVX: EMI0CF[4:2] = 001, 010, or 011

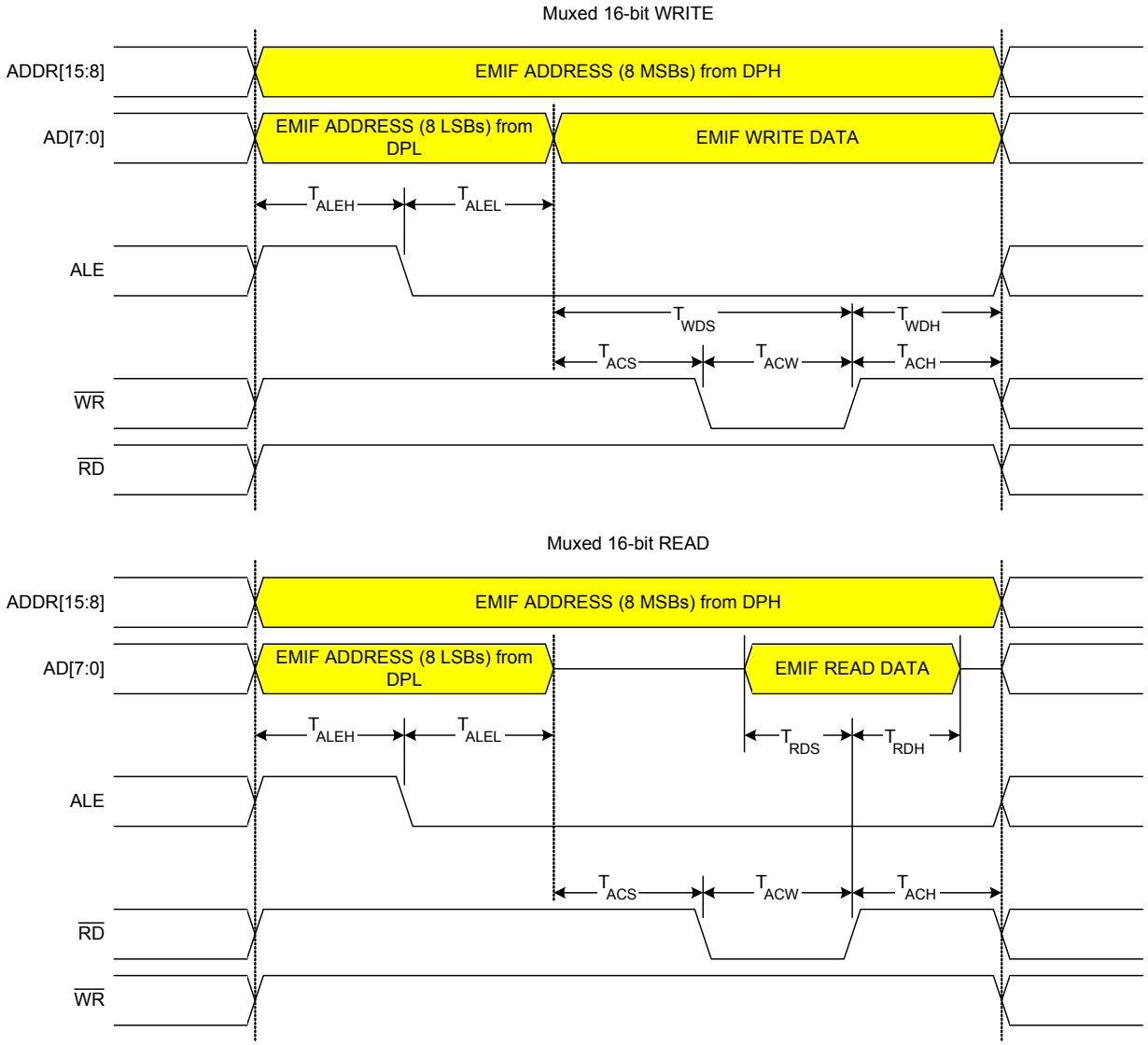


Figure 16.7. Multiplexed 16-bit MOVX Timing

16.6.2.2. 8-bit MOVX without Bank Select: EMI0CF[4:2] = 001 or 011

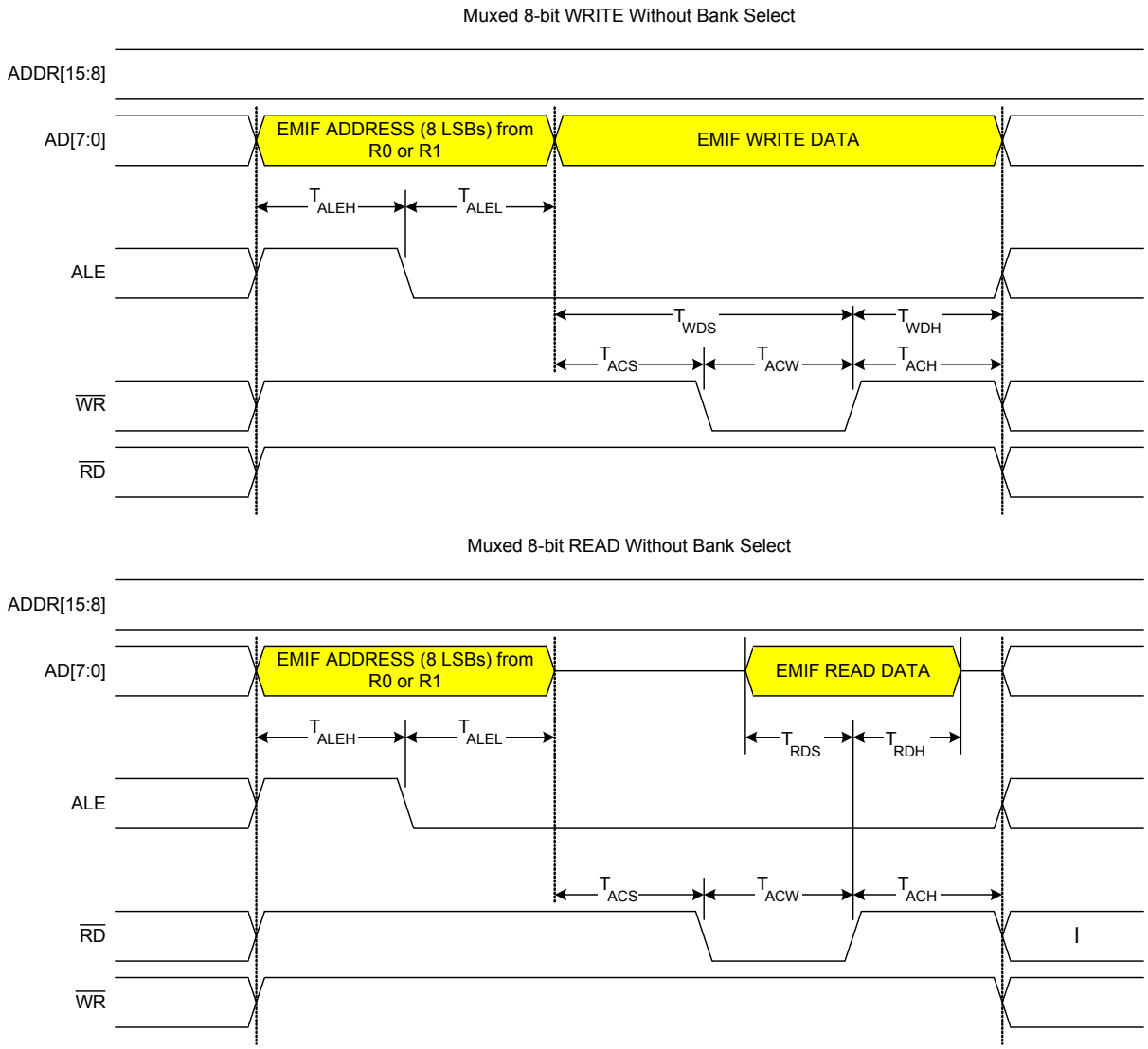


Figure 16.8. Multiplexed 8-bit MOVX without Bank Select Timing

16.6.2.3. 8-bit MOVX with Bank Select: EMI0CF[4:2] = 010

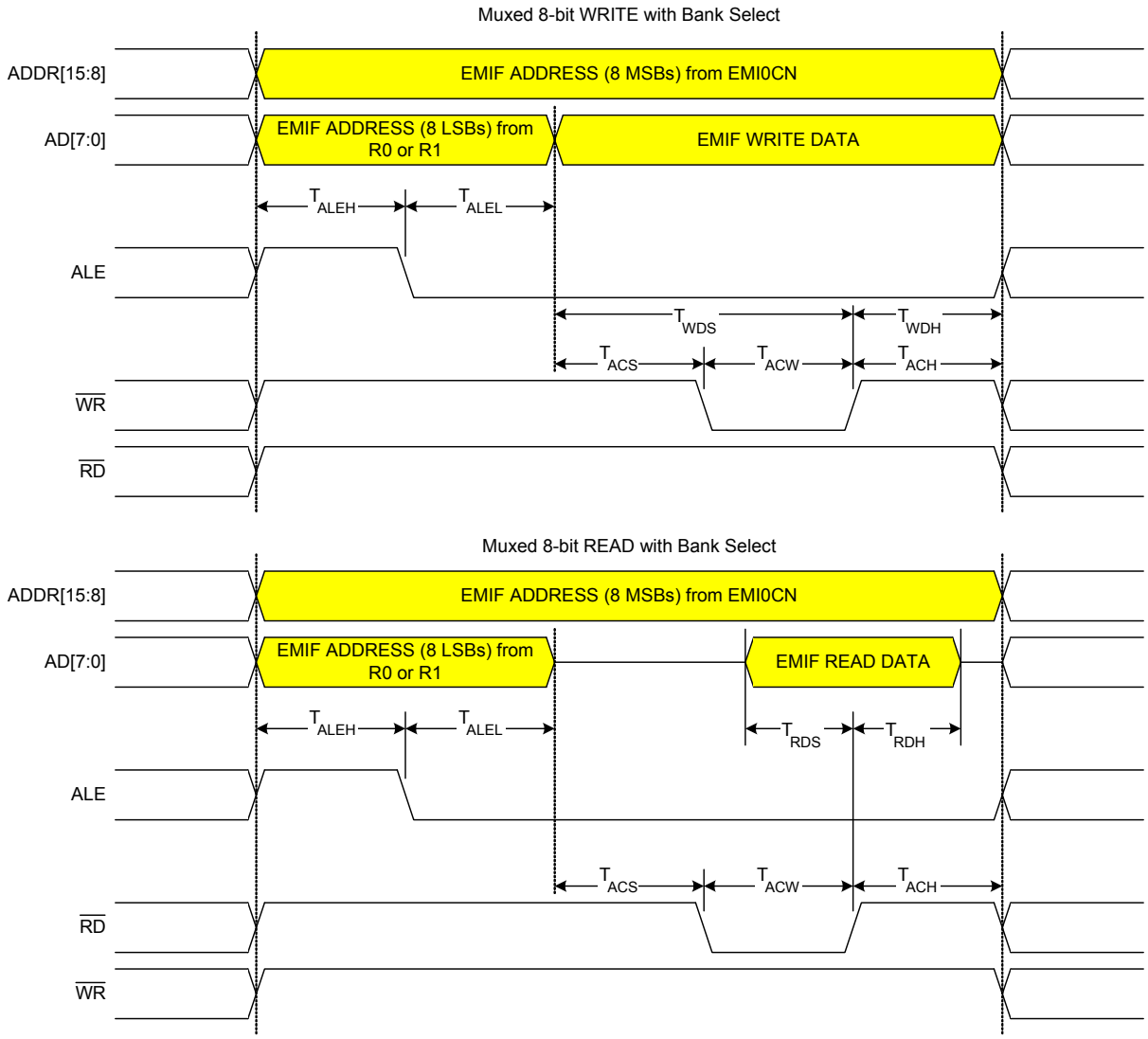


Figure 16.9. Multiplexed 8-bit MOVX with Bank Select Timing

Table 16.1. AC Parameters for External Memory Interface

Parameter	Description	Min*	Max*	Units
T_{ACS}	Address/Control Setup Time	0	3 x T_{SYSCLK}	ns
T_{ACW}	Address/Control Pulse Width	T_{SYSCLK}	16 x T_{SYSCLK}	ns
T_{ACH}	Address/Control Hold Time	0	3 x T_{SYSCLK}	ns
T_{ALEH}	Address Latch Enable High Time	T_{SYSCLK}	4 x T_{SYSCLK}	ns
T_{ALEL}	Address Latch Enable Low Time	T_{SYSCLK}	4 x T_{SYSCLK}	ns
T_{WDS}	Write Data Setup Time	T_{SYSCLK}	19 x T_{SYSCLK}	ns
T_{WDH}	Write Data Hold Time	0	3 x T_{SYSCLK}	ns
T_{RDS}	Read Data Setup Time	20	—	ns
T_{RDH}	Read Data Hold Time	0	—	ns

Note: T_{SYSCLK} is equal to one period of the device system clock (SYSCLK).

Table 16.2. EMIF Pinout (C8051F70x/71x)

Multiplexed Mode	
Signal Name	Port Pin
$\overline{RD}$	P6.1
$\overline{WR}$	P6.0
ALE	P6.2
D0/A0	P5.0
D1/A1	P5.1
D2/A2	P5.2
D3/A3	P5.3
D4/A4	P5.4
D5/A5	P5.5
D6/A6	P5.6
D7/A7	P5.7
A8	P4.0
A9	P4.1
A10	P4.2
A11	P4.3
A12	P4.4
A13	P4.5
A14	P4.6
A15	P4.7
—	—
—	—
—	—
—	—
—	—
—	—
—	—

Non Multiplexed Mode	
Signal Name	Port Pin
$\overline{RD}$	P6.1
$\overline{WR}$	P6.0
D0	P5.0
D1	P5.1
D2	P5.2
D3	P5.3
D4	P5.4
D5	P5.5
D6	P5.6
D7	P5.7
A0	P4.0
A1	P4.1
A2	P4.2
A3	P4.3
A4	P4.4
A5	P4.5
A6	P4.6
A7	P4.7
A8	P3.0
A9	P3.1
A10	P3.2
A11	P3.3
A12	P3.4
A13	P3.5
A14	P3.6
A15	P3.7

17. In-System Device Identification

The C8051F70x/71x has SFRs that identify the device family and derivative. 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 changing functionality to suit the capabilities of that MCU.

In order for firmware to identify the MCU, it must read three SFRs. HWID describes the MCU's family, DERIVID describes the specific derivative within that device family, and REVID describes the hardware revision of the MCU.

SFR Definition 17.1. HWID: Hardware Identification Byte

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

SFR Address = 0xC4; SFR Page = F

Bit	Name	Description
7:0	**HWID[7:0]**	**Hardware Identification byte** Describes the MCU family. 0xE1: Devices covered in this document (C8051F70x/71x)

SFR Definition 17.2. DERIVID: Derivative Identification Byte

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

SFR Address = 0xEC; SFR Page = F

Bit	Name	Description
7:0	**DERIVID[7:0]**	**Derivative Identification byte** Shows the C8051F70x/71x derivative being used. 0xD0: C8051F700; 0xD1: C8051F701; 0xD2: C8051F702; 0xD3: C8051F703 0xD4: C8051F704; 0xD5: C8051F705; 0xD6: C8051F706; 0xD7: C8051F707 0xD8: C8051F708; 0xD9: C8051F709; 0xDA: C8051F710; 0xDB: C8051F711 0xDC: C8051F712; 0xDD: C8051F713; 0xDE: C8051F714; 0xDF: C8051F715

SFR Definition 17.3. REVID: Hardware Revision Identification Byte

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

SFR Address = 0xAD; SFR Page = F

Bit	Name	Description
7:0	**REVID[7:0]**	**Hardware Revision Identification byte** Shows the C8051F70x/71x hardware revision being used. For example, 0x00 = Revision A.

18. 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 C8051F70x/71x's resources and peripherals. The CIP-51 controller core duplicates the SFRs found in a typical 8051 implementation as well as implementing additional SFRs used to configure and access the sub-systems unique to the C8051F70x/71x. This allows the addition of new functionality while retaining compatibility with the MCS-51™ instruction set. Table 18.1 lists the SFRs implemented in the C8051F70x/71x device family.

The SFR registers are accessed anytime the direct addressing mode is used to access memory locations from 0x80 to 0xFF. SFRs with addresses ending in 0x0 or 0x8 (e.g. P0, TCON, SCON0, IE, etc.) are bit-addressable as well as byte-addressable. All other SFRs are byte-addressable only. Unoccupied addresses in the SFR space are reserved for future use. Accessing these areas will have an indeterminate effect and should be avoided. Refer to the corresponding pages of the data sheet, as indicated in Table 18.2, for a detailed description of each register.

Table 18.1. Special Function Register (SFR) Memory Map

Addr	SFR Page	0(8)	1(9)	2(A)	3(B)	4(C)	5(D)	6(E)	7(F)
F8	0 F	**SPI0CN**	PCA0L P0DRV	PCA0H P1DRV	PCA0CPL0 P2DRV	PCA0CPH0 P3DRV	P4DRV	P5DRV	**VDM0CN**
F0	0 F	**B**	P0MDIN	P1MDIN	P0MAT P2MDIN	P0MASK P3MDIN	P4MDIN	P5MDIN	
E8	0 F	**ADC0CN**	PCA0CPL1	PCA0CPH1	PCA0CPL2	PCA0CPH2 DERIVID	PCA0MD	EMI0TC	**RSTSRC**
E0	0 F	**ACC**	P1MAT XBR0	P1MASK XBR1	**WDTCN**	IT01CF		**EIE1**	**EIE2**
D8	0 F	**PCA0CN**	CRC0DATA	PCA0CPM0	PCA0CPM1	PCA0CPM2			
D0	0 F	**PSW**	**EEDATA**	REF0CN		P0SKIP	P1SKIP	P2SKIP	
C8	0 F	**TMR2CN**		TMR2RLL	TMR2RLH	TMR2L	TMR2H	EIP1	EIP2
C0	0 F	**SMB0CN**	SMB0CF P6DRV	SMB0DAT	ADC0GTL	ADC0GTH HWID	ADC0LTL EECNTL	ADC0LTH EEKEY	EMI0CF
B8	0 F	**IP**	REG0CN	SMB0ADR	ADC0MX SMB0ADM	ADC0CF	ADC0L CLKSEL	ADC0H	OSCICL
B0	0 F	**P3**		**P6**	**P5**		OSCXCN	**EEADDR**	**FLKEY**
A8	0 F	**IE**	CS0DL OSCICN	CS0DH EMI0CN		**P4**	REVID		P3MDOUT
A0	0 F	**P2**	SPI0CFG PCA0PWM	SPI0CKR	SPI0DAT	P0MDOUT	P1MDOUT	P2MDOUT	**SFRPAGE**
98	0 F	**SCON0**	**SBUF0**	CS0CN P4MDOUT	CPT0CN P5MDOUT	CS0MX P6MDOUT	CPT0MD	CS0CF	CPT0MX
90	0 F	**P1**	TMR3CN CRC0CN	TMR3RLL CS0SS	TMR3RLH CS0SE	TMR3L CRC0IN	TMR3H CRC0FLIP	CS0THL CRC0AUTO	CS0THH CRC0CNT
88	0 F	**TCON**	**TMOD**	**TL0**	**TL1**	**TH0**	**TH1**	**CKCON**	**PSCTL**
80	0 F	**P0**	**SP**	**DPL**	**DPH**				**PCON**
		0(8)	1(9)	2(A)	3(B)	4(C)	5(D)	6(E)	7(F)

Notes:

1. SFR addresses ending in 0x0 or 0x8 (leftmost column) are bit-addressable.
2. SFRs indicated with bold lettering and shaded cells are available on both SFR Page 0 and F.

SFR Definition 18.1. SFRPAGE: SFR Page

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

SFR Address = 0xA7; SFR Page = All Pages

Bit	Name	Description
7:0	SFRPAGE[7:0]	**SFR Page Bits.** Represents the SFR Page the C8051 core uses when reading or modifying SFRs. Write: Sets the SFR Page. Read: Byte is the SFR page the C8051 core is using.

Table 18.2. Special Function Registers

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

Register	Address	Page	Description	Page
ACC	0xE0	All Pages	Accumulator	90
ADC0CF	0xBC	F	ADC0 Configuration	50
ADC0CN	0xE8	All Pages	ADC0 Control	52
ADC0GTH	0xC4	0	ADC0 Greater-Than Compare High	53
ADC0GTL	0xC3	0	ADC0 Greater-Than Compare Low	53
ADC0H	0xBE	0	ADC0 High	51
ADC0L	0xBD	0	ADC0 Low	51
ADC0LTH	0xC6	0	ADC0 Less-Than Compare Word High	54
ADC0LTL	0xC5	0	ADC0 Less-Than Compare Word Low	54
ADC0MX	0xBB	0	AMUX0 Multiplexer Channel Select	57
B	0xF0	All Pages	B Register	91
CKCON	0x8E	All Pages	Clock Control	245
CLKSEL	0xBD	F	Clock Select	245
CPT0CN	0x9B	0	Comparator0 Control	68
CPT0MD	0x9D	0	Comparator0 Mode Selection	69
CPT0MX	0x9F	0	Comparator0 MUX Selection	71
CRC0AUTO	0x96	F	CRC0 Automatic Control Register	199
CRC0CN	0x91	F	CRC0 Control	197
CRC0CNT	0x97	F	CRC0 Automatic Flash Sector Count	199

Table 18.2. Special Function Registers (Continued)

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

Register	Address	Page	Description	Page
CRC0DATA	0xD9	F	CRC0 Data Output	198
CRC0FLIP	0x95	F	CRC0 Bit Flip	200
CRC0IN	0x94	F	CRC Data Input	198
CS0THH	0x97	0	CS0 Digital Compare Threshold High	80
CS0THL	0x96	0	CS0 Digital Compare Threshold High	80
CS0CN	0x9A	0	CS0 Control	76
CS0DH	0xAA	0	CS0 Data High	78
CS0DL	0xA9	0	CS0 Data Low	78
CS0CF	0x9E	0	CS0 Configuration	77
CS0MX	0x9C	0	CS0 Mux	82
CS0SE	0x93	F	Auto Scan End Channel	79
CS0SS	0x92	F	Auto Scan Start Channel	79
DERIVID	0xEC	F	Derivative Identification	113
DPH	0x83	All Pages	Data Pointer High	89
DPL	0x82	All Pages	Data Pointer Low	89
EEADDR	0xB6	All Pages	EEPROM Byte Address	141
EECNTL	0xC5	F	EEPROM Control	143
EEDATA	0xD1	All Pages	EEPROM Byte Data	142
EEKEY	0xC6	F	EEPROM Protect Key	144
EIE1	0xE6	All Pages	Extended Interrupt Enable 1	127
EIE2	0xE7	All Pages	Extended Interrupt Enable 2	128
EIP1	0xCE	F	Extended Interrupt Priority 1	129
EIP2	0xCF	F	Extended Interrupt Priority 2	130
EMI0CF	0xC7	F	EMIF Configuration	99
EMI0CN	0xAA	F	EMIF Control	98
EMI0TC	0xEE	F	EMIF Timing Control	104
FLKEY	0xB7	All Pages	Flash Lock And Key	139
HWID	0xC4	F	Hardware Identification	113
IE	0xA8	All Pages	Interrupt Enable	125
IP	0xB8	All Pages	Interrupt Priority	126
IT01CF	0xE4	F	INT0/INT1 Configuration	132
OSCICL	0xBF	F	Internal Oscillator Calibration	158
OSCICN	0xA9	F	Internal Oscillator Control	159
OSCXCN	0xB5	F	External Oscillator Control	161

Table 18.2. Special Function Registers (Continued)

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

Register	Address	Page	Description	Page
P0	0x80	All Pages	Port 0 Latch	178
P0DRV	0xF9	F	Port 0 Drive Strength	180
P0MASK	0xF4	0	Port 0 Mask	175
P0MAT	0xF3	0	Port 0 Match	176
P0MDIN	0xF1	F	Port 0 Input Mode Configuration	178
P0MDOUT	0xA4	F	Port 0 Output Mode Configuration	179
P0SKIP	0xD4	F	Port 0 Skip	179
P1	0x90	All Pages	Port 1 Latch	180
P1DRV	0xFA	F	Port 1 Drive Strength	182
P1MASK	0xE2	0	P0 Mask	176
P1MAT	0xE1	0	P1 Match	177
P1MDIN	0xF2	F	Port 1 Input Mode Configuration	181
P1MDOUT	0xA5	F	Port 1 Output Mode Configuration	181
P1SKIP	0xD5	F	Port 1 Skip	182
P2	0xA0	All Pages	Port 2 Latch	183
P2DRV	0xFB	F	Port 2 Drive Strength	185
P2MDIN	0xF3	F	Port 2 Input Mode Configuration	183
P2MDOUT	0xA6	F	Port 2 Output Mode Configuration	184
P2SKIP	0xD6	F	Port 2 Skip	184
P3	0xB0	All Pages	Port 3 Latch	185
P3DRV	0xFC	F	Port 3 Drive Strength	187
P3MDIN	0xF4	F	Port 3 Input Mode Configuration	186
P3MDOUT	0xAF	F	Port 3 Output Mode Configuration	186
P4	0xAC	All Pages	Port 4 Latch	187
P4DRV	0xFD	F	Port 4 Drive Strength	189
P4MDIN	0xF5	F	Port 4 Input Mode Configuration	188
P4MDOUT	0x9A	F	Port 4 Output Mode Configuration	188
P5	0xB3	All Pages	Port 5 Latch	189
P5DRV	0xFE	F	Port 5 Drive Strength	191
P5MDIN	0xF6	F	Port 5 Input Mode Configuration	190
P5MDOUT	0x9B	F	Port 5 Output Mode Configuration	190
P6	0xB2	All Pages	Port 6 Latch	191
P6DRV	0xC1	F	Port 6 Drive Strength	192
P6MDOUT	0x9C	F	Port 6 Output Mode Configuration	192

Table 18.2. Special Function Registers (Continued)

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

Register	Address	Page	Description	Page
PCA0CN	0xD8	All Pages	PCA Control	278
PCA0CPH0	0xFC	0	PCA Capture 0 High	283
PCA0CPH1	0xEA	0	PCA Capture 1 High	283
PCA0CPH2	0xEC	0	PCA Capture 2 High	283
PCA0CPL0	0xFB	0	PCA Capture 0 Low	283
PCA0CPL1	0xE9	0	PCA Capture 1 Low	283
PCA0CPL2	0xEB	0	PCA Capture 2 Low	283
PCA0CPM0	0xDA	F	PCA Module 0 Mode Register	281
PCA0CPM1	0xDB	F	PCA Module 1 Mode Register	281
PCA0CPM2	0xDC	F	PCA Module 2 Mode Register	281
PCA0H	0xFA	0	PCA Counter High	282
PCA0L	0xF9	0	PCA Counter Low	282
PCA0MD	0xED	F	PCA Mode	279
PCA0PWM	0xA1	F	PCA PWM Configuration	280
PCON	0x87	All Pages	Power Control	147
PSCTL	0x8F	All Pages	Program Store R/W Control	138
PSW	0xD0	All Pages	Program Status Word	92
REF0CN	0xD2	F	Voltage Reference Control	62
REG0CN	0xB9	F	Voltage Regulator Control	64
REVID	0xAD	F	Revision ID	114
RSTSRC	0xEF	All Pages	Reset Source Configuration/Status	153
SBUF0	0x99	All Pages	UART0 Data Buffer	242
SCON0	0x98	All Pages	UART0 Control	241
SFRPAGE	0xA7	All Pages	SFR Page	117
SMB0ADM	0xBB	F	SMBus Slave Address mask	212
SMB0ADR	0xBA	F	SMBus Slave Address	211
SMB0CF	0xC1	0	SMBus Configuration	207
SMB0CN	0xC0	All Pages	SMBus Control	209
SMB0DAT	0xC2	0	SMBus Data	213
SP	0x81	All Pages	Stack Pointer	90
SPI0CFG	0xA1	0	SPI0 Configuration	230
SPI0CKR	0xA2	F	SPI0 Clock Rate Control	232
SPI0CN	0xF8	All Pages	SPI0 Control	231
SPI0DAT	0xA3	0	SPI0 Data	232

Table 18.2. Special Function Registers (Continued)

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

Register	Address	Page	Description	Page
TCON	0x88	All Pages	Timer/Counter Control	250
TH0	0x8C	All Pages	Timer/Counter 0 High	253
TH1	0x8D	All Pages	Timer/Counter 1 High	253
TL0	0x8A	All Pages	Timer/Counter 0 Low	252
TL1	0x8B	All Pages	Timer/Counter 1 Low	252
TMOD	0x89	All Pages	Timer/Counter Mode	251
TMR2CN	0xC8	All Pages	Timer/Counter 2 Control	257
TMR2H	0xCD	0	Timer/Counter 2 High	259
TMR2L	0xCC	0	Timer/Counter 2 Low	259
TMR2RLH	0xCB	0	Timer/Counter 2 Reload High	258
TMR2RLL	0xCA	0	Timer/Counter 2 Reload Low	258
TMR3CN	0x91	0	Timer/Counter 3 Control	263
TMR3H	0x95	0	Timer/Counter 3 High	265
TMR3L	0x94	0	Timer/Counter 3 Low	265
TMR3RLH	0x93	0	Timer/Counter 3 Reload High	264
TMR3RLL	0x92	0	Timer/Counter 3 Reload Low	264
VDM0CN	0xFF	All Pages	VDD Monitor Control	151
WDTCN	0xE3	All Pages	Watchdog Timer Control	155
XBR0	0xE1	F	Port I/O Crossbar Control 0	173
XBR1	0xE2	F	Port I/O Crossbar Control 1	174
All other SFR Locations			Reserved	

19. Interrupts

The C8051F70x/71x includes an extended interrupt system supporting a total of 16 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–EIE1). However, interrupts must first be globally enabled by setting the EA bit (IE.7) to logic 1 before the individual interrupt enables are recognized. Setting the EA bit to logic 0 disables all interrupt sources regardless of the individual interrupt-enable settings.

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.

19.1. MCU Interrupt Sources and Vectors

The C8051F70x/71x MCUs support 12 interrupt sources. Software can simulate an interrupt by setting an 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 19.1. 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).

19.1.1. 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 19.1.

19.1.2. 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.

Table 19.1. Interrupt Summary

Interrupt Source	Interrupt Vector	Priority Order	Pending Flag	Bit addressable?	Cleared by HW?	Enable Flag	Priority Control
Reset	0x0000	Top	None	N/A	N/A	Always Enabled	Always Highest
External Interrupt 0 ($\overline{\text{INT0}}$)	0x0003	0	IE0 (TCON.1)	Y	Y	EX0 (IE.0)	PX0 (IP.0)
Timer 0 Overflow	0x000B	1	TF0 (TCON.5)	Y	Y	ET0 (IE.1)	PT0 (IP.1)
External Interrupt 1 ($\overline{\text{INT1}}$)	0x0013	2	IE1 (TCON.3)	Y	Y	EX1 (IE.2)	PX1 (IP.2)
Timer 1 Overflow	0x001B	3	TF1 (TCON.7)	Y	Y	ET1 (IE.3)	PT1 (IP.3)
UART0	0x0023	4	RI0 (SCON0.0) TI0 (SCON0.1)	Y	N	ES0 (IE.4)	PS0 (IP.4)
Timer 2 Overflow	0x002B	5	TF2H (TMR2CN.7) TF2L (TMR2CN.6)	Y	N	ET2 (IE.5)	PT2 (IP.5)
SPI0	0x0033	6	SPIF (SPI0CN.7) WCOL (SPI0CN.6) MODF (SPI0CN.5) RXOVRN (SPI0CN.4)	Y		ESPI0 (IE.6)	PSPI0 (IP.6)
SMB0	0x003B	7	SI (SMB0CN.0)	Y	N	ESMB0 (EIE1.0)	PSMB0 (EIP1.0)
Port Match	0x0043	8	None	N/A	N/A	EMAT (EIE1.1)	PMAT (EIP1.1)
ADC0 Window Compare	0x0043	9	AD0WINT (ADC0CN.3)	Y	N	EWADC0 (EIE1.2)	PWADC0 (EIP1.2)
ADC0 Conversion Complete	0x004B	10	AD0INT (ADC0CN.5)	Y	N	EADC0 (EIE1.3)	PADC0 (EIP1.3)
Programmable Counter Array	0x0053	11	CF (PCA0CN.7) CCFn (PCA0CN.n)	Y	N	EPCA0 (EIE1.4)	PPCA0 (EIP1.4)
Comparator0	0x005B	12	CP0FIF (CPT0CN.4) CP0RIF (CPT0CN.5)	N	N	ECP0 (EIE1.5)	PCP0 (EIP1.5)
RESERVED							
Timer 3 Overflow	0x0073	14	TF3H (TMR3CN.7) TF3L (TMR3CN.6)	N	N	ET3 (EIE1.7)	PT3 (EIP1.7)
CS0 Conversion Complete	0x007B	15	CS0INT (CS0CN.5)	N	N	ECSCPT (EIE2.0)	PSCCPT (EIP2.0)
CS0 Greater Than	0x0083	16	CS0CMPF (CS0CN.0)	N	N	ECSGRT (EIE2.1)	PSCGRT (EIP2.1)

19.2. Interrupt Register Descriptions

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

SFR Definition 19.1. IE: Interrupt Enable

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

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

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

SFR Definition 19.2. IP: Interrupt Priority

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

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

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

SFR Definition 19.3. EIE1: Extended Interrupt Enable 1

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

SFR Address = 0xE6; SFR Page = All Pages

Bit	Name	Function
7	ET3	**Enable Timer 3 Interrupt.** This bit sets the masking of the Timer 3 interrupt. 0: Disable Timer 3 interrupt. 1: Enable interrupt requests generated by the TF3L or TF3H flags.
6	Reserved	Reserved. Must write 0.
5	ECP0	**Enable Comparator0 (CP0) Interrupt.** This bit sets the masking of the CP0 rising edge or falling edge interrupt. 0: Disable CP0 interrupts. 1: Enable interrupt requests generated by the CP0RIF and CP0FIF flags.
4	EADC0	**Enable ADC0 Conversion Complete Interrupt.** This bit sets the masking of the ADC0 Conversion Complete interrupt. 0: Disable ADC0 Conversion Complete interrupt. 1: Enable interrupt requests generated by the AD0INT flag.
3	EPCA0	**Enable Programmable Counter Array (PCA0) Interrupt.** This bit sets the masking of the PCA0 interrupts. 0: Disable all PCA0 interrupts. 1: Enable interrupt requests generated by PCA0.
2	EWADC0	**Enable Window Comparison ADC0 interrupt.** This bit sets the masking of ADC0 Window Comparison interrupt. 0: Disable ADC0 Window Comparison interrupt. 1: Enable interrupt requests generated by ADC0 Window Compare flag (AD0WINT).
1	EMAT	**Enable Port Match Interrupts.** This bit sets the masking of the Port Match event interrupt. 0: Disable all Port Match interrupts. 1: Enable interrupt requests generated by a Port Match.
0	ESMB0	**Enable SMBus (SMB0) Interrupt.** This bit sets the masking of the SMB0 interrupt. 0: Disable all SMB0 interrupts. 1: Enable interrupt requests generated by SMB0.

SFR Definition 19.4. EIE2: Extended Interrupt Enable 2

Bit	7	6	5	4	3	2	1	0
Name							ECSGRT	ECSCPT
Type	R	R	R	R	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xE7; SFR Page = All Pages

Bit	Name	Function
7:2	Unused	Read = 000000b; Write = don't care.
1	ECSGRT	**Enable Capacitive Sense Greater Than Comparator Interrupt.** 0: Disable Capacitive Sense Greater Than Comparator interrupt. 1: Enable interrupt requests generated by CS0CMPF.
0	ECSCPT	**Enable Capacitive Sense Conversion Complete Interrupt.** 0: Disable Capacitive Sense Conversion Complete interrupt. 1: Enable interrupt requests generated by CS0INT.

SFR Definition 19.5. EIP1: Extended Interrupt Priority 1

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

SFR Address = 0xCE; SFR Page = F

Bit	Name	Function
7	PT3	**Timer 3 Interrupt Priority Control.** This bit sets the priority of the Timer 3 interrupt. 0: Timer 3 interrupt set to low priority level. 1: Timer 3 interrupt set to high priority level.
6	Reserved	Reserved. Must write to 0.
5	PCP0	**Comparator0 (CP0) Interrupt Priority Control.** This bit sets the priority of the CP0 rising edge or falling edge interrupt. 0: CP0 interrupt set to low priority level. 1: CP0 interrupt set to high priority level.
4	PPCA0	**Programmable Counter Array (PCA0) Interrupt Priority Control.** This bit sets the priority of the PCA0 interrupt. 0: PCA0 interrupt set to low priority level. 1: PCA0 interrupt set to high priority level.
3	PADC0	**ADC0 Conversion Complete Interrupt Priority Control.** This bit sets the priority of the ADC0 Conversion Complete interrupt. 0: ADC0 Conversion Complete interrupt set to low priority level. 1: ADC0 Conversion Complete interrupt set to high priority level.
2	PWADC0	**ADC0 Window Comparator Interrupt Priority Control.** This bit sets the priority of the ADC0 Window interrupt. 0: ADC0 Window interrupt set to low priority level. 1: ADC0 Window interrupt set to high priority level.
1	PMAT	**Port Match Interrupt Priority Control.** This bit sets the priority of the Port Match Event interrupt. 0: Port Match interrupt set to low priority level. 1: Port Match interrupt set to high priority level.
0	PSMB0	**SMBus (SMB0) Interrupt Priority Control.** This bit sets the priority of the SMB0 interrupt. 0: SMB0 interrupt set to low priority level. 1: SMB0 interrupt set to high priority level.

SFR Definition 19.6. EIP2: Extended Interrupt Priority 2

Bit	7	6	5	4	3	2	1	0
Name	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	PSCGRT	PSCCPT
Type	R	R	R	R	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xCF; SFR Page = F

Bit	Name	Function
7:2	Reserved	Reserved. Must write to 0.
1	PSCGRT	**Capacitive Sense Greater Than Comparator Priority Control.** This bit sets the priority of the Capacitive Sense Greater Than Comparator interrupt. 0: CS0 Greater Than Comparator interrupt set to low priority level. 1: CS0 Greater Than Comparator set to high priority level.
0	PSCCPT	**Capacitive Sense Conversion Complete Priority Control.** This bit sets the priority of the Capacitive Sense Conversion Complete interrupt. 0: CS0 Conversion Complete set to low priority level. 1: CS0 Conversion Complete set to high priority level.

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

The $\overline{\text{INT0}}$ and $\overline{\text{INT1}}$ external interrupt sources are configurable as active high or low, edge or level sensitive. The IN0PL ($\overline{\text{INT0}}$ Polarity) and IN1PL ($\overline{\text{INT1}}$ Polarity) bits in the IT01CF register select active high or active low; the IT0 and IT1 bits in TCON (Section “31.1. Timer 0 and Timer 1” on page 246) select level or edge sensitive. The table below lists the possible configurations.

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

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

$\overline{\text{INT0}}$ and $\overline{\text{INT1}}$ are assigned to Port pins as defined in the IT01CF register (see SFR Definition 19.7). Note that $\overline{\text{INT0}}$ and $\overline{\text{INT0}}$ Port pin assignments are independent of any Crossbar assignments. $\overline{\text{INT0}}$ and $\overline{\text{INT1}}$ will monitor their assigned Port pins without disturbing the peripheral that was assigned the Port pin via the Crossbar. To assign a Port pin only to $\overline{\text{INT0}}$ and/or $\overline{\text{INT1}}$, configure the Crossbar to skip the selected pin(s). This is accomplished by setting the associated bit in register XBR0 (see Section “26.3. Priority Crossbar Decoder” on page 170 for complete details on configuring the Crossbar).

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

SFR Definition 19.7. IT01CF: $\overline{\text{INT0}}$/$\overline{\text{INT1}}$ Configuration

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

SFR Address = 0xE4; SFR Page = F

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

20. Flash Memory

On-chip, re-programmable Flash memory is included for program code and non-volatile data storage. The Flash memory can be programmed in-system through the C2 interface or by software using the MOVX write instruction. Once cleared to logic 0, a Flash bit must be erased to set it back to logic 1. Flash bytes would typically be erased (set to 0xFF) before being reprogrammed. The write and erase operations are automatically timed by hardware for proper execution; data polling to determine the end of the write/erase operations is not required. Code execution is stalled during Flash write/erase operations. Refer to Table 7.6 for complete Flash memory electrical characteristics.

20.1. Programming The Flash Memory

The simplest means of programming the Flash memory is through the C2 interface using programming tools provided by Silicon Laboratories or a third party vendor. This is the only means for programming a non-initialized device. For details on the C2 commands to program Flash memory, see Section “33. C2 Interface” on page 284.

The Flash memory can be programmed by software using the MOVX write instruction with the address and data byte to be programmed provided as normal operands. Before programming Flash memory using MOVX, Flash programming operations must be enabled by: (1) setting the PSWE Program Store Write Enable bit (PSCTL.0) to logic 1 (this directs the MOVX writes to target Flash memory); and (2) Writing the Flash key codes in sequence to the Flash Lock register (FLKEY). The PSWE bit remains set until cleared by software. For detailed guidelines on programming Flash from firmware, please see Section “20.4. Flash Write and Erase Guidelines” on page 135.

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

20.1.1. Flash Lock and Key Functions

Flash writes and erases by user software are protected with a lock and key function. The Flash Lock and Key Register (FLKEY) must be written with the correct key codes, in sequence, before Flash operations may be performed. The key codes are: 0xA5, 0xF1. The timing does not matter, but the codes must be written in order. If the key codes are written out of order, or the wrong codes are written, Flash writes and erases will be disabled until the next system reset. Flash writes and erases will also be disabled if a Flash write or erase is attempted before the key codes have been written properly. The Flash lock resets after each write or erase; the key codes must be written again before a following Flash operation can be performed. The FLKEY register is detailed in SFR Definition 20.2.

20.1.2. Flash Erase Procedure

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

1. Save current interrupt state and disable interrupts.
2. Set the PSEE bit (register PSCTL).
3. Set the PSWE bit (register PSCTL).
4. Write the first key code to FLKEY: 0xA5.
5. Write the second key code to FLKEY: 0xF1.
6. Using the MOVX instruction, write a data byte to any location within the 512-byte page to be erased.

7. Clear the PSWE and PSEE bits.
8. Restore previous interrupt state.

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

Note: Flash security settings may prevent erasure of some Flash pages, such as the reserved area and the page containing the lock bytes. For a summary of Flash security settings and restrictions affecting Flash erase operations, please see Section “20.3. Security Options” on page 134.

20.1.3. Flash Write Procedure

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

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

1. Save current interrupt state and disable interrupts.
2. Ensure that the Flash byte has been erased (has a value of 0xFF).
3. Set the PSWE bit (register PSCTL).
4. Clear the PSEE bit (register PSCTL).
5. Write the first key code to FLKEY: 0xA5.
6. Write the second key code to FLKEY: 0xF1.
7. Using the MOVX instruction, write a single data byte to the desired location within the 512-byte sector.
8. Clear the PSWE bit.
9. Restore previous interrupt state.

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

Note: Flash security settings may prevent writes to some areas of Flash, such as the reserved area. For a summary of Flash security settings and restrictions affecting Flash write operations, please see Section “20.3. Security Options” on page 134.

20.2. Non-volatile Data Storage

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

Note: MOVX read instructions always target XRAM.

20.3. Security Options

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

A Security Lock Byte located at the last byte of Flash user space offers protection of the Flash program memory from access (reads, writes, and erases) by unprotected code or the C2 interface. The Flash secu-

rity mechanism allows the user to lock all Flash pages, starting at page 0, by writing a non-0xFF value to the lock byte. **Note that writing a non-0xFF value to the lock byte will lock all pages of FLASH from reads, writes, and erases, including the page containing the lock byte.**

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 20.1 summarizes the Flash security features of the C8051F70x/71x devices.

Table 20.1. Flash Security Summary

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

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

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

20.4. Flash Write and Erase Guidelines

Any system which contains routines which write or erase Flash memory from software involves some risk that the write or erase routines will execute unintentionally if the CPU is operating outside its specified operating range of VDD, system clock frequency, or temperature. This accidental execution of Flash modi-

fying code can result in alteration of Flash memory contents causing a system failure that is only recoverable by re-Flashing the code in the device.

To help prevent the accidental modification of Flash by firmware, the VDD Monitor must be enabled and enabled as a reset source on C8051F70x/71x devices for the Flash to be successfully modified. **If either the VDD Monitor or the VDD Monitor reset source is not enabled, a Flash Error Device Reset will be generated when the firmware attempts to modify the Flash.**

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

20.4.1. VDD Maintenance and the VDD Monitor

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

Note: On C8051F70x/71x devices, both the VDD Monitor and the VDD Monitor reset source must be enabled to write or erase Flash without generating a Flash Error Device Reset.

On C8051F70x/71x devices, both the VDD Monitor and the VDD Monitor reset source are enabled by hardware after a power-on reset.

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

20.4.2. PSWE Maintenance

7. Reduce the number of places in code where the PSWE bit (b0 in PSCTL) is set to a 1. There should be exactly one routine in code that sets PSWE to a 1 to write Flash bytes and one routine in code that sets both PSWE and PSEE both to a 1 to erase Flash pages.
8. Minimize the number of variable accesses while PSWE is set to a 1. Handle pointer address updates and loop maintenance outside the "PSWE = 1;... PSWE = 0;" area. Code examples showing this can be found in AN201, *"Writing to Flash from Firmware"*, available from the Silicon Laboratories web site.

9. Disable interrupts prior to setting PSWE to a 1 and leave them disabled until after PSWE has been reset to 0. Any interrupts posted during the Flash write or erase operation will be serviced in priority order after the Flash operation has been completed and interrupts have been re-enabled by software.
10. Make certain that the Flash write and erase pointer variables are not located in XRAM. See your compiler documentation for instructions regarding how to explicitly locate variables in different memory areas.
11. Add address bounds checking to the routines that write or erase Flash memory to ensure that a routine called with an illegal address does not result in modification of the Flash.

20.4.3. System Clock

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

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

SFR Definition 20.1. PSCTL: Program Store R/W Control

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

SFR Address =0x8F; SFR Page = All Pages

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

SFR Definition 20.2. FLKEY: Flash Lock and Key

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

SFR Address = 0xB7; SFR Page = All Pages

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

21. EEPROM

C8051F700/1/4/5/8/9 and C8051F712/3 devices have hardware which emulates 32 bytes of non-volatile, byte-programmable EEPROM data space. The module mirrors each non-volatile byte through 32 bytes of volatile data space. This data space can be accessed indirectly through EEADDR and EEDATA. Users can copy the complete 32-byte image between EEPROM space and volatile space using controls in the EECNTL SFR.

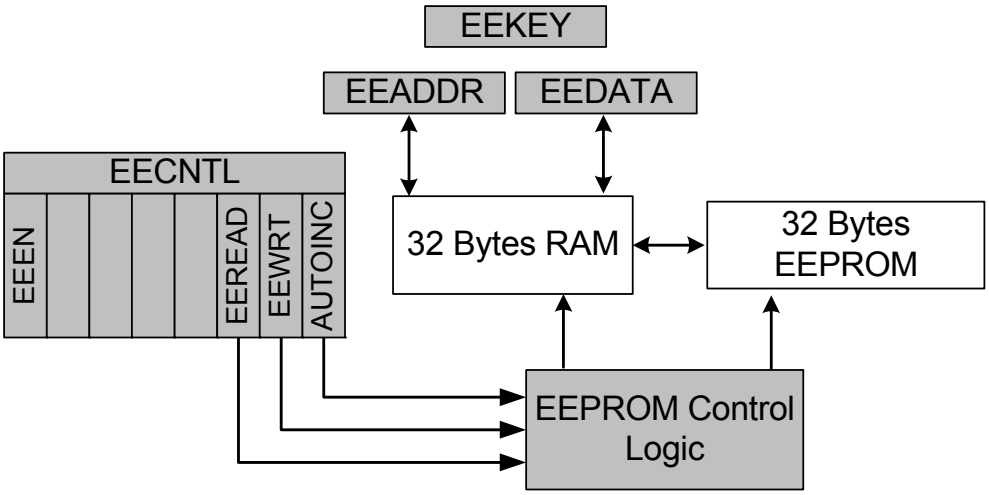


Figure 21.1. EEPROM Block Diagram

21.1. RAM Reads and Writes

In order to perform EEPROM reads and writes, the EEPROM control logic must be enabled by setting EEEN (EECNTL.7).

32 bytes of RAM can be accessed indirectly through EEADDR and EEDATA. To write to a byte of RAM, write address of byte to EEADDR and then write the value to be written to EEDATA. To read a byte from RAM, write address of byte to be read to EEADDR. The value stored at that address can then be read from EEDATA.

21.2. Auto Increment

When AUTOINC (EECNTL.0) is set, EEADDR will increment by one after each write to EEDATA and each read from EEDATA. When Auto Increment is enabled and EEADDR reaches the top address of dedicated RAM space, the next write to or read from EEDATA will cause EEADDR to wrap along the address boundary, which will set the address to 0.

21.3. Interfacing with the EEPROM

The EEPROM is accessed through the dedicated 32 bytes of RAM. Writes to EEPROM are allowed only after writes have been enabled (see “21.4. EEPROM Security”). The contents of the EEPROM can be uploaded to the RAM by setting EEREAD (EECNTL.2). Contents of RAM can be downloaded to EEPROM by setting EEWRT (EENTL.1).

21.4. EEPROM Security

RAM can only be downloaded to EEPROM after firmware writes a sequence of two bytes to EEKEY. In order to enable EEPROM writes:

1. Write the first EEPROM key code byte to EEKEY: 0x55
2. Write the second EEPROM key code byte to EEKEY: 0xAA

After a EEPROM writes have been enabled and a single write has executed, the control logic locks EEPROM writes until the two-byte unlock sequence has been entered into EEKEY again.

The protection state of the EEPROM can be observed by reading EEPSTATE (EEKEY2:0). This state can be read at any time without affecting the EEPROM's protection state.

if the two-byte unlock sequence is entered incorrectly, or if a write is attempted without first entering the two-byte sequence, EEPROM writes will be locked until the next power-on reset.

SFR Definition 21.1. EEADDR: EEPROM Byte Address

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

SFR Address = 0xB6; SFR Page = All Pages

Bit	Name	Description
7:5	Unused	Read = 000b; Write = Don't Care
4:0	EEADDR[4:0]	**EEPROM Byte Address** Selects one of 32 EEPROM bytes to read/write.

SFR Definition 21.2. EEDATA: EEPROM Byte Data

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

SFR Address = 0xD1; SFR Page = All Pages

Bit	Name	Description	Write	Read
7:0	EEDATA[7:0]	**E2PROM Data** The EEDATA register is used to read bytes from the EEPROM space and write bytes to EEPROM space.	Writes byte to location stored in EEADDR.	Returns contents at location stored in EEADDR.

SFR Definition 21.3. EECNTL: EEPROM Control

Bit	7	6	5	4	3	2	1	0
Name	EEEN					EEREAD	EEWRT	AUTOINC
Type	R/W	R				R/W		
Reset	0	0	0	0	0	0	0	1

SFR Address = 0xC5; SFR Page = F

Bit	Name	Description
7	EEEN	**EEPROM Enable** 0: EEPROM control logic disabled. 1: EEPROM control logic enabled. EEPROM reads and writes can be performed.
6:4	Reserved	**Reserved. Read = variable; Write = Don't Care**
3	Reserved	**Reserved. Read = 0b, Write = 0**
2	EEREAD	**EEPROM 32-Byte Read** 0: Does nothing. 1: 32 bytes of EEPROM Data will be read from Flash to internal RAM.
1	EEWRITE	**EEPROM 32-Byte Write** 0: Does nothing. 1: 32 bytes of EEPROM Data will be written from internal RAM to Flash.
0	AUTOINC	**Auto Increment** 0: Disable auto-increment. 1: Enable auto-increment.

SFR Definition 21.4. EEKEY: EEPROM Protect Key

Bit	7	6	5	4	3	2	1	0
Name	EEKEY						EEPSTATE/EEKEY	
Type	W						R/W	
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xC6; SFR Page = F

Bit	Name	Description	Write	Read
7:0	EEKEY	**EEPROM Key** Protects the EEPROM from inadvertent writes and erases.	The sequence 0x55 0xAA must be written to enable EEPROM writes and erases	
1:0	EEPSTATE	**EEPROM Protection State** These bytes show whether Flash writes/erases have been enabled, disabled, or locked.		00: Write/Erase is not enabled 01: The first key has been written 10: Write/Erase is enabled 11: EEPROM is locked from further writes/erases

22. Power Management Modes

The C8051F70x/71x devices have three software programmable power management modes: Idle, Stop, and Suspend. Idle mode and Stop mode are part of the standard 8051 architecture, while Suspend mode is an enhanced power-saving mode implemented by the high-speed oscillator peripheral.

Idle mode halts the CPU while leaving the peripherals and clocks active. In Stop mode, the CPU is halted, all interrupts and timers (except the Missing Clock Detector) are inactive, and the internal oscillator is stopped (analog peripherals remain in their selected states; the external oscillator is not affected). Suspend mode is similar to Stop mode in that the internal oscillator and CPU are halted, but the device can wake on events such as a Port Mismatch, Comparator low output, or a Timer 3 overflow. 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 and Suspend mode consume the least power because the majority of the device is shut down with no clocks active. SFR Definition 22.1 describes the Power Control Register (PCON) used to control the C8051F70x/71x's Stop and Idle power management modes. Suspend mode is controlled by the SUSPEND bit in the OSCICN register (SFR Definition 25.3).

Although the C8051F70x/71x has Idle, Stop, and Suspend 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 or serial buses, draw little power when they are not in use. Turning off oscillators lowers power consumption considerably, at the expense of reduced functionality.

22.1. 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 indefi-

nitely, waiting for an external stimulus to wake up the system. Refer to Section “24. Watchdog Timer” on page 154 for more information on the use and configuration of the WDT.

22.2. 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. 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 of 100 µs.

22.3. Suspend Mode

Suspend mode allows a system running from the internal oscillator to go to a very low power state similar to Stop mode, but the processor can be awakened by certain events without requiring a reset of the device. Setting the SUSPEND bit (OSCICN.5) causes the hardware to halt the CPU and the high-frequency internal oscillator, and go into Suspend mode as soon as the instruction that sets the bit completes execution. All internal registers and memory maintain their original data. Most digital peripherals are not active in Suspend mode. The exception to this is the Port Match feature and Timer 3, when it is run from an external oscillator source.

Note that the clock divider bits CLKDIV[2:0] in register CLKSEL must be set to "divide by 1" when entering Suspend mode.

Suspend mode can be terminated by five types of events, a port match (described in Section “26.5. Port Match” on page 175), a Timer 3 overflow (described in Section “31.3. Timer 3” on page 260), a comparator low output (if enabled), a capacitive sense greater-than comparator interrupt, or a device reset event. In order to run Timer 3 in Suspend mode, the timer must be configured to clock from the external clock source. When Suspend mode is terminated, the device will continue execution on the instruction following the one that set the SUSPEND bit. If the wake event (port match or Timer 3 overflow) was configured to generate an interrupt, the interrupt will be serviced upon waking the device. If Suspend mode is terminated by an internal or external reset, the CIP-51 performs a normal reset sequence and begins program execution at address 0x0000.

SFR Definition 22.1. PCON: Power Control

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

SFR Address = 0x87; SFR Page = All Pages

Bit	Name	Function
7:2	GF[5:0]	**General Purpose Flags 5–0.** These are general purpose flags for use under software control.
1	STOP	**Stop Mode Select.** Setting this bit will place the CIP-51 in Stop mode. This bit will always be read as 0. 1: CPU goes into Stop mode (internal oscillator stopped).
0	IDLE	**IDLE: Idle Mode Select.** Setting this bit will place the CIP-51 in Idle mode. This bit will always be read as 0. 1: CPU goes into Idle mode. (Shuts off clock to CPU, but clock to Timers, Interrupts, Serial Ports, and Analog Peripherals are still active.)

23. Reset Sources

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

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

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

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

On exit from the reset state, the program counter (PC) is reset, and the system clock defaults to the internal oscillator. The Watchdog Timer is enabled with the system clock divided by 12 as its clock source. Program execution begins at location 0x0000.

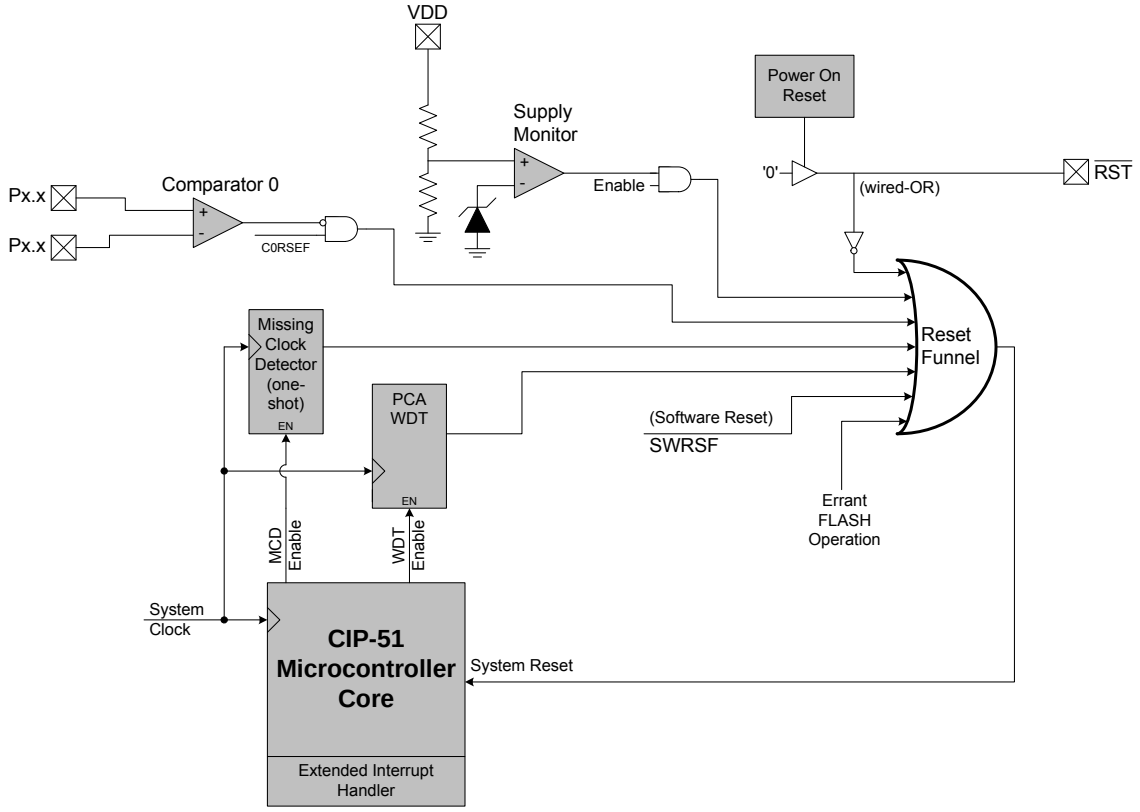


Figure 23.1. Reset Sources

23.1. Power-On Reset

During power-up, the device is held in a reset state and the $\overline{RST}$ pin is driven low until V_{DD} settles above V_{RST}. A delay occurs before the device is released from reset; the delay decreases as the V_{DD} ramp time increases (V_{DD} ramp time is defined as how fast V_{DD} ramps from 0 V to V_{RST}). Figure 23.2. plots the power-on and V_{DD} monitor reset timing. The maximum V_{DD} ramp time is 1 ms; slower ramp times may cause the device to be released from reset before V_{DD} reaches the V_{RST} level. For ramp times less than 1 ms, the power-on reset delay ($T_{PORDelay}$) is typically less than 10 ms.

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.

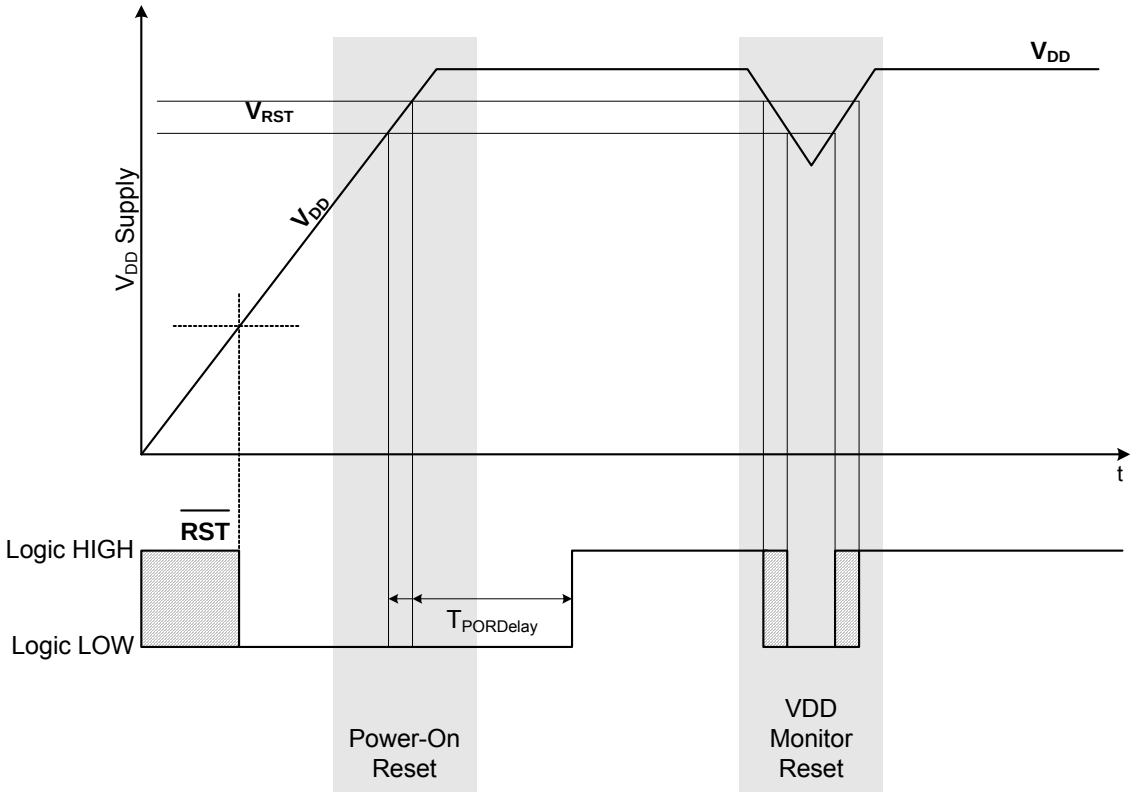


Figure 23.2. Power-On and V_{DD} Monitor Reset Timing

23.2. Power-Fail Reset / V_{DD} Monitor

When a power-down transition or power irregularity causes V_{DD} to drop below V_{RST}, the power supply monitor will drive the $\overline{RST}$ pin low and hold the CIP-51 in a reset state (see Figure 23.2). When V_{DD} returns to a level above V_{RST}, the CIP-51 will be released from the reset state. Even though internal data memory contents are not altered by the power-fail reset, it is impossible to determine if V_{DD} dropped below the level required for data retention. If the PORSF flag reads 1, the data may no longer be valid. The V_{DD} monitor is enabled after power-on resets. Its defined state (enabled/disabled) is not altered by any other reset source. For example, if the V_{DD} monitor is disabled by code and a software reset is performed, the V_{DD} monitor will still be disabled after the reset.

Important Note: If the V_{DD} monitor is being turned on from a disabled state, it should be enabled before it is selected as a reset source. Selecting the V_{DD} monitor as a reset source before it is enabled and stabilized may cause a system reset. In some applications, this reset may be undesirable. If this is not desirable in the application, a delay should be introduced between enabling the monitor and selecting it as a reset source. The procedure for enabling the V_{DD} monitor and configuring it as a reset source from a disabled state is shown below:

1. Enable the V_{DD} monitor (VDMEN bit in VDM0CN = 1).
2. If necessary, wait for the V_{DD} monitor to stabilize.
3. Select the V_{DD} monitor as a reset source (PORSF bit in RSTSRC = 1).

See Figure 23.2 for V_{DD} monitor timing; note that the power-on-reset delay is not incurred after a V_{DD} monitor reset. See Section “7. Electrical Characteristics” on page 39 for complete electrical characteristics of the V_{DD} monitor.

SFR Definition 23.1. VDM0CN: V_{DD} Monitor Control

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

SFR Address = 0xFF; SFR Page = All Pages

Bit	Name	Function
7	VDMEN	**V_{DD} Monitor Enable.** This bit turns the V_{DD} monitor circuit on/off. The V_{DD} Monitor cannot generate system resets until it is also selected as a reset source in register RSTSRC (SFR Definition 23.2). Selecting the V_{DD} monitor as a reset source before it has stabilized may generate a system reset. In systems where this reset would be undesirable, a delay should be introduced between enabling the V_{DD} Monitor and selecting it as a reset source. 0: V_{DD} Monitor Disabled. 1: V_{DD} Monitor Enabled.
6	VDDSTAT	**V_{DD} Status.** This bit indicates the current power supply status (V_{DD} Monitor output). 0: V_{DD} is at or below the V_{DD} monitor threshold. 1: V_{DD} is above the V_{DD} monitor threshold.
5:0	Unused	Read = 000000b; Write = Don't care.

23.3. External Reset

The external $\overline{RST}$ pin provides a means for external circuitry to force the device into a reset state. Asserting an active-low signal on the $\overline{RST}$ pin generates a reset; an external pullup and/or decoupling of the $\overline{RST}$ pin may be necessary to avoid erroneous noise-induced resets. See Section "7. Electrical Characteristics" on page 39 for complete $\overline{RST}$ pin specifications. The PINRSF flag (RSTSRC.0) is set on exit from an external reset.

23.4. Missing Clock Detector Reset

The Missing Clock Detector (MCD) is a one-shot circuit that is triggered by the system clock. If the system clock remains high or low for more than 100 µs, the one-shot will time out and generate a reset. After a MCD reset, the MCDRSF flag (RSTSRC.2) will read 1, signifying the MCD as the reset source; otherwise, this bit reads 0. Writing a 1 to the MCDRSF bit enables the Missing Clock Detector; writing a 0 disables it. The state of the $\overline{RST}$ pin is unaffected by this reset.

23.5. Comparator0 Reset

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

23.6. 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 Section “24. Watchdog Timer” on page 154. If a system malfunction prevents user software from updating the WDT, a reset is generated and the WDTRSF bit (RSTSRC.3) is set to 1. The state of the $\overline{RST}$ pin is unaffected by this reset.

23.7. Flash Error Reset

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

- A Flash write or erase is attempted above user code space. This occurs when PSWE is set to 1 and a MOVX write operation targets an address above address 0x3DFF.
- A Flash read is attempted above user code space. This occurs when a MOVC operation targets an address above address 0x3DFF.
- A Program read is attempted above user code space. This occurs when user code attempts to branch to an address above 0x3DFF.
- A Flash read, write or erase attempt is restricted due to a Flash security setting (see Section “20.3. Security Options” on page 134).

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

23.8. Software Reset

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

SFR Definition 23.2. RSTSRC: Reset Source

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

SFR Address = 0xEF; SFR Page = All Pages

Bit	Name	Description	Write	Read
7	Unused	**Unused.**	Don't care.	0
6	FERROR	**Flash Error Reset Flag.**	N/A	Set to 1 if Flash read/write/erase error caused the last reset.
5	C0RSEF	**Comparator0 Reset Enable and Flag.**	Writing a 1 enables Comparator0 as a reset source (active-low).	Set to 1 if Comparator0 caused the last reset.
4	SWRSF	**Software Reset Force and Flag.**	Writing a 1 forces a system reset.	Set to 1 if last reset was caused by a write to SWRSF.
3	WDTRSF	**Watchdog Timer Reset Flag.**	N/A	Set to 1 if Watchdog Timer overflow caused the last reset.
2	MCDRSF	**Missing Clock Detector Enable and Flag.**	Writing a 1 enables the Missing Clock Detector. The MCD triggers a reset if a missing clock condition is detected.	Set to 1 if Missing Clock Detector timeout caused the last reset.
1	PORSF	**Power-On / V_{DD} Monitor Reset Flag, and V_{DD} monitor Reset Enable.**	Writing a 1 enables the V_{DD} monitor as a reset source. **Writing 1 to this bit before the V_{DD} monitor is enabled and stabilized may cause a system reset.**	Set to 1 anytime a power-on or V_{DD} monitor reset occurs. **When set to 1 all other RSTSRC flags are indeterminate.**
0	PINRSF	**HW Pin Reset Flag.**	N/A	Set to 1 if $\overline{RST}$ pin caused the last reset.
Note: Do not use read-modify-write operations on this register				

24. Watchdog Timer

The MCU includes a programmable Watchdog Timer (WDT) running off the system clock. 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 /RST pin is unaffected by this reset.

The WDT consists of a 21-bit timer running from the programmed system clock. 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. Watchdog features are controlled via the Watchdog Timer Control Register (WDTCN) shown in SFR Definition 24.1.

24.1. Enable/Reset WDT

The watchdog timer is both enabled and reset by 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 WDT is enabled and reset as a result of any system reset.

24.2. Disable 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
SETB EA          ; re-enable interrupts
```

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.

24.3. Disable 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.

24.4. Setting WDT Interval

WDTCN.[2:0] control the watchdog timeout interval. The interval is given by the following equation:

4^(3+WDTCN[2-0]) x Tsysclk ;where T_{sysclk} is the system clock period.

For a 3 MHz system clock, this provides an interval range of 0.021 to 349.5 ms. 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.

SFR Definition 24.1. WDTCN: Watchdog Timer Control

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

SFR Address = 0xE3; SFR Page = All Pages

Bit	Name	Description	Write	Read
7:0	WDT[7:0]	**WDT Control.**	Writing 0xA5 both enables and reloads the WDT. Writing 0xDE followed within 4 system clocks by 0xAD disables the WDT. Writing 0xFF locks out the disable feature.	
4	WDTSTATUS	**Watchdog Status Bit.**		0: WDT is inactive 1: WDT is active
2:0	WDTTIMEOUT	**Watchdog Timeout Interval Bits.**	WDTCN[2:0] bits set the Watchdog Timeout Interval. When writing these bits, WDTCN[7] must be set to 0.	

25. Oscillators and Clock Selection

C8051F70x/71x devices include a programmable internal high-frequency oscillator and an external oscillator drive circuit. The internal high-frequency oscillator can be enabled/disabled and calibrated using the OSCICN and OSCICL registers, as shown in Figure 25.1. The system clock can be sourced by the external oscillator circuit or the internal oscillator (default). The internal oscillator offers a selectable post-scaling feature, which is initially set to divide the clock by 8.

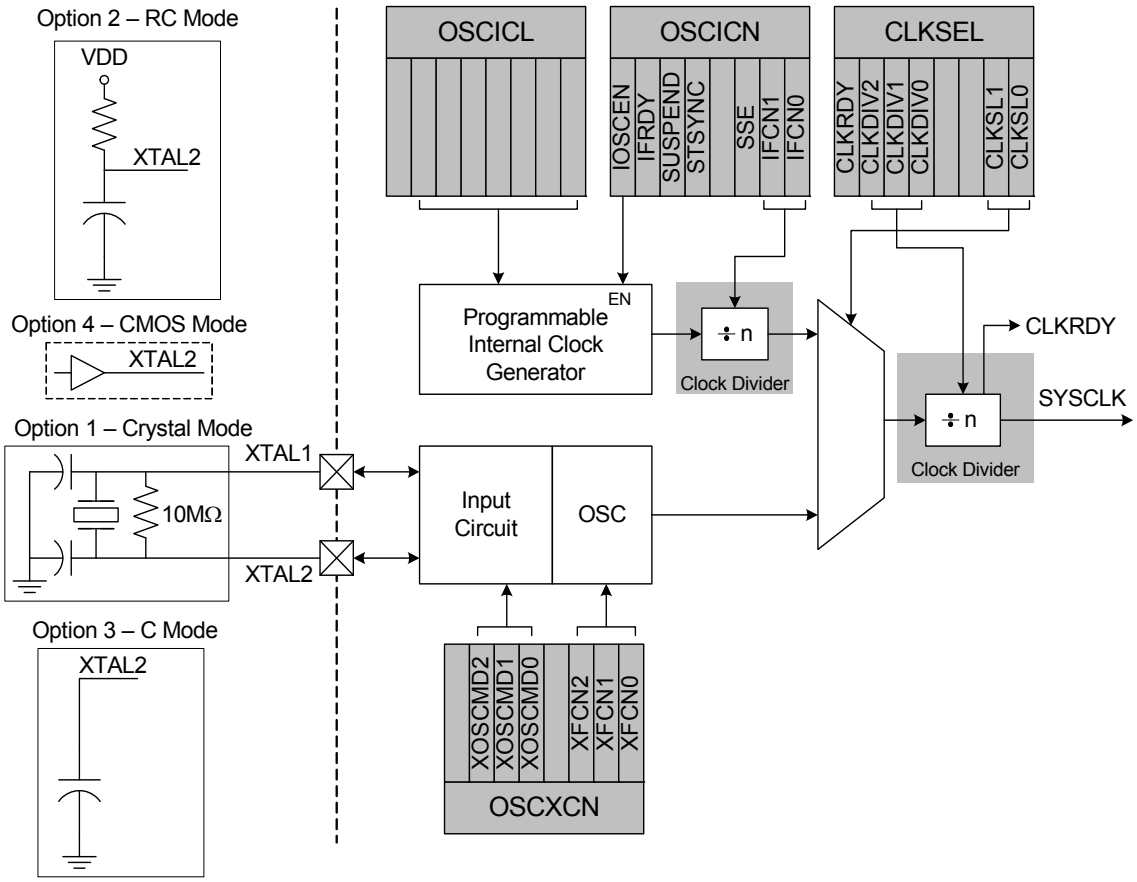


Figure 25.1. Oscillator Options

25.1. System Clock Selection

The system clock source for the MCU can be selected using the CLKSEL register. The clock selected as the system clock can be divided by 1, 2, 4, 8, 16, 32, 64, or 128. When switching between two clock divide values, the transition may take up to 128 cycles of the undivided clock source. The CLKRDY flag can be polled to determine when the new clock divide value has been applied. The clock divider must be set to "divide by 1" when entering Suspend mode. The system clock source may also be switched on-the-fly. The switchover takes effect after one clock period of the slower oscillator.

SFR Definition 25.1. CLKSEL: Clock Select

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

SFR Address = 0xBD; SFR Page= F

Bit	Name	Function
7	CLKRDY	**System Clock Divider Clock Ready Flag.** 0: The selected clock divide setting has not been applied to the system clock. 1: The selected clock divide setting has been applied to the system clock.
6:4	CLKDIV	**System Clock Divider Bits.** Selects the clock division to be applied to the selected source (internal or external). 000: Selected clock is divided by 1. 001: Selected clock is divided by 2. 010: Selected clock is divided by 4. 011: Selected clock is divided by 8. 100: Selected clock is divided by 16. 101: Selected clock is divided by 32. 110: Selected clock is divided by 64. 111: Selected clock is divided by 128.
3	Reserved	Read = 0b. Must write 0b.
2:0	CLKSEL[2:0]	**System Clock Select.** Selects the oscillator to be used as the undivided system clock source. 000: Internal Oscillator 001: External Oscillator All other values reserved.

25.2. Programmable Internal High-Frequency (H-F) Oscillator

All C8051F70x/71x devices include a programmable internal high-frequency oscillator that defaults as the system clock after a system reset. The internal oscillator period can be adjusted via the OSCICL register as defined by SFR Definition 25.2.

On C8051F70x/71x devices, OSCICL is factory calibrated to obtain a 24.5 MHz base frequency.

The internal oscillator output frequency may be divided by 1, 2, 4, or 8, as defined by the IFCN bits in register OSCICN. The divide value defaults to 8 following a reset.

The precision oscillator supports a spread spectrum mode which modulates the output frequency in order to reduce the EMI generated by the system. When enabled (SSE = 1), the oscillator output frequency is modulated by a stepped triangle wave whose frequency is equal to the oscillator frequency divided by 384 (63.8 kHz using the factory calibration). The maximum deviation from the center frequency is ±0.75%. The output frequency updates occur every 32 cycles and the step size is typically 0.25% of the center frequency.

SFR Definition 25.2. OSCICL: Internal H-F Oscillator Calibration

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

SFR Address = 0xBF; SFR Page = F

Bit	Name	Function
6:0	OSCICL[7:0]	**Internal Oscillator Calibration Bits.** These bits determine the internal oscillator period. When set to 00000000b, the H-F oscillator operates at its fastest setting. When set to 11111111b, the H-F oscillator operates at its slowest setting. The reset value is factory calibrated to generate an internal oscillator frequency of 24.5 MHz.

SFR Definition 25.3. OSCICN: Internal H-F Oscillator Control

Bit	7	6	5	4	3	2	1	0
Name	IOSCEN	IFRDY	SUSPEND	STSYNC	SSE		IFCN[1:0]	
Type	R/W	R	R/W	R	R/W	R	R/W	
Reset	1	1	0	0	0	0	0	0

SFR Address = 0xA9; SFR Page = F

Bit	Name	Function
7	IOSCEN	**Internal H-F Oscillator Enable Bit.** 0: Internal H-F Oscillator Disabled. 1: Internal H-F Oscillator Enabled.
6	IFRDY	**Internal H-F Oscillator Frequency Ready Flag.** 0: Internal H-F Oscillator is not running at programmed frequency. 1: Internal H-F Oscillator is running at programmed frequency.
5	SUSPEND	**Internal Oscillator Suspend Enable Bit.** Setting this bit to logic 1 places the internal oscillator in SUSPEND mode. The internal oscillator resumes operation when one of the SUSPEND mode awakening events occurs.
4	STSYNC	**Suspend Timer Synchronization Bit.** This bit is used to indicate when it is safe to read and write the registers associated with the suspend wake-up timer. If a suspend wake-up source other than Timer 3 has brought the oscillator out of suspend mode, it make take up to three timer clocks before the timer can be read or written. 0: Timer 3 registers can be read safely. 1: Timer 3 register reads and writes should not be performed.
3	SSE	**Spread Spectrum Enable.** Spread spectrum enable bit. 0: Spread Spectrum clock dithering disabled. 1: Spread Spectrum clock dithering enabled.
2	Unused	Read = 0b; Write = Don't Care
1:0	IFCN[1:0]	**Internal H-F Oscillator Frequency Divider Control Bits.** 00: SYSCLK derived from Internal H-F Oscillator divided by 8. 01: SYSCLK derived from Internal H-F Oscillator divided by 4. 10: SYSCLK derived from Internal H-F Oscillator divided by 2. 11: SYSCLK derived from Internal H-F Oscillator divided by 1.

25.3. External Oscillator Drive Circuit

The external oscillator circuit may drive an external crystal, ceramic resonator, capacitor, or RC network. A CMOS clock may also provide a clock input. For a crystal or ceramic resonator configuration, the crystal/resonator must be wired across the XTAL1 and XTAL2 pins as shown in Option 1 of Figure 25.1. A 10 MΩ resistor also must be wired across the XTAL2 and XTAL1 pins for the crystal/resonator configuration. In RC, capacitor, or CMOS clock configuration, the clock source should be wired to the XTAL2 pin as shown in Option 2, 3, or 4 of Figure 25.1. The type of external oscillator must be selected in the OSCXCN register, and the frequency control bits (XFCN) must be selected appropriately (see SFR Definition 25.4).

Important Note on External Oscillator Usage: Port pins must be configured when using the external oscillator circuit. When the external oscillator drive circuit is enabled in crystal/resonator mode, Port pins P0.2 and P0.3 are used as XTAL1 and XTAL2 respectively. When the external oscillator drive circuit is enabled in capacitor, RC, or CMOS clock mode, Port pin P0.3 is used as XTAL2. The Port I/O Crossbar should be configured to skip the Port pins used by the oscillator circuit; see Section “26.3. Priority Crossbar Decoder” on page 170 for Crossbar configuration. Additionally, when using the external oscillator circuit in crystal/resonator, capacitor, or RC mode, the associated Port pins should be configured as **analog inputs**. In CMOS clock mode, the associated pin should be configured as a **digital input**. See Section “26.4. Port I/O Initialization” on page 172 for details on Port input mode selection.

SFR Definition 25.4. OSCXCN: External Oscillator Control

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

SFR Address = 0xB5; SFR Page = F

Bit	Name	Function
7	XTLVLD	**Crystal Oscillator Valid Flag.** (Read only when XOSCMD = 11x.) 0: Crystal Oscillator is unused or not yet stable. 1: Crystal Oscillator is running and stable.
6:4	XOSCMD[2:0]	**External Oscillator Mode Select.** 00x: External Oscillator circuit off. 010: External CMOS Clock Mode. 011: External CMOS Clock Mode with divide by 2 stage. 100: RC Oscillator Mode. 101: Capacitor Oscillator Mode. 110: Crystal Oscillator Mode. 111: Crystal Oscillator Mode with divide by 2 stage.
3	UNUSED	Read = 0; Write = Don't Care
2:0	XFCN[2:0]	**External Oscillator Frequency Control Bits.** Set according to the desired frequency for Crystal or RC mode. Set according to the desired K Factor for C mode.

XFCN	Crystal Mode	RC Mode	C Mode
000	f ≤ 32 kHz	f ≤ 25 kHz	K Factor = 0.87
001	32 kHz < f ≤ 84 kHz	25 kHz < f ≤ 50 kHz	K Factor = 2.6
010	84 kHz < f ≤ 225 kHz	50 kHz < f ≤ 100 kHz	K Factor = 7.7
011	225 kHz < f ≤ 590 kHz	100 kHz < f ≤ 200 kHz	K Factor = 22
100	590 kHz < f ≤ 1.5 MHz	200 kHz < f ≤ 400 kHz	K Factor = 65
101	1.5 MHz < f ≤ 4 MHz	400 kHz < f ≤ 800 kHz	K Factor = 180
110	4 MHz < f ≤ 10 MHz	800 kHz < f ≤ 1.6 MHz	K Factor = 664
111	10 MHz < f ≤ 30 MHz	1.6 MHz < f ≤ 3.2 MHz	K Factor = 1590

25.3.1. External Crystal Example

If a crystal or ceramic resonator is used as an external oscillator source for the MCU, the circuit should be configured as shown in Figure 25.1, Option 1. The External Oscillator Frequency Control value (XFCN) should be chosen from the Crystal column of the table in SFR Definition 25.4 (OSCXCN register). For example, an 11.0592 MHz crystal requires an XFCN setting of 111b and a 32.768 kHz Watch Crystal requires an XFCN setting of 001b. After an external 32.768 kHz oscillator is stabilized, the XFCN setting can be switched to 000 to save power. It is recommended to enable the missing clock detector before switching the system clock to any external oscillator source.

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

1. Force XTAL1 and XTAL2 to a low state. This involves enabling the Crossbar and writing 0 to the port pins associated with XTAL1 and XTAL2.
2. Configure XTAL1 and XTAL2 as analog inputs.
3. Enable the external oscillator.
4. Wait at least 1 ms.
5. Poll for XTLVLD = 1.
6. If desired, enable the Missing Clock Detector.
7. Switch the system clock to the external oscillator.

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

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

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

For example, a tuning-fork crystal of 32.768 kHz with a recommended load capacitance of 12.5 pF should use the configuration shown in Figure 25.1, Option 1. The total value of the capacitors and the stray capacitance of the XTAL pins should equal 25 pF. With a stray capacitance of 3 pF per pin, the 22 pF capacitors yield an equivalent capacitance of 12.5 pF across the crystal, as shown in Figure 25.2.

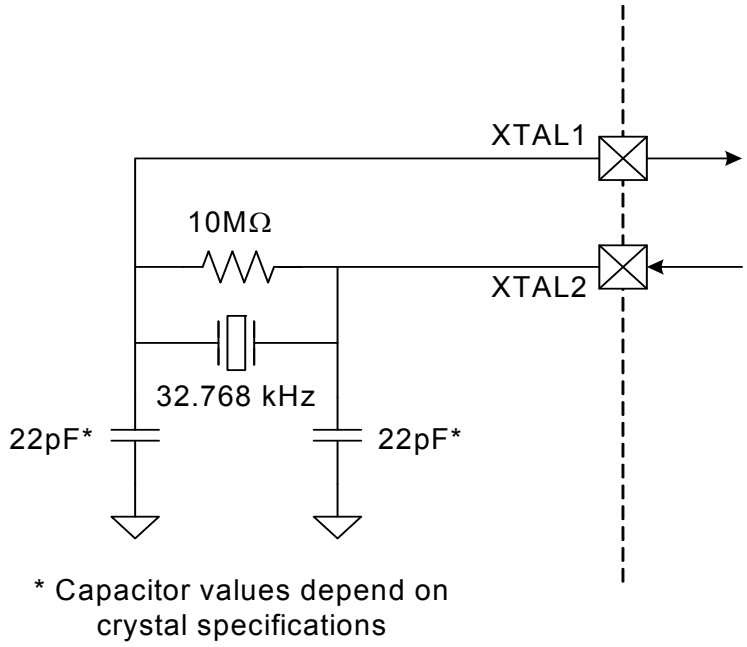


Figure 25.2. External 32.768 kHz Quartz Crystal Oscillator Connection Diagram

25.3.2. External RC Example

If an RC network is used as an external oscillator source for the MCU, the circuit should be configured as shown in Figure 25.1, Option 2. The capacitor should be no greater than 100 pF; however for very small capacitors, the total capacitance may be dominated by parasitic capacitance in the PCB layout. To determine the required External Oscillator Frequency Control value (XFCN) in the OSCXCN Register, first select the RC network value to produce the desired frequency of oscillation, according to Equation 25.1, where f = the frequency of oscillation in MHz, C = the capacitor value in pF, and R = the pull-up resistor value in kΩ.

Equation 25.1. RC Mode Oscillator Frequency

$$f = 1.23 \times 10^3 / (R \times C)$$

For example: If the frequency desired is 100 kHz, let R = 246 kΩ and C = 50 pF:

f = 1.23(10^3) / RC = 1.23 (10^3) / [246 x 50] = 0.1 MHz = 100 kHz

Referring to the table in SFR Definition 25.4, the required XFCN setting is 010b.

25.3.3. External Capacitor Example

If a capacitor is used as an external oscillator for the MCU, the circuit should be configured as shown in Figure 25.1, Option 3. The capacitor should be no greater than 100 pF; however for very small capacitors, the total capacitance may be dominated by parasitic capacitance in the PCB layout. To determine the required External Oscillator Frequency Control value (XFCN) in the OSCXCN Register, select the capacitor to be used and find the frequency of oscillation according to Equation 25.2, where f = the frequency of oscillation in MHz, C = the capacitor value in pF, and V_{DD} = the MCU power supply in Volts.

Equation 25.2. C Mode Oscillator Frequency

$$f = (KF)/(R \times V_{DD})$$

For example: Assume V_{DD} = 3.0 V and f = 150 kHz:

f = KF / (C x VDD)
0.150 MHz = KF / (C x 3.0)

Since the frequency of roughly 150 kHz is desired, select the K Factor from the table in SFR Definition 25.4 (OSCXCN) as KF = 22:

0.150 MHz = 22 / (C x 3.0)
C x 3.0 = 22 / 0.150 MHz
C = 146.6 / 3.0 pF = 48.8 pF

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

26. Port Input/Output

Digital and analog resources are available through 64 I/O pins. Each of the Port pins P0.0-P2.7 can be defined as general-purpose I/O (GPIO), assigned to one of the internal digital resources, or assigned to an analog function as shown in Figure 26.4. 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. 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 26.4 and Figure 26.5). The registers XBR0 and XBR1, defined in SFR Definition 26.1 and SFR Definition 26.2, are used to select internal digital functions.

All Port I/Os are tolerant of voltages up to 2 V above the V_{DD} supply (refer to Figure 26.2 for the Port cell circuit). The Port I/O cells are configured as either push-pull or open-drain in the Port Output Mode registers (PnMDOUT, where n = 0,1). Complete Electrical Specifications for Port I/O are given in Section "7. Electrical Characteristics" on page 39.

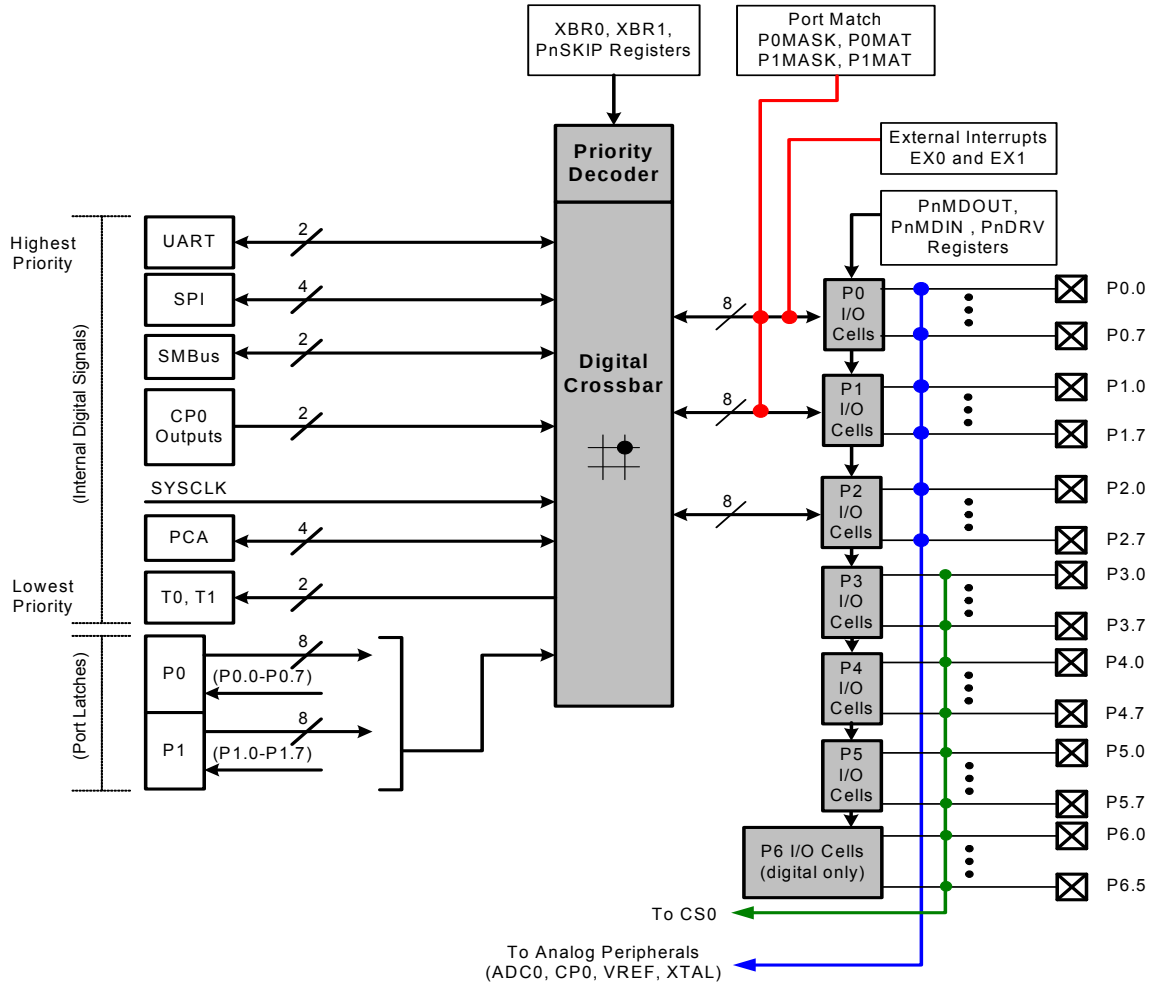


Figure 26.1. Port I/O Functional Block Diagram

26.1. Port I/O Modes of Operation

Port pins P0.0 - P6.5 use the Port I/O cell shown in Figure 26.2. Each Port I/O cell can be configured by software for analog I/O or digital I/O using the PnMDIN registers. On reset, all Port I/O cells default to a high impedance state with weak pull-ups enabled. Until the crossbar is enabled (XBARE = 1), both the high and low port I/O drive circuits are explicitly disabled on all crossbar pins.

26.1.1. Port Pins Configured for Analog I/O

Any pins to be used as Comparator or ADC input, Capacitive Sense input, external oscillator input/output, VREF output, or AGND connection should be configured for analog I/O (PnMDIN.n = 0). When a pin is configured for analog I/O, its weak pullup, digital driver, and digital receiver are disabled. Port pins configured for analog I/O will always read back a value of 0.

Configuring pins as analog I/O saves power and isolates the Port pin from digital interference. 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.

26.1.2. Port Pins Configured For Digital I/O

Any pins to be used by digital peripherals (UART, SPI, SMBus, etc.), external event trigger functions, or as GPIO should be configured as digital I/O (PnMDIN.n = 1). For digital I/O pins, one of two output modes (push-pull or open-drain) must be selected using the PnMDOUT registers.

Push-pull outputs (PnMDOUT.n = 1) drive the Port pad to the VDD or GND supply rails based on the output logic value of the Port pin. Open-drain outputs have the high side driver disabled; therefore, they only drive the Port pad to GND when the output logic value is 0 and become high impedance inputs (both high low drivers turned off) when the output logic value is 1.

When a digital I/O cell is placed in the high impedance state, a weak pull-up transistor pulls the Port pad to the VDD supply voltage to ensure the digital input is at a defined logic state. Weak pull-ups are disabled when the I/O cell is driven to GND to minimize power consumption, and 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.

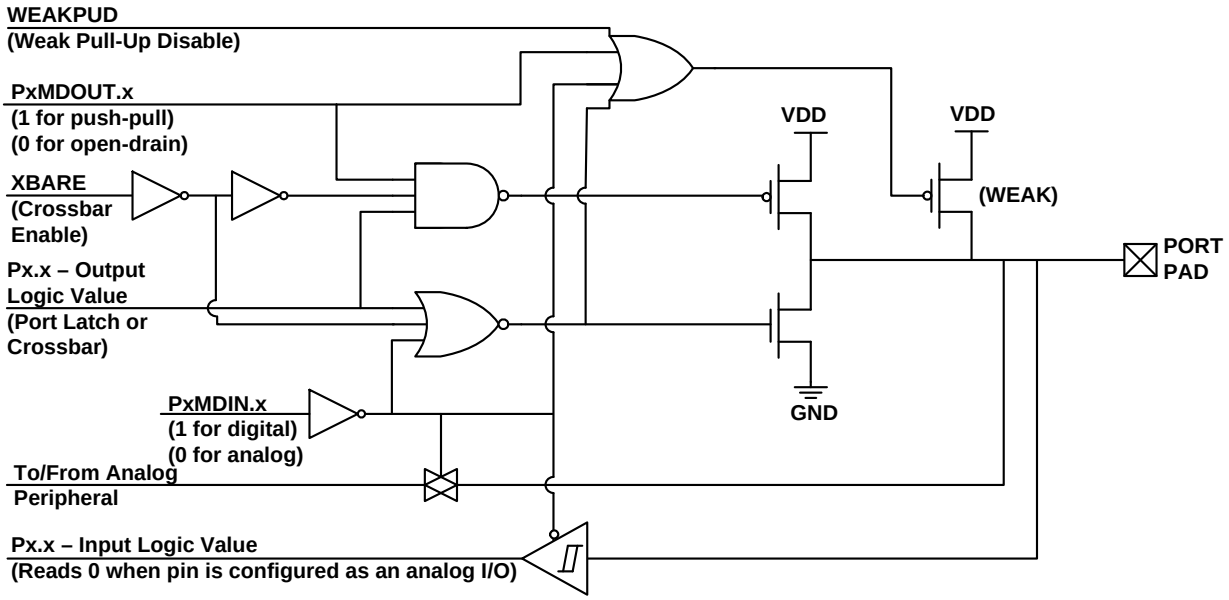
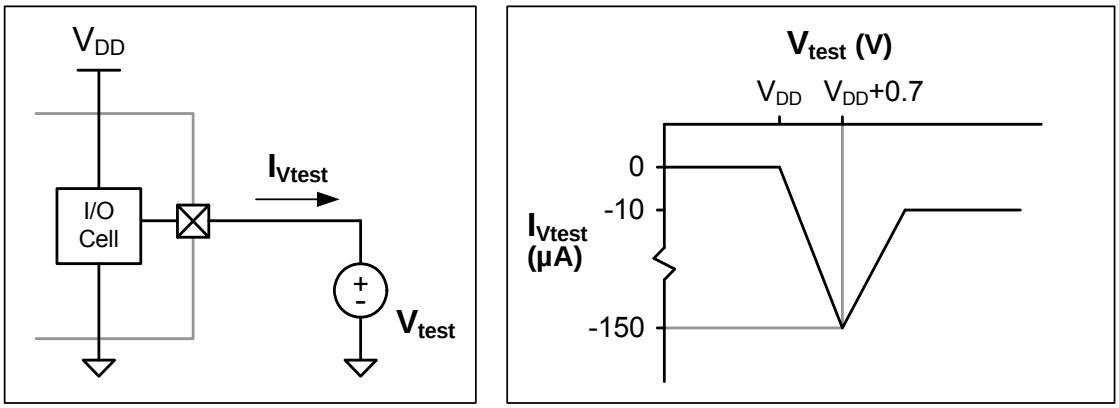


Figure 26.2. Port I/O Cell Block Diagram

26.1.3. Interfacing Port I/O to 5 V Logic

All Port I/O configured for digital, open-drain operation are capable of interfacing to digital logic operating at a supply voltage up to 2 V higher than VDD and less than 5.25 V. An external pull-up resistor to the higher supply voltage is typically required for most systems.

Important Note: In a multi-voltage interface, the external pull-up resistor should be sized to allow a current of at least 150 µA to flow into the Port pin when the supply voltage is between (VDD + 0. 6V) and (VDD + 1.0V). Once the Port pin voltage increases beyond this range, the current flowing into the Port pin is minimal. Figure 26.3 shows the input current characteristics of port pins driven above VDD. The port pin requires 150 µA peak overdrive current when its voltage reaches approximately (VDD + 0.7 V).



Port I/O Overdrive Test Circuit **Port I/O Overdrive Current vs. Voltage**

Figure 26.3. Port I/O Overdrive Current

26.1.4. Increasing Port I/O Drive Strength

Port I/O output drivers support a high and low drive strength; the default is low drive strength. The drive strength of a Port I/O can be configured using the PnDRV registers. See Section “7. Electrical Characteristics” on page 39 for the difference in output drive strength between the two modes.

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

Port I/O pins P0.0–P2.7 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.

26.2.1. Assigning Port I/O Pins to Analog Functions

Table 26.1 shows all available analog functions that require Port I/O assignments. **Port pins selected for these analog functions should have their corresponding bit in PnSKIP set to 1.** This reserves the pin for use by the analog function and does not allow it to be claimed by the Crossbar. Table 26.1 shows the potential mapping of Port I/O to each analog function.

Table 26.1. Port I/O Assignment for Analog Functions

Analog Function	Potentially Assignable Port Pins	SFR(s) used for Assignment
ADC Input	P0.0–P2.7	AMX0P, AMX0N, PnSKIP, PnMDIN
Comparator0 Input	P0.0–P1.7	CPT0MX, PnSKIP, PnMDIN
CS0 Input	P2.0–P5.7	PnMDIN
Voltage Reference (VREF0)	P0.0	REF0CN, P0SKIP, PnMDIN
Ground Reference (AGND)	P0.1	REF0CN, P0SKIP
External Oscillator in Crystal Mode (XTAL1)	P0.2	OSCXCN, P0SKIP, P0MDIN
External Oscillator in RC, C, or Crystal Mode (XTAL2)	P0.3	OSCXCN, P0SKIP, P0MDIN

26.2.2. Assigning Port I/O Pins to Digital Functions

Any Port pins not assigned to analog functions may be assigned to digital functions or used as GPIO. Most digital functions rely on the Crossbar for pin assignment; however, some digital functions bypass the Crossbar in a manner similar to the analog functions listed above. **Port pins used by these digital functions and any Port pins selected for use as GPIO should have their corresponding bit in PnSKIP set to 1.** Table 26.2 shows all available digital functions and the potential mapping of Port I/O to each digital function.

Table 26.2. Port I/O Assignment for Digital Functions

Digital Function	Potentially Assignable Port Pins	SFR(s) used for Assignment
UART0, SPI0, SMBus, CP0, CP0A, SYSCLK, PCA0 (CEX0-2 and ECI), T0 or T1.	Any Port pin available for assignment by the Crossbar. This includes P0.0–P2.7 pins which have their PnSKIP bit set to 0. **Note:** The Crossbar will always assign UART0 pins to P0.4 and P0.5.	XBR0, XBR1
Any pin used for GPIO	P0.0–P6.5	P0SKIP, P1SKIP, P2SKIP
External Memory Interface	P3.0–P6.2	EMI0CF

26.2.3. Assigning Port I/O Pins to External Event Trigger Functions

External event trigger functions can be used to trigger an interrupt or wake the device from a low power mode when a transition occurs on a digital I/O pin. The event trigger functions do not require dedicated pins and will function on both GPIO pins (PnSKIP = 1) and pins in use by the Crossbar (PnSKIP = 0). External event trigger functions cannot be used on pins configured for analog I/O. Table 26.3 shows all available external event trigger functions.

Table 26.3. Port I/O Assignment for External Event Trigger Functions

Event Trigger Function	Potentially Assignable Port Pins	SFR(s) used for Assignment
External Interrupt 0	P0.0–P0.7	IT01CF
External Interrupt 1	P0.0–P0.7	IT01CF
Port Match	P0.0–P1.7	P0MASK, P0MAT P1MASK, P1MAT

26.3. Priority Crossbar Decoder

The Priority Crossbar Decoder (Figure 26.4) assigns a priority to each I/O function, starting at the top with UART0. When a digital resource is selected, the least-significant unassigned Port pin is assigned to that resource (excluding UART0, which is always at pins 4 and 5). If a Port pin is assigned, the Crossbar skips that pin when assigning the next selected resource. Additionally, the Crossbar will skip Port pins whose associated bits in the PnSKIP registers are set. The PnSKIP registers allow software to skip Port pins that are to be used for analog input, dedicated functions, or GPIO.

Important Note on Crossbar Configuration: If a Port pin is claimed by a peripheral without use of the Crossbar, its corresponding PnSKIP bit should be set. This applies to P0.0 if VREF is used, P0.1 if AGND is used, P0.3 and/or P0.2 if the external oscillator circuit 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 26.5 shows the Crossbar Decoder priority with pins P0.2 and P0.6 skipped (P0SKIP = 0x44).

	P0								P1								P2							
SF Signals	VREF	AGND	XTAL1	XTAL2			CNVSTR																	
PIN I/O	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
TX0					■																			
RX0						■																		
SCK	■																							
MISO		■																						
MOSI			■																					
NSS[1]				■																				
SDA	■	■	■	■	■																			
SCL		■	■	■	■	■																		
CP0	■		■	■	■	■	■																	
CP0A	■	■	■	■	■	■	■	■																
SYSCLK	■	■	■	■	■	■	■	■	■															
CEX0	■	■	■	■	■	■	■	■	■	■														
CEX1		■	■	■	■	■	■	■	■	■	■													
CEX2			■	■	■	■	■	■	■	■	■	■												
ECI	■	■	■	■	■	■	■	■	■	■	■	■	■											
T0	■	■	■	■	■	■	■	■	■	■	■	■	■	■										
T1	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■									

■ Port pin potentially available to peripheral

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

Notes:
1. NSS is only pinned out in 4-wire SPI Mode

Figure 26.4. Crossbar Priority Decoder—Possible Pin Assignments

	P0								P1								P2							
SF Signals	VREF	AGND	XTAL1	XTAL2			CNVSTR																	
PIN I/O	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
TX0																								
RX0																								
SCK																								
MISO																								
MOSI																								
NSS[1]																								
SDA																								
SCL																								
CP0																								
CP0A																								
SYSCLK																								
CEX0																								
CEX1																								
CEX2																								
ECI																								
T0																								
T1																								
	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	P0SKIP[0:7]								P1SKIP[0:7]								P2SKIP[0:7]							

Port pin potentially available to peripheral

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

Notes:
1. NSS is only pinned out in 4-wire SPI Mode

Figure 26.5. Crossbar Priority Decoder Example

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

Important 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.

26.4. 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. See SFR Definition 26.8 for the PnMDIN register details.

The output driver characteristics of the I/O pins are defined using the Port Output Mode registers (PnMD-OUT). 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 XBR1 to 1 enables the Crossbar. Until the Crossbar is enabled, the external pins remain as standard Port I/O (in input mode), regardless of the XBRn Register settings. For given XBRn Register settings, one can determine the I/O pin-out using the Priority Decode Table; as an alternative, the Configuration Wizard utility of the Silicon Labs IDE software will determine the Port I/O pin-assignments based on the XBRn Register settings.

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

SFR Definition 26.1. XBR0: Port I/O Crossbar Register 0

Bit	7	6	5	4	3	2	1	0
Name			CP0AE	CP0E	SYSCKE	SMB0E	SPI0E	URT0E
Type	R	R	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xE1; SFR Page = F

Bit	Name	Function
7:6	Unused	Read = 00b; Write = Don't Care.
5	CP0AE	**Comparator0 Asynchronous Output Enable.** 0: Asynchronous CP0 unavailable at Port pin. 1: Asynchronous CP0 routed to Port pin.
4	CP0E	**Comparator0 Output Enable.** 0: CP0 unavailable at Port pin. 1: CP0 routed to Port pin.
3	SYSCKE	**$\overline{\text{SYSCLK}}$ Output Enable.** 0: $\overline{\text{SYSCLK}}$ unavailable at Port pin. 1: $\overline{\text{SYSCLK}}$ output routed to Port pin.
2	SMB0E	**SMBus I/O Enable.** 0: SMBus I/O unavailable at Port pins. 1: SMBus I/O routed to Port pins.
1	SPI0E	**SPI I/O Enable.** 0: SPI I/O unavailable at Port pins. 1: SPI I/O routed to Port pins. Note that the SPI can be assigned either 3 or 4 GPIO pins.
0	URT0E	**UART I/O Output Enable.** 0: UART I/O unavailable at Port pin. 1: UART TX0, RX0 routed to Port pins P0.4 and P0.5.

SFR Definition 26.2. XBR1: Port I/O Crossbar Register 1

Bit	7	6	5	4	3	2	1	0
Name	WEAKPUD	XBARE	T1E	T0E	ECIE		PCA0ME[1:0]	
Type	R/W	R/W	R/W	R/W	R/W	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xE2; SFR Page = F

Bit	Name	Function
7	WEAKPUD	**Port I/O Weak Pullup Disable.** 0: Weak Pullups enabled (except for Ports whose I/O are configured for analog mode). 1: Weak Pullups disabled.
6	XBARE	**Crossbar Enable.** 0: Crossbar disabled. 1: Crossbar enabled.
5	T1E	**T1 Enable.** 0: T1 unavailable at Port pin. 1: T1 routed to Port pin.
4	T0E	**T0 Enable.** 0: T0 unavailable at Port pin. 1: T0 routed to Port pin.
3	ECIE	**PCA0 External Counter Input Enable.** 0: ECI unavailable at Port pin. 1: ECI routed to Port pin.
2	Unused	Read = 0b; Write = Don’t Care.
1:0	PCA0ME[1:0]	**PCA Module I/O Enable Bits.** 00: All PCA I/O unavailable at Port pins. 01: CEX0 routed to Port pin. 10: CEX0, CEX1 routed to Port pins. 11: CEX0, CEX1, CEX2 routed to Port pins.

26.5. Port Match

Port match functionality allows system events to be triggered by a logic value change on P0 or P1. A software controlled value stored in the PnMATCH registers specifies the expected or normal logic values of P0 and P1. A Port mismatch event occurs if the logic levels of the Port's input pins no longer match the software controlled value. This allows Software to be notified if a certain change or pattern occurs on P0 or P1 input pins regardless of the XBRn settings.

The PnMASK registers can be used to individually select which P0 and P1 pins should be compared against the PnMATCH registers. A Port mismatch event is generated if (P0 & P0MASK) does not equal (P0MATCH & P0MASK) or if (P1 & P1MASK) does not equal (P1MATCH & P1MASK).

A Port mismatch event may be used to generate an interrupt or wake the device from a low power mode, such as IDLE or SUSPEND. See the Interrupts and Power Options chapters for more details on interrupt and wake-up sources.

SFR Definition 26.3. P0MASK: Port 0 Mask Register

Bit	7	6	5	4	3	2	1	0
Name	P0MASK[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xF4; SFR Page = 0

Bit	Name	Function
7:0	P0MASK[7:0]	**Port 0 Mask Value.** Selects P0 pins to be compared to the corresponding bits in P0MAT. 0: P0.n pin logic value is ignored and cannot cause a Port Mismatch event. 1: P0.n pin logic value is compared to P0MAT.n.

SFR Definition 26.4. P0MAT: Port 0 Match Register

Bit	7	6	5	4	3	2	1	0
Name	P0MAT[7:0]							
Type	R/W							
Reset	1	1	1	1	1	1	1	1

SFR Address = 0xF3; SFR Page = 0

Bit	Name	Function
7:0	P0MAT[7:0]	**Port 0 Match Value.** Match comparison value used on Port 0 for bits in P0MASK which are set to 1. 0: P0.n pin logic value is compared with logic LOW. 1: P0.n pin logic value is compared with logic HIGH.

SFR Definition 26.5. P1MASK: Port 1 Mask Register

Bit	7	6	5	4	3	2	1	0
Name	P1MASK[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xE2; SFR Page = 0

Bit	Name	Function
7:0	P1MASK[7:0]	**Port 1 Mask Value.** Selects P1 pins to be compared to the corresponding bits in P1MAT. 0: P1.n pin logic value is ignored and cannot cause a Port Mismatch event. 1: P1.n pin logic value is compared to P1MAT.n.

SFR Definition 26.6. P1MAT: Port 1 Match Register

Bit	7	6	5	4	3	2	1	0
Name	P1MAT[7:0]							
Type	R/W							
Reset	1	1	1	1	1	1	1	1

SFR Address = 0xE1; SFR Page = 0

Bit	Name	Function
7:0	P1MAT[7:0]	**Port 1 Match Value.** Match comparison value used on Port 1 for bits in P1MASK which are set to 1. 0: P1.n pin logic value is compared with logic LOW. 1: P1.n pin logic value is compared with logic HIGH.

26.6. Special Function Registers for Accessing and Configuring Port I/O

All Port I/O are accessed through corresponding special function registers (SFRs) that are both byte addressable and bit addressable. When writing to a Port, the value written to the SFR is latched to maintain the output data value at each pin. When reading, the logic levels of the Port's input pins are returned regardless of the XBRn settings (i.e., even when the pin is assigned to another signal by the Crossbar, the Port register can always read its corresponding Port I/O pin). The exception to this is the execution of the read-modify-write instructions that target a Port Latch register as the destination. The read-modify-write instructions when operating on a Port SFR are the following: ANL, ORL, XRL, JBC, CPL, INC, DEC, DJNZ and MOV, CLR or SETB, when the destination is an individual bit in a Port SFR. For these instructions, the value of the latch register (not the pin) is read, modified, and written back to the SFR.

Each Port has a corresponding PnSKIP register which allows its individual Port pins to be assigned to digital functions or skipped by the Crossbar. All Port pins used for analog functions, GPIO, or dedicated digital functions such as the EMIF should have their PnSKIP bit set to 1.

The Port input mode of the I/O pins is defined using the Port Input Mode registers (PnMDIN). Each Port cell can be configured for analog or digital I/O. This selection is required even for the digital resources selected in the XBRn registers, and is not automatic.

The 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.

SFR Definition 26.7. P0: Port 0

Bit	7	6	5	4	3	2	1	0
Name	P0[7:0]							
Type	R/W							
Reset	1	1	1	1	1	1	1	1

SFR Address = 0x80; SFR Page = All Pages; Bit Addressable

Bit	Name	Description	Write	Read
7:0	P0[7:0]	**Port 0 Data.** Sets the Port latch logic value or reads the Port pin logic state in Port cells configured for digital I/O.	0: Set output latch to logic LOW. 1: Set output latch to logic HIGH.	0: P0.n Port pin is logic LOW. 1: P0.n Port pin is logic HIGH.

SFR Definition 26.8. P0MDIN: Port 0 Input Mode

Bit	7	6	5	4	3	2	1	0
Name	P0MDIN[7:0]							
Type	R/W							
Reset	1	1	1	1	1	1	1	1

SFR Address = 0xF1; SFR Page = F

Bit	Name	Function
7:0	P0MDIN[7:0]	**Analog Configuration Bits for P0.7–P0.0 (respectively).** Port pins configured for analog mode have their weak pullup, digital driver, and digital receiver disabled. 0: Corresponding P0.n pin is configured for analog mode. 1: Corresponding P0.n pin is not configured for analog mode.

SFR Definition 26.9. P0MDOUT: Port 0 Output Mode

Bit	7	6	5	4	3	2	1	0
Name	P0MDOUT[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xA4; SFR Page = F

Bit	Name	Function
7:0	P0MDOUT[7:0]	**Output Configuration Bits for P0.7–P0.0 (respectively).** These bits are ignored if the corresponding bit in register P0MDIN is logic 0. 0: Corresponding P0.n Output is open-drain. 1: Corresponding P0.n Output is push-pull.

SFR Definition 26.10. P0SKIP: Port 0 Skip

Bit	7	6	5	4	3	2	1	0
Name	P0SKIP[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xD4; SFR Page = F

Bit	Name	Function
7:0	P0SKIP[7:0]	**Port 0 Crossbar Skip Enable Bits.** These bits select Port 0 pins to be skipped by the Crossbar Decoder. Port pins used for analog, special functions or GPIO should be skipped by the Crossbar. 0: Corresponding P0.n pin is not skipped by the Crossbar. 1: Corresponding P0.n pin is skipped by the Crossbar.

SFR Definition 26.11. P0DRV: Port 0 Drive Strength

Bit	7	6	5	4	3	2	1	0
Name	P0DRV[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xF9; SFR Page = F

Bit	Name	Function
7:0	P0DRV[7:0]	**Drive Strength Configuration Bits for P0.7-P0.0 (respectively).** Configures digital I/O Port cells to high or low output drive strength. 0: Corresponding P0.n Output has low output drive strength. 1: Corresponding P0.n Output has high output drive strength.

SFR Definition 26.12. P1: Port 1

Bit	7	6	5	4	3	2	1	0
Name	P1[7:0]							
Type	R/W							
Reset	1	1	1	1	1	1	1	1

SFR Address = 0x90; SFR Page = All Pages; Bit Addressable

Bit	Name	Description	Write	Read
7:0	P1[7:0]	**Port 1 Data.** Sets the Port latch logic value or reads the Port pin logic state in Port cells configured for digital I/O.	0: Set output latch to logic LOW. 1: Set output latch to logic HIGH.	0: P1.n Port pin is logic LOW. 1: P1.n Port pin is logic HIGH.

SFR Definition 26.13. P1MDIN: Port 1 Input Mode

Bit	7	6	5	4	3	2	1	0
Name	P1MDIN[7:0]							
Type	R/W							
Reset	1	1	1	1	1	1	1	1

SFR Address = 0xF2; SFR Page = F

Bit	Name	Function
7:0	P1MDIN[7:0]	**Analog Configuration Bits for P1.7–P1.0 (respectively).** Port pins configured for analog mode have their weak pullup, digital driver, and digital receiver disabled. 0: Corresponding P1.n pin is configured for analog mode. 1: Corresponding P1.n pin is not configured for analog mode.

SFR Definition 26.14. P1MDOUT: Port 1 Output Mode

Bit	7	6	5	4	3	2	1	0
Name	P1MDOUT[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xA5; SFR Page = F

Bit	Name	Function
7:0	P1MDOUT[7:0]	**Output Configuration Bits for P1.7–P1.0 (respectively).** These bits are ignored if the corresponding bit in register P1MDIN is logic 0. 0: Corresponding P1.n Output is open-drain. 1: Corresponding P1.n Output is push-pull.

SFR Definition 26.15. P1SKIP: Port 1 Skip

Bit	7	6	5	4	3	2	1	0
Name	P1SKIP[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xD5; SFR Page = F

Bit	Name	Function
7:0	P1SKIP[7:0]	**Port 1 Crossbar Skip Enable Bits.** These bits select Port 1 pins to be skipped by the Crossbar Decoder. Port pins used for analog, special functions or GPIO should be skipped by the Crossbar. 0: Corresponding P1.n pin is not skipped by the Crossbar. 1: Corresponding P1.n pin is skipped by the Crossbar.

SFR Definition 26.16. P1DRV: Port 1 Drive Strength

Bit	7	6	5	4	3	2	1	0
Name	P1DRV[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xFA; SFR Page = F

Bit	Name	Function
7:0	P1DRV[7:0]	**Drive Strength Configuration Bits for P1.7–P1.0 (respectively).** Configures digital I/O Port cells to high or low output drive strength. 0: Corresponding P1.n Output has low output drive strength. 1: Corresponding P1.n Output has high output drive strength.

SFR Definition 26.17. P2: Port 2

Bit	7	6	5	4	3	2	1	0
Name	P2[7:0]							
Type	R/W							
Reset	1	1	1	1	1	1	1	1

SFR Address = 0xA0; SFR Page = All Pages; Bit Addressable

Bit	Name	Description	Write	Read
7:0	P2[7:0]	**Port 2 Data.** Sets the Port latch logic value or reads the Port pin logic state in Port cells configured for digital I/O.	0: Set output latch to logic LOW. 1: Set output latch to logic HIGH.	0: P2.n Port pin is logic LOW. 1: P2.n Port pin is logic HIGH.

SFR Definition 26.18. P2MDIN: Port 2 Input Mode

Bit	7	6	5	4	3	2	1	0
Name	P2MDIN[7:0]							
Type	R/W							
Reset	1	1	1	1	1	1	1	1

SFR Address = 0xF3; SFR Page = F

Bit	Name	Function
7:0	P2MDIN[7:0]	**Analog Configuration Bits for P2.7–P2.0 (respectively).** Port pins configured for analog mode have their weak pullup, digital driver, and digital receiver disabled. 0: Corresponding P2.n pin is configured for analog mode. 1: Corresponding P2.n pin is not configured for analog mode.

SFR Definition 26.19. P2MDOUT: Port 2 Output Mode

Bit	7	6	5	4	3	2	1	0
Name	P2MDOUT[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xA6; SFR Page = F

Bit	Name	Function
7:0	P2MDOUT[7:0]	**Output Configuration Bits for P2.7–P2.0 (respectively).** These bits are ignored if the corresponding bit in register P2MDIN is logic 0. 0: Corresponding P2.n Output is open-drain. 1: Corresponding P2.n Output is push-pull.

SFR Definition 26.20. P2SKIP: Port 2 Skip

Bit	7	6	5	4	3	2	1	0
Name	P2SKIP[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xD6; SFR Page = F

Bit	Name	Function
7:0	P2SKIP[3:0]	**Port 2 Crossbar Skip Enable Bits.** These bits select Port 2 pins to be skipped by the Crossbar Decoder. Port pins used for analog, special functions or GPIO should be skipped by the Crossbar. 0: Corresponding P2.n pin is not skipped by the Crossbar. 1: Corresponding P2.n pin is skipped by the Crossbar.

SFR Definition 26.21. P2DRV: Port 2 Drive Strength

Bit	7	6	5	4	3	2	1	0
Name	P2DRV[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xFB; SFR Page = F

Bit	Name	Function
7:0	P2DRV[7:0]	**Drive Strength Configuration Bits for P2.7-P2.0 (respectively).** Configures digital I/O Port cells to high or low output drive strength. 0: Corresponding P2.n Output has low output drive strength. 1: Corresponding P2.n Output has high output drive strength.

SFR Definition 26.22. P3: Port 3

Bit	7	6	5	4	3	2	1	0
Name	P3[7:0]							
Type	R/W							
Reset	1	1	1	1	1	1	1	1

SFR Address = 0xB0; SFR Page = All Pages; Bit Addressable

Bit	Name	Description	Write	Read
7:0	P3[7:0]	**Port 3 Data.** Sets the Port latch logic value or reads the Port pin logic state in Port cells configured for digital I/O.	0: Set output latch to logic LOW. 1: Set output latch to logic HIGH.	0: P3.n Port pin is logic LOW. 1: P3.n Port pin is logic HIGH.

SFR Definition 26.23. P3MDIN: Port 3 Input Mode

Bit	7	6	5	4	3	2	1	0
Name	P3MDIN[7:0]							
Type	R/W							
Reset	1	1	1	1	1	1	1	1

SFR Address = 0xF4; SFR Page = F

Bit	Name	Function
7:0	P3MDIN[7:0]	**Analog Configuration Bits for P3.7–P3.0 (respectively).** Port pins configured for analog mode have their weak pullup, digital driver, and digital receiver disabled. 0: Corresponding P3.n pin is configured for analog mode. 1: Corresponding P3.n pin is not configured for analog mode.

SFR Definition 26.24. P3MDOUT: Port 3 Output Mode

Bit	7	6	5	4	3	2	1	0
Name	P3MDOUT[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xAF; SFR Page = F

Bit	Name	Function
7:0	P3MDOUT[7:0]	**Output Configuration Bits for P3.7–P3.0 (respectively).** These bits are ignored if the corresponding bit in register P3MDIN is logic 0. 0: Corresponding P3.n Output is open-drain. 1: Corresponding P3.n Output is push-pull.

SFR Definition 26.25. P3DRV: Port 3 Drive Strength

Bit	7	6	5	4	3	2	1	0
Name	P3DRV[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xFC; SFR Page = F

Bit	Name	Function
7:0	P3DRV[7:0]	**Drive Strength Configuration Bits for P3.7-P3.0 (respectively).** Configures digital I/O Port cells to high or low output drive strength. 0: Corresponding P3.n Output has low output drive strength. 1: Corresponding P3.n Output has high output drive strength.

SFR Definition 26.26. P4: Port 4

Bit	7	6	5	4	3	2	1	0
Name	P4[7:0]							
Type	R/W							
Reset	1	1	1	1	1	1	1	1

SFR Address = 0xAC; SFR Page = All Pages

Bit	Name	Description	Write	Read
7:0	P4[7:0]	**Port 4 Data.** Sets the Port latch logic value or reads the Port pin logic state in Port cells configured for digital I/O.	0: Set output latch to logic LOW. 1: Set output latch to logic HIGH.	0: P4.n Port pin is logic LOW. 1: P4.n Port pin is logic HIGH.

SFR Definition 26.27. P4MDIN: Port 4 Input Mode

Bit	7	6	5	4	3	2	1	0
Name	P4MDIN[7:0]							
Type	R/W							
Reset	1	1	1	1	1	1	1	1

SFR Address = 0xF5; SFR Page = F

Bit	Name	Function
7:0	P4MDIN[7:0]	**Analog Configuration Bits for P4.7–P4.0 (respectively).** Port pins configured for analog mode have their weak pullup, digital driver, and digital receiver disabled. 0: Corresponding P4.n pin is configured for analog mode. 1: Corresponding P4.n pin is not configured for analog mode.

SFR Definition 26.28. P4MDOUT: Port 4 Output Mode

Bit	7	6	5	4	3	2	1	0
Name	P4MDOUT[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x9A; SFR Page = F

Bit	Name	Function
7:0	P4MDOUT[7:0]	**Output Configuration Bits for P4.7–P4.0 (respectively).** These bits are ignored if the corresponding bit in register P4MDIN is logic 0. 0: Corresponding P4.n Output is open-drain. 1: Corresponding P4.n Output is push-pull.

SFR Definition 26.29. P4DRV: Port 4 Drive Strength

Bit	7	6	5	4	3	2	1	0
Name	P4DRV[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xFD; SFR Page = F

Bit	Name	Function
7:0	P4DRV[7:0]	**Drive Strength Configuration Bits for P4.7-P4.0 (respectively).** Configures digital I/O Port cells to high or low output drive strength. 0: Corresponding P4.n Output has low output drive strength. 1: Corresponding P4.n Output has high output drive strength.

SFR Definition 26.30. P5: Port 5

Bit	7	6	5	4	3	2	1	0
Name	P5[7:0]							
Type	R/W							
Reset	1	1	1	1	1	1	1	1

SFR Address = 0xB3; SFR Page = All Pages

Bit	Name	Description	Write	Read
7:0	P5[7:0]	**Port 5 Data.** Sets the Port latch logic value or reads the Port pin logic state in Port cells configured for digital I/O.	0: Set output latch to logic LOW. 1: Set output latch to logic HIGH.	0: P5.n Port pin is logic LOW. 1: P5.n Port pin is logic HIGH.

SFR Definition 26.31. P5MDIN: Port 5 Input Mode

Bit	7	6	5	4	3	2	1	0
Name	P5MDIN[7:0]							
Type	R/W							
Reset	1	1	1	1	1	1	1	1

SFR Address = 0xF6; SFR Page = F

Bit	Name	Function
7:0	P5MDIN[7:0]	**Analog Configuration Bits for P5.7–P5.0 (respectively).** Port pins configured for analog mode have their weak pullup, digital driver, and digital receiver disabled. 0: Corresponding P5.n pin is configured for analog mode. 1: Corresponding P5.n pin is not configured for analog mode.

SFR Definition 26.32. P5MDOUT: Port 5 Output Mode

Bit	7	6	5	4	3	2	1	0
Name	P5MDOUT[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x9B; SFR Page = F

Bit	Name	Function
7:0	P5MDOUT[7:0]	**Output Configuration Bits for P5.7–P5.0 (respectively).** These bits are ignored if the corresponding bit in register P5MDIN is logic 0. 0: Corresponding P5.n Output is open-drain. 1: Corresponding P5.n Output is push-pull.

SFR Definition 26.33. P5DRV: Port 5 Drive Strength

Bit	7	6	5	4	3	2	1	0
Name	P5DRV[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xFE; SFR Page = F

Bit	Name	Function
7:0	P5DRV[7:0]	**Drive Strength Configuration Bits for P5.7-P5.0 (respectively).** Configures digital I/O Port cells to high or low output drive strength. 0: Corresponding P5.n Output has low output drive strength. 1: Corresponding P5.n Output has high output drive strength.

SFR Definition 26.34. P6: Port 6

Bit	7	6	5	4	3	2	1	0
Name			P6[5:0]					
Type	R	R	R/W					
Reset	0	0	1	1	1	1	1	1

SFR Address = 0xB2; SFR Page = All Pages

Bit	Name	Description	Write	Read
7:6	Unused	Read = 00b; Write = Don’t Care		
5:0	P6[5:0]	**Port 6 Data.** Sets the Port latch logic value or reads the Port pin logic state in Port cells configured for digital I/O.	0: Set output latch to logic LOW. 1: Set output latch to logic HIGH.	0: P6.n Port pin is logic LOW. 1: P6.n Port pin is logic HIGH.

SFR Definition 26.35. P6MDOUT: Port 6 Output Mode

Bit	7	6	5	4	3	2	1	0
Name			P6MDOUT[5:0]					
Type	R	R	R/W					
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x9C; SFR Page = F

Bit	Name	Function
7:6	Unused	Read = 00b; Write = Don't Care
5:0	P6MDOUT[5:0]	**Output Configuration Bits for P6.5–P6.0 (respectively).** These bits are ignored if the corresponding bit in register P6MDIN is logic 0. 0: Corresponding P6.n Output is open-drain. 1: Corresponding P6.n Output is push-pull.

SFR Definition 26.36. P6DRV: Port 6 Drive Strength

Bit	7	6	5	4	3	2	1	0
Name			P6DRV[5:0]					
Type			R/W					
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xC1; SFR Page = F

Bit	Name	Function
7:6	Unused	Read = 00b; Write = Don't Care
5:0	P6DRV[5:0]	**Drive Strength Configuration Bits for P6.5-P6.0 (respectively).** Configures digital I/O Port cells to high or low output drive strength. 0: Corresponding P6.n Output has low output drive strength. 1: Corresponding P6.n Output has high output drive strength.

27. Cyclic Redundancy Check Unit (CRC0)

C8051F70x/71x devices include a cyclic redundancy check unit (CRC0) that can perform a CRC using a 16-bit or 32-bit polynomial. CRC0 accepts a stream of 8-bit data written to the CRC0IN register. CRC0 posts the 16-bit or 32-bit result to an internal register. The internal result register may be accessed indirectly using the CRC0PNT bits and CRC0DAT register, as shown in Figure 27.1. CRC0 also has a bit reverse register for quick data manipulation.

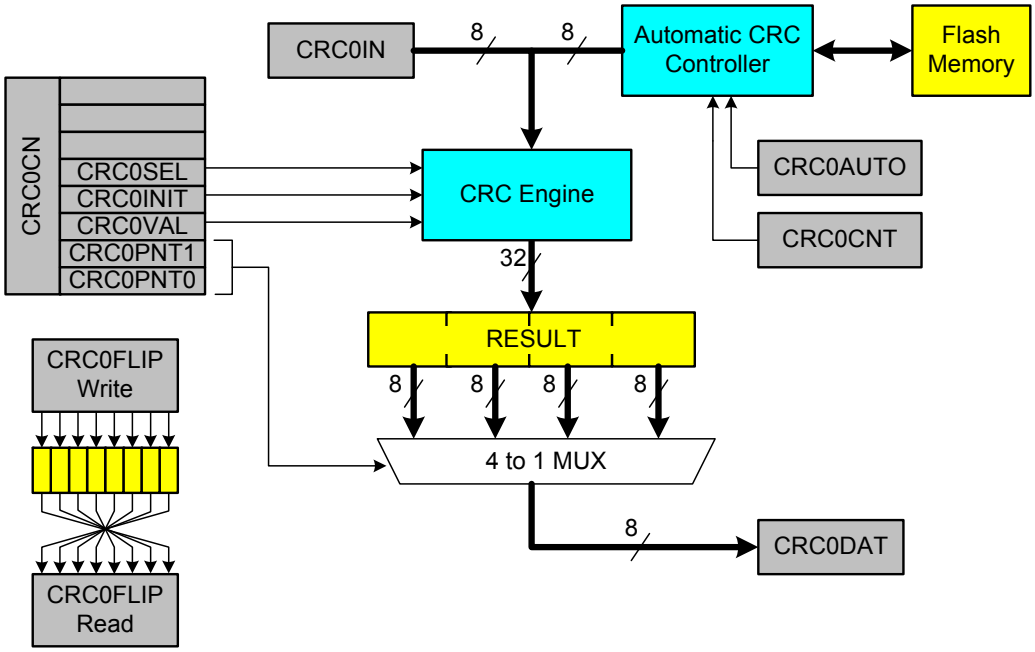


Figure 27.1. CRC0 Block Diagram

27.1. 16-bit CRC Algorithm

The C8051F70x/71x CRC unit calculates the 16-bit CRC MSB-first, using a poly of 0x1021. The following describes the 16-bit CRC algorithm performed by the hardware:

1. XOR the most-significant byte of the current CRC result with the input byte. If this is the first iteration of the CRC unit, the current CRC result will be the set initial value (0x0000 or 0xFFFF).
2. If the MSB of the CRC result is set, left-shift the CRC result, and then XOR the CRC result with the polynomial (0x1021).
3. If the MSB of the CRC result is not set, left-shift the CRC result.
4. Repeat at Step 2 for the number of input bits (8).

For example, the 16-bit C8051F70x/71x 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 CRC_acc; // Return the final remainder (CRC value)
}
```

Table 27.1 lists example input values and the associated outputs using the 16-bit C8051F70x/71x CRC algorithm (an initial value of 0xFFFF is used):

Table 27.1. Example 16-bit CRC Outputs

Input	Output
0x63	0xBD35
0xAA, 0xBB, 0xCC	0x6CF6
0x00, 0x00, 0xAA, 0xBB, 0xCC	0xB166

27.2. 32-bit CRC Algorithm

The C8051F70x/71x CRC unit calculates the 32-bit CRC using a poly of 0x04C11DB7. The CRC-32 algorithm is "reflected", meaning that all of the input bytes and the final 32-bit output are bit-reversed in the processing engine. The following is a description of a simplified CRC algorithm that produces results identical to the hardware:

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

For example, the 32-bit C8051F70x/71x CRC algorithm can be described by the following code:

```
unsigned long UpdateCRC (unsigned long CRC_acc, unsigned char CRC_input){
   unsigned char i; // loop counter
   #define POLY 0xEDB88320 // bit-reversed version of the poly 0x04C11DB7
   // Create the CRC "dividend" for polynomial arithmetic (binary arithmetic
   // with no carries)
   CRC_acc = CRC_acc ^ CRC_input;
   // "Divide" the poly into the dividend using CRC XOR subtraction
   // CRC_acc holds the "remainder" of each divide
   // Only complete this division for 8 bits since input is 1 byte
   for (i = 0; i < 8; i++)
   {
      // Check if the MSB is set (if MSB is 1, then the POLY can "divide"
      // into the "dividend")
      if ((CRC_acc & 0x00000001) == 0x00000001)
      {
         // if so, shift the CRC value, and XOR "subtract" the poly
         CRC_acc = CRC_acc >> 1;
         CRC_acc ^= POLY;
      }
      else
      {
         // if not, just shift the CRC value
         CRC_acc = CRC_acc >> 1;
      }
   }
   return CRC_acc; // Return the final remainder (CRC value)
}
```

Table 27.2 lists example input values and the associated outputs using the 32-bit C8051F70x/71x CRC algorithm (an initial value of 0xFFFFFFFF is used):

Table 27.2. Example 32-bit CRC Outputs

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

27.3. Preparing for a CRC Calculation

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

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

27.4. Performing a CRC Calculation

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

1. Prepare CRC0 for a CRC calculation as shown above.
2. Write the index of the starting page to CRC0AUTO.
3. Set the AUTOEN bit in CRC0AUTO.
4. Write the number of Flash sectors to perform in the CRC calculation to CRC0CNT.

Note: Each Flash sector is 512 bytes.

5. Write any value to CRC0CN (or OR its contents with 0x00) to initiate the CRC calculation. The CPU will not execute code any additional code until the CRC operation completes.
6. Clear the AUTOEN bit in CRC0AUTO.
7. Read the CRC result using the procedure below.

27.5. Accessing the CRC0 Result

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

SFR Definition 27.1. CRC0CN: CRC0 Control

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

SFR Address = 0x91; SFR Page = F

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

SFR Definition 27.2. CRC0IN: CRC Data Input

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

SFR Address = 0x94; SFR Page = F

Bit	Name	Function
7:0	CRC0IN[7:0]	**CRC0 Data Input.** Each write to CRC0IN results in the written data being computed into the existing CRC result according to the CRC algorithm described in Section 27.1

SFR Definition 27.3. CRC0DATA: CRC Data Output

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

SFR Address = 0xD9; SFR Page = F

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

SFR Definition 27.4. CRC0AUTO: CRC Automatic Control

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

SFR Address = 0x96; SFR Page = F

Bit	Name	Function
7	AUTOEN	**Automatic CRC Calculation Enable.** When AUTOEN is set to 1, any write to CRC0CN will initiate an automatic CRC starting at Flash sector CRC0ST and continuing for CRC0CNT sectors.
6	CRCCPT	**Automatic CRC Calculation Complete.** Set to 0 when a CRC calculation is in progress. Code execution is stopped during a CRC calculation, therefore reads from firmware will always return 1.
5	Reserved	**Reserved. Must write 0.**
4:0	CRC0ST[4:0]	**Automatic CRC Calculation Starting Flash Sector.** These bits specify the Flash sector to start the automatic CRC calculation. The starting address of the first Flash sector included in the automatic CRC calculation is CRC0ST x 512.

SFR Definition 27.5. CRC0CNT: CRC Automatic Flash Sector Count

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

SFR Address = 0x97; SFR Page = F

Bit	Name	Function
7:6	Unused	Read = 00b; Write = Don’t Care.
5:0	CRC0CNT[5:0]	**Automatic CRC Calculation Flash Sector Count.** These bits specify the number of Flash sectors to include when performing an automatic CRC calculation. The base address of the last flash sector included in the automatic CRC calculation is equal to (CRC0ST + CRC0CNT) x 512.

27.6. CRC0 Bit Reverse Feature

CRC0 includes hardware to reverse the bit order of each bit in a byte as shown in Figure 27.1. 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.

SFR Definition 27.6. CRC0FLIP: CRC Bit Flip

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

SFR Address = 0x95; SFR Page = F

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

28. SMBus

The SMBus I/O interface is a two-wire, bi-directional serial bus. The SMBus is compliant with the System Management Bus Specification, version 1.1, and compatible with the I^2C serial bus. Reads and writes to the interface by the system controller are byte oriented with the SMBus interface autonomously controlling the serial transfer of the data. Data can be transferred at up to 1/20th of the system clock as a master or slave (this can be faster than allowed by the SMBus specification, depending on the system clock used). A method of extending the clock-low duration is available to accommodate devices with different speed capabilities on the same bus.

The SMBus interface may operate as a master and/or slave, and may function on a bus with multiple masters. The SMBus provides control of SDA (serial data), SCL (serial clock) generation and synchronization, arbitration logic, and START/STOP control and generation. The SMBus peripheral can be fully driven by software (i.e., software accepts/rejects slave addresses, and generates ACKs), or hardware slave address recognition and automatic ACK generation can be enabled to minimize software overhead. A block diagram of the SMBus peripheral and the associated SFRs is shown in Figure 28.1.

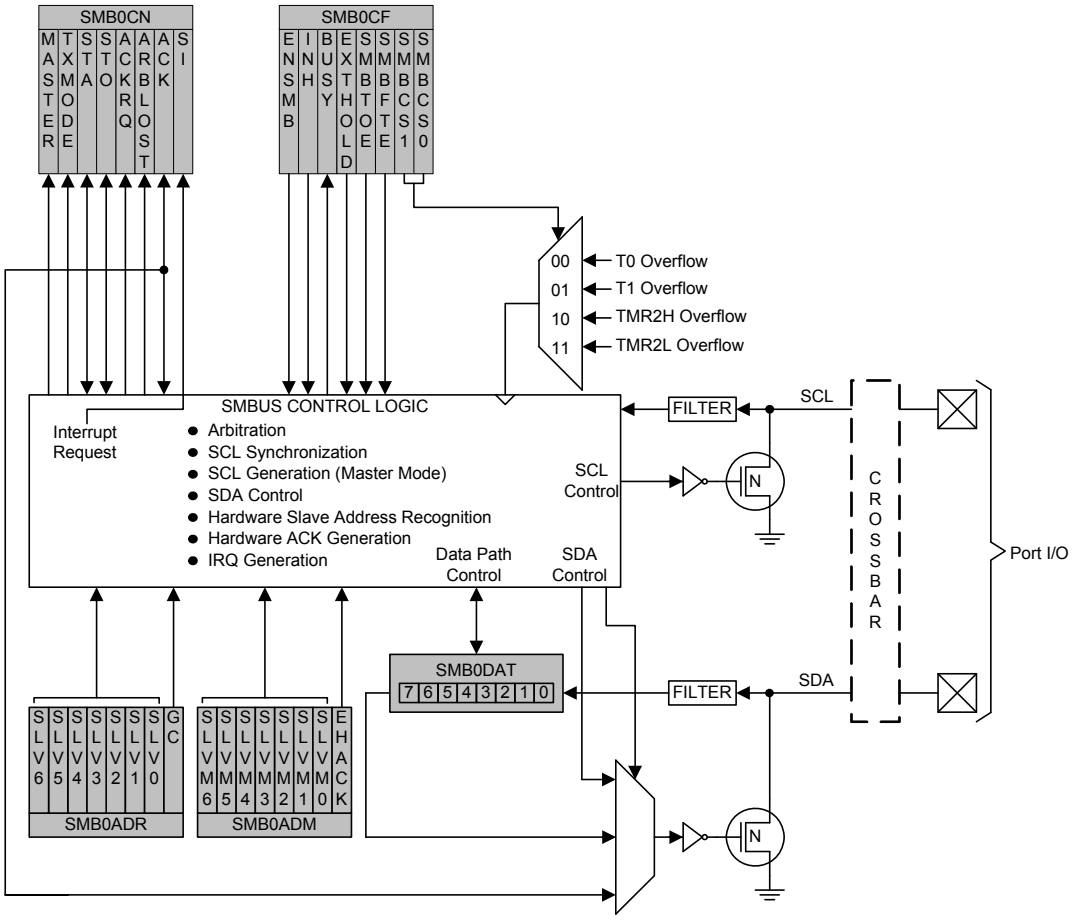


Figure 28.1. SMBus Block Diagram

28.1. 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.

28.2. SMBus Configuration

Figure 28.2 shows a typical SMBus configuration. The SMBus specification allows any recessive voltage between 3.0 V and 5.0 V; different devices on the bus may operate at different voltage levels. The bi-directional SCL (serial clock) and SDA (serial data) lines must be connected to a positive power supply voltage through a pullup resistor or similar circuit. Every device connected to the bus must have an open-drain or open-collector output for both the SCL and SDA lines, so that both are pulled high (recessive state) when the bus is free. The maximum number of devices on the bus is limited only by the requirement that the rise and fall times on the bus not exceed 300 ns and 1000 ns, respectively.

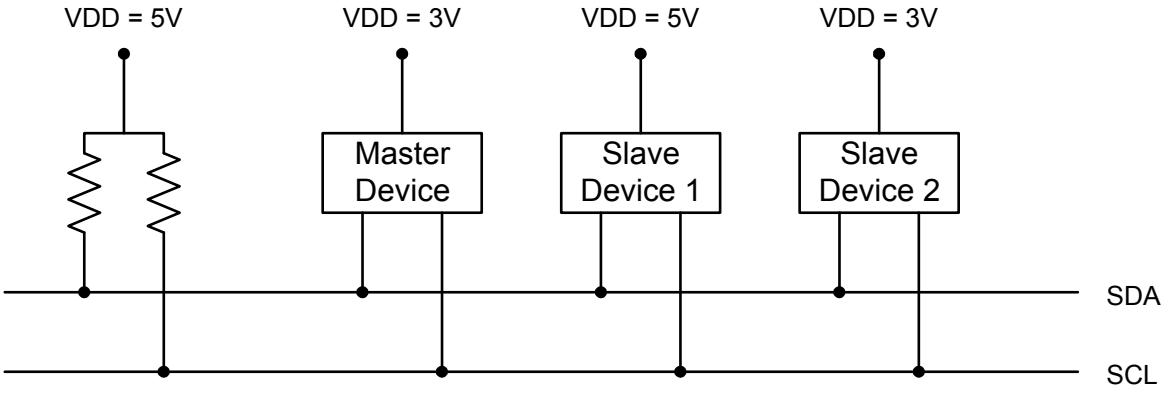


Figure 28.2. Typical SMBus Configuration

28.3. 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 28.3). 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 28.3 illustrates a typical SMBus transaction.

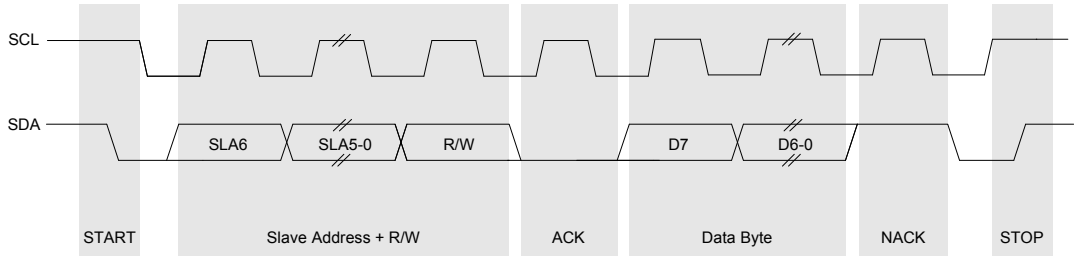


Figure 28.3. SMBus Transaction

28.3.1. 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.

28.3.2. 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 “28.3.5. SCL High (SMBus Free) Timeout” on page 204). 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.

28.3.3. 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.

28.3.4. SCL Low Timeout

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

When the SMBTOE bit in SMB0CF is set, Timer 3 is used to detect SCL low timeouts. Timer 3 is forced to reload when SCL is high, and allowed to count when SCL is low. With Timer 3 enabled and configured to overflow after 25 ms (and SMBTOE set), the Timer 3 interrupt service routine can be used to reset (disable and re-enable) the SMBus in the event of an SCL low timeout.

28.3.5. SCL High (SMBus Free) Timeout

The SMBus specification stipulates that if the SCL and SDA lines remain high for more that 50 µs, the bus is designated as free. When the SMBFTE bit in SMB0CF is set, the bus will be considered free if SCL and SDA remain high for more than 10 SMBus clock source periods (as defined by the timer configured for the SMBus clock source). If the SMBus is waiting to generate a Master START, the START will be generated following this timeout. A clock source is required for free timeout detection, even in a slave-only implementation.

28.4. 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 28.5 for more details on transmission sequences.

Interrupts are also generated to indicate the beginning of a transfer when a master (START generated), or the end of a transfer when a slave (STOP detected). Software should read the SMB0CN (SMBus Control register) to find the cause of the SMBus interrupt. The SMB0CN register is described in Section 28.4.2; Table 28.5 provides a quick SMB0CN decoding reference.

28.4.1. 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 28.1. SMBus Clock Source Selection

SMBCS1	SMBCS0	SMBus Clock Source
0	0	Timer 0 Overflow
0	1	Timer 1 Overflow
1	0	Timer 2 High Byte Overflow
1	1	Timer 2 Low Byte Overflow

The SMBCS1–0 bits select the SMBus clock source, which is used only when operating as a master or when the Free Timeout detection is enabled. When operating as a master, overflows from the selected source determine the absolute minimum SCL low and high times as defined in Equation 28.1.The selected clock source may be shared by other peripherals so long as the timer is left running at all times. For example, Timer 1 overflows may generate the SMBus and UART baud rates simultaneously. Timer configuration is covered in Section “31. Timers” on page 244.

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

Equation 28.1. 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 28.1. 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 28.2.

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

Equation 28.2. Typical SMBus Bit Rate

Figure 28.4 shows the typical SCL generation described by Equation 28.2. 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 28.1.

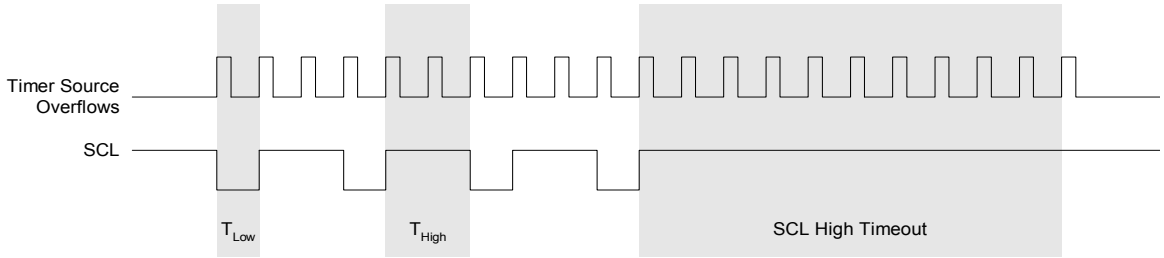


Figure 28.4. Typical SMBus SCL Generation

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 28.2 shows the minimum setup and hold times for the two EXTHOLD settings. Setup and hold time extensions are typically necessary when SYSCLK is above 10 MHz.

Table 28.2. Minimum SDA Setup and Hold Times

EXTHOLD	Minimum SDA Setup Time	Minimum SDA Hold Time
0	T_{low} – 4 system clocks or 1 system clock + s/w delay*	3 system clocks
1	11 system clocks	12 system clocks
Note: Setup Time for ACK bit transmissions and the MSB of all data transfers. When using software acknowledgement, the s/w delay occurs between the time SMB0DAT or ACK is written and when SI is cleared. Note that if SI is cleared in the same write that defines the outgoing ACK value, s/w delay is zero.		

With the SMBTOE bit set, Timer 3 should be configured to overflow after 25 ms in order to detect SCL low timeouts (see Section “28.3.4. SCL Low Timeout” on page 203). The SMBus interface will force Timer 3 to reload while SCL is high, and allow Timer 3 to count when SCL is low. The Timer 3 interrupt service routine should be used to reset SMBus communication by disabling and re-enabling the SMBus.

SMBus Free Timeout detection can be enabled by setting the SMBFTE bit. When this bit is set, the bus will be considered free if SDA and SCL remain high for more than 10 SMBus clock source periods (see Figure 28.4).

SFR Definition 28.1. SMB0CF: SMBus Clock/Configuration

Bit	7	6	5	4	3	2	1	0
Name	ENSMB	INH	BUSY	EXTHOLD	SMBTOE	SMBFTE	SMBCS[1:0]	
Type	R/W	R/W	R	R/W	R/W	R/W	R/W	
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xC1; SFR Page = 0

Bit	Name	Function
7	ENSMB	**SMBus Enable.** This bit enables the SMBus interface when set to 1. When enabled, the interface constantly monitors the SDA and SCL pins.
6	INH	**SMBus Slave Inhibit.** When this bit is set to logic 1, the SMBus does not generate an interrupt when slave events occur. This effectively removes the SMBus slave from the bus. Master Mode interrupts are not affected.
5	BUSY	**SMBus Busy Indicator.** This bit is set to logic 1 by hardware when a transfer is in progress. It is cleared to logic 0 when a STOP or free-timeout is sensed.
4	EXTHOLD	**SMBus Setup and Hold Time Extension Enable.** This bit controls the SDA setup and hold times according to Table 28.2. 0: SDA Extended Setup and Hold Times disabled. 1: SDA Extended Setup and Hold Times enabled.
3	SMBTOE	**SMBus SCL Timeout Detection Enable.** This bit enables SCL low timeout detection. If set to logic 1, the SMBus forces Timer 3 to reload while SCL is high and allows Timer 3 to count when SCL goes low. If Timer 3 is configured to Split Mode, only the High Byte of the timer is held in reload while SCL is high. Timer 3 should be programmed to generate interrupts at 25 ms, and the Timer 3 interrupt service routine should reset SMBus communication.
2	SMBFTE	**SMBus Free Timeout Detection Enable.** When this bit is set to logic 1, the bus will be considered free if SCL and SDA remain high for more than 10 SMBus clock source periods.
1:0	SMBCS[1:0]	**SMBus Clock Source Selection.** These two bits select the SMBus clock source, which is used to generate the SMBus bit rate. The selected device should be configured according to Equation 28.1. 00: Timer 0 Overflow 01: Timer 1 Overflow 10: Timer 2 High Byte Overflow 11: Timer 2 Low Byte Overflow

28.4.2. SMB0CN Control Register

SMB0CN is used to control the interface and to provide status information (see SFR Definition 28.2). The higher four bits of SMB0CN (MASTER, TXMODE, STA, and STO) form a status vector that can be used to jump to service routines. MASTER indicates whether a device is the master or slave during the current transfer. TXMODE indicates whether the device is transmitting or receiving data for the current byte.

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

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

The SI bit (SMBus Interrupt Flag) is set at the beginning and end of each transfer, after each byte frame, or when an arbitration is lost; see Table 28.3 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.

28.4.2.1. Software ACK Generation

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

28.4.2.2. Hardware ACK Generation

When the EHACK bit in register SMB0ADM is set to 1, automatic slave address recognition and ACK generation is enabled. More detail about automatic slave address recognition can be found in Section 28.4.3. As a receiver, the value currently specified by the ACK bit will be automatically sent on the bus during the ACK cycle of an incoming data byte. As a transmitter, reading the ACK bit indicates the value received on the last ACK cycle. The ACKRQ bit is not used when hardware ACK generation is enabled. If a received slave address is NACKed by hardware, further slave events will be ignored until the next START is detected, and no interrupt will be generated.

Table 28.3 lists all sources for hardware changes to the SMB0CN bits. Refer to Table 28.5 for SMBus status decoding using the SMB0CN register.

SFR Definition 28.2. SMB0CN: SMBus Control

Bit	7	6	5	4	3	2	1	0
Name	MASTER	TXMODE	STA	STO	ACKRQ	ARBLOST	ACK	SI
Type	R	R	R/W	R/W	R	R	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xC0; SFR Page = All Pages; Bit-Addressable

Bit	Name	Description	Read	Write
7	MASTER	**SMBus Master/Slave Indicator.** This read-only bit indicates when the SMBus is operating as a master.	0: SMBus operating in slave mode. 1: SMBus operating in master mode.	N/A
6	TXMODE	**SMBus Transmit Mode Indicator.** This read-only bit indicates when the SMBus is operating as a transmitter.	0: SMBus in Receiver Mode. 1: SMBus in Transmitter Mode.	N/A
5	STA	**SMBus Start Flag.**	0: No Start or repeated Start detected. 1: Start or repeated Start detected.	0: No Start generated. 1: When Configured as a Master, initiates a START or repeated START.
4	STO	**SMBus Stop Flag.**	0: No Stop condition detected. 1: Stop condition detected (if in Slave Mode) or pending (if in Master Mode).	0: No STOP condition is transmitted. 1: When configured as a Master, causes a STOP condition to be transmitted after the next ACK cycle. Cleared by Hardware.
3	ACKRQ	**SMBus Acknowledge Request.**	0: No Ack requested 1: ACK requested	N/A
2	ARBLOST	**SMBus Arbitration Lost Indicator.**	0: No arbitration error. 1: Arbitration Lost	N/A
1	ACK	**SMBus Acknowledge.**	0: NACK received. 1: ACK received.	0: Send NACK 1: Send ACK
0	SI	**SMBus Interrupt Flag.** This bit is set by hardware under the conditions listed in Table 15.3. SI must be cleared by software. While SI is set, SCL is held low and the SMBus is stalled.	0: No interrupt pending 1: Interrupt Pending	0: Clear interrupt, and initiate next state machine event. 1: Force interrupt.

Table 28.3. Sources for Hardware Changes to SMB0CN

Bit	Set by Hardware When:	Cleared by Hardware When:
MASTER	■ A START is generated.	■ A STOP is generated. ■ Arbitration is lost.
TXMODE	■ START is generated. ■ SMB0DAT is written before the start of an SMBus frame.	■ A START is detected. ■ Arbitration is lost. ■ SMB0DAT is not written before the start of an SMBus frame.
STA	■ A START followed by an address byte is received.	■ Must be cleared by software.
STO	■ A STOP is detected while addressed as a slave. ■ 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). ■ SCL is sensed low while attempting to generate a STOP or repeated START condition. ■ SDA is sensed low while transmitting a 1 (excluding ACK bits).	■ Each time SI is cleared.
ACK	■ The incoming ACK value is low (ACKNOWLEDGE).	■ The incoming ACK value is high (NOT ACKNOWLEDGE).
SI	■ A START has been generated. ■ Lost arbitration. ■ A byte has been transmitted and an ACK/NACK received. ■ A byte has been received. ■ A START or repeated START followed by a slave address + R/W has been received. ■ A STOP has been received.	■ Must be cleared by software.

28.4.3. Hardware Slave Address Recognition

The SMBus hardware has the capability to automatically recognize incoming slave addresses and send an ACK without software intervention. Automatic slave address recognition is enabled by setting the EHACK bit in register SMB0ADM to 1. This will enable both automatic slave address recognition and automatic hardware ACK generation for received bytes (as a master or slave). More detail on automatic hardware ACK generation can be found in Section 28.4.2.2.

The registers used to define which address(es) are recognized by the hardware are the SMBus Slave Address register (SFR Definition 28.3) and the SMBus Slave Address Mask register (SFR Definition 28.4). A single address or range of addresses (including the General Call Address 0x00) can be specified using these two registers. The most-significant seven bits of the two registers are used to define which addresses will be ACKed. A 1 in bit positions of the slave address mask SLVM[6:0] enable a comparison between the received slave address and the hardware's slave address SLV[6:0] for those bits. A 0 in a bit

of the slave address mask means that bit will be treated as a “don’t care” for comparison purposes. In this case, either a 1 or a 0 value are acceptable on the incoming slave address. Additionally, if the GC bit in register SMB0ADR is set to 1, hardware will recognize the General Call Address (0x00). Table 28.4 shows some example parameter settings and the slave addresses that will be recognized by hardware under those conditions.

Table 28.4. Hardware Address Recognition Examples (EHACK = 1)

Hardware Slave Address SLV[6:0]	Slave Address Mask SLVM[6:0]	GC bit	Slave Addresses Recognized by Hardware
0x34	0x7F	0	0x34
0x34	0x7F	1	0x34, 0x00 (General Call)
0x34	0x7E	0	0x34, 0x35
0x34	0x7E	1	0x34, 0x35, 0x00 (General Call)
0x70	0x73	0	0x70, 0x74, 0x78, 0x7C

SFR Definition 28.3. SMB0ADR: SMBus Slave Address

Bit	7	6	5	4	3	2	1	0
Name	SLV[6:0]							GC
Type	R/W							R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xBA; SFR Page = F

Bit	Name	Function
7:1	SLV[6:0]	**SMBus Hardware Slave Address.** Defines the SMBus Slave Address(es) for automatic hardware acknowledgement. Only address bits which have a 1 in the corresponding bit position in SLVM[6:0] are checked against the incoming address. This allows multiple addresses to be recognized.
0	GC	**General Call Address Enable.** When hardware address recognition is enabled (EHACK = 1), this bit will determine whether the General Call Address (0x00) is also recognized by hardware. 0: General Call Address is ignored. 1: General Call Address is recognized.

SFR Definition 28.4. SMB0ADM: SMBus Slave Address Mask

Bit	7	6	5	4	3	2	1	0
Name	SLVM[6:0]							EHACK
Type	R/W							R/W
Reset	1	1	1	1	1	1	1	0

SFR Address = 0xBB; SFR Page = F

Bit	Name	Function
7:1	SLVM[6:0]	**SMBus Slave Address Mask.** Defines which bits of register SMB0ADR are compared with an incoming address byte, and which bits are ignored. Any bit set to 1 in SLVM[6:0] enables comparisons with the corresponding bit in SLV[6:0]. Bits set to 0 are ignored (can be either 0 or 1 in the incoming address).
0	EHACK	**Hardware Acknowledge Enable.** Enables hardware acknowledgement of slave address and received data bytes. 0: Firmware must manually acknowledge all incoming address and data bytes. 1: Automatic Slave Address Recognition and Hardware Acknowledge is Enabled.

28.4.4. Data Register

The SMBus Data register SMB0DAT holds a byte of serial data to be transmitted or one that has just been received. Software may safely read or write to the data register when the SI flag is set. Software should not attempt to access the SMB0DAT register when the SMBus is enabled and the SI flag is cleared to logic 0, as the interface may be in the process of shifting a byte of data into or out of the register.

Data in SMB0DAT is always shifted out MSB first. After a byte has been received, the first bit of received data is located at the MSB of SMB0DAT. While data is being shifted out, data on the bus is simultaneously being shifted in. SMB0DAT always contains the last data byte present on the bus. In the event of lost arbitration, the transition from master transmitter to slave receiver is made with the correct data or address in SMB0DAT.

SFR Definition 28.5. SMB0DAT: SMBus Data

Bit	7	6	5	4	3	2	1	0
Name	SMB0DAT[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xC2; SFR Page = 0

Bit	Name	Function
7:0	SMB0DAT[7:0]	**SMBus Data.** The SMB0DAT register contains a byte of data to be transmitted on the SMBus serial interface or a byte that has just been received on the SMBus serial interface. The CPU can read from or write to this register whenever the SI serial interrupt flag (SMB0CN.0) is set to logic 1. The serial data in the register remains stable as long as the SI flag is set. When the SI flag is not set, the system may be in the process of shifting data in/out and the CPU should not attempt to access this register.

28.5. 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.

28.5.1. 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 28.5 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.

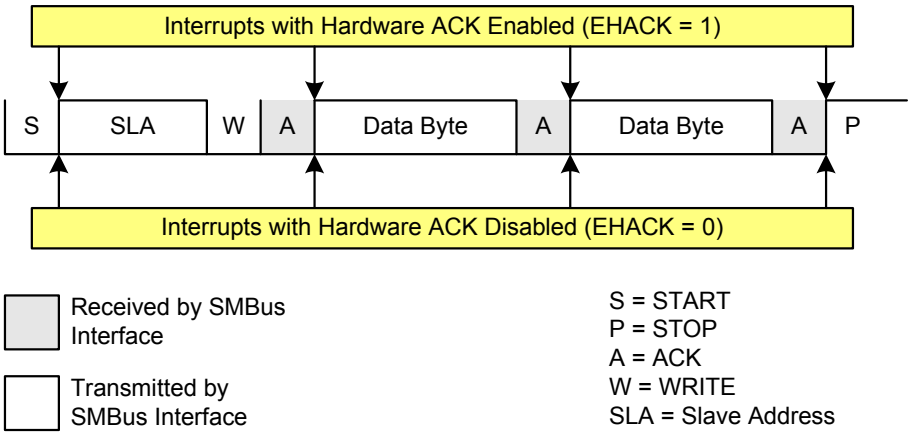


Figure 28.5. Typical Master Write Sequence

28.5.2. 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 28.6 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.

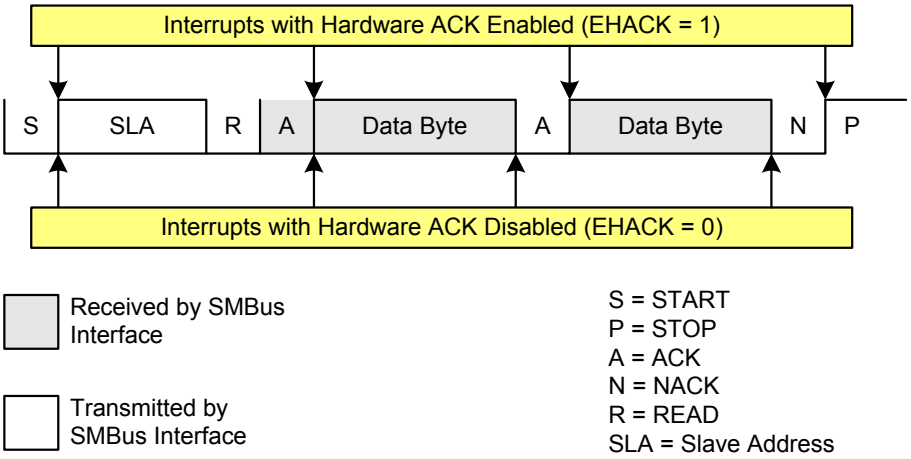


Figure 28.6. Typical Master Read Sequence

28.5.3. 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 28.7 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.

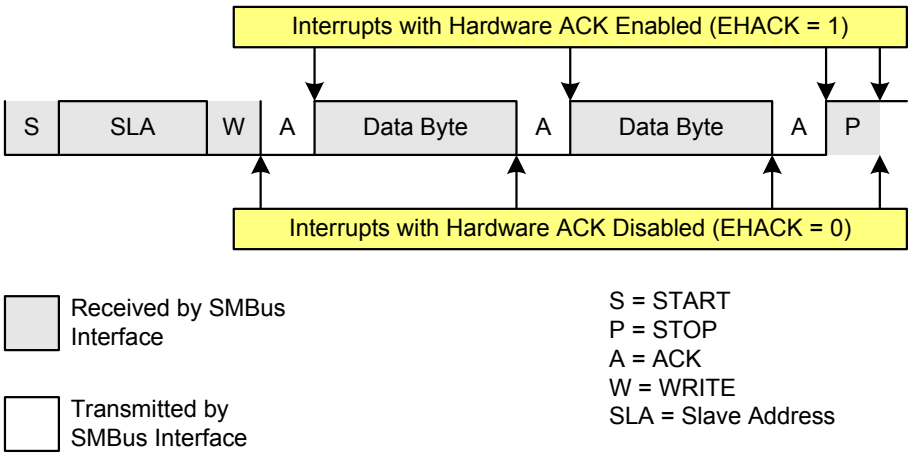


Figure 28.7. Typical Slave Write Sequence

28.5.4. 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 28.8 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.

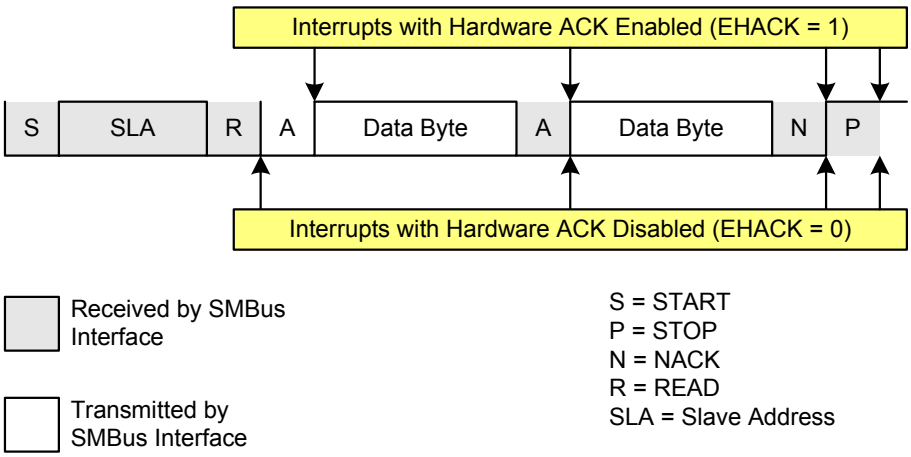


Figure 28.8. Typical Slave Read Sequence

28.6. 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 28.5 describes the typical actions when hardware slave address recognition and ACK generation is disabled. Table 28.6 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 28.5. SMBus Status Decoding With Hardware ACK Generation Disabled (EHACK = 0)

Mode	Values Read				Current SMbus State	Typical Response Options	Values to Write			Next Status Vector Expected
	Status Vector	ACKRQ	ARBLOST	ACK			STA	STO	ACK	
Master Transmitter	1110	0	0	X	A master START was generated.	Load slave address + R/W into SMB0DAT.	0	0	X	1100
	1100	0	0	0	A master data or address byte was transmitted; NACK received.	Set STA to restart transfer.	1	0	X	1110
						Abort transfer.	0	1	X	—
		0	0	1	A master data or address byte was transmitted; ACK received.	Load next data byte into SMB0DAT.	0	0	X	1100
						End transfer with STOP.	0	1	X	—
						End transfer with STOP and start another transfer.	1	1	X	—
						Send repeated START.	1	0	X	1110
						Switch to Master Receiver Mode (clear SI without writing new data to SMB0DAT).	0	0	X	1000
Master Receiver	1000	1	0	X	A master data byte was received; ACK requested.	Acknowledge received byte; Read SMB0DAT.	0	0	1	1000
						Send NACK to indicate last byte, and send STOP.	0	1	0	—
						Send NACK to indicate last byte, and send STOP followed by START.	1	1	0	1110
						Send ACK followed by repeated START.	1	0	1	1110
						Send NACK to indicate last byte, and send repeated START.	1	0	0	1110
						Send ACK and switch to Master Transmitter Mode (write to SMB0DAT before clearing SI).	0	0	1	1100
						Send NACK and switch to Master Transmitter Mode (write to SMB0DAT before clearing SI).	0	0	0	1100

Table 28.5. SMBus Status Decoding With Hardware ACK Generation Disabled (EHACK = 0) (Continued)

Mode	Values Read				Current SMbus State	Typical Response Options	Values to Write			Next Status Vector Expected
	Status Vector	ACKRQ	ARBLOST	ACK			STA	STO	ACK	
Slave Transmitter	0100	0	0	0	A slave byte was transmitted; NACK received.	No action required (expecting STOP condition).	0	0	X	0001
		0	0	1	A slave byte was transmitted; ACK received.	Load SMB0DAT with next data byte to transmit.	0	0	X	0100
		0	1	X	A Slave byte was transmitted; error detected.	No action required (expecting Master to end transfer).	0	0	X	0001
	0101	0	X	X	An illegal STOP or bus error was detected while a Slave Transmission was in progress.	Clear STO.	0	0	X	—
Slave Receiver	0010	1	0	X	A slave address + R/W was received; ACK requested.	If Write, Acknowledge received address	0	0	1	0000
						If Read, Load SMB0DAT with data byte; ACK received address	0	0	1	0100
						NACK received address.	0	0	0	—
		1	1	X	Lost arbitration as master; slave address + R/W received; ACK requested.	If Write, Acknowledge received address	0	0	1	0000
						If Read, Load SMB0DAT with data byte; ACK received address	0	0	1	0100
						NACK received address.	0	0	0	—
						Reschedule failed transfer; NACK received address.	1	0	0	1110
	0001	0	0	X	A STOP was detected while addressed as a Slave Transmitter or Slave Receiver.	Clear STO.	0	0	X	—
		1	1	X	Lost arbitration while attempting a STOP.	No action required (transfer complete/aborted).	0	0	0	—
	0000	1	0	X	A slave byte was received; ACK requested.	Acknowledge received byte; Read SMB0DAT.	0	0	1	0000
						NACK received byte.	0	0	0	—

Table 28.5. SMBus Status Decoding With Hardware ACK Generation Disabled (EHACK = 0) (Continued)

Mode	Values Read				Current SMbus State	Typical Response Options	Values to Write			Next Status Vector Expected
	Status Vector	ACKRQ	ARBLOST	ACK			STA	STO	ACK	
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 28.6. SMBus Status Decoding With Hardware ACK Generation Enabled (EHACK = 1)

Mode	Values Read				Current SMbus State	Typical Response Options	Values to Write			Next Status Vector Expected
	Status Vector	ACKRQ	ARBLOST	ACK			STA	STO	ACK	
Master Transmitter	1110	0	0	X	A master START was generated.	Load slave address + R/W into SMB0DAT.	0	0	X	1100
	1100	0	0	0	A master data or address byte was transmitted; NACK received.	Set STA to restart transfer.	1	0	X	1110
						Abort transfer.	0	1	X	—
		0	0	1	A master data or address byte was transmitted; ACK received.	Load next data byte into SMB0DAT.	0	0	X	1100
						End transfer with STOP.	0	1	X	—
						End transfer with STOP and start another transfer.	1	1	X	—
						Send repeated START.	1	0	X	1110
						Switch to Master Receiver Mode (clear SI without writing new data to SMB0DAT). Set ACK for initial data byte.	0	0	1	1000

Table 28.6. SMBus Status Decoding With Hardware ACK Generation Enabled (EHACK = 1) (Continued)

Mode	Values Read				Current SMbus State	Typical Response Options	Values to Write			Next Status Vector Expected
	Status Vector	ACKRQ	ARBLOST	ACK			STA	STO	ACK	
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 28.6. SMBus Status Decoding With Hardware ACK Generation Enabled (EHACK = 1) (Continued)

Mode	Values Read				Current SMbus State	Typical Response Options	Values to Write			Next Status Vector Expected
	Status Vector	ACKRQ	ARBLOST	ACK			STA	STO	ACK	
Slave 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

29. Enhanced Serial Peripheral Interface (SPI0)

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

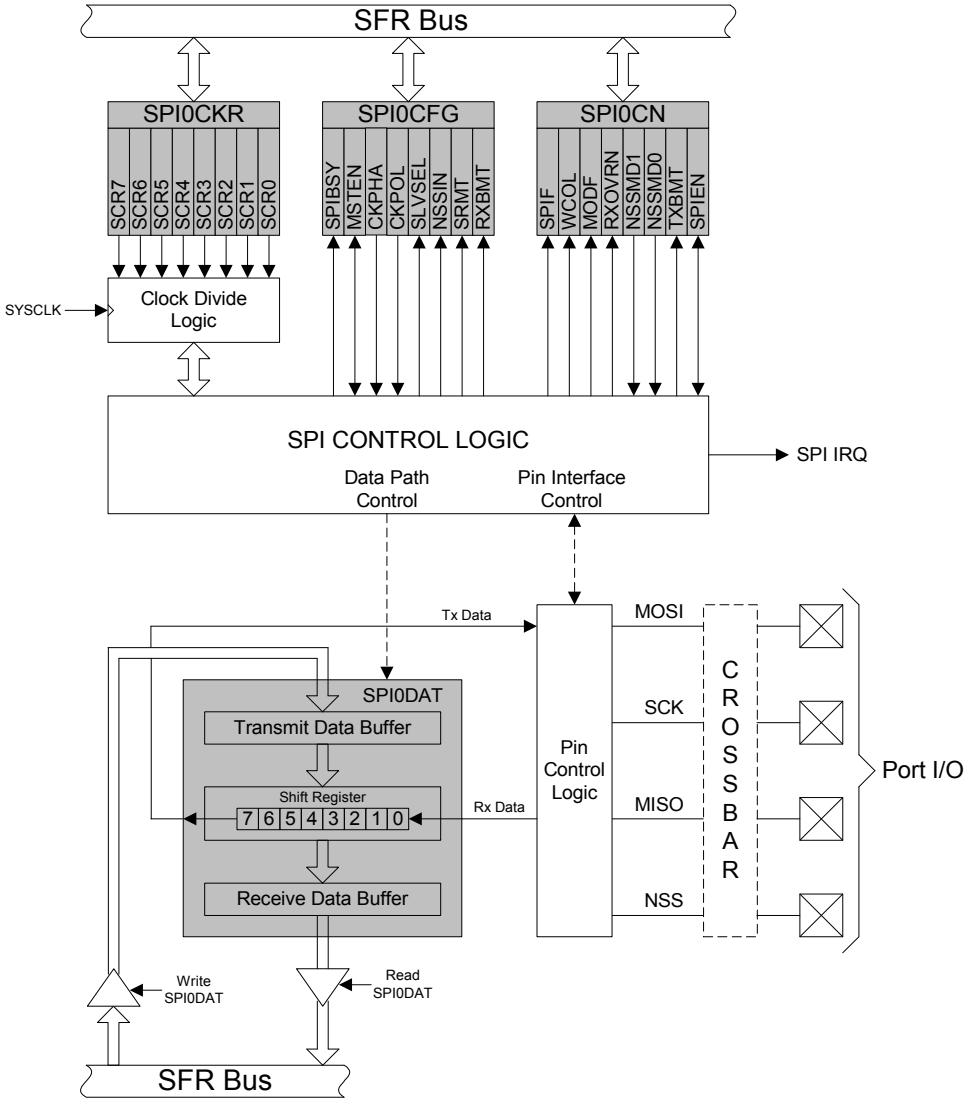


Figure 29.1. SPI Block Diagram

29.1. Signal Descriptions

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

29.1.1. 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.

29.1.2. 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.

29.1.3. 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.

29.1.4. 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 29.2, Figure 29.3, and Figure 29.4 for typical connection diagrams of the various operational modes. **Note that the setting of NSSMD bits affects the pinout of the device.** When in 3-wire master or 3-wire slave mode, the NSS pin will not be mapped by the crossbar. In all other modes, the NSS signal will be mapped to a pin on the device. See Section “26. Port Input/Output” on page 165 for general purpose port I/O and crossbar information.

29.2. 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 29.2 shows a connection diagram between two master devices in multiple-master mode.

3-wire single-master mode is active when NSSMD1 (SPI0CN.3) = 0 and NSSMD0 (SPI0CN.2) = 0. In this mode, NSS is not used, and is not mapped to an external port pin through the crossbar. Any slave devices that must be addressed in this mode should be selected using general-purpose I/O pins. Figure 29.3 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 29.4 shows a connection diagram for a master device in 4-wire master mode and two slave devices.

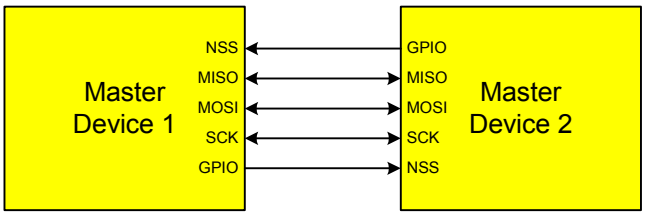


Figure 29.2. Multiple-Master Mode Connection Diagram

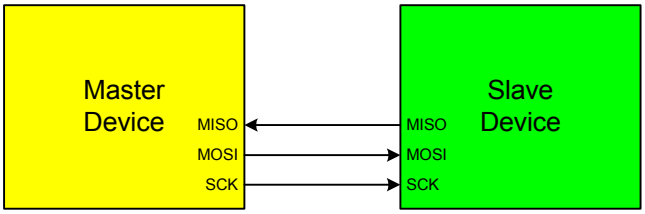


Figure 29.3. 3-Wire Single Master and 3-Wire Single Slave Mode Connection Diagram

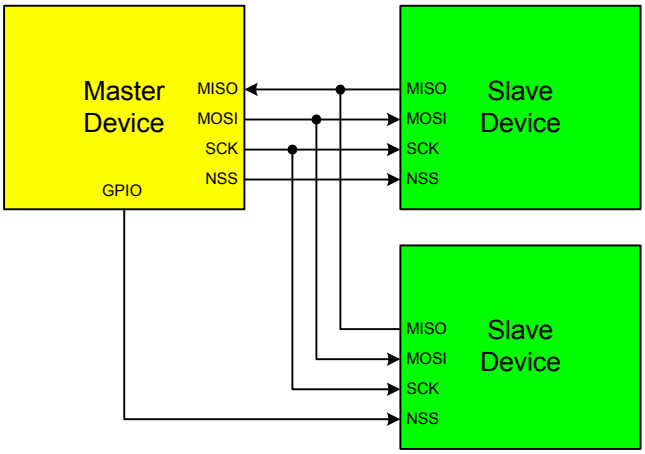


Figure 29.4. 4-Wire Single Master Mode and 4-Wire Slave Mode Connection Diagram

29.3. 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. The NSS signal must be driven low at least 2 system clocks before the first active edge of SCK for each byte transfer. Figure 29.4 shows a connection diagram between two slave devices in 4-wire slave mode and a master device.

3-wire slave mode is active when NSSMD1 (SPI0CN.3) = 0 and NSSMD0 (SPI0CN.2) = 0. NSS is not used in this mode, and is not mapped to an external port pin through the crossbar. Since there is no way of uniquely addressing the device in 3-wire slave mode, SPI0 must be the only slave device present on the bus. It is important to note that in 3-wire slave mode there is no external means of resetting the bit counter that determines when a full byte has been received. The bit counter can only be reset by disabling and re-enabling SPI0 with the SPIEN bit. Figure 29.3 shows a connection diagram between a slave device in 3-wire slave mode and a master device.

29.4. 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.

29.5. 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 29.5. For slave mode, the clock and data relationships are shown in Figure 29.6 and Figure 29.7. CKPHA should be set to 0 on both the master and slave SPI when communicating between two Silicon Labs C8051 devices.

The SPI0 Clock Rate Register (SPI0CKR) as shown in SFR Definition 29.3 controls the master mode serial clock frequency. This register is ignored when operating in slave mode. When the SPI is configured as a master, the maximum data transfer rate (bits/sec) is one-half the system clock frequency or 12.5 MHz, whichever is slower. When the SPI is configured as a slave, the maximum data transfer rate (bits/sec) for full-duplex operation is 1/10 the system clock frequency, provided that the master issues SCK, NSS (in 4-wire slave mode), and the serial input data synchronously with the slave's system clock. If the master issues SCK, NSS, and the serial input data asynchronously, the maximum data transfer rate (bits/sec) must be less than 1/10 the system clock frequency. In the special case where the master only wants to transmit data to the slave and does not need to receive data from the slave (i.e. half-duplex operation), the SPI slave can receive data at a maximum data transfer rate (bits/sec) of 1/4 the system clock frequency. This is provided that the master issues SCK, NSS, and the serial input data synchronously with the slave's system clock.

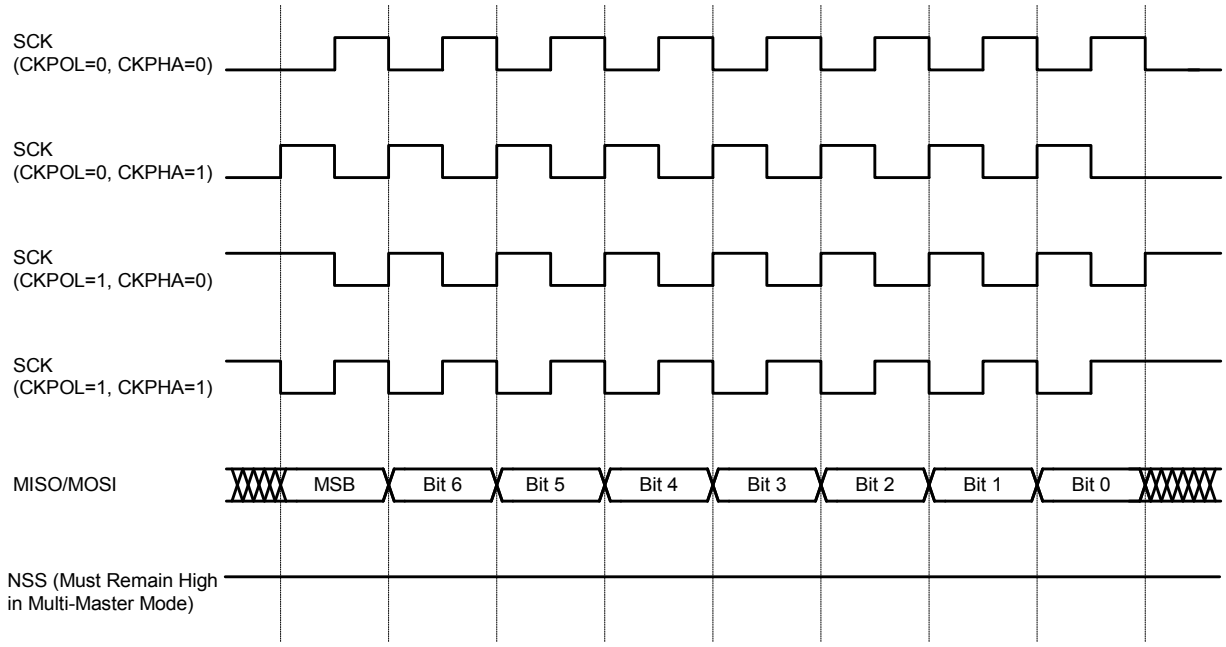


Figure 29.5. Master Mode Data/Clock Timing

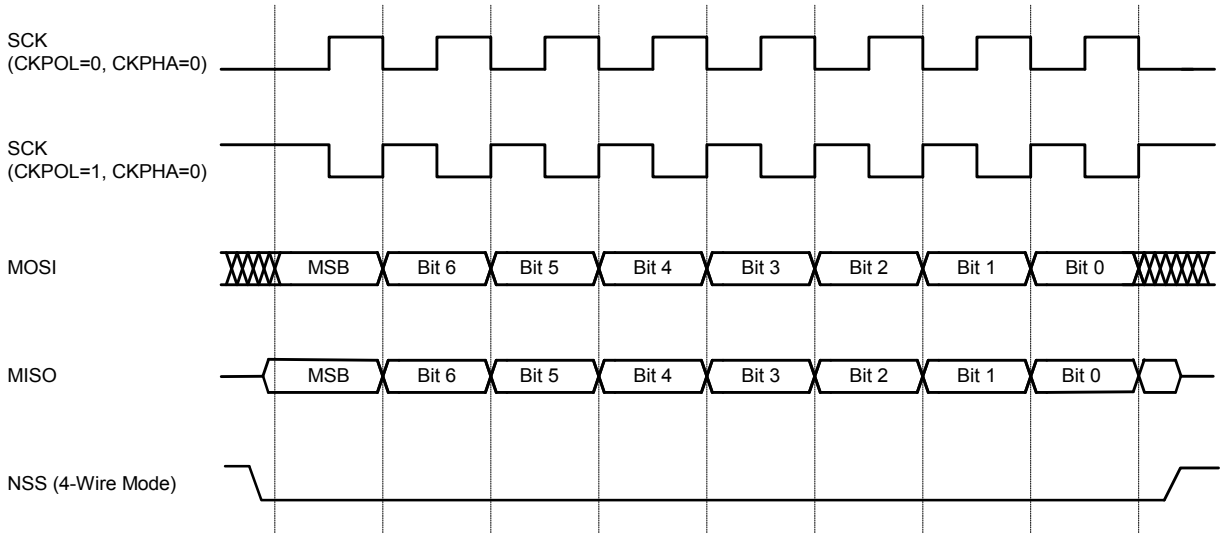


Figure 29.6. Slave Mode Data/Clock Timing (CKPHA = 0)

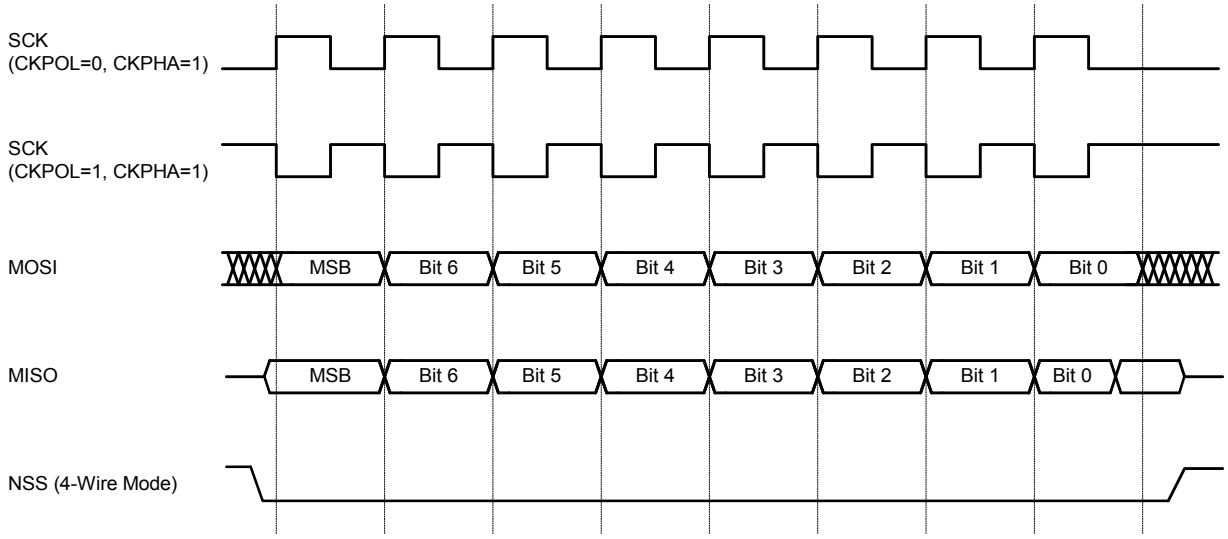


Figure 29.7. Slave Mode Data/Clock Timing (CKPHA = 1)

29.6. SPI Special Function Registers

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

SFR Definition 29.1. SPI0CFG: SPI0 Configuration

Bit	7	6	5	4	3	2	1	0
Name	SPIBSY	MSTEN	CKPHA	CKPOL	SLVSEL	NSSIN	SRMT	RXBMT
Type	R	R/W	R/W	R/W	R	R	R	R
Reset	0	0	0	0	0	1	1	1

SFR Address = 0xA1; SFR Page = 0

Bit	Name	Function
7	SPIBSY	**SPI Busy.** This bit is set to logic 1 when a SPI transfer is in progress (master or slave mode).
6	MSTEN	**Master Mode Enable.** 0: Disable master mode. Operate in slave mode. 1: Enable master mode. Operate as a master.
5	CKPHA	**SPI0 Clock Phase.** 0: Data centered on first edge of SCK period.* 1: Data centered on second edge of SCK period.*
4	CKPOL	**SPI0 Clock Polarity.** 0: SCK line low in idle state. 1: SCK line high in idle state.
3	SLVSEL	**Slave Selected Flag.** 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.** 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).** This bit will be set to logic 1 when all data has been transferred in/out of the shift register, and there is no new information available to read from the transmit buffer or write to the receive buffer. It returns to logic 0 when a data byte is transferred to the shift register from the transmit buffer or by a transition on SCK. SRMT = 1 when in Master Mode.
0	RXBMT	**Receive Buffer Empty (valid in slave mode only).** This bit will be set to logic 1 when the receive buffer has been read and contains no new information. If there is new information available in the receive buffer that has not been read, this bit will return to logic 0. RXBMT = 1 when in Master Mode.

Note: In slave mode, data on MOSI is sampled in the center of each data bit. In master mode, data on MISO is sampled one SYSCLK before the end of each data bit, to provide maximum settling time for the slave device. See Table 29.1 for timing parameters.

SFR Definition 29.2. SPI0CN: SPI0 Control

Bit	7	6	5	4	3	2	1	0
Name	SPIF	WCOL	MODF	RXOVRN	NSSMD[1:0]		TXBMT	SPIEN
Type	R/W	R/W	R/W	R/W	R/W		R	R/W
Reset	0	0	0	0	0	1	1	0

SFR Address = 0xF8; SFR Page = All Pages; Bit-Addressable

Bit	Name	Function
7	SPIF	**SPI0 Interrupt Flag.** 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.** 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.** This bit is set to logic 1 by hardware when a master mode collision is detected (NSS is low, MSTEN = 1, and NSSMD[1:0] = 01). If SPI interrupts are enabled, an interrupt will be generated. This bit is not automatically cleared by hardware, and must be cleared by software.
4	RXOVRN	**Receive Overrun Flag (valid in slave mode only).** This bit is set to logic 1 by hardware when the receive buffer still holds unread data from a previous transfer and the last bit of the current transfer is shifted into the SPI0 shift register. If SPI interrupts are enabled, an interrupt will be generated. This bit is not automatically cleared by hardware, and must be cleared by software.
3:2	NSSMD[1:0]	**Slave Select Mode.** Selects between the following NSS operation modes: (See Section 29.2 and Section 29.3). 00: 3-Wire Slave or 3-Wire Master Mode. NSS signal is not routed to a port pin. 01: 4-Wire Slave or Multi-Master Mode (Default). NSS is an input to the device. 1x: 4-Wire Single-Master Mode. NSS signal is mapped as an output from the device and will assume the value of NSSMD0.
1	TXBMT	**Transmit Buffer Empty.** This bit will be set to logic 0 when new data has been written to the transmit buffer. When data in the transmit buffer is transferred to the SPI shift register, this bit will be set to logic 1, indicating that it is safe to write a new byte to the transmit buffer.
0	SPIEN	**SPI0 Enable.** 0: SPI disabled. 1: SPI enabled.

SFR Definition 29.3. SPI0CKR: SPI0 Clock Rate

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

SFR Address = 0xA2; SFR Page = F

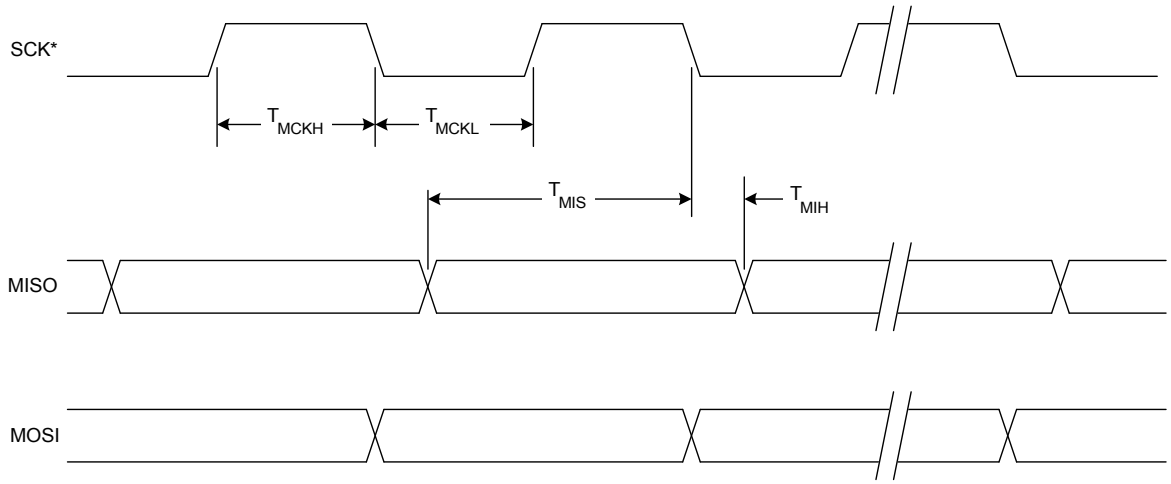
Bit	Name	Function
7:0	SCR[7:0]	**SPI0 Clock Rate.** These bits determine the frequency of the SCK output when the SPI0 module is configured for master mode operation. The SCK clock frequency is a divided version of the system clock, and is given in the following equation, where *SYSCLK* is the system clock frequency and *SPI0CKR* is the 8-bit value held in the SPI0CKR register. $f_{SCK} = \frac{SYSCLK}{2 \times (SPI0CKR[7:0] + 1)}$ for 0 <= SPI0CKR <= 255 Example: If SYSCLK = 2 MHz and SPI0CKR = 0x04, $f_{SCK} = \frac{2000000}{2 \times (4 + 1)}$ $f_{SCK} = 200kHz$

SFR Definition 29.4. SPI0DAT: SPI0 Data

Bit	7	6	5	4	3	2	1	0
Name	SPI0DAT[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

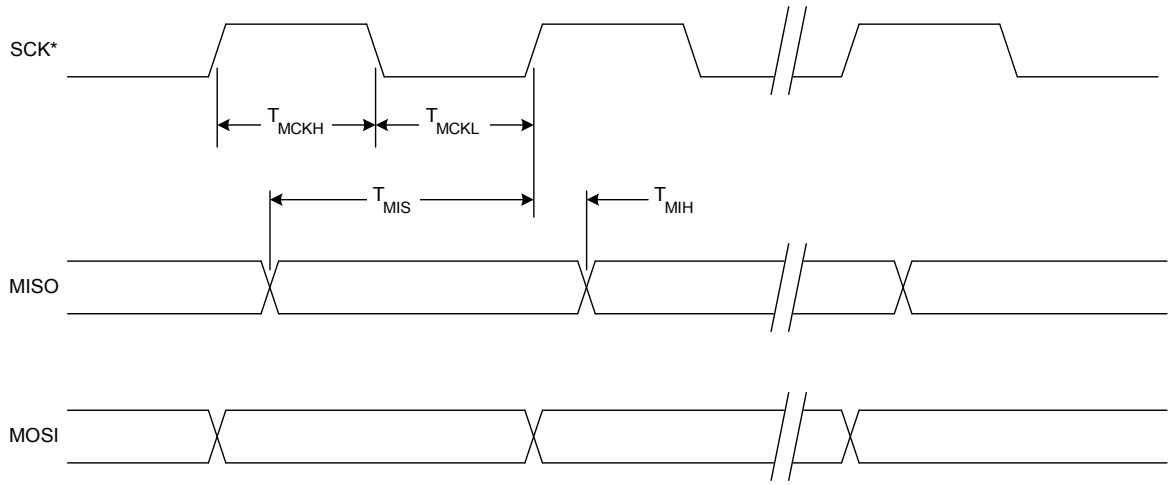
SFR Address = 0xA3; SFR Page = 0

Bit	Name	Function
7:0	SPI0DAT[7:0]	**SPI0 Transmit and Receive Data.** The SPI0DAT register is used to transmit and receive SPI0 data. Writing data to SPI0DAT places the data into the transmit buffer and initiates a transfer when in Master Mode. A read of SPI0DAT returns the contents of the receive buffer.



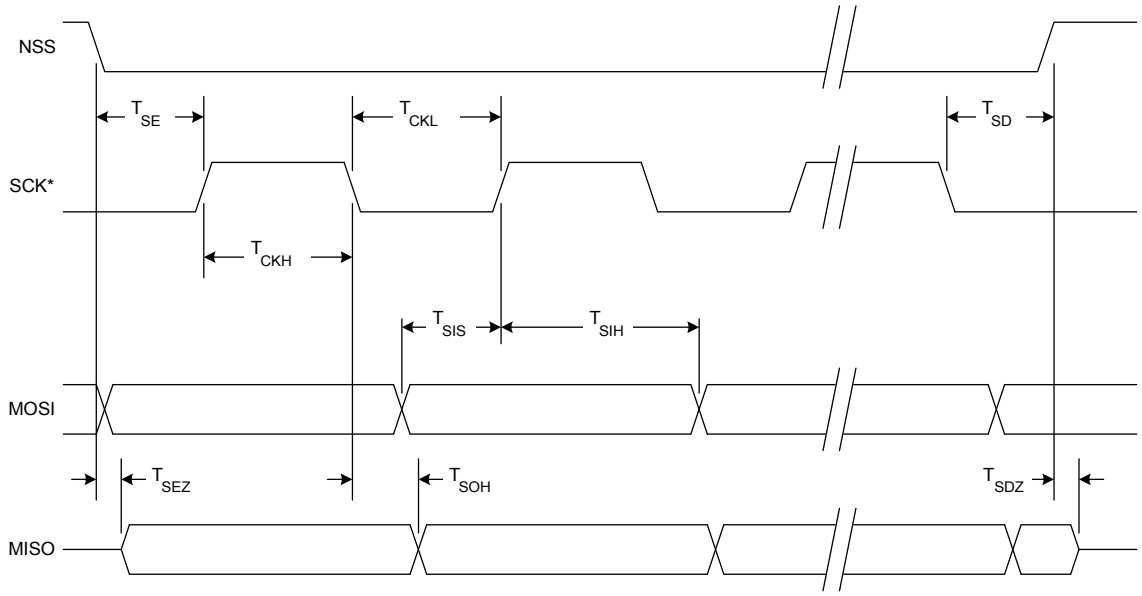
* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

Figure 29.8. SPI Master Timing (CKPHA = 0)



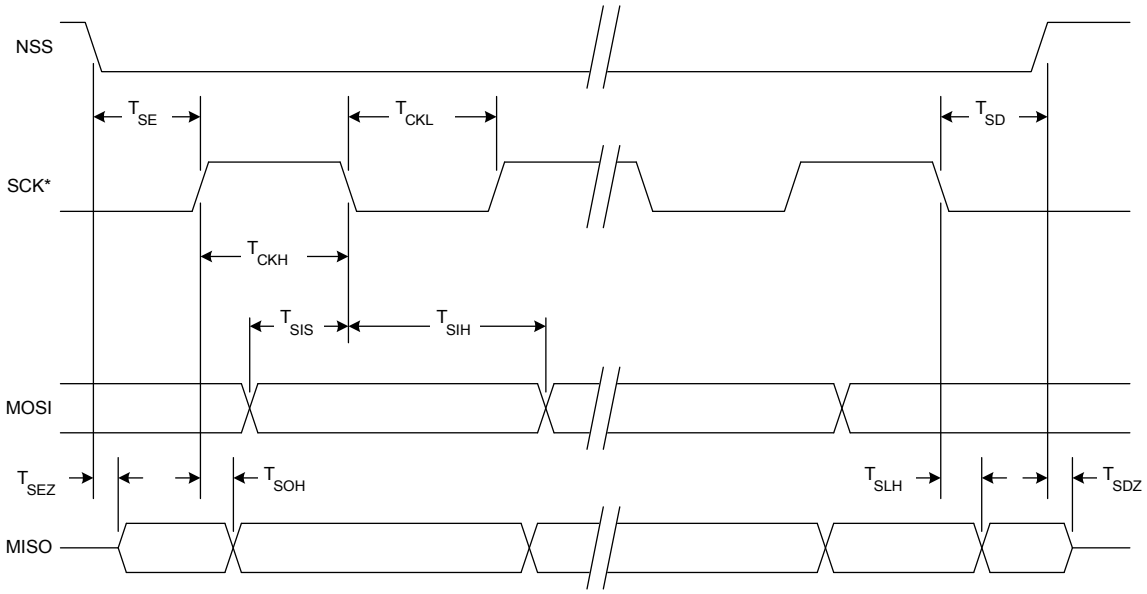
* SCK is shown for CKPOL = 0. SCK is the opposite polarity for CKPOL = 1.

Figure 29.9. SPI Master Timing (CKPHA = 1)



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

Figure 29.10. SPI Slave Timing (CKPHA = 0)



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

Figure 29.11. SPI Slave Timing (CKPHA = 1)

Table 29.1. SPI Slave Timing Parameters

Parameter	Description	Min	Max	Units
Master Mode Timing (See Figure 29.8 and Figure 29.9)				
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 29.10 and Figure 29.11)				
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).				

30. 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 “30.1. Enhanced Baud Rate Generation” on page 237). Received data buffering allows UART0 to start reception of a second incoming data byte before software has finished reading the previous data byte.

UART0 has two associated SFRs: Serial Control Register 0 (SCON0) and Serial Data Buffer 0 (SBUF0). The single SBUF0 location provides access to both transmit and receive registers. **Writes to SBUF0 always access the Transmit register. Reads of SBUF0 always access the buffered Receive register; it is not possible to read data from the Transmit register.**

With UART0 interrupts enabled, an interrupt is generated each time a transmit is completed (TI0 is set in SCON0), or a data byte has been received (RI0 is set in SCON0). The UART0 interrupt flags are not cleared by hardware when the CPU vectors to the interrupt service routine. They must be cleared manually by software, allowing software to determine the cause of the UART0 interrupt (transmit complete or receive complete).

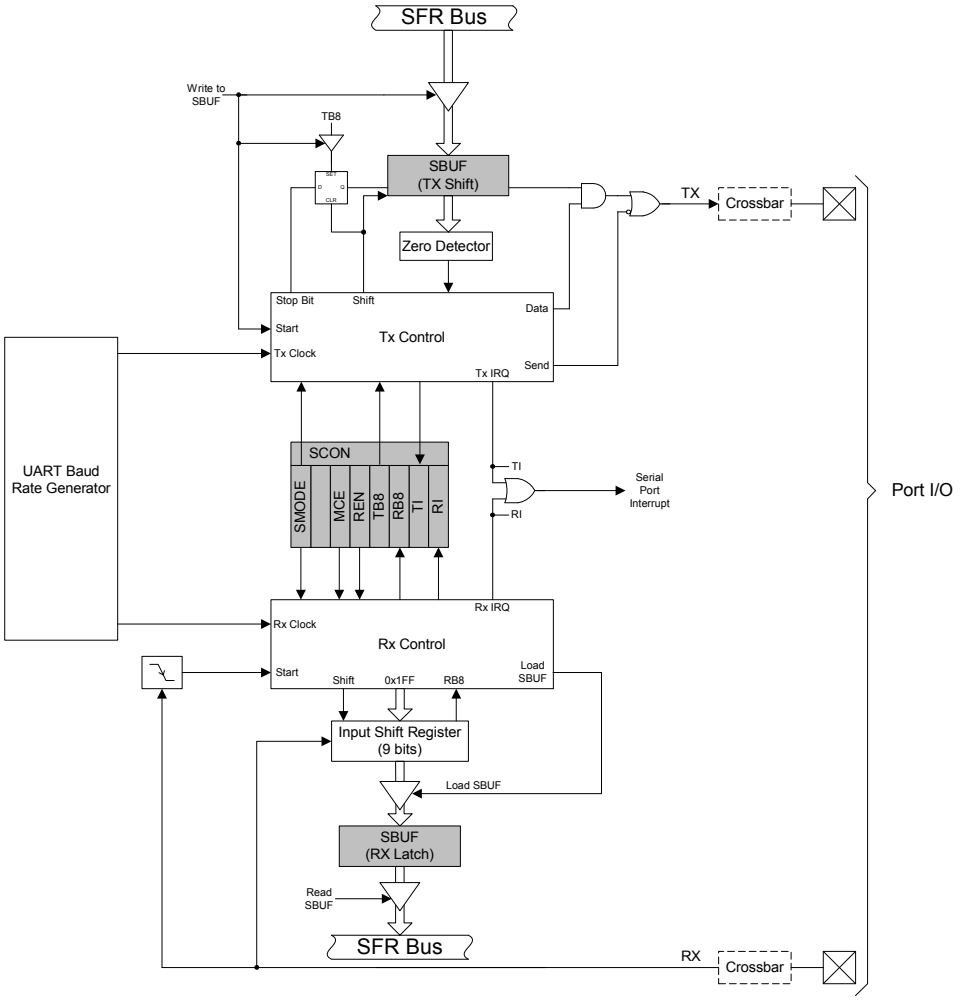


Figure 30.1. UART0 Block Diagram

30.1. 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 30.2), 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.

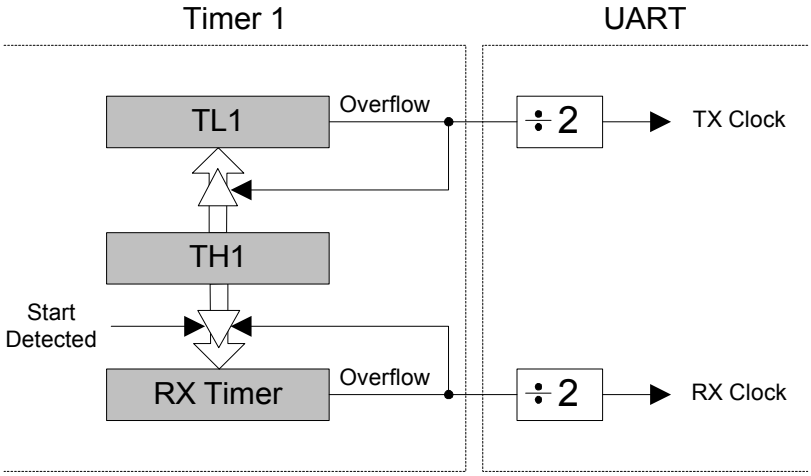


Figure 30.2. UART0 Baud Rate Logic

Timer 1 should be configured for Mode 2, 8-bit auto-reload (see Section “31.1.3. Mode 2: 8-bit Counter/Timer with Auto-Reload” on page 247). The Timer 1 reload value should be set so that overflows will occur at two times the desired UART baud rate frequency. Timer 1 may be clocked by one of six sources: SYSCLK, SYSCLK/4, SYSCLK/12, SYSCLK/48, the external oscillator clock/8, or an external input T1. For any given Timer 1 clock source, the UART0 baud rate is determined by Equation 30.1-A and Equation 30.1-B.

A) $$\text{UartBaudRate} = \frac{1}{2} \times \text{T1_Overflow_Rate}$$

B) $$\text{T1_Overflow_Rate} = \frac{\text{T1}_{\text{CLK}}}{256 - \text{TH1}}$$

Equation 30.1. UART0 Baud Rate

Where $T1_{CLK}$ is the frequency of the clock supplied to Timer 1, and *T1H* is the high byte of Timer 1 (reload value). Timer 1 clock frequency is selected as described in Section “31. Timers” on page 244. A quick reference for typical baud rates and system clock frequencies is given in Table 30.1 through Table 30.2. The internal oscillator may still generate the system clock when the external oscillator is driving Timer 1.

30.2. Operational Modes

UART0 provides standard asynchronous, full duplex communication. The UART mode (8-bit or 9-bit) is selected by the S0MODE bit (SCON0.7). Typical UART connection options are shown in Figure 30.3.

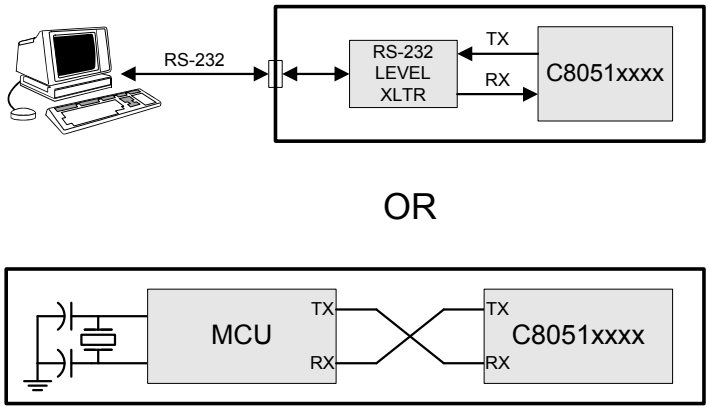


Figure 30.3. UART Interconnect Diagram

30.2.1. 8-Bit UART

8-Bit UART mode uses a total of 10 bits per data byte: one start bit, eight data bits (LSB first), and one stop bit. Data are transmitted LSB first from the TX0 pin and received at the RX0 pin. On receive, the eight data bits are stored in SBUF0 and the stop bit goes into RB80 (SCON0.2).

Data transmission begins when software writes a data byte to the SBUF0 register. The TI0 Transmit Interrupt Flag (SCON0.1) is set at the end of the transmission (the beginning of the stop-bit time). Data reception can begin any time after the REN0 Receive Enable bit (SCON0.4) is set to logic 1. After the stop bit is received, the data byte will be loaded into the SBUF0 receive register if the following conditions are met: RI0 must be logic 0, and if MCE0 is logic 1, the stop bit must be logic 1. In the event of a receive data overrun, the first received 8 bits are latched into the SBUF0 receive register and the following overrun data bits are lost.

If these conditions are met, the eight bits of data is stored in SBUF0, the stop bit is stored in RB80 and the RI0 flag is set. If these conditions are not met, SBUF0 and RB80 will not be loaded and the RI0 flag will not be set. An interrupt will occur if enabled when either TI0 or RI0 is set.

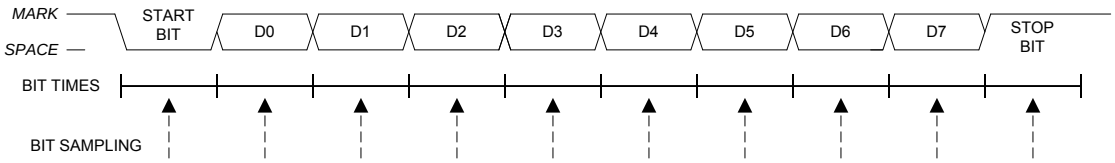


Figure 30.4. 8-Bit UART Timing Diagram

30.2.2. 9-Bit UART

9-bit UART mode uses a total of eleven bits per data byte: a start bit, 8 data bits (LSB first), a programmable ninth data bit, and a stop bit. The state of the ninth transmit data bit is determined by the value in TB80 (SCON0.3), which is assigned by user software. It can be assigned the value of the parity flag (bit P in register PSW) for error detection, or used in multiprocessor communications. On receive, the ninth data bit goes into RB80 (SCON0.2) and the stop bit is ignored.

Data transmission begins when an instruction writes a data byte to the SBUF0 register. The TI0 Transmit Interrupt Flag (SCON0.1) is set at the end of the transmission (the beginning of the stop-bit time). Data reception can begin any time after the REN0 Receive Enable bit (SCON0.4) is set to 1. After the stop bit is received, the data byte will be loaded into the SBUF0 receive register if the following conditions are met: (1) RI0 must be logic 0, and (2) if MCE0 is logic 1, the 9th bit must be logic 1 (when MCE0 is logic 0, the state of the ninth data bit is unimportant). If these conditions are met, the eight bits of data are stored in SBUF0, the ninth bit is stored in RB80, and the RI0 flag is set to 1. If the above conditions are not met, SBUF0 and RB80 will not be loaded and the RI0 flag will not be set to 1. A UART0 interrupt will occur if enabled when either TI0 or RI0 is set to 1.

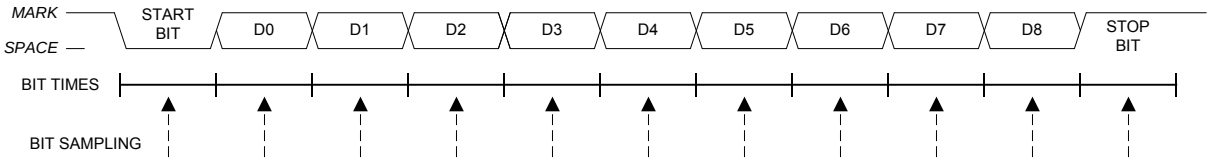


Figure 30.5. 9-Bit UART Timing Diagram

30.3. Multiprocessor Communications

9-Bit UART mode supports multiprocessor communication between a master processor and one or more slave processors by special use of the ninth data bit. When a master processor wants to transmit to one or more slaves, it first sends an address byte to select the target(s). An address byte differs from a data byte in that its ninth bit is logic 1; in a data byte, the ninth bit is always set to logic 0.

Setting the MCE0 bit (SCON0.5) of a slave processor configures its UART such that when a stop bit is received, the UART will generate an interrupt only if the ninth bit is logic 1 (RB80 = 1) signifying an address byte has been received. In the UART interrupt handler, software will compare the received address with the slave's own assigned 8-bit address. If the addresses match, the slave will clear its MCE0 bit to enable interrupts on the reception of the following data byte(s). Slaves that weren't addressed leave their MCE0 bits set and do not generate interrupts on the reception of the following data bytes, thereby ignoring the data. Once the entire message is received, the addressed slave resets its MCE0 bit to ignore all transmissions until it receives the next address byte.

Multiple addresses can be assigned to a single slave and/or a single address can be assigned to multiple slaves, thereby enabling "broadcast" transmissions to more than one slave simultaneously. The master processor can be configured to receive all transmissions or a protocol can be implemented such that the master/slave role is temporarily reversed to enable half-duplex transmission between the original master and slave(s).

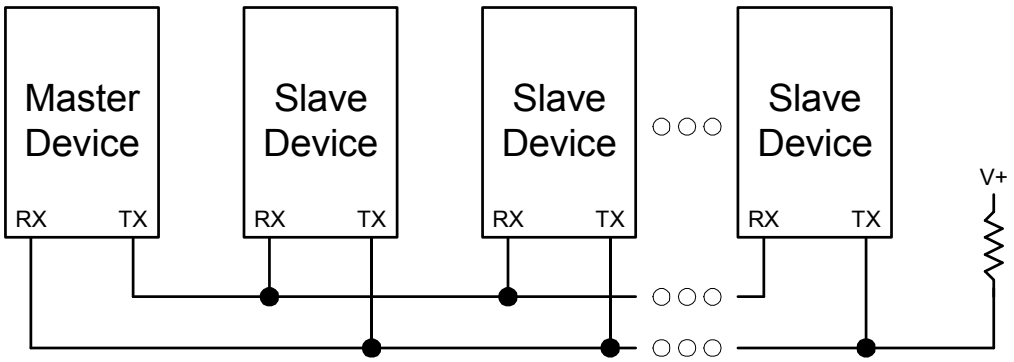


Figure 30.6. UART Multi-Processor Mode Interconnect Diagram

SFR Definition 30.1. SCON0: Serial Port 0 Control

Bit	7	6	5	4	3	2	1	0
Name	S0MODE		MCE0	REN0	TB80	RB80	TI0	RI0
Type	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	1	0	0	0	0	0	0

SFR Address = 0x98; SFR Page = All Pages; Bit-Addressable

Bit	Name	Function
7	S0MODE	**Serial Port 0 Operation Mode.** Selects the UART0 Operation Mode. 0: 8-bit UART with Variable Baud Rate. 1: 9-bit UART with Variable Baud Rate.
6	Unused	Read = 1b, Write = Don't Care.
5	MCE0	**Multiprocessor Communication Enable.** The function of this bit is dependent on the Serial Port 0 Operation Mode: **Mode 0: Checks for valid stop bit.** 0: Logic level of stop bit is ignored. 1: RI0 will only be activated if stop bit is logic level 1. **Mode 1: Multiprocessor Communications Enable.** 0: Logic level of ninth bit is ignored. 1: RI0 is set and an interrupt is generated only when the ninth bit is logic 1.
4	REN0	**Receive Enable.** 0: UART0 reception disabled. 1: UART0 reception enabled.
3	TB80	**Ninth Transmission Bit.** The logic level of this bit will be sent as the ninth transmission bit in 9-bit UART Mode (Mode 1). Unused in 8-bit mode (Mode 0).
2	RB80	**Ninth Receive Bit.** RB80 is assigned the value of the STOP bit in Mode 0; it is assigned the value of the 9th data bit in Mode 1.
1	TI0	**Transmit Interrupt Flag.** Set by hardware when a byte of data has been transmitted by UART0 (after the 8th bit in 8-bit UART Mode, or at the beginning of the STOP bit in 9-bit UART Mode). When the UART0 interrupt is enabled, setting this bit causes the CPU to vector to the UART0 interrupt service routine. This bit must be cleared manually by software.
0	RI0	**Receive Interrupt Flag.** Set to 1 by hardware when a byte of data has been received by UART0 (set at the STOP bit sampling time). When the UART0 interrupt is enabled, setting this bit to 1 causes the CPU to vector to the UART0 interrupt service routine. This bit must be cleared manually by software.

SFR Definition 30.2. SBUF0: Serial (UART0) Port Data Buffer

Bit	7	6	5	4	3	2	1	0
Name	SBUF0[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x99; SFR Page = All Pages

Bit	Name	Function
7:0	SBUF0[7:0]	**Serial Data Buffer Bits 7–0 (MSB–LSB).** This SFR accesses two registers; a transmit shift register and a receive latch register. When data is written to SBUF0, it goes to the transmit shift register and is held for serial transmission. Writing a byte to SBUF0 initiates the transmission. A read of SBUF0 returns the contents of the receive latch.

Table 30.1. Timer Settings for Standard Baud Rates Using The Internal 24.5 MHz Oscillator

	Frequency: 24.5 MHz						
	Target Baud Rate (bps)	Baud Rate% Error	Oscillator Divide Factor	Timer Clock Source	SCA1–SCA0 (pre-scale select)[1]	T1M[1]	Timer 1 Reload Value (hex)
SYSCLK from Internal Osc.	230400	–0.32%	106	SYSCLK	XX[2]	1	0xCB
	115200	–0.32%	212	SYSCLK	XX	1	0x96
	57600	0.15%	426	SYSCLK	XX	1	0x2B
	28800	–0.32%	848	SYSCLK/4	01	0	0x96
	14400	0.15%	1704	SYSCLK/12	00	0	0xB9
	9600	–0.32%	2544	SYSCLK/12	00	0	0x96
	2400	–0.32%	10176	SYSCLK/48	10	0	0x96
	1200	0.15%	20448	SYSCLK/48	10	0	0x2B

Notes:
1. SCA1–SCA0 and T1M bit definitions can be found in Section 31.1.
2. X = Don't care.

Table 30.2. Timer Settings for Standard Baud Rates Using an External 22.1184 MHz Oscillator

	Frequency: 22.1184 MHz						
	Target Baud Rate (bps)	Baud Rate% Error	Oscillator Divide Factor	Timer Clock Source	SCA1–SCA0 (pre-scale select)[1]	T1M[1]	Timer 1 Reload Value (hex)
SYSCLK from External Osc.	230400	0.00%	96	SYSCLK	XX[2]	1	0xD0
	115200	0.00%	192	SYSCLK	XX	1	0xA0
	57600	0.00%	384	SYSCLK	XX	1	0x40
	28800	0.00%	768	SYSCLK / 12	00	0	0xE0
	14400	0.00%	1536	SYSCLK / 12	00	0	0xC0
	9600	0.00%	2304	SYSCLK / 12	00	0	0xA0
	2400	0.00%	9216	SYSCLK / 48	10	0	0xA0
	1200	0.00%	18432	SYSCLK / 48	10	0	0x40
SYSCLK from Internal Osc.	230400	0.00%	96	EXTCLK / 8	11	0	0xFA
	115200	0.00%	192	EXTCLK / 8	11	0	0xF4
	57600	0.00%	384	EXTCLK / 8	11	0	0xE8
	28800	0.00%	768	EXTCLK / 8	11	0	0xD0
	14400	0.00%	1536	EXTCLK / 8	11	0	0xA0
	9600	0.00%	2304	EXTCLK / 8	11	0	0x70

Notes:
1. SCA1–SCA0 and T1M bit definitions can be found in Section 31.1.
2. X = Don't care.

31. Timers

Each MCU includes four counter/timers: two are 16-bit counter/timers compatible with those found in the standard 8051, and two are 16-bit auto-reload timer for use with the ADC, SMBus, or for general purpose use. These timers can be used to measure time intervals, count external events and generate periodic interrupt requests. Timer 0 and Timer 1 are nearly identical and have four primary modes of operation. Timer 2 and Timer 3 offer 16-bit and split 8-bit timer functionality with auto-reload. Additionally, Timer 3 offers the ability to be clocked from the external oscillator while the device is in Suspend mode, and can be used as a wake-up source. This allows for implementation of a very low-power system, including RTC capability.

Timer 0 and Timer 1 Modes:	Timer 2 Modes:	Timer 3 Modes:
13-bit counter/timer	16-bit timer with auto-reload	16-bit timer with auto-reload
16-bit counter/timer		
8-bit counter/timer with auto-reload	Two 8-bit timers with auto-reload	Two 8-bit timers with auto-reload
Two 8-bit counter/timers (Timer 0 only)		

Timers 0 and 1 may be clocked by one of five sources, determined by the Timer Mode Select bits (T1M–T0M) and the Clock Scale bits (SCA1–SCA0). The Clock Scale bits define a pre-scaled clock from which Timer 0 and/or Timer 1 may be clocked (See SFR Definition 31.1 for pre-scaled clock selection).

Timer 0/1 may then be configured to use this pre-scaled clock signal or the system clock. Timer 2 and Timer 3 may be clocked by the system clock, the system clock divided by 12, 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 should be held at a given level for at least two full system clock cycles to ensure the level is properly sampled.

SFR Definition 31.1. CKCON: Clock Control

Bit	7	6	5	4	3	2	1	0
Name	T3MH	T3ML	T2MH	T2ML	T1M	T0M	SCA[1:0]	
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x8E; SFR Page = All Pages

Bit	Name	Function
7	T3MH	**Timer 3 High Byte Clock Select.** Selects the clock supplied to the Timer 3 high byte (split 8-bit timer mode only). 0: Timer 3 high byte uses the clock defined by the T3XCLK bit in TMR3CN. 1: Timer 3 high byte uses the system clock.
6	T3ML	**Timer 3 Low Byte Clock Select.** Selects the clock supplied to Timer 3. Selects the clock supplied to the lower 8-bit timer in split 8-bit timer mode. 0: Timer 3 low byte uses the clock defined by the T3XCLK bit in TMR3CN. 1: Timer 3 low byte uses the system clock.
5	T2MH	**Timer 2 High Byte Clock Select.** Selects the clock supplied to the Timer 2 high byte (split 8-bit timer mode only). 0: Timer 2 high byte uses the clock defined by the T2XCLK bit in TMR2CN. 1: Timer 2 high byte uses the system clock.
4	T2ML	**Timer 2 Low Byte Clock Select.** Selects the clock supplied to Timer 2. If Timer 2 is configured in split 8-bit timer mode, this bit selects the clock supplied to the lower 8-bit timer. 0: Timer 2 low byte uses the clock defined by the T2XCLK bit in TMR2CN. 1: Timer 2 low byte uses the system clock.
3	T1	**Timer 1 Clock Select.** Selects the clock source supplied to Timer 1. Ignored when C/T1 is set to 1. 0: Timer 1 uses the clock defined by the prescale bits SCA[1:0]. 1: Timer 1 uses the system clock.
2	T0	**Timer 0 Clock Select.** Selects the clock source supplied to Timer 0. Ignored when C/T0 is set to 1. 0: Counter/Timer 0 uses the clock defined by the prescale bits SCA[1:0]. 1: Counter/Timer 0 uses the system clock.
1:0	SCA[1:0]	**Timer 0/1 Prescale Bits.** These bits control the Timer 0/1 Clock Prescaler: 00: System clock divided by 12 01: System clock divided by 4 10: System clock divided by 48 11: External clock divided by 8 (synchronized with the system clock)

31.1. Timer 0 and Timer 1

Each timer is implemented as a 16-bit register accessed as two separate bytes: a low byte (TL0 or TL1) and a high byte (TH0 or TH1). The Counter/Timer Control register (TCON) is used to enable Timer 0 and Timer 1 as well as indicate status. Timer 0 interrupts can be enabled by setting the ET0 bit in the IE register (Section “19.2. Interrupt Register Descriptions” on page 124); Timer 1 interrupts can be enabled by setting the ET1 bit in the IE register (Section “19.2. Interrupt Register Descriptions” on page 124). Both counter/timers operate in one of four primary modes selected by setting the Mode Select bits T1M1–T0M0 in the Counter/Timer Mode register (TMOD). Each timer can be configured independently. Each operating mode is described below.

31.1.1. 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 C/T0 bit in the TMOD register selects the counter/timer's clock source. When C/T0 is set to logic 1, high-to-low transitions at the selected Timer 0 input pin (T0) increment the timer register (Refer to Section “26.3. Priority Crossbar Decoder” on page 170 for information on selecting and configuring external I/O pins). Clearing C/T 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 (see SFR Definition 31.1).

Setting the TR0 bit (TCON.4) enables the timer when either GATE0 in the TMOD register is logic 0 or the input signal $\overline{\text{INT0}}$ is active as defined by bit IN0PL in register IT01CF (see SFR Definition 19.7). Setting GATE0 to 1 allows the timer to be controlled by the external input signal $\overline{\text{INT0}}$ (see Section “19.2. Interrupt Register Descriptions” on page 124), facilitating pulse width measurements

TR0	GATE0	$\overline{\text{INT0}}$	Counter/Timer
0	X	X	Disabled
1	0	X	Enabled
1	1	0	Disabled
1	1	1	Enabled
Note: X = Don't Care			

Setting TR0 does not force the timer to reset. The timer registers should be loaded with the desired initial value before the timer is enabled.

TL1 and TH1 form the 13-bit register for Timer 1 in the same manner as described above for TL0 and TH0. Timer 1 is configured and controlled using the relevant TCON and TMOD bits just as with Timer 0. The input signal $\overline{\text{INT0}}$ is used with Timer 1; the /INT1 polarity is defined by bit IN1PL in register IT01CF (see SFR Definition 19.7).

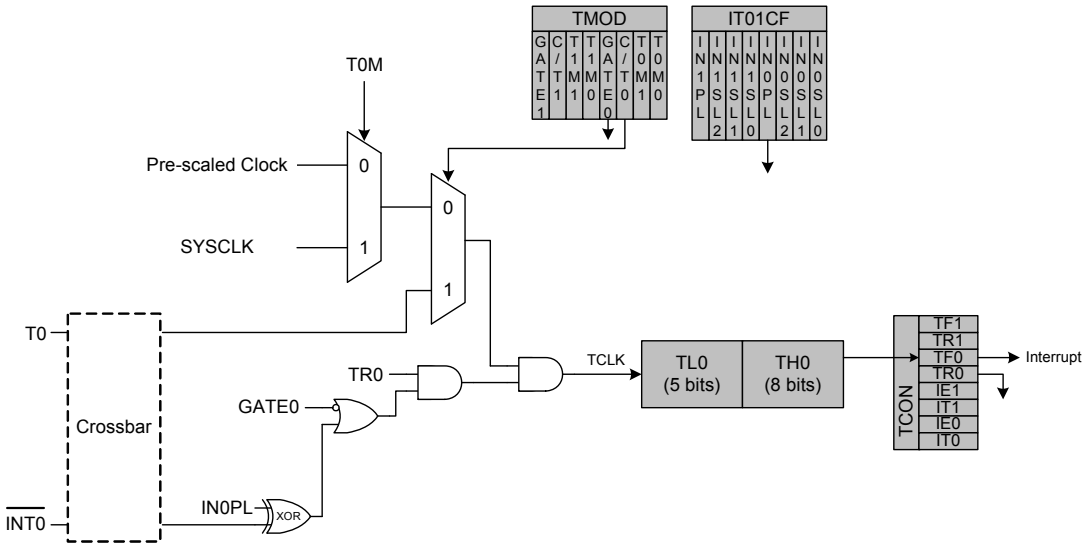


Figure 31.1. T0 Mode 0 Block Diagram

31.1.2. 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.

31.1.3. 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 (TCON.4) enables the timer when either GATE0 in the TMOD register is logic 0 or when the input signal $\overline{\text{INT0}}$ is active as defined by bit IN0PL in register IT01CF (see Section “19.3. INT0 and INT1 External Interrupts” on page 131 for details on the external input signals $\overline{\text{INT0}}$ and $\overline{\text{INT1}}$).

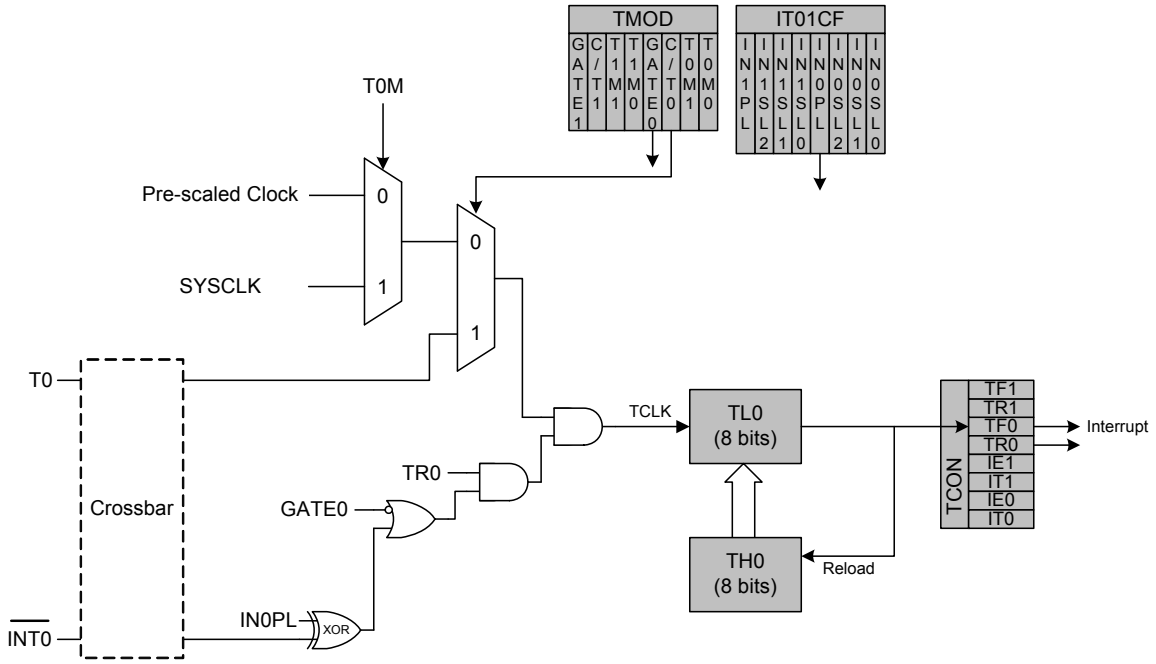


Figure 31.2. T0 Mode 2 Block Diagram

31.1.4. Mode 3: Two 8-bit Counter/Timers (Timer 0 Only)

In Mode 3, Timer 0 is configured as two separate 8-bit counter/timers held in TL0 and TH0. The counter/timer in TL0 is controlled using the Timer 0 control/status bits in TCON and TMOD: TR0, C/T0, GATE0 and TF0. TL0 can use either the system clock or an external input signal as its timebase. The TH0 register is restricted to a timer function sourced by the system clock or prescaled clock. TH0 is enabled using the Timer 1 run control bit TR1. TH0 sets the Timer 1 overflow flag TF1 on overflow and thus controls the Timer 1 interrupt.

Timer 1 is inactive in Mode 3. When Timer 0 is operating in Mode 3, Timer 1 can be operated in Modes 0, 1 or 2, but cannot be clocked by external signals nor set the TF1 flag and generate an interrupt. However, the Timer 1 overflow can be used to generate baud rates for the SMBus and/or UART, and/or initiate ADC conversions. While Timer 0 is operating in Mode 3, Timer 1 run control is handled through its mode settings. To run Timer 1 while Timer 0 is in Mode 3, set the Timer 1 Mode as 0, 1, or 2. To disable Timer 1, configure it for Mode 3.

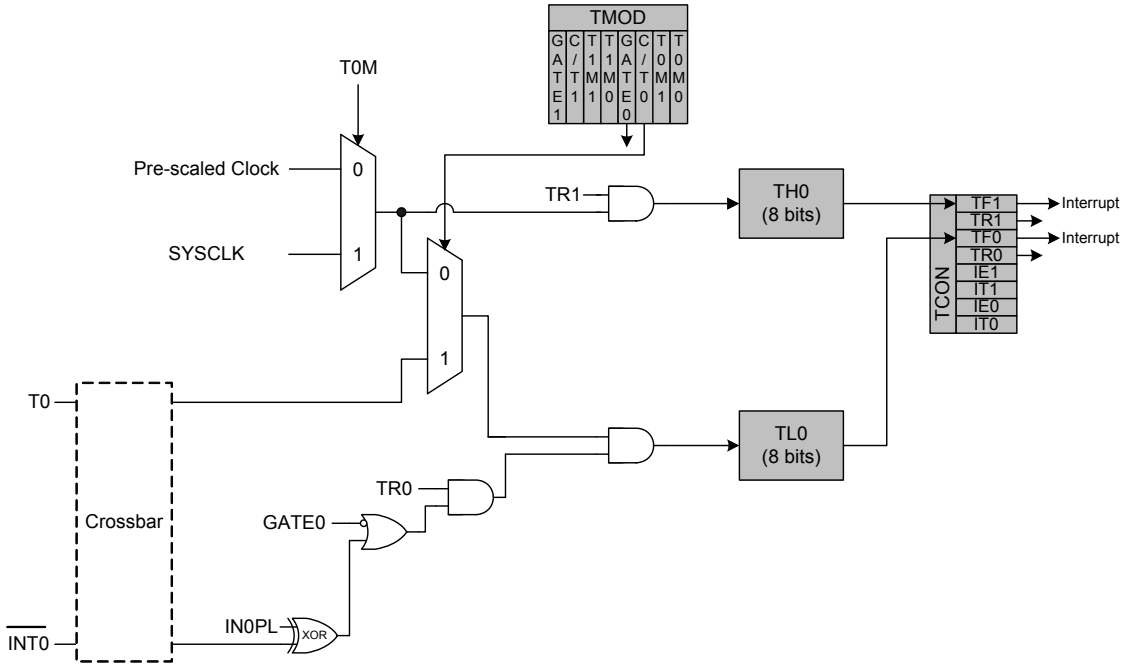


Figure 31.3. T0 Mode 3 Block Diagram

SFR Definition 31.2. TCON: Timer Control

Bit	7	6	5	4	3	2	1	0
Name	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x88; SFR Page = All Pages; Bit-Addressable

Bit	Name	Function
7	TF1	**Timer 1 Overflow Flag.** 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.** Timer 1 is enabled by setting this bit to 1.
5	TF0	**Timer 0 Overflow Flag.** 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.** Timer 0 is enabled by setting this bit to 1.
3	IE1	**External Interrupt 1.** This flag is set by hardware when an edge/level of type defined by IT1 is detected. It can be cleared by software but is automatically cleared when the CPU vectors to the External Interrupt 1 service routine in edge-triggered mode.
2	IT1	**Interrupt 1 Type Select.** This bit selects whether the configured /INT1 interrupt will be edge or level sensitive. /INT1 is configured active low or high by the IN1PL bit in the IT01CF register (see SFR Definition 19.7). 0: /INT1 is level triggered. 1: /INT1 is edge triggered.
1	IE0	**External Interrupt 0.** This flag is set by hardware when an edge/level of type defined by IT1 is detected. It can be cleared by software but is automatically cleared when the CPU vectors to the External Interrupt 0 service routine in edge-triggered mode.
0	IT0	**Interrupt 0 Type Select.** This bit selects whether the configured $\overline{\text{INT0}}$ interrupt will be edge or level sensitive. $\overline{\text{INT0}}$ is configured active low or high by the IN0PL bit in register IT01CF (see SFR Definition 19.7). 0: $\overline{\text{INT0}}$ is level triggered. 1: $\overline{\text{INT0}}$ is edge triggered.

SFR Definition 31.3. TMOD: Timer Mode

Bit	7	6	5	4	3	2	1	0
Name	GATE1	C/T1	T1M[1:0]		GATE0	C/T0	T0M[1:0]	
Type	R/W	R/W	R/W		R/W	R/W	R/W	
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x89; SFR Page = All Pages

Bit	Name	Function
7	GATE1	**Timer 1 Gate Control.** 0: Timer 1 enabled when TR1 = 1 irrespective of /INT1 logic level. 1: Timer 1 enabled only when TR1 = 1 AND /INT1 is active as defined by bit IN1PL in register IT01CF (see SFR Definition 19.7).
6	C/T1	**Counter/Timer 1 Select.** 0: Timer: Timer 1 incremented by clock defined by T1M bit in register CKCON. 1: Counter: Timer 1 incremented by high-to-low transitions on external pin (T1).
5:4	T1M[1:0]	**Timer 1 Mode Select.** These bits select the Timer 1 operation mode. 00: Mode 0, 13-bit Counter/Timer 01: Mode 1, 16-bit Counter/Timer 10: Mode 2, 8-bit Counter/Timer with Auto-Reload 11: Mode 3, Timer 1 Inactive
3	GATE0	**Timer 0 Gate Control.** 0: Timer 0 enabled when TR0 = 1 irrespective of $\overline{\text{INT0}}$ logic level. 1: Timer 0 enabled only when TR0 = 1 AND $\overline{\text{INT0}}$ is active as defined by bit IN0PL in register IT01CF (see SFR Definition 19.7).
2	C/T0	**Counter/Timer 0 Select.** 0: Timer: Timer 0 incremented by clock defined by T0M bit in register CKCON. 1: Counter: Timer 0 incremented by high-to-low transitions on external pin (T0).
1:0	T0M[1:0]	**Timer 0 Mode Select.** These bits select the Timer 0 operation mode. 00: Mode 0, 13-bit Counter/Timer 01: Mode 1, 16-bit Counter/Timer 10: Mode 2, 8-bit Counter/Timer with Auto-Reload 11: Mode 3, Two 8-bit Counter/Timers

SFR Definition 31.4. TL0: Timer 0 Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TL0[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x8A; SFR Page = All Pages

Bit	Name	Function
7:0	TL0[7:0]	**Timer 0 Low Byte.** The TL0 register is the low byte of the 16-bit Timer 0.

SFR Definition 31.5. TL1: Timer 1 Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TL1[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x8B; SFR Page = All Pages

Bit	Name	Function
7:0	TL1[7:0]	**Timer 1 Low Byte.** The TL1 register is the low byte of the 16-bit Timer 1.

SFR Definition 31.6. TH0: Timer 0 High Byte

Bit	7	6	5	4	3	2	1	0
Name	TH0[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x8C; SFR Page = All Pages

Bit	Name	Function
7:0	TH0[7:0]	**Timer 0 High Byte.** The TH0 register is the high byte of the 16-bit Timer 0.

SFR Definition 31.7. TH1: Timer 1 High Byte

Bit	7	6	5	4	3	2	1	0
Name	TH1[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x8D; SFR Page = All Pages

Bit	Name	Function
7:0	TH1[7:0]	**Timer 1 High Byte.** The TH1 register is the high byte of the 16-bit Timer 1.

31.2. Timer 2

Timer 2 is a 16-bit timer formed by two 8-bit SFRs: TMR2L (low byte) and TMR2H (high byte). Timer 2 may operate in 16-bit auto-reload mode or (split) 8-bit auto-reload mode. The T2SPLIT bit (TMR2CN.3) defines the Timer 2 operation mode. Timer 2 can also be used in capture mode to capture rising edges of the Comparator 0 output.

Timer 2 may be clocked by the system clock, the system clock divided by 12, or the external oscillator source divided by 8. The external clock mode is ideal for real-time clock (RTC) functionality, where the internal oscillator drives the system clock while Timer 2 (and/or the PCA) is clocked by an external precision oscillator. The external oscillator source divided by 8 is synchronized with the system clock.

31.2.1. 16-bit Timer with Auto-Reload

When T2SPLIT (TMR2CN.3) is zero, Timer 2 operates as a 16-bit timer with auto-reload. Timer 2 can be clocked by SYSCLK, SYSCLK divided by 12, or the external oscillator clock source divided by 8. As the 16-bit timer register increments and overflows from 0xFFFF to 0x0000, the 16-bit value in the Timer 2 reload registers (TMR2RLH and TMR2RLL) is loaded into the Timer 2 register as shown in Figure 31.4, and the Timer 2 High Byte Overflow Flag (TMR2CN.7) is set. If Timer 2 interrupts are enabled (if IE.5 is set), an interrupt will be generated on each Timer 2 overflow. Additionally, if Timer 2 interrupts are enabled and the TF2LEN bit is set (TMR2CN.5), an interrupt will be generated each time the lower 8 bits (TMR2L) overflow from 0xFF to 0x00.

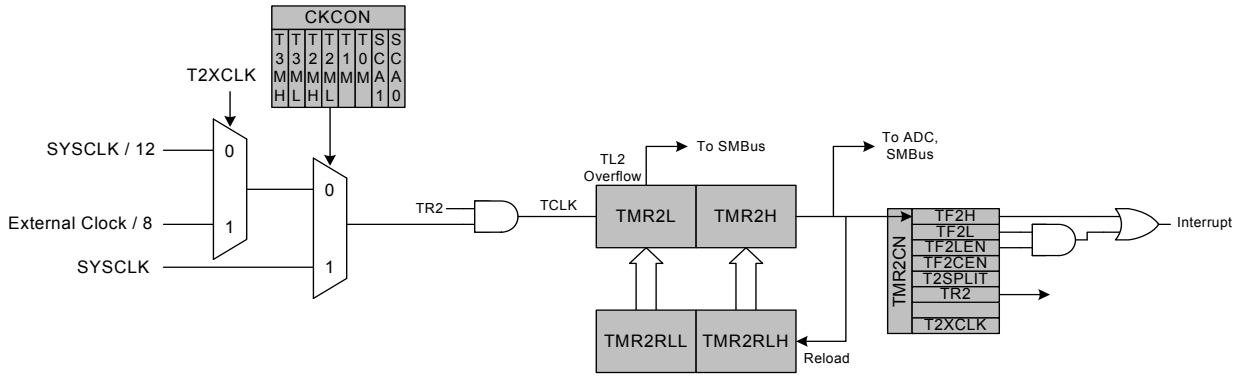


Figure 31.4. Timer 2 16-Bit Mode Block Diagram

31.2.2. 8-bit Timers with Auto-Reload

When T2SPLIT is set, Timer 2 operates as two 8-bit timers (TMR2H and TMR2L). Both 8-bit timers operate in auto-reload mode as shown in Figure 31.5. TMR2RLL holds the reload value for TMR2L; TMR2RLH holds the reload value for TMR2H. The TR2 bit in TMR2CN handles the run control for TMR2H. TMR2L is always running when configured for 8-bit Mode.

Each 8-bit timer may be configured to use SYSCLK, SYSCLK divided by 12, or the external oscillator clock source divided by 8. The Timer 2 Clock Select bits (T2MH and T2ML in CKCON) select either SYSCLK or the clock defined by the Timer 2 External Clock Select bit (T2XCLK in TMR2CN), as follows:

T2MH	T2XCLK	TMR2H Clock Source
0	0	SYSCLK / 12
0	1	External Clock / 8
1	X	SYSCLK

T2ML	T2XCLK	TMR2L Clock Source
0	0	SYSCLK / 12
0	1	External Clock / 8
1	X	SYSCLK

The TF2H bit is set when TMR2H overflows from 0xFF to 0x00; the TF2L bit is set when TMR2L overflows from 0xFF to 0x00. When Timer 2 interrupts are enabled (IE.5), an interrupt is generated each time TMR2H overflows. If Timer 2 interrupts are enabled and TF2LEN (TMR2CN.5) is set, an interrupt is generated each time either TMR2L or TMR2H overflows. When TF2LEN is enabled, software must check the TF2H and TF2L flags to determine the source of the Timer 2 interrupt. The TF2H and TF2L interrupt flags are not cleared by hardware and must be manually cleared by software.

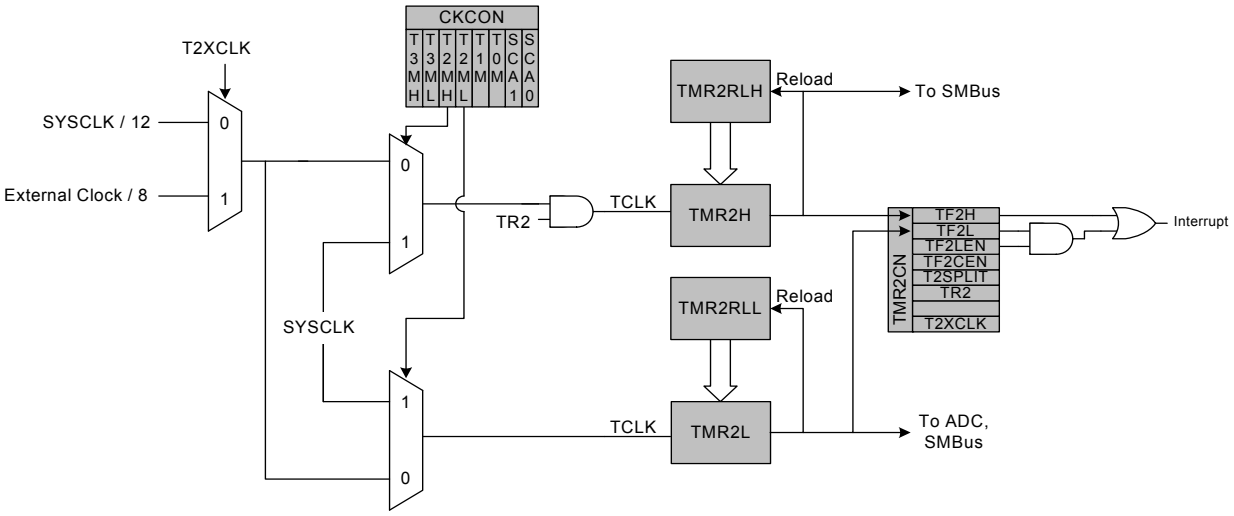


Figure 31.5. Timer 2 8-Bit Mode Block Diagram

31.2.3. Comparator 0 Capture Mode

The capture mode in Timer 2 allows Comparator 0 rising edges to be captured with the timer clocking from the system clock or the system clock divided by 12. Timer 2 capture mode is enabled by setting TF2CEN to 1 and T2SPLIT to 0.

When capture mode is enabled, a capture event will be generated on every Comparator 0 rising edge. When the capture event occurs, the contents of Timer 2 (TMR2H:TMR2L) are loaded into the Timer 2 reload registers (TMR2RLH:TMR2RLL) and the TF2H flag is set (triggering an interrupt if Timer 2 interrupts are enabled). By recording the difference between two successive timer capture values, the Comparator 0 period can be determined with respect to the Timer 2 clock. The Timer 2 clock should be much faster than the capture clock to achieve an accurate reading.

This mode allows software to determine the time between consecutive Comparator 0 rising edges, which can be used for detecting changes in the capacitance of a capacitive switch, or measuring the frequency of a low-level analog signal.

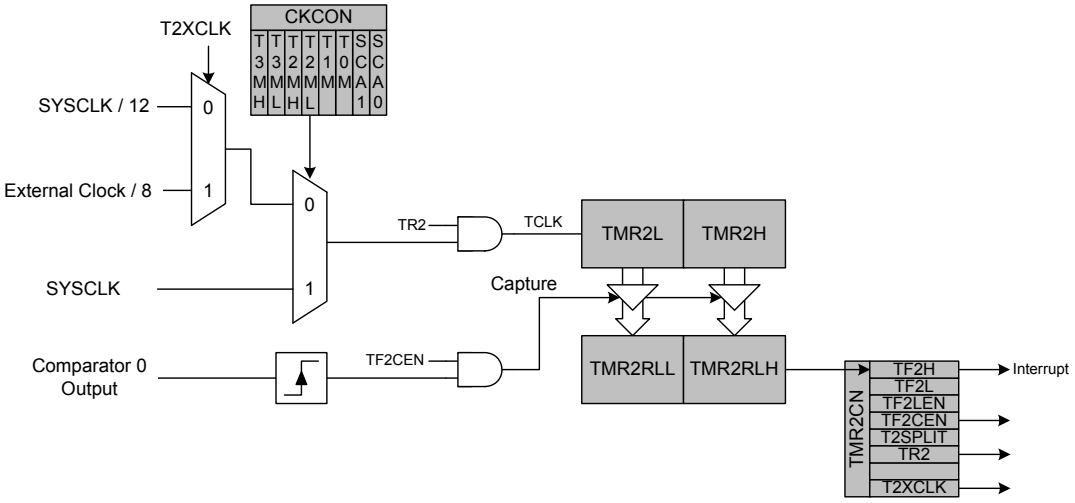


Figure 31.6. Timer 2 Capture Mode Block Diagram

SFR Definition 31.8. TMR2CN: Timer 2 Control

Bit	7	6	5	4	3	2	1	0
Name	TF2H	TF2L	TF2LEN	TF2CEN	T2SPLIT	TR2		T2XCLK
Type	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xC8; SFR Page = All Pages; Bit-Addressable

Bit	Name	Function
7	TF2H	**Timer 2 High Byte Overflow Flag.** Set by hardware when the Timer 2 high byte overflows from 0xFF to 0x00. In 16 bit mode, this will occur when Timer 2 overflows from 0xFFFF to 0x0000. When the Timer 2 interrupt is enabled, setting this bit causes the CPU to vector to the Timer 2 interrupt service routine. This bit is not automatically cleared by hardware.
6	TF2L	**Timer 2 Low Byte Overflow Flag.** 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.** 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 Comparator Capture Enable.** When set to 1, this bit enables Timer 2 Comparator Capture Mode. If TF2CEN is set, on a rising edge of the Comparator0 output the current 16-bit timer value in TMR2H:TMR2L will be copied to TMR2RLH:TMR2RLL. If Timer 2 interrupts are also enabled, an interrupt will be generated on this event.
3	T2SPLIT	**Timer 2 Split Mode Enable.** When this bit is set, Timer 2 operates as two 8-bit timers with auto-reload. 0: Timer 2 operates in 16-bit auto-reload mode. 1: Timer 2 operates as two 8-bit auto-reload timers.
2	TR2	**Timer 2 Run Control.** 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	Unused	Read = 0b; Write = Don't Care.
0	T2XCLK	**Timer 2 External Clock Select.** This bit selects the external clock source for Timer 2. If Timer 2 is in 8-bit mode, this bit selects the external oscillator clock source for both timer bytes. However, the Timer 2 Clock Select bits (T2MH and T2ML in register CKCON) may still be used to select between the external clock and the system clock for either timer. 0: Timer 2 clock is the system clock divided by 12. 1: Timer 2 clock is the external clock divided by 8 (synchronized with SYSCLK).

SFR Definition 31.9. TMR2RLL: Timer 2 Reload Register Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR2RLL[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xCA; SFR Page = 0

Bit	Name	Function
7:0	TMR2RLL[7:0]	**Timer 2 Reload Register Low Byte.** TMR2RLL holds the low byte of the reload value for Timer 2.

SFR Definition 31.10. TMR2RLH: Timer 2 Reload Register High Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR2RLH[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xCB; SFR Page = 0

Bit	Name	Function
7:0	TMR2RLH[7:0]	**Timer 2 Reload Register High Byte.** TMR2RLH holds the high byte of the reload value for Timer 2.

SFR Definition 31.11. TMR2L: Timer 2 Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR2L[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xCC; SFR Page = 0

Bit	Name	Function
7:0	TMR2L[7:0]	**Timer 2 Low Byte.** In 16-bit mode, the TMR2L register contains the low byte of the 16-bit Timer 2. In 8-bit mode, TMR2L contains the 8-bit low byte timer value.

SFR Definition 31.12. TMR2H Timer 2 High Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR2H[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xCD; SFR Page = 0

Bit	Name	Function
7:0	TMR2H[7:0]	**Timer 2 Low Byte.** In 16-bit mode, the TMR2H register contains the high byte of the 16-bit Timer 2. In 8-bit mode, TMR2H contains the 8-bit high byte timer value.

31.3. Timer 3

Timer 3 is a 16-bit timer formed by two 8-bit SFRs: TMR3L (low byte) and TMR3H (high byte). Timer 3 may operate in 16-bit auto-reload mode or (split) 8-bit auto-reload mode. The T3SPLIT bit (TMR3CN.3) defines the Timer 3 operation mode.

Timer 3 may be clocked by the system clock, the system clock divided by 12, or the external oscillator source divided by 8. The external clock mode is ideal for real-time clock (RTC) functionality, where the internal high-frequency oscillator drives the system clock while Timer 3 is clocked by an external oscillator source. The external oscillator source divided by 8 is synchronized with the system clock when in all operating modes except suspend. When the internal oscillator is placed in suspend mode, The external clock/8 signal can directly drive the timer. This allows the use of an external clock to wake up the device from suspend mode. The timer will continue to run in suspend mode and count up. When the timer overflow occurs, the device will wake from suspend mode, and begin executing code again. The timer value may be set prior to entering suspend, to overflow in the desired amount of time (number of clocks) to wake the device. If a wake-up source other than the timer wakes the device from suspend mode, it may take up to three timer clocks before the timer registers can be read or written. During this time, the STSYNC bit in register OSCICN will be set to 1, to indicate that it is not safe to read or write the timer registers.

31.3.1. 16-bit Timer with Auto-Reload

When T3SPLIT (TMR3CN.3) is zero, Timer 3 operates as a 16-bit timer with auto-reload. Timer 3 can be clocked by SYSCLK, SYSCLK divided by 12, or the external oscillator clock source divided by 8. As the 16-bit timer register increments and overflows from 0xFFFF to 0x0000, the 16-bit value in the Timer 3 reload registers (TMR3RLH and TMR3RLL) is loaded into the Timer 3 register as shown in Figure 31.7, and the Timer 3 High Byte Overflow Flag (TMR3CN.7) is set. If Timer 3 interrupts are enabled (if EIE1.7 is set), an interrupt will be generated on each Timer 3 overflow. Additionally, if Timer 3 interrupts are enabled and the TF3LEN bit is set (TMR3CN.5), an interrupt will be generated each time the lower 8 bits (TMR3L) overflow from 0xFF to 0x00.

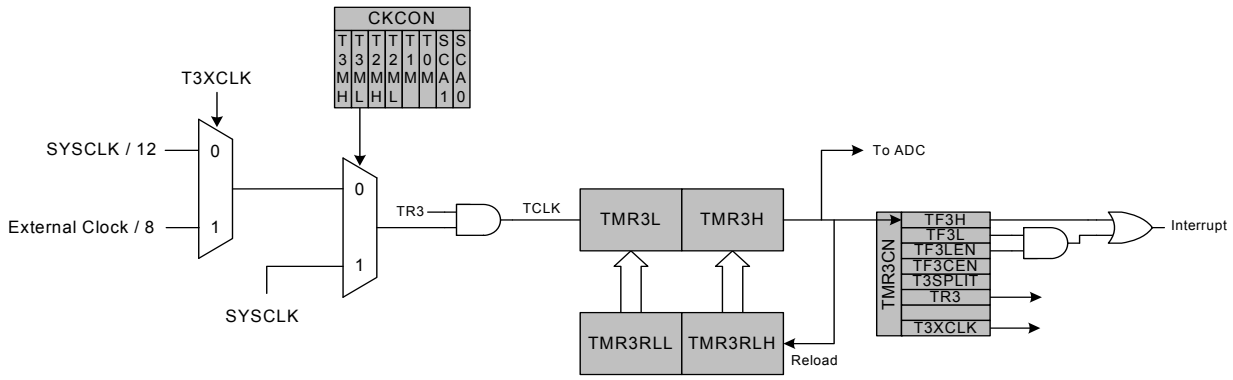


Figure 31.7. Timer 3 16-Bit Mode Block Diagram

31.3.2. 8-bit Timers with Auto-Reload

When T3SPLIT is set, Timer 3 operates as two 8-bit timers (TMR3H and TMR3L). Both 8-bit timers operate in auto-reload mode as shown in Figure 31.8. TMR3RLL holds the reload value for TMR3L; TMR3RLH holds the reload value for TMR3H. The TR3 bit in TMR3CN handles the run control for TMR3H. TMR3L is always running when configured for 8-bit Mode. Timer 3 can also be used in capture mode to capture rising edges of the Comparator 0 output.

Each 8-bit timer may be configured to use SYSCLK, SYSCLK divided by 12, or the external oscillator clock source divided by 8. The Timer 3 clock select bits (T3MH and T3ML in CKCON) select either SYSCLK or the clock defined by the Timer 3 External Clock Select bits (T3XCLK in TMR3CN), as follows:

T3MH	T3XCLK	TMR3H Clock Source
0	0	SYSCLK / 12
0	1	External Clock / 8
1	X	SYSCLK

T3ML	T3XCLK	TMR3L Clock Source
0	0	SYSCLK / 12
0	1	External Clock / 8
1	X	SYSCLK

The TF3H bit is set when TMR3H overflows from 0xFF to 0x00; the TF3L bit is set when TMR3L overflows from 0xFF to 0x00. When Timer 3 interrupts are enabled, an interrupt is generated each time TMR3H overflows. If Timer 3 interrupts are enabled and TF3LEN (TMR3CN.5) is set, an interrupt is generated each time either TMR3L or TMR3H overflows. When TF3LEN is enabled, software must check the TF3H and TF3L flags to determine the source of the Timer 3 interrupt. The TF3H and TF3L interrupt flags are not cleared by hardware and must be manually cleared by software.

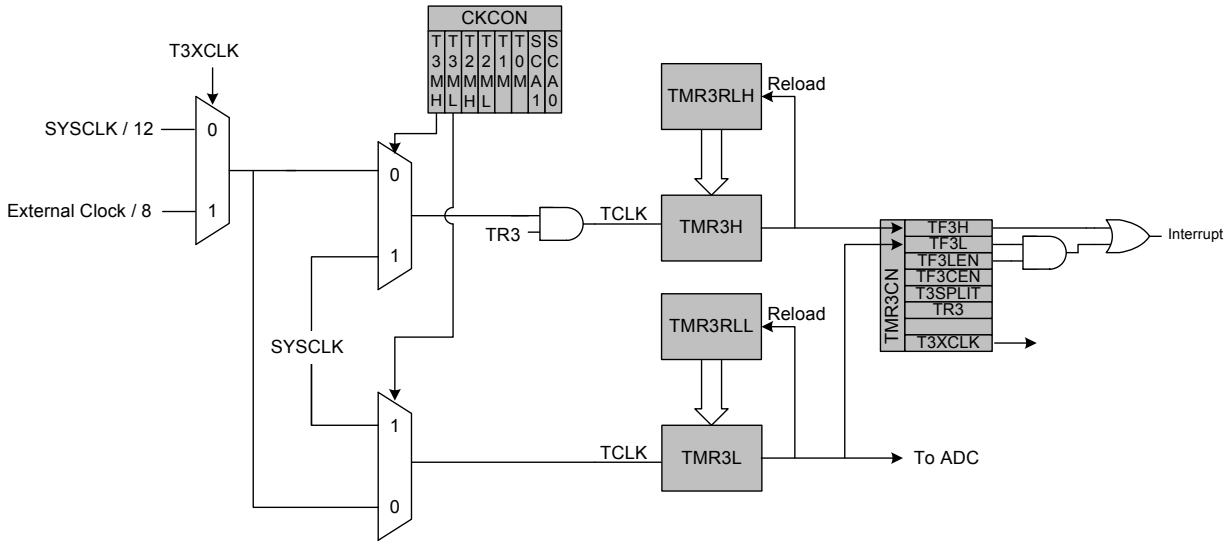


Figure 31.8. Timer 3 8-Bit Mode Block Diagram

31.3.3. Comparator 0 Capture Mode

The capture mode in Timer 3 allows Comparator 0 rising edges to be captured with the timer clocking from the system clock or the system clock divided by 12. Timer 3 capture mode is enabled by setting TF3CEN to 1 and T3SPLIT to 0.

When capture mode is enabled, a capture event will be generated on every Comparator 0 rising edge. When the capture event occurs, the contents of Timer 3 (TMR3H:TMR3L) are loaded into the Timer 3 reload registers (TMR3RLH:TMR3RLL) and the TF3H flag is set (triggering an interrupt if Timer 3 interrupts are enabled). By recording the difference between two successive timer capture values, the Comparator 0 period can be determined with respect to the Timer 3 clock. The Timer 3 clock should be much faster than the capture clock to achieve an accurate reading.

This mode allows software to determine the time between consecutive Comparator 0 rising edges, which can be used for detecting changes in the capacitance of a capacitive switch, or measuring the frequency of a low-level analog signal.

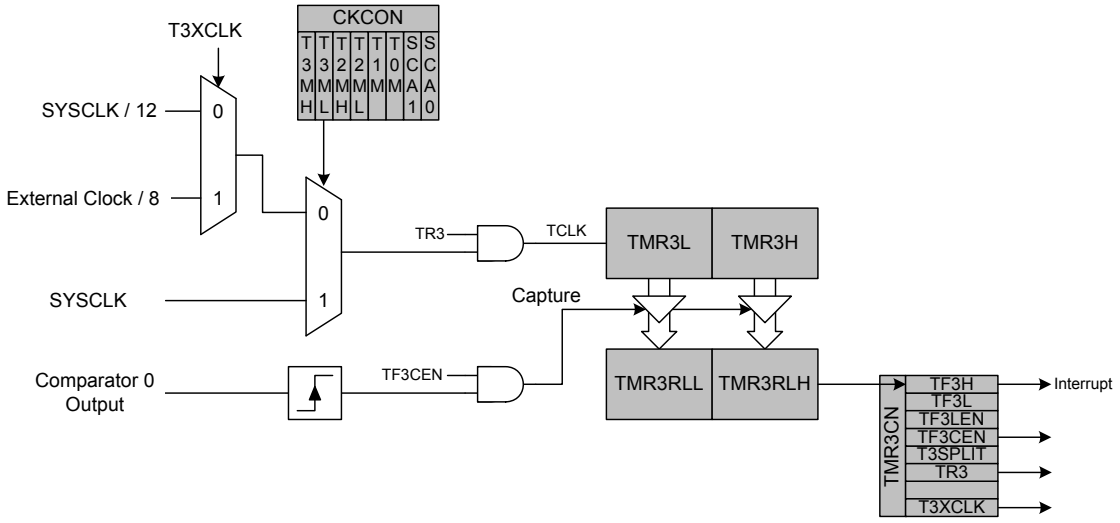


Figure 31.9. Timer 3 Capture Mode Block Diagram

SFR Definition 31.13. TMR3CN: Timer 3 Control

Bit	7	6	5	4	3	2	1	0
Name	TF3H	TF3L	TF3LEN	TF3CEN	T3SPLIT	TR3		T3XCLK
Type	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x91; SFR Page = 0

Bit	Name	Function
7	TF3H	**Timer 3 High Byte Overflow Flag.** Set by hardware when the Timer 3 high byte overflows from 0xFF to 0x00. In 16 bit mode, this will occur when Timer 3 overflows from 0xFFFF to 0x0000. When the Timer 3 interrupt is enabled, setting this bit causes the CPU to vector to the Timer 3 interrupt service routine. This bit is not automatically cleared by hardware.
6	TF3L	**Timer 3 Low Byte Overflow Flag.** 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.** When set to 1, this bit enables Timer 3 Low Byte interrupts. If Timer 3 interrupts are also enabled, an interrupt will be generated when the low byte of Timer 3 overflows.
4	TF3CEN	**Timer 3 Comparator Capture Enable.** When set to 1, this bit enables Timer 3 Comparator Capture Mode. If TF3CEN is set, on a rising edge of the Comparator0 output the current 16-bit timer value in TMR3H:TMR3L will be copied to TMR3RLH:TMR3RLL. If Timer 3 interrupts are also enabled, an interrupt will be generated on this event.
3	T3SPLIT	**Timer 3 Split Mode Enable.** When this bit is set, Timer 3 operates as two 8-bit timers with auto-reload. 0: Timer 3 operates in 16-bit auto-reload mode. 1: Timer 3 operates as two 8-bit auto-reload timers.
2	TR3	**Timer 3 Run Control.** 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	Unused	Read = 0b; Write = Don't Care.
0	T3XCLK	**Timer 3 External Clock Select.** This bit selects the external clock source for Timer 3. If Timer 3 is in 8-bit mode, this bit selects the external oscillator clock source for both timer bytes. However, the Timer 3 Clock Select bits (T3MH and T3ML in register CKCON) may still be used to select between the external clock and the system clock for either timer. 0: System clock divided by 12. 1: External clock divided by 8 (synchronized with SYSCLK when not in suspend).

SFR Definition 31.14. TMR3RLL: Timer 3 Reload Register Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR3RLL[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x92; SFR Page = 0

Bit	Name	Function
7:0	TMR3RLL[7:0]	**Timer 3 Reload Register Low Byte.** TMR3RLL holds the low byte of the reload value for Timer 3.

SFR Definition 31.15. TMR3RLH: Timer 3 Reload Register High Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR3RLH[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x93; SFR Page = 0

Bit	Name	Function
7:0	TMR3RLH[7:0]	**Timer 3 Reload Register High Byte.** TMR3RLH holds the high byte of the reload value for Timer 3.

SFR Definition 31.16. TMR3L: Timer 3 Low Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR3L[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x94; SFR Page = 0

Bit	Name	Function
7:0	TMR3L[7:0]	**Timer 3 Low Byte.** In 16-bit mode, the TMR3L register contains the low byte of the 16-bit Timer 3. In 8-bit mode, TMR3L contains the 8-bit low byte timer value.

SFR Definition 31.17. TMR3H Timer 3 High Byte

Bit	7	6	5	4	3	2	1	0
Name	TMR3H[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

SFR Address = 0x95; SFR Page = 0

Bit	Name	Function
7:0	TMR3H[7:0]	**Timer 3 High Byte.** In 16-bit mode, the TMR3H register contains the high byte of the 16-bit Timer 3. In 8-bit mode, TMR3H contains the 8-bit high byte timer value.

32. Programmable Counter Array

The Programmable Counter Array (PCA0) provides enhanced timer functionality while requiring less CPU intervention than the standard 8051 counter/timers. The PCA consists of a dedicated 16-bit counter/timer and three 16-bit capture/compare modules. Each capture/compare module has its own associated I/O line (CEXn) which is routed through the Crossbar to Port I/O when enabled. The counter/timer is driven by a programmable timebase that can select between seven sources: system clock, system clock divided by four, system clock divided by twelve, the external oscillator clock source divided by 8, Timer 0 overflows, or an external clock signal on the ECI input pin. Each capture/compare module may be configured to operate independently in one of six modes: Edge-Triggered Capture, Software Timer, High-Speed Output, Frequency Output, 8 to 11-Bit PWM, or 16-Bit PWM (each mode is described in Section "32.3. Capture/Compare Modules" on page 268). The external oscillator clock option is ideal for real-time clock (RTC) functionality, allowing the PCA to be clocked by a precision external oscillator while the internal oscillator drives the system clock. The PCA is configured and controlled through the system controller's Special Function Registers. The PCA block diagram is shown in Figure 32.1

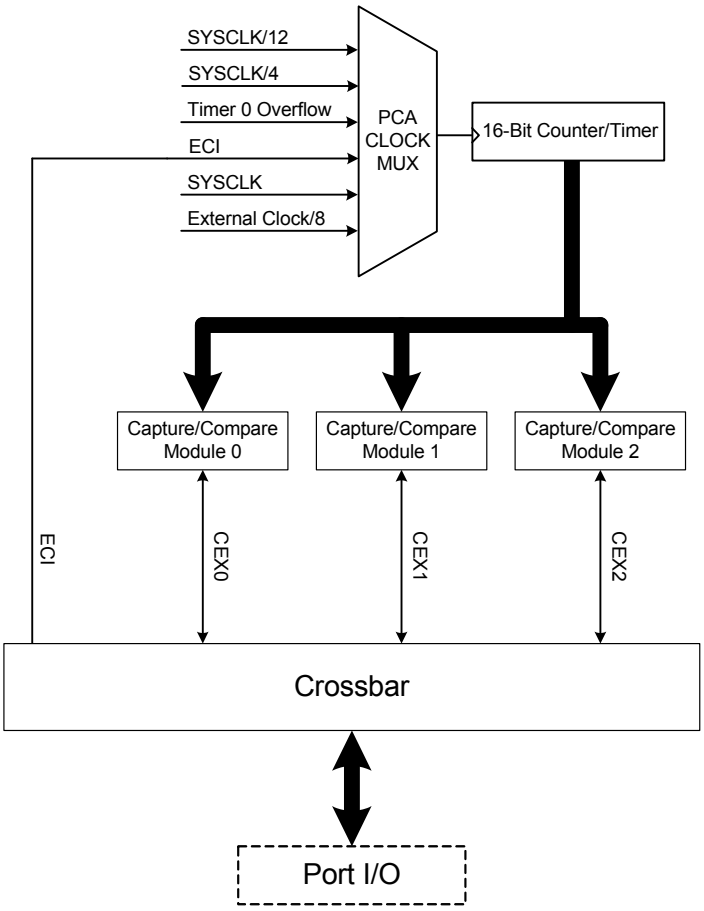


Figure 32.1. PCA Block Diagram

32.1. PCA Counter/Timer

The 16-bit PCA counter/timer consists of two 8-bit SFRs: PCA0L and PCA0H. PCA0H is the high byte (MSB) of the 16-bit counter/timer and PCA0L is the low byte (LSB). Reading PCA0L automatically latches the value of PCA0H into a “snapshot” register; the following PCA0H read accesses this “snapshot” register. **Reading the PCA0L Register first guarantees an accurate reading of the entire 16-bit PCA0 counter.** Reading PCA0H or PCA0L does not disturb the counter operation. The CPS2–CPS0 bits in the PCA0MD register select the timebase for the counter/timer as shown in Table 32.1.

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 32.1. 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	x	Reserved
Note: External oscillator source divided by 8 is synchronized with the system clock.			

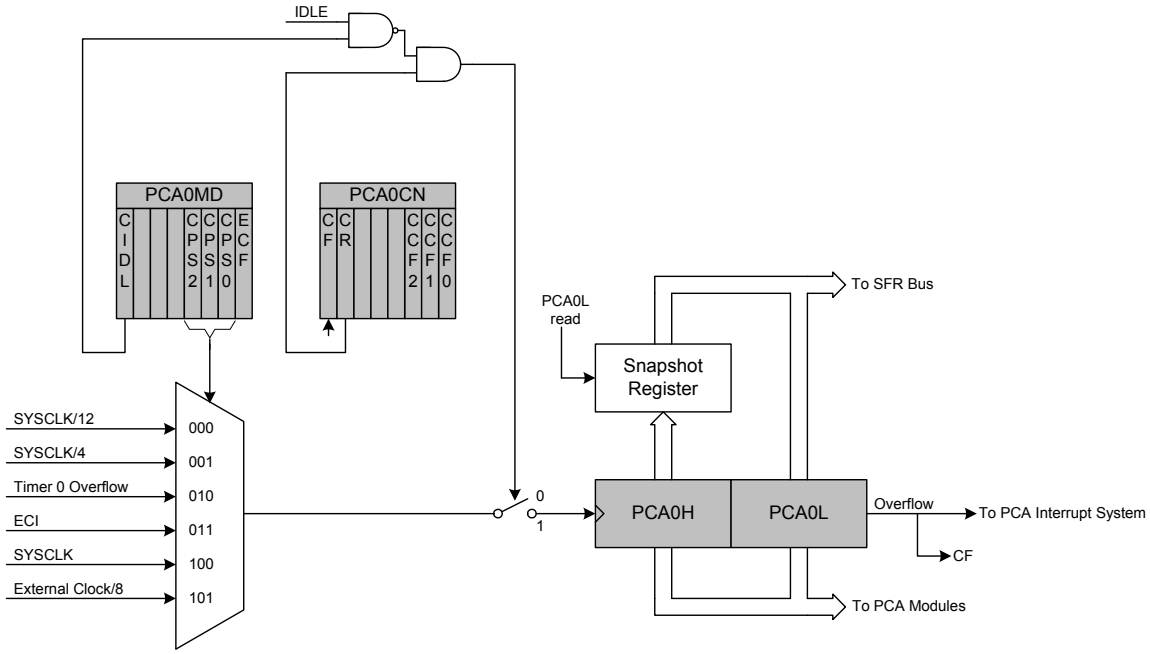


Figure 32.2. PCA Counter/Timer Block Diagram

32.2. PCA0 Interrupt Sources

Figure 32.3 shows a diagram of the PCA interrupt tree. There are eight independent event flags that can be used to generate a PCA0 interrupt. They are: the main PCA counter overflow flag (CF), which is set upon a 16-bit overflow of the PCA0 counter, an intermediate overflow flag (COVF), which can be set on an overflow from the 8th, 9th, 10th, or 11th bit of the PCA0 counter, and the individual flags for each PCA channel (CCF0, CCF1, and CCF2), which are set according to the operation mode of that module. These event flags are always set when the trigger condition occurs. Each of these flags can be individually selected to generate a PCA0 interrupt, using the corresponding interrupt enable flag (ECF for CF, ECOV for COVF, and ECCFn for each CCFn). PCA0 interrupts must be globally enabled before any individual interrupt sources are recognized by the processor. PCA0 interrupts are globally enabled by setting the EA bit and the EPCA0 bit to logic 1.

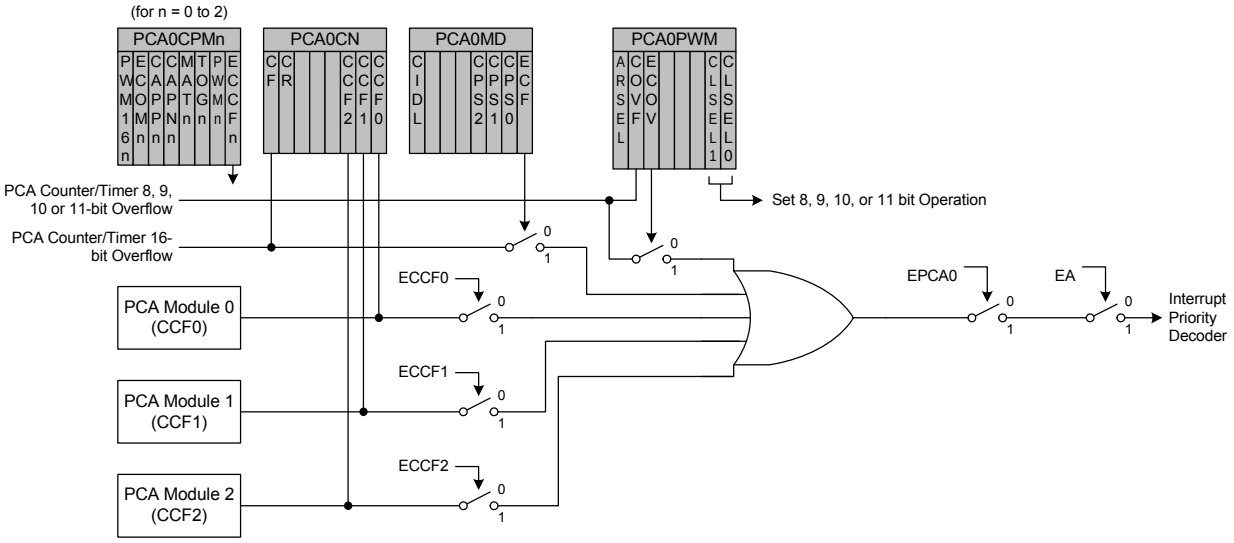


Figure 32.3. PCA Interrupt Block Diagram

32.3. Capture/Compare Modules

Each module can be configured to operate independently in one of six operation modes: Edge-triggered Capture, Software Timer, High Speed Output, Frequency Output, 8 to 11-Bit Pulse Width Modulator, or 16-Bit Pulse Width Modulator. Each module has Special Function Registers (SFRs) associated with it in the CIP-51 system controller. These registers are used to exchange data with a module and configure the module's mode of operation. Table 32.2 summarizes the bit settings in the PCA0CPMn and PCA0PWM registers used to select the PCA capture/compare module’s operating mode. 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 32.2. PCA0CPM and PCA0PWM Bit Settings for PCA Capture/Compare Modules

Operational Mode	**PCA0CPMn**								**PCA0PWM**				
Bit Number	7	6	5	4	3	2	1	0	7	6	5	4-2	1–0
Capture triggered by positive edge on CEXn	X	X	1	0	0	0	0	A	0	X	B	XXX	XX
Capture triggered by negative edge on CEXn	X	X	0	1	0	0	0	A	0	X	B	XXX	XX
Capture triggered by any transition on CEXn	X	X	1	1	0	0	0	A	0	X	B	XXX	XX
Software Timer	X	C	0	0	1	0	0	A	0	X	B	XXX	XX
High Speed Output	X	C	0	0	1	1	0	A	0	X	B	XXX	XX
Frequency Output	X	C	0	0	0	1	1	A	0	X	B	XXX	XX
8-Bit Pulse Width Modulator (Note 7)	0	C	0	0	E	0	1	A	0	X	B	XXX	00
9-Bit Pulse Width Modulator (Note 7)	0	C	0	0	E	0	1	A	D	X	B	XXX	01
10-Bit Pulse Width Modulator (Note 7)	0	C	0	0	E	0	1	A	D	X	B	XXX	10
11-Bit Pulse Width Modulator (Note 7)	0	C	0	0	E	0	1	A	D	X	B	XXX	11
16-Bit Pulse Width Modulator	1	C	0	0	E	0	1	A	0	X	B	XXX	XX

Notes:

1. X = Don't Care (no functional difference for individual module if 1 or 0).
2. A = Enable interrupts for this module (PCA interrupt triggered on CCFn set to 1).
3. B = Enable 8th, 9th, 10th or 11th bit overflow interrupt (Depends on setting of CLSEL[1:0]).
4. C = When set to 0, the digital comparator is off. For high speed and frequency output modes, the associated pin will not toggle. In any of the PWM modes, this generates a 0% duty cycle (output = 0).
5. D = Selects whether the Capture/Compare register (0) or the Auto-Reload register (1) for the associated channel is accessed via addresses PCA0CPHn and PCA0CPLn.
6. E = When set, a match event will cause the CCFn flag for the associated channel to be set.
7. All modules set to 8, 9, 10 or 11-bit PWM mode use the same cycle length setting.

32.3.1. Edge-triggered Capture Mode

In this mode, a valid transition on the CEXn pin causes the PCA to capture the value of the PCA counter/timer and load it into the corresponding module's 16-bit capture/compare register (PCA0CPLn and PCA0CPHn). The CAPPn and CAPNn bits in the PCA0CPMn register are used to select the type of transition that triggers the capture: low-to-high transition (positive edge), high-to-low transition (negative edge), or either transition (positive or negative edge). When a capture occurs, the Capture/Compare Flag (CCFn) in PCA0CN is set to logic 1. An interrupt request is generated if the CCFn interrupt for that module is enabled. The CCFn bit is not automatically cleared by hardware when the CPU vectors to the interrupt service routine, and must be cleared by software. If both CAPPn and CAPNn bits are set to logic 1, then the state of the Port pin associated with CEXn can be read directly to determine whether a rising-edge or falling-edge caused the capture.

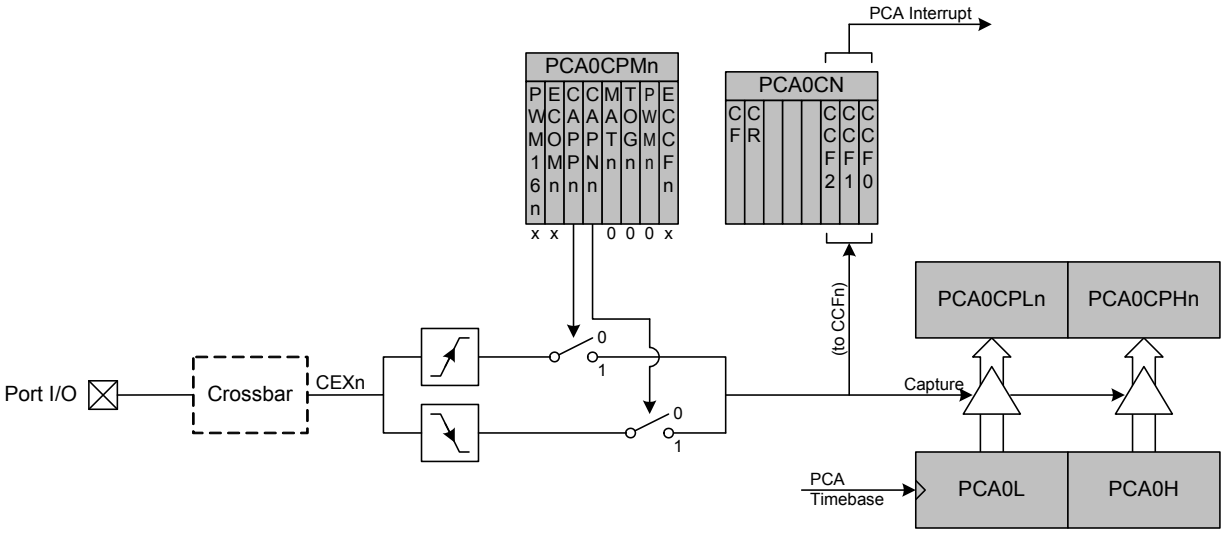


Figure 32.4. PCA Capture Mode Diagram

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

32.3.2. Software Timer (Compare) Mode

In Software Timer mode, the PCA counter/timer value is compared to the module's 16-bit capture/compare register (PCA0CPHn and PCA0CPLn). When a match occurs, the Capture/Compare Flag (CCFn) in PCA0CN is set to logic 1. An interrupt request is generated if the CCFn interrupt for that module is enabled. The CCFn bit is not automatically cleared by hardware when the CPU vectors to the interrupt service routine, and must be cleared by software. Setting the ECOMn and MATn bits in the PCA0CPMn register enables Software Timer mode.

Important Note About Capture/Compare Registers: When writing a 16-bit value to the PCA0 Capture/Compare registers, the low byte should always be written first. Writing to PCA0CPLn clears the ECOMn bit to 0; writing to PCA0CPHn sets ECOMn to 1.

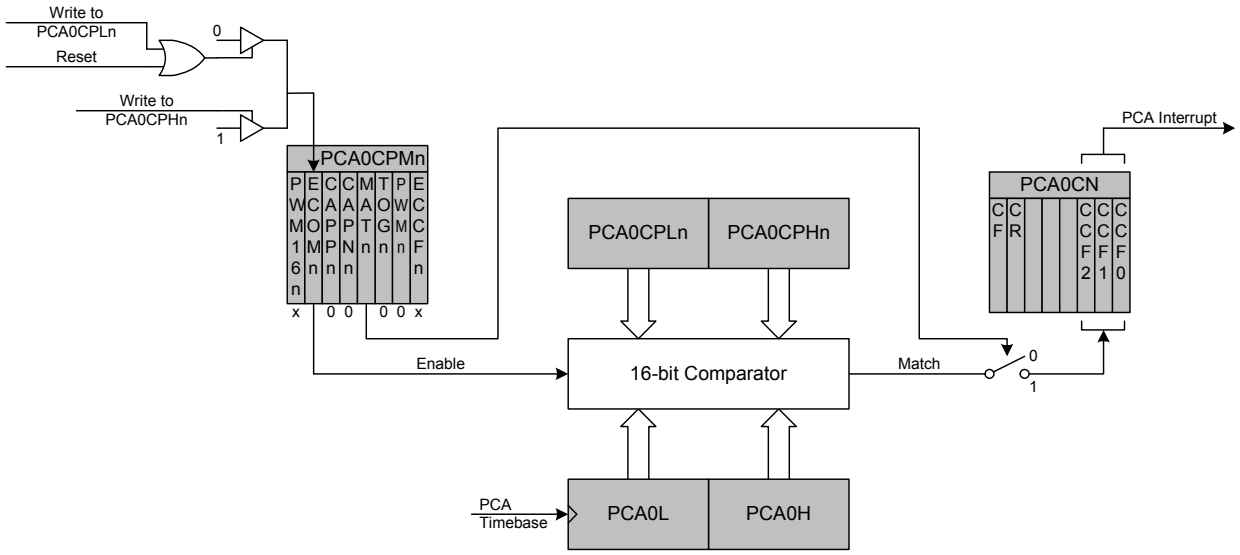


Figure 32.5. PCA Software Timer Mode Diagram

32.3.3. High-Speed Output Mode

In High-Speed Output mode, a module's associated CEXn pin is toggled each time a match occurs between the PCA Counter and the module's 16-bit capture/compare register (PCA0CPHn and PCA0CPLn). When a match occurs, the Capture/Compare Flag (CCFn) in PCA0CN is set to logic 1. An interrupt request is generated if the CCFn interrupt for that module is enabled. The CCFn bit is not automatically cleared by hardware when the CPU vectors to the interrupt service routine, and must be cleared by software. Setting the TOGn, MATn, and ECOMn bits in the PCA0CPMn register enables the High-Speed Output mode. If ECOMn is cleared, the associated pin will retain its state, and not toggle on the next match event.

Important Note About Capture/Compare Registers: When writing a 16-bit value to the PCA0 Capture/Compare registers, the low byte should always be written first. Writing to PCA0CPLn clears the ECOMn bit to 0; writing to PCA0CPHn sets ECOMn to 1.

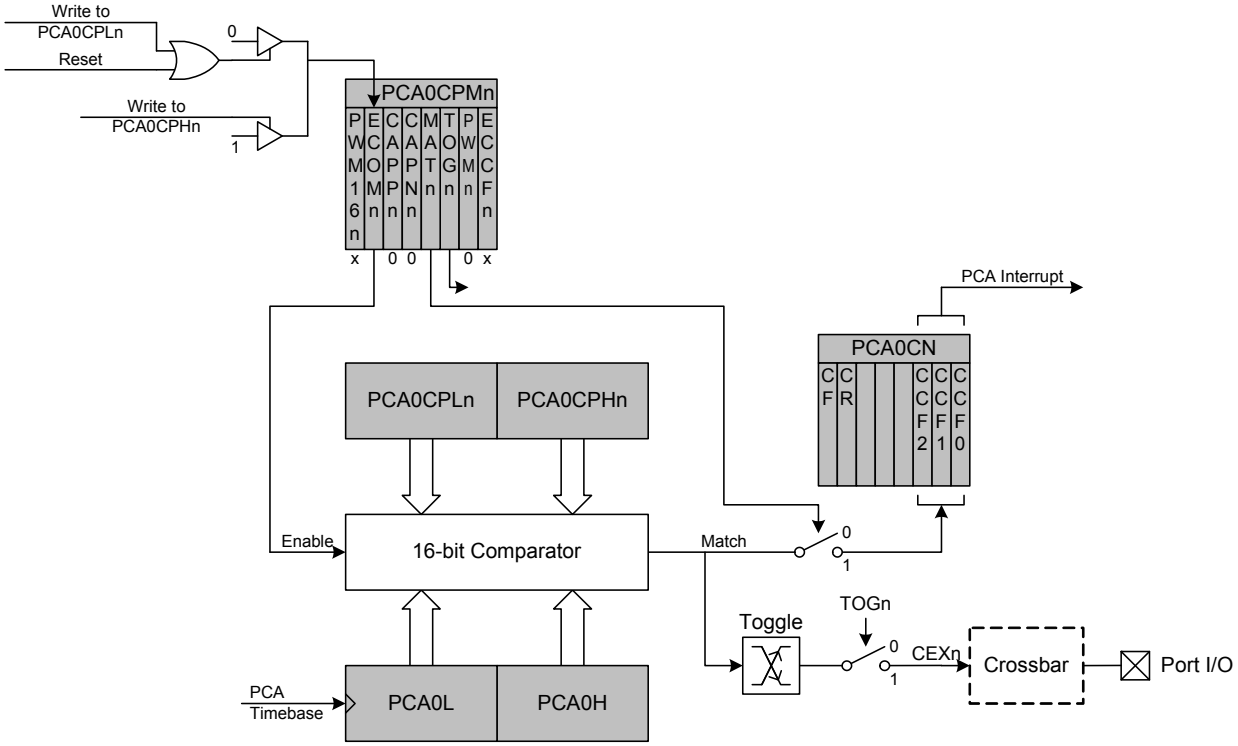


Figure 32.6. PCA High-Speed Output Mode Diagram

32.3.4. Frequency Output Mode

Frequency Output Mode produces a programmable-frequency square wave on the module's associated CEXn pin. The capture/compare module high byte holds the number of PCA clocks to count before the output is toggled. The frequency of the square wave is then defined by Equation 32.1.

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

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

Equation 32.1. Square Wave Frequency Output

Where F_{PCA} is the frequency of the clock selected by the CPS2–0 bits in the PCA mode register, PCA0MD. The lower byte of the capture/compare module is compared to the PCA counter low byte; on a match, CEXn is toggled and the offset held in the high byte is added to the matched value in PCA0CPLn. Frequency Output Mode is enabled by setting the ECOMn, TOGn, and PWMn bits in the PCA0CPMn register. 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.

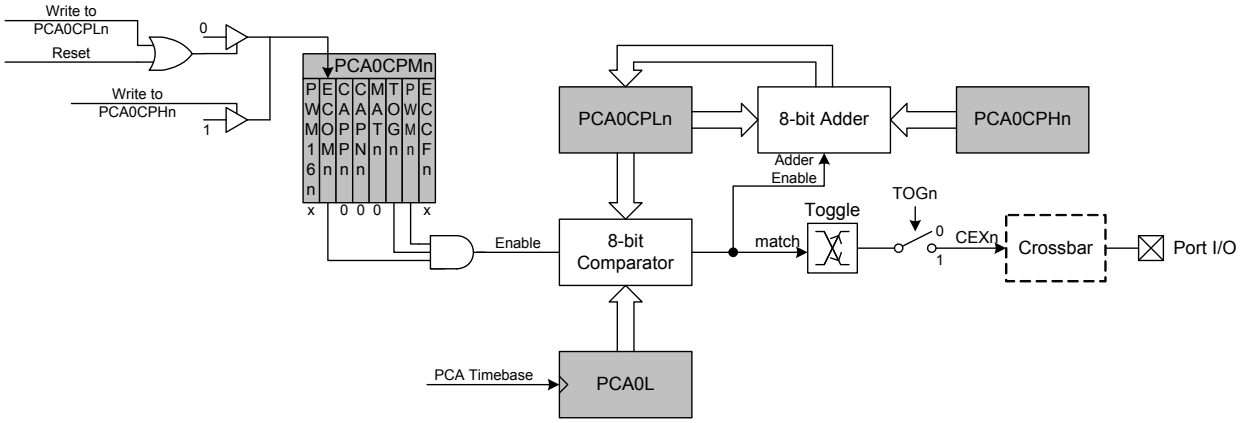


Figure 32.7. PCA Frequency Output Mode

32.3.5. 8-bit, 9-bit, 10-bit and 11-bit Pulse Width Modulator Modes

Each module can be used independently to generate a pulse width modulated (PWM) output on its associated CEXn pin. The frequency of the output is dependent on the timebase for the PCA counter/timer, and the setting of the PWM cycle length (8, 9, 10 or 11-bits). For backwards-compatibility with the 8-bit PWM mode available on other devices, the 8-bit PWM mode operates slightly different than 9, 10 and 11-bit PWM modes. **It is important to note that all channels configured for 8/9/10/11-bit PWM mode will use the same cycle length.** It is not possible to configure one channel for 8-bit PWM mode and another for 11-bit mode (for example). However, other PCA channels can be configured to Pin Capture, High-Speed Output, Software Timer, Frequency Output, or 16-bit PWM mode independently.

32.3.5.1. 8-bit Pulse Width Modulator Mode

The duty cycle of the PWM output signal in 8-bit PWM mode is varied using the module's PCA0CPLn capture/compare register. When the value in the low byte of the PCA counter/timer (PCA0L) is equal to the value in PCA0CPLn, the output on the CEXn pin will be set. When the count value in PCA0L overflows, the CEXn output will be reset (see Figure 32.8). Also, when the counter/timer low byte (PCA0L) overflows from 0xFF to 0x00, PCA0CPLn is reloaded automatically with the value stored in the module’s capture/compare high byte (PCA0CPHn) without software intervention. Setting the ECOMn and PWMn bits in the PCA0CPMn register, and setting the CLSEL bits in register PCA0PWM to 00b enables 8-Bit Pulse Width Modulator mode. If the MATn bit is set to 1, the CCFn flag for the module will be set each time an 8-bit comparator match (rising edge) occurs. The COVF flag in PCA0PWM can be used to detect the overflow (falling edge), which will occur every 256 PCA clock cycles. The duty cycle for 8-Bit PWM Mode is given in Equation 32.2.

Important Note About Capture/Compare Registers: When writing a 16-bit value to the PCA0 Capture/Compare registers, the low byte should always be written first. Writing to PCA0CPLn clears the ECOMn bit to 0; writing to PCA0CPHn sets ECOMn to 1.

$$\text{Duty Cycle} = \frac{(256 - PCA0CPHn)}{256}$$

Equation 32.2. 8-Bit PWM Duty Cycle

Using Equation 32.2, the largest duty cycle is 100% (PCA0CPHn = 0), and the smallest duty cycle is 0.39% (PCA0CPHn = 0xFF). A 0% duty cycle may be generated by clearing the ECOMn bit to 0.

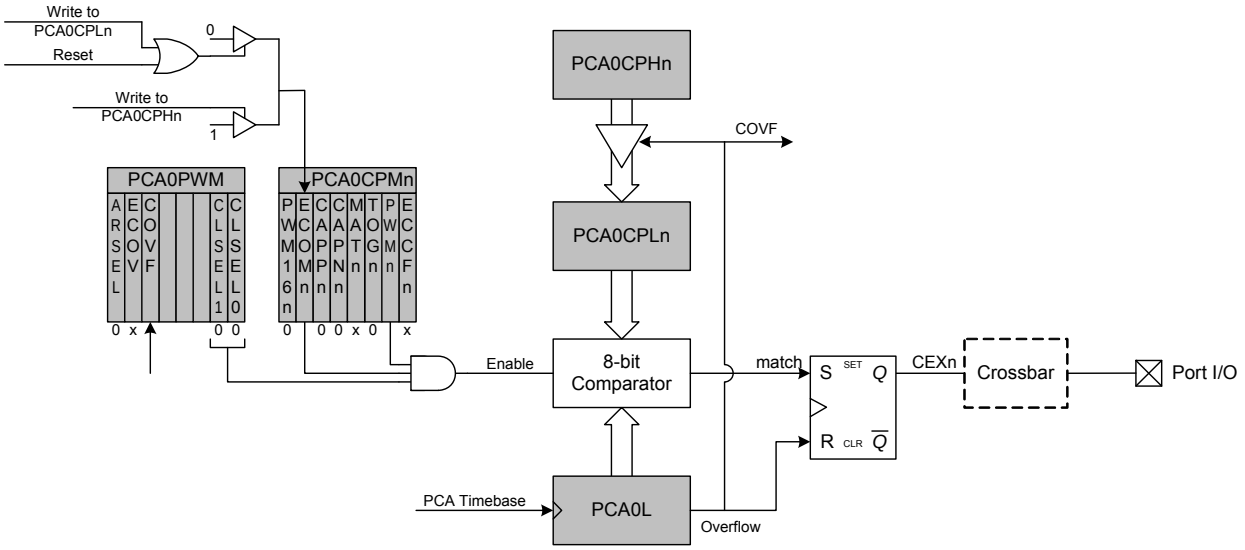


Figure 32.8. PCA 8-Bit PWM Mode Diagram

32.3.5.2. 9/10/11-bit Pulse Width Modulator Mode

The duty cycle of the PWM output signal in 9/10/11-bit PWM mode should be varied by writing to an “Auto-Reload” Register, which is dual-mapped into the PCA0CPHn and PCA0CPLn register locations. The data written to define the duty cycle should be right-justified in the registers. The auto-reload registers are accessed (read or written) when the bit ARSEL in PCA0PWM is set to 1. The capture/compare registers are accessed when ARSEL is set to 0.

When the least-significant N bits of the PCA0 counter match the value in the associated module’s capture/compare register (PCA0CPn), the output on CEXn is asserted high. When the counter overflows from the Nth bit, CEXn is asserted low (see Figure 32.9). Upon an overflow from the Nth bit, the COVF flag is set, and the value stored in the module’s auto-reload register is loaded into the capture/compare register. The value of N is determined by the CLSEL bits in register PCA0PWM.

The 9, 10 or 11-bit PWM mode is selected by setting the ECOMn and PWMn bits in the PCA0CPMn register, and setting the CLSEL bits in register PCA0PWM to the desired cycle length (other than 8-bits). If the MATn bit is set to 1, the CCFn flag for the module will be set each time a comparator match (rising edge) occurs. The COVF flag in PCA0PWM can be used to detect the overflow (falling edge), which will occur every 512 (9-bit), 1024 (10-bit) or 2048 (11-bit) PCA clock cycles. The duty cycle for 9/10/11-Bit PWM Mode is given in Equation 32.2, where N is the number of bits in the PWM cycle.

Important Note About PCA0CPHn and PCA0CPLn Registers: When writing a 16-bit value to the PCA0CPn registers, the low byte should always be written first. Writing to PCA0CPLn clears the ECOMn bit to 0; writing to PCA0CPHn sets ECOMn to 1.

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

Equation 32.3. 9, 10, and 11-Bit PWM Duty Cycle

A 0% duty cycle may be generated by clearing the ECOMn bit to 0.

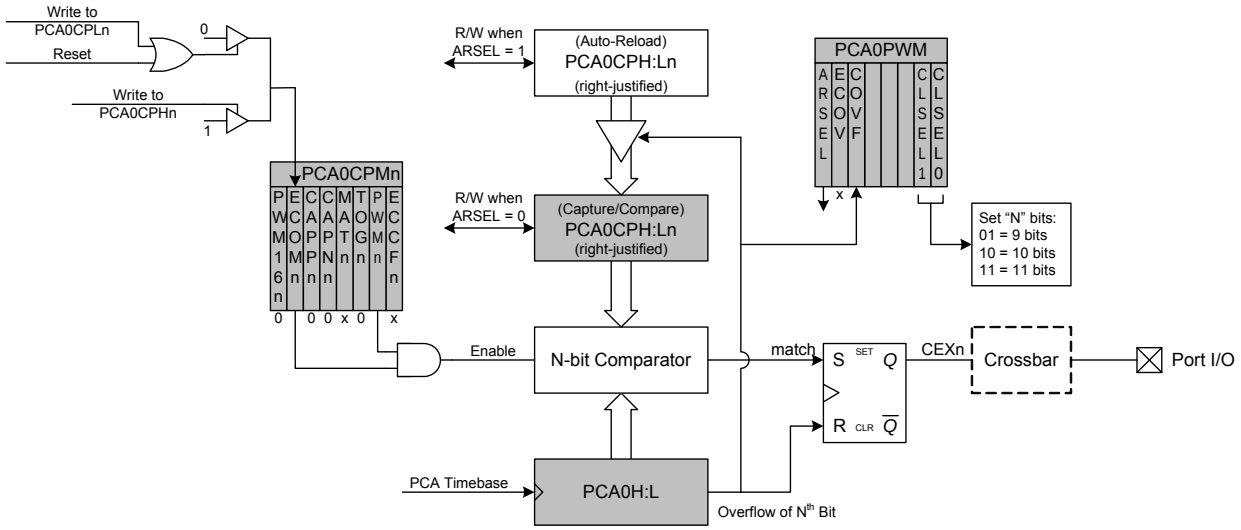


Figure 32.9. PCA 9, 10 and 11-Bit PWM Mode Diagram

32.3.6. 16-Bit Pulse Width Modulator Mode

A PCA module may also be operated in 16-Bit PWM mode. 16-bit PWM mode is independent of the other (8/9/10/11-bit) PWM modes. In this mode, the 16-bit capture/compare module defines the number of PCA clocks for the low time of the PWM signal. When the PCA counter matches the module contents, the output on CEXn is asserted high; when the 16-bit counter overflows, CEXn is asserted low. To output a varying duty cycle, new value writes should be synchronized with PCA CCFn match interrupts. 16-Bit PWM Mode is enabled by setting the ECOMn, PWMn, and PWM16n bits in the PCA0CPMn register. For a varying duty cycle, match interrupts should be enabled (ECCFn = 1 AND MATn = 1) to help synchronize the capture/compare register writes. If the MATn bit is set to 1, the CCFn flag for the module will be set each time a 16-bit comparator match (rising edge) occurs. The CF flag in PCA0CN can be used to detect the overflow (falling edge). The duty cycle for 16-Bit PWM Mode is given by Equation 32.4.

Important Note About Capture/Compare Registers: When writing a 16-bit value to the PCA0 Capture/Compare registers, the low byte should always be written first. Writing to PCA0CPLn clears the ECOMn bit to 0; writing to PCA0CPHn sets ECOMn to 1.

$$\text{Duty Cycle} = \frac{(65536 - PCA0CPn)}{65536}$$

Equation 32.4. 16-Bit PWM Duty Cycle

Using Equation 32.4, the largest duty cycle is 100% (PCA0CPn = 0), and the smallest duty cycle is 0.0015% (PCA0CPn = 0xFFFF). A 0% duty cycle may be generated by clearing the ECOMn bit to 0.

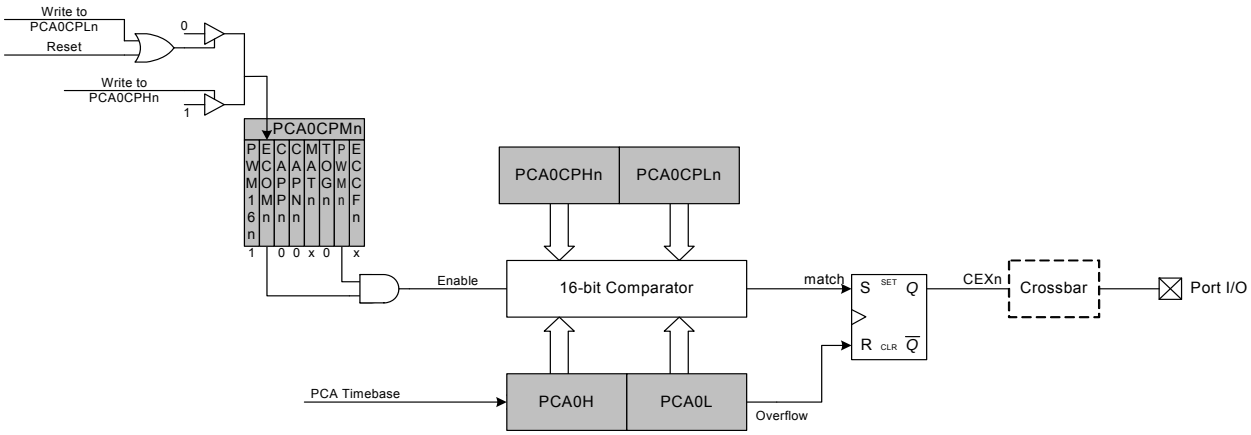


Figure 32.10. PCA 16-Bit PWM Mode

32.4. Register Descriptions for PCA0

Following are detailed descriptions of the special function registers related to the operation of the PCA.

SFR Definition 32.1. PCA0CN: PCA Control

Bit	7	6	5	4	3	2	1	0
Name	CF	CR				CCF2	CCF1	CCF0
Type	R/W	R/W	R	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xD8; SFR Page = All Pages; Bit-Addressable

Bit	Name	Function
7	CF	**PCA Counter/Timer Overflow Flag.** Set by hardware when the PCA Counter/Timer overflows from 0xFFFF to 0x0000. When the Counter/Timer Overflow (CF) interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by software.
6	CR	**PCA Counter/Timer Run Control.** This bit enables/disables the PCA Counter/Timer. 0: PCA Counter/Timer disabled. 1: PCA Counter/Timer enabled.
5:3	Unused	Read = 000b; Write = Don’t care
2:0	CCF[2:0]	**PCA Module n Capture/Compare Flag.** These bits are set by hardware when a match or capture occurs in the associated PCA Module n. When the CCFn interrupt is enabled, setting this bit causes the CPU to vector to the PCA interrupt service routine. This bit is not automatically cleared by hardware and must be cleared by software.

SFR Definition 32.2. PCA0MD: PCA Mode

Bit	7	6	5	4	3	2	1	0
Name	CIDL				CPS2	CPS1	CPS0	ECF
Type	R/W	R	R	R	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xED; SFR Page = F

Bit	Name	Function
7	CIDL	**PCA Counter/Timer Idle Control.** Specifies PCA behavior when CPU is in Idle Mode. 0: PCA continues to function normally while the system controller is in Idle Mode. 1: PCA operation is suspended while the system controller is in Idle Mode.
6:4	Unused	Read = 000b, Write = don't care.
3:1	CPS[2:0]	**PCA Counter/Timer Pulse Select.** These bits select the timebase source for the PCA counter 000: System clock divided by 12 001: System clock divided by 4 010: Timer 0 overflow 011: High-to-low transitions on ECI (max rate = system clock divided by 4) 100: System clock 101: External clock divided by 8 (synchronized with the system clock) 110-111: Reserved
0	ECF	**PCA Counter/Timer Overflow Interrupt Enable.** This bit sets the masking of the PCA Counter/Timer Overflow (CF) interrupt. 0: Disable the CF interrupt. 1: Enable a PCA Counter/Timer Overflow interrupt request when CF (PCA0CN.7) is set.

SFR Definition 32.3. PCA0PWM: PCA PWM Configuration

Bit	7	6	5	4	3	2	1	0
Name	ARSEL	ECOV	COVF				CLSEL[1:0]	
Type	R/W	R/W	R/W	R	R	R	R/W	
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xA1; SFR Page = F

Bit	Name	Function
7	ARSEL	**Auto-Reload Register Select.** This bit selects whether to read and write the normal PCA capture/compare registers (PCA0CPn), or the Auto-Reload registers at the same SFR addresses. This function is used to define the reload value for 9, 10, and 11-bit PWM modes. In all other modes, the Auto-Reload registers have no function. 0: Read/Write Capture/Compare Registers at PCA0CPHn and PCA0CPLn. 1: Read/Write Auto-Reload Registers at PCA0CPHn and PCA0CPLn.
6	ECOV	**Cycle Overflow Interrupt Enable.** This bit sets the masking of the Cycle Overflow Flag (COVF) interrupt. 0: COVF will not generate PCA interrupts. 1: A PCA interrupt will be generated when COVF is set.
5	COVF	**Cycle Overflow Flag.** This bit indicates an overflow of the 8th, 9th, 10th, or 11th bit of the main PCA counter (PCA0). The specific bit used for this flag depends on the setting of the Cycle Length Select bits. The bit can be set by hardware or software, but must be cleared by software. 0: No overflow has occurred since the last time this bit was cleared. 1: An overflow has occurred since the last time this bit was cleared.
4:2	Unused	Read = 000b; Write = don’t care.
1:0	CLSEL[1:0]	**Cycle Length Select.** When 16-bit PWM mode is not selected, these bits select the length of the PWM cycle, between 8, 9, 10, or 11 bits. This affects all channels configured for PWM which are not using 16-bit PWM mode. These bits are ignored for individual channels configured to16-bit PWM mode. 00: 8 bits. 01: 9 bits. 10: 10 bits. 11: 11 bits.

SFR Definition 32.4. PCA0CPMn: PCA Capture/Compare Mode

Bit	7	6	5	4	3	2	1	0
Name	PWM16n	ECOMn	CAPPn	CAPNn	MATn	TOGn	PWMn	ECCFn
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Addresses: PCA0CPM0 = 0xDA, PCA0CPM1 = 0xDB, PCA0CPM2 = 0xDC

SFR Pages: PCA0CPM0 = F, PCA0CPM1 = F, PCA0CPM2 = F

Bit	Name	Function
7	PWM16n	**16-bit Pulse Width Modulation Enable.** This bit enables 16-bit mode when Pulse Width Modulation mode is enabled. 0: 8 to 11-bit PWM selected. 1: 16-bit PWM selected.
6	ECOMn	**Comparator Function Enable.** This bit enables the comparator function for PCA module n when set to 1.
5	CAPPn	**Capture Positive Function Enable.** This bit enables the positive edge capture for PCA module n when set to 1.
4	CAPNn	**Capture Negative Function Enable.** This bit enables the negative edge capture for PCA module n when set to 1.
3	MATn	**Match Function Enable.** This bit enables the match function for PCA module n when set to 1. When enabled, matches of the PCA counter with a module's capture/compare register cause the CCFn bit in PCA0MD register to be set to logic 1.
2	TOGn	**Toggle Function Enable.** This bit enables the toggle function for PCA module n when set to 1. When enabled, matches of the PCA counter with a module's capture/compare register cause the logic level on the CEXn pin to toggle. If the PWMn bit is also set to logic 1, the module operates in Frequency Output Mode.
1	PWMn	**Pulse Width Modulation Mode Enable.** This bit enables the PWM function for PCA module n when set to 1. When enabled, a pulse width modulated signal is output on the CEXn pin. 8 to 11-bit PWM is used if PWM16n is cleared; 16-bit mode is used if PWM16n is set to logic 1. If the TOGn bit is also set, the module operates in Frequency Output Mode.
0	ECCFn	**Capture/Compare Flag Interrupt Enable.** This bit sets the masking of the Capture/Compare Flag (CCFn) interrupt. 0: Disable CCFn interrupts. 1: Enable a Capture/Compare Flag interrupt request when CCFn is set.

SFR Definition 32.5. PCA0L: PCA Counter/Timer Low Byte

Bit	7	6	5	4	3	2	1	0
Name	PCA0[7:0]							
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xF9; SFR Page = 0

Bit	Name	Function
7:0	PCA0[7:0]	**PCA Counter/Timer Low Byte.** The PCA0L register holds the low byte (LSB) of the 16-bit PCA Counter/Timer.

SFR Definition 32.6. PCA0H: PCA Counter/Timer High Byte

Bit	7	6	5	4	3	2	1	0
Name	PCA0[15:8]							
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Address = 0xFA; SFR Page = 0

Bit	Name	Function
7:0	PCA0[15:8]	**PCA Counter/Timer High Byte.** The PCA0H register holds the high byte (MSB) of the 16-bit PCA Counter/Timer. Reads of this register will read the contents of a “snapshot” register, whose contents are updated only when the contents of PCA0L are read (see Section 32.1).

SFR Definition 32.7. PCA0CPLn: PCA Capture Module Low Byte

Bit	7	6	5	4	3	2	1	0
Name	PCA0CPn[7:0]							
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Addresses: PCA0CPL0 = 0xFB, PCA0CPL1 = 0xE9, PCA0CPL2 = 0xEB,

SFR Pages: PCA0CPL0 = 0, PCA0CPL1 = 0, PCA0CPL2 = 0,

Bit	Name	Function
7:0	PCA0CPn[7:0]	**PCA Capture Module Low Byte.** The PCA0CPLn register holds the low byte (LSB) of the 16-bit capture module n. This register address also allows access to the low byte of the corresponding PCA channel's auto-reload value for 9, 10, or 11-bit PWM mode. The ARSEL bit in register PCA0PWM controls which register is accessed.
Note: A write to this register will clear the module's ECOMn bit to a 0.		

SFR Definition 32.8. PCA0CPHn: PCA Capture Module High Byte

Bit	7	6	5	4	3	2	1	0
Name	PCA0CPn[15:8]							
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

SFR Addresses: PCA0CPH0 = 0xFC, PCA0CPH1 = 0xEA, PCA0CPH2 = 0xEC,

SFR Pages: PCA0CPH0 = 0, PCA0CPH1 = 0, PCA0CPH2 = 0,

Bit	Name	Function
7:0	PCA0CPn[15:8]	**PCA Capture Module High Byte.** The PCA0CPHn register holds the high byte (MSB) of the 16-bit capture module n. This register address also allows access to the high byte of the corresponding PCA channel's auto-reload value for 9, 10, or 11-bit PWM mode. The ARSEL bit in register PCA0PWM controls which register is accessed.
Note: A write to this register will set the module's ECOMn bit to a 1.		

33. C2 Interface

C8051F70x/71x 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 operates using only two pins: a bi-directional data signal (C2D), and a clock input (C2CK). See the C2 Interface Specification for details on the C2 protocol.

33.1. C2 Interface Registers

The following describes the C2 registers necessary to perform Flash programming functions through the C2 interface. All C2 registers are accessed through the C2 interface as described in the C2 Interface Specification.

C2 Register Definition 33.1. C2ADD: C2 Address

Bit	7	6	5	4	3	2	1	0
Name	C2ADD[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

Bit	Name	Function		
7:0	C2ADD[7:0]	**C2 Address.** The C2ADD register is accessed via the C2 interface to select the target Data register for C2 Data Read and Data Write commands.		
		Address	**Name**	**Description**
		0x00	DEVICEID	Selects the Device ID Register (read only)
		0x01	REVID	Selects the Revision ID Register (read only)
		0x02	FPCTL	Selects the C2 Flash Programming Control Register
		0xBF	FPDAT	Selects the C2 Flash Data Register
		0x96	CRC0AUTO*	Selects the CRC0AUTO Register
		0x97	CRC0CNT*	Selects the CRC0CNT Register
		0x91	CRC0CN*	Selects the CRC0CN Register
		0xD9	CRC0DATA*	Selects the CRC0DATA Register
		0x95	CRC0FLIP*	Selects the CRC0FLIP Register
		0x94	CRC0IN*	Selects the CRC0IN Register
Note: CRC registers and functions are described in Section “27. Cyclic Redundancy Check Unit (CRC0)” on page 193.				

C2 Register Definition 33.2. DEVICEID: C2 Device ID

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

C2 Address: 0x00

Bit	Name	Function
7:0	DEVICEID[7:0]	**Device ID.** This read-only register returns the 8-bit device ID: 0x1E (C8051F70x/71x).

C2 Register Definition 33.3. REVID: C2 Revision ID

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

C2 Address: 0x01

Bit	Name	Function
7:0	REVID[7:0]	**Revision ID.** This read-only register returns the 8-bit revision ID. For example: 0x00 = Revision A.

C2 Register Definition 33.4. FPCTL: C2 Flash Programming Control

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

C2 Address: 0x02

Bit	Name	Function
7:0	FPCTL[7:0]	**C2 Flash Programming Control Register.** This register is used to enable Flash programming via the C2 interface. To enable C2 Flash programming, the following codes must be written in order: 0x02, 0x01. Once C2 Flash programming is enabled, a system reset must be issued to resume normal operation.

C2 Register Definition 33.5. FPDAT: C2 Flash Programming Data

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

C2 Address: 0xBF

Bit	Name	Function	
7:0	FPDAT[7:0]	**C2 Flash Programming Data Register.** This register is used to pass Flash commands, addresses, and data during C2 Flash accesses. Valid commands are listed below.	
		Code	**Command**
		0x06	Flash Block Read
		0x07	Flash Block Write
		0x08	Flash Page Erase
		0x03	Device Erase

33.2. C2CK Pin Sharing

The C2CK pin is shared with the $\overline{RST}$ signal on this device family. If the $\overline{RST}$ pin is used by other parts of the system, debugging and programming the device can still be accomplished without disrupting the rest of the system. If this is desired, it is normally necessary to add a resistor to isolate the system's reset line from the C2CK signal. This external resistors would not be necessary for production boards, where debugging capabilities are not needed. A typical isolation configuration is shown in Figure 33.1.

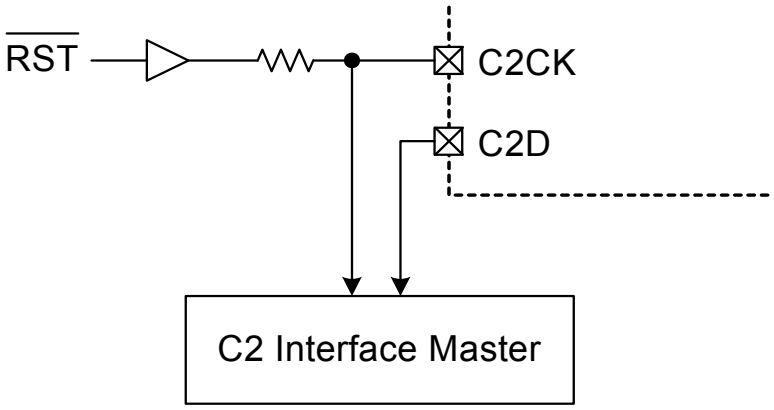


Figure 33.1. Typical C2CK Pin Sharing

The configuration in Figure 33.1 assumes the $\overline{RST}$ pin on the target device is used as an input only. Additional resistors may be necessary depending on the specific application.

DOCUMENT CHANGE LIST

Revision 0.2 to Revision 0.3

- Corrected Dimension D in the QFN-48 Package Specifications.
- Updated Table 7.1 on page 39.
- Updated Register 8.1, “ADC0CF: ADC0 Configuration,” on page 50.
- Updated Register 12.3, “CPT0MX: Comparator0 MUX Selection,” on page 71.
- Updated “Port Pins Configured for Analog I/O” on page 166.
- Updated Register 33.2, “DEVICEID: C2 Device ID,” on page 285.

NOTES:

CONTACT INFORMATION

Silicon Laboratories Inc.

Silicon Laboratories Inc.
400 West Cesar Chavez
Austin, TX 78701
Tel: 1+(512) 416-8500
Fax: 1+(512) 416-9669
Toll Free: 1+(877) 444-3032

Email: MCUinfo@silabs.com
Internet: www.silabs.com

The information in this document is believed to be accurate in all respects at the time of publication but is subject to change without notice. Silicon Laboratories assumes no responsibility for errors and omissions, and disclaims responsibility for any consequences resulting from the use of information included herein. Additionally, Silicon Laboratories assumes no responsibility for the functioning of undescribed features or parameters. Silicon Laboratories reserves the right to make changes without further notice. Silicon Laboratories makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Silicon Laboratories assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. Silicon Laboratories products are not designed, intended, or authorized for use in applications intended to support or sustain life, or for any other application in which the failure of the Silicon Laboratories product could create a situation where personal injury or death may occur. Should Buyer purchase or use Silicon Laboratories products for any such unintended or unauthorized application, Buyer shall indemnify and hold Silicon Laboratories harmless against all claims and damages.

Silicon Laboratories and Silicon Labs are trademarks of Silicon Laboratories Inc.

Other products or brandnames mentioned herein are trademarks or registered trademarks of their respective holders