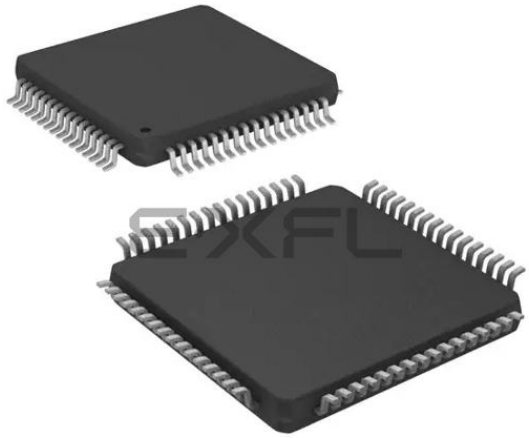

**Welcome to E-XFL.COM**

## What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

## Applications of "Embedded - Microcontrollers"

| Details | |
|---|---|
| Product Status | Obsolete |
| Core Processor | 8051 |
| Core Size | 8-Bit |
| Speed | 25MHz |
| Connectivity | CANbus, EBI/EMI, SMBus (2-Wire/I²C), SPI, UART/USART |
| Peripherals | Brown-out Detect/Reset, POR, PWM, Temp Sensor, WDT |
| Number of I/O | 32 |
| Program Memory Size | 64KB (64K x 8) |
| Program Memory Type | FLASH |
| EEPROM Size | - |
| RAM Size | 4.25K x 8 |
| Voltage - Supply (Vcc/Vdd) | 2.7V ~ 3.6V |
| Data Converters | A/D 8x8b, 13x10b; D/A 2x10b, 2x12b |
| Oscillator Type | Internal |
| Operating Temperature | -40°C ~ 85°C (TA) |
| Mounting Type | Surface Mount |
| Package / Case | 64-TQFP |
| Supplier Device Package | 64-TQFP (10x10) |
| Purchase URL | https://www.e-xfl.com/product-detail/silicon-labs/c8051f043 |

## NOTES:

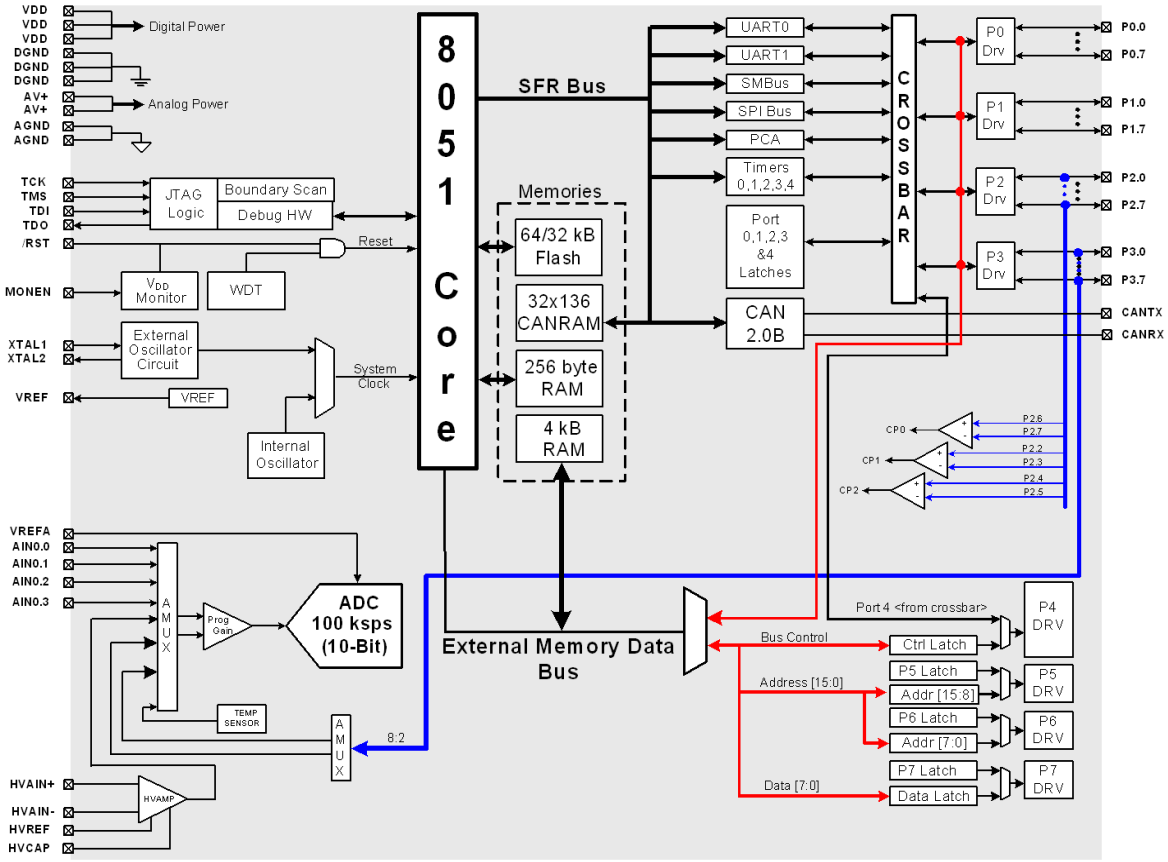


**Figure 1.4. C8051F045/7 Block Diagram**

## 5.3. ADC Modes of Operation

ADC0 has a maximum conversion speed of 100 ksps. The ADC0 conversion clock is derived from the system clock divided by the value held in the ADC0SC bits of register ADC0CF.

### 5.3.1. Starting a Conversion

A conversion can be initiated in one of four ways, depending on the programmed states of the ADC0 Start of Conversion Mode bits (AD0CM1, AD0CM0) in ADC0CN. Conversions may be initiated by the following:

- Writing a '1' to the AD0BUSY bit of ADC0CN;
- A Timer 3 overflow (i.e., timed continuous conversions);
- A rising edge detected on the external ADC convert start signal, CNVSTR0;
- A Timer 2 overflow (i.e., timed continuous conversions).

The AD0BUSY bit is set to logic 1 during conversion and restored to logic 0 when conversion is complete. The falling edge of AD0BUSY triggers an interrupt (when enabled) and sets the AD0INT interrupt flag (ADC0CN.5). Converted data is available in the ADC0 data word MSB and LSB registers, ADC0H, ADC0L. Converted data can be either left or right justified in the ADC0H:ADC0L register pair (see example in Figure 5.7) depending on the programmed state of the AD0LJST bit in the ADC0CN register.

When initiating conversions by writing a '1' to AD0BUSY, the AD0INT bit should be polled to determine when a conversion has completed (ADC0 interrupts may also be used). The recommended polling procedure is shown below.

Step 1. Write a '0' to AD0INT;
Step 2. Write a '1' to AD0BUSY;
Step 3. Poll AD0INT for '1';
Step 4. Process ADC0 data.

### 5.3.2. Tracking Modes

According to Table 5.2, each ADC0 conversion must be preceded by a minimum tracking time for the converted result to be accurate. The AD0TM bit in register ADC0CN controls the ADC0 track-and-hold mode. In its default state, the ADC0 input is continuously tracked when a conversion is not in progress. When the AD0TM bit is logic 1, ADC0 operates in low-power tracking mode. In this mode, each conversion is preceded by a tracking period of 3 SAR clocks after the start-of-conversion signal. When the CNVSTR0 signal is used to initiate conversions in low-power tracking mode, ADC0 tracks only when CNVSTR0 is low; conversion begins on the rising edge of CNVSTR0 (see Figure 5.4). Tracking can also be disabled when the entire chip is in low power standby or sleep modes. Low-power tracking mode is also useful when AMUX or PGA settings are frequently changed, to ensure that settling time requirements are met (see **Section "5.3.3. Settling Time Requirements" on page 56**).

## SFR Definition 5.6. ADC0CN: ADC0 Control

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| AD0EN | AD0TM | AD0INT | AD0BUSY | AD0CM1 | AD0CM0 | AD0WINT | AD0LJST | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | Bit Addressable |

SFR Address: 0xE8
SFR Page: 0

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

Bit6: AD0TM: ADC Track Mode Bit
0: When the ADC is enabled, tracking is continuous unless a conversion is in process
1: Tracking Defined by AD0CM1-0 bits

Bit5: AD0INT: ADC0 Conversion Complete Interrupt Flag.
This flag must be cleared by software.
0: ADC0 has not completed a data conversion since the last time this flag was cleared.
1: ADC0 has completed a data conversion.

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

Bit3-2: AD0CM1-0: ADC0 Start of Conversion Mode Select.
If AD0TM = 0:
00: ADC0 conversion initiated on every write of ‘1’ to AD0BUSY.
01: ADC0 conversion initiated on overflow of Timer 3.
10: ADC0 conversion initiated on rising edge of external CNVSTR0.
11: ADC0 conversion initiated on overflow of Timer 2.
If AD0TM = 1:
00: Tracking starts with the write of ‘1’ to AD0BUSY and lasts for 3 SAR clocks, followed by conversion.
01: Tracking started by the overflow of Timer 3 and last for 3 SAR clocks, followed by conversion.
10: ADC0 tracks only when CNVSTR0 input is logic low; conversion starts on rising CNVSTR0 edge.
11: Tracking started by the overflow of Timer 2 and last for 3 SAR clocks, followed by conversion.

Bit1: AD0WINT: ADC0 Window Compare Interrupt Flag.
This bit must be cleared by software.
0: ADC0 Window Comparison Data match has not occurred since this flag was last cleared.
1: ADC0 Window Comparison Data match has occurred.

Bit0: AD0LJST: ADC0 Left Justify Select.
0: Data in ADC0H:ADC0L registers are right-justified.
1: Data in ADC0H:ADC0L registers are left-justified.

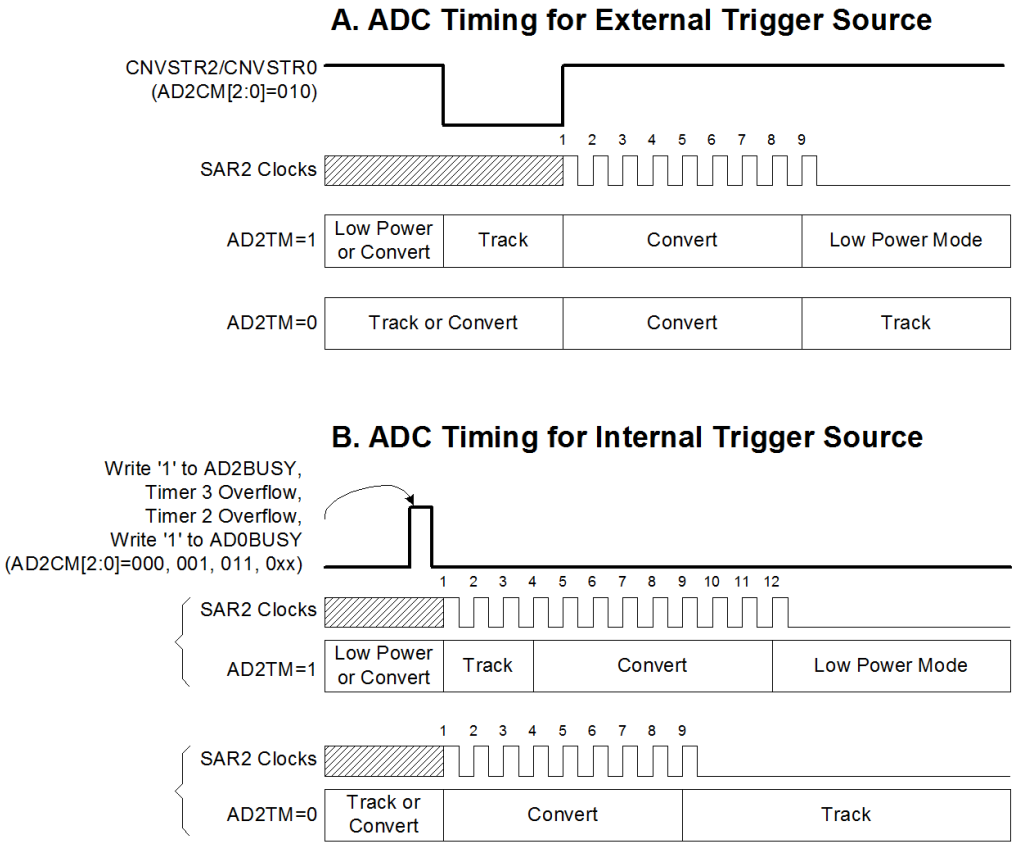


**Figure 7.2. ADC2 Track and Conversion Example Timing**

## SFR Definition 11.1. CPTnCN: Comparator 0, 1, and 2 Control

| R/W | R | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| CPnEN | CPnOUT | CPnRIF | CPnFIF | CPnHYP1 | CPnHYP0 | CPnHYN1 | CPnHYN0 | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: CPT0CN: 0x88; CPT1CN: 0x88; CPT2CN: 0x88
SFR Pages: CPT0CN:page 1;CPT1CN:page 2; CPT2CN:page 3

Bit7: CPnEN: Comparator Enable Bit. **(Please see note below.)**
0: Comparator Disabled.
1: Comparator Enabled.

Bit6: CPnOUT: Comparator Output State Flag.
0: Voltage on CPn+ < CPn–.
1: Voltage on CPn+ > CPn–.

Bit5: CPnRIF: Comparator Rising-Edge Interrupt Flag.
0: No Comparator Rising Edge Interrupt has occurred since this flag was last cleared.
1: Comparator Rising Edge Interrupt has occurred. Must be cleared by software.

Bit4: CPnFIF: Comparator Falling-Edge Interrupt Flag.
0: No Comparator Falling-Edge Interrupt has occurred since this flag was last cleared.
1: Comparator Falling-Edge Interrupt has occurred. Must be cleared by software.

Bits3-2: CPnHYP1-0: Comparator Positive Hysteresis Control Bits.
00: Positive Hysteresis Disabled.
01: Positive Hysteresis = 5 mV.
10: Positive Hysteresis = 10 mV.
11: Positive Hysteresis = 20 mV.

Bits1-0: CPnHYN1-0: Comparator Negative Hysteresis Control Bits.
00: Negative Hysteresis Disabled.
01: Negative Hysteresis = 5 mV.
10: Negative Hysteresis = 10 mV.
11: Negative Hysteresis = 20 mV.

NOTE: Upon enabling a comparator, the output of the comparator is not immediately valid. Before using a comparator as an interrupt or reset source, software should wait for a minimum of the specified “Power-up time” as specified in Table 11.1, “Comparator Electrical Characteristics,” on page 126.

## Table 11.1. Comparator Electrical Characteristics

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

| Parameter | Conditions | Min | Typ | Max | Units |
|---|---|---|---|---|---|
| Response Time, Mode 0 | CPn+ – CPn– = 100 mV | — | 100 | — | ns |
| | CPn+ – CPn– = 10 mV | — | 250 | — | ns |
| Response Time, Mode 1 | CPn+ – CPn– = 100 mV | — | 175 | — | ns |
| | CPn+ – CPn– = 10 mV | — | 500 | — | ns |
| Response Time, Mode 2 | CPn+ – CPn– = 100 mV | — | 320 | — | ns |
| | CPn+ – CPn– = 10 mV | — | 1100 | — | ns |
| Response Time, Mode 3 | CPn+ – CPn– = 100 mV | — | 1050 | — | ns |
| | CPn+ – CPn– = 10 mV | — | 5200 | — | ns |
| Common-Mode Rejection Ratio | | — | 1.5 | 4 | mV/V |
| Positive Hysteresis 1 | CPnHYP1-0 = 00 | — | 0 | 1 | mV |
| Positive Hysteresis 2 | CPnHYP1-0 = 01 | 2 | 4.5 | 7 | mV |
| Positive Hysteresis 3 | CPnHYP1-0 = 10 | 4 | 9 | 13 | mV |
| Positive Hysteresis 4 | CPnHYP1-0 = 11 | 10 | 17 | 25 | mV |
| Negative Hysteresis 1 | CPnHYN1-0 = 00 | | 0 | 1 | mV |
| Negative Hysteresis 2 | CPnHYN1-0 = 01 | 2 | 4.5 | 7 | mV |
| Negative Hysteresis 3 | CPnHYN1-0 = 10 | 4 | 9 | 13 | mV |
| Negative Hysteresis 4 | CPnHYN1-0 = 11 | 10 | 17 | 25 | mV |
| Inverting or Non-Inverting Input Voltage Range | | –0.25 | | $V_{DD}$ + 0.25 | V |
| Input Capacitance | | — | 7 | — | pF |
| Input Bias Current | | –5 | 0.001 | +5 | nA |
| Input Offset Voltage | | –5 | | +5 | mV |
| **Power Supply** | | | | | |
| Power Supply Rejection | | — | 0.1 | 1 | mV/V |
| Power-up Time | | — | 10 | — | µs |
| Supply Current at DC | Mode 0 | — | 7.6 | — | µA |
| | Mode 1 | — | 3.2 | — | µA |
| | Mode 2 | — | 1.3 | — | µA |
| | Mode 3 | — | 0.4 | — | µA |

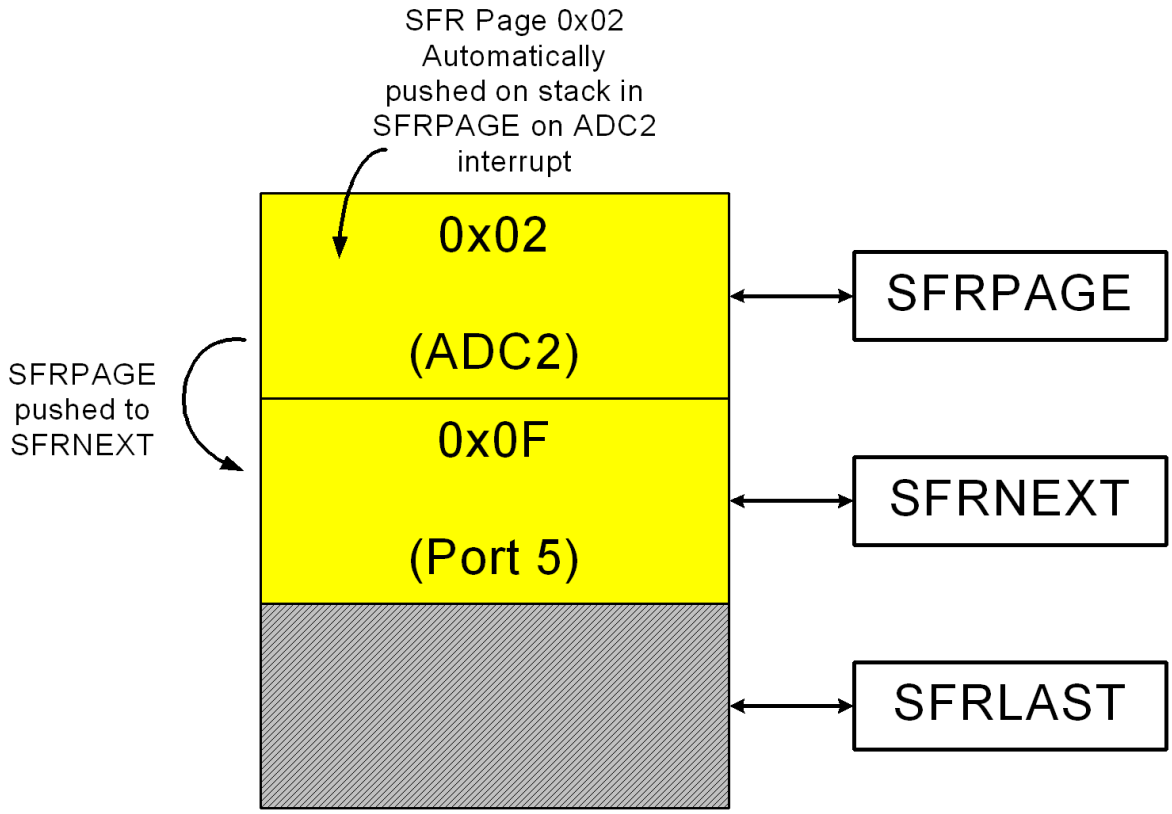


**Figure 12.5. SFR Page Stack After ADC2 Window Comparator Interrupt Occurs**

While in the ADC2 ISR, a PCA interrupt occurs. Recall the PCA interrupt is configured as a *high* priority interrupt, while the ADC2 interrupt is configured as a *low* priority interrupt. Thus, the CIP-51 will now vector to the high priority PCA ISR. Upon doing so, the CIP-51 will automatically place the SFR page needed to access the PCA's special function registers into the SFRPAGE register, SFR Page 0x00. The value that was in the SFRPAGE register before the PCA interrupt (SFR Page 2 for ADC2) is pushed down the stack into SFRNEXT. Likewise, the value that was in the SFRNEXT register before the PCA interrupt (in this case SFR Page 0x0F for Port 5) is pushed down to the SFRLAST register, the "bottom" of the stack. Note that a value stored in SFRLAST (via a previous software write to the SFRLAST register) will be overwritten. See Figure 12.6 below.

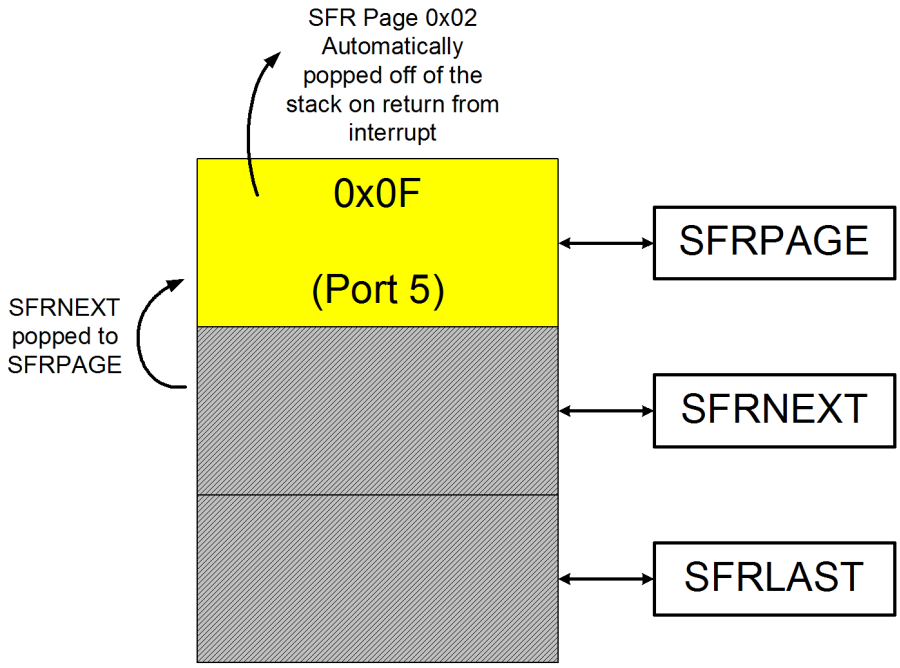


**Figure 12.8. SFR Page Stack Upon Return From ADC2 Window Interrupt**

Note that in the above example, all three bytes in the SFR Page Stack are accessible via the SFRPAGE, SFRNEXT, and SFRLAST special function registers. If the stack is altered while servicing an interrupt, it is possible to return to a different SFR Page upon interrupt exit than selected prior to the interrupt call. Direct access to the SFR Page stack can be useful to enable real-time operating systems to control and manage context switching between multiple tasks.

Push operations on the SFR Page Stack only occur on interrupt service, and pop operations only occur on interrupt exit (execution on the RETI instruction). The automatic switching of the SFRPAGE and operation of the SFR Page Stack as described above can be disabled in software by clearing the SFR Automatic Page Enable Bit (SFRPGEN) in the SFR Page Control Register (SFRPGCN). See SFR Definition 12.1.

## SFR Definition 12.11. IE: Interrupt Enable

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| EA | IEGF0 | ET2 | ES0 | ET1 | EX1 | ET0 | EX0 | 00000000 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | Bit Addressable |

SFR Address: 0xA8
SFR Page: All Pages

Bit7: EA: Enable All Interrupts.
This bit globally enables/disables all interrupts. It overrides the individual interrupt mask settings.
0: Disable all interrupt sources.
1: Enable each interrupt according to its individual mask setting.

Bit6: IEGF0: General Purpose Flag 0.
This is a general purpose flag for use under software control.

Bit5: ET2: Enabler 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 TF2 flag.

Bit4: ES0: Enable UART0 Interrupt.
This bit sets the masking of the UART0 interrupt.
0: Disable UART0 interrupt.
1: Enable UART0 interrupt.

Bit3: 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.

Bit2: EX1: Enable External Interrupt 1.
This bit sets the masking of external interrupt 1.
0: Disable external interrupt 1.
1: Enable interrupt requests generated by the /INT1 pin.

Bit1: 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.

Bit0: EX0: Enable External Interrupt 0.
This bit sets the masking of external interrupt 0.
0: Disable external interrupt 0.
1: Enable interrupt requests generated by the /INT0 pin.

**16.6.1.2.8-bit MOVX without Bank Select: EMI0CF[4:2] = '101' or '111'.**

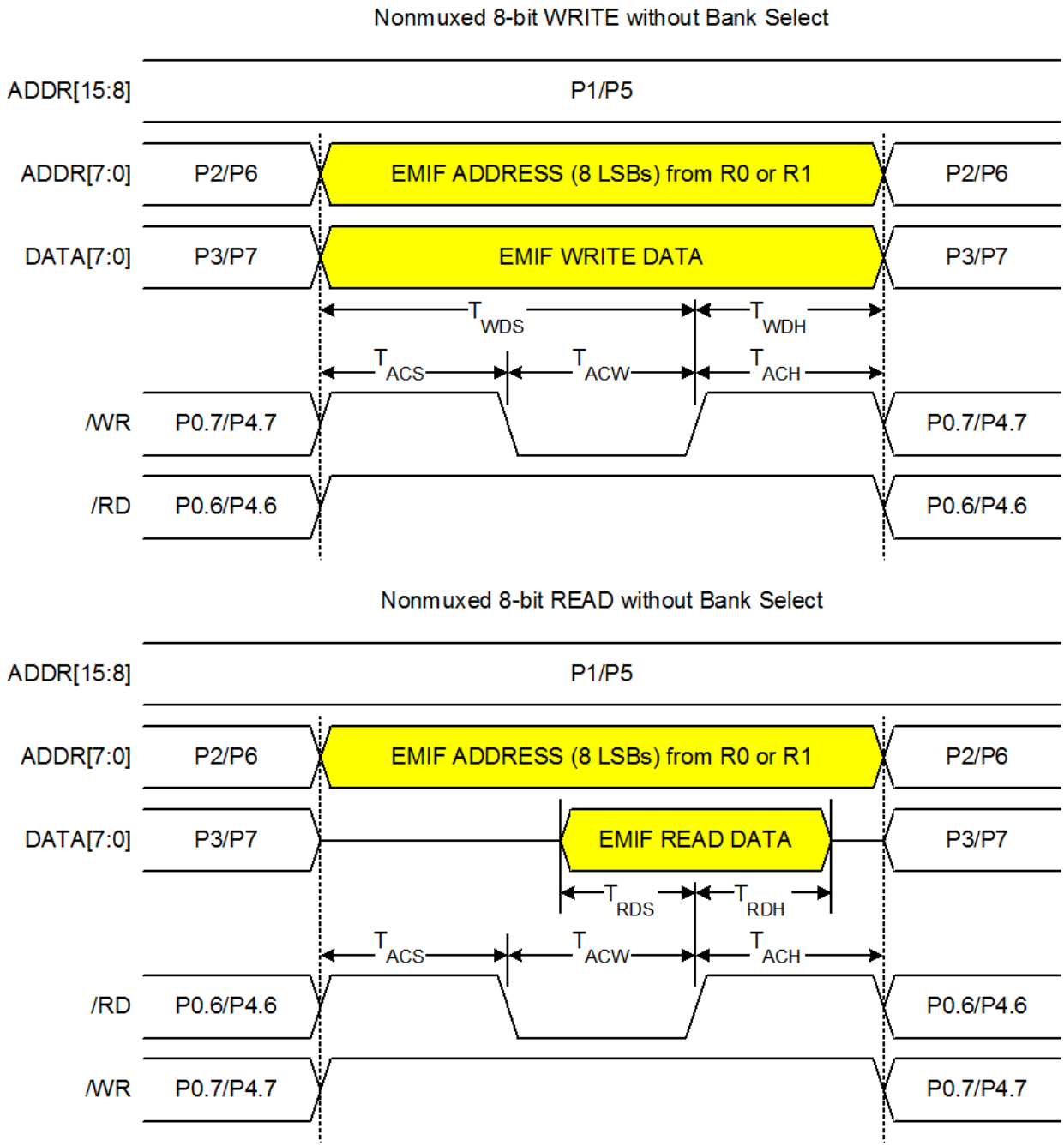


**Figure 16.5. Non-multiplexed 8-bit MOVX without Bank Select Timing**

### 16.6.2. Multiplexed Mode

#### 16.6.2.1.16-bit MOVX: EMI0CF[4:2] = ‘001’, ‘010’, or ‘011’.

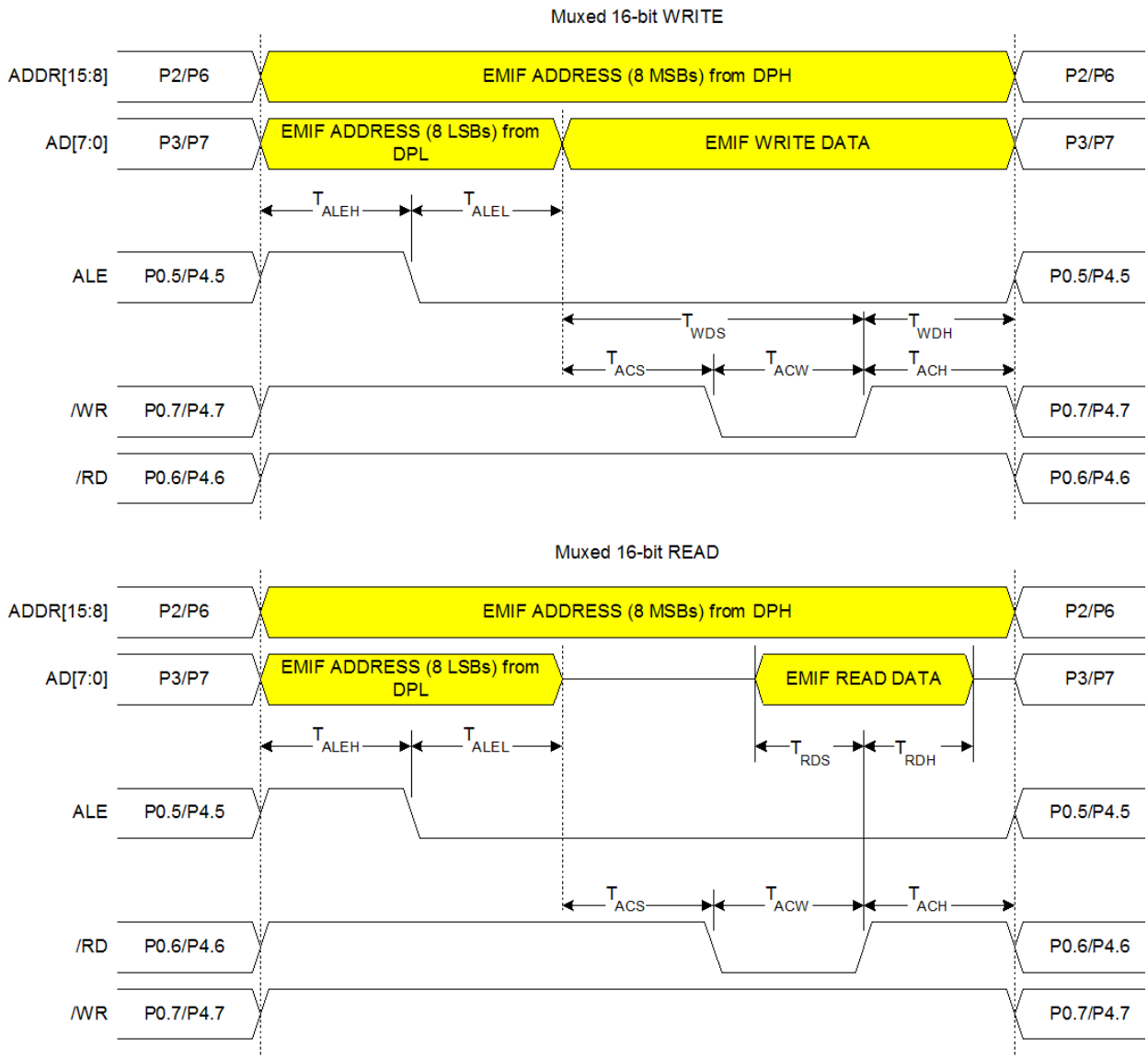


**Figure 16.7. Multiplexed 16-bit MOVX Timing**

### 17.1.7. Crossbar Pin Assignment Example

In this example (Figure 17.6), we configure the Crossbar to allocate Port pins for UART0, the SMBus, UART1, /INT0, and /INT1 (8 pins total). Additionally, we configure the External Memory Interface to operate in Multiplexed mode and to appear on the Low ports. Further, we configure P1.2, P1.3, and P1.4 for Analog Input mode so that the voltages at these pins can be measured by ADC2. The configuration steps are as follows:

1. XBR0, XBR1, and XBR2 are set such that UART0EN = 1, SMB0EN = 1, INT0E = 1, INT1E = 1, and EMIFLE = 1. Thus: XBR0 = 0x05, XBR1 = 0x14, and XBR2 = 0x02.
2. We configure the External Memory Interface to use Multiplexed mode and to appear on the Low ports. PRTSEL = 0, EMD2 = 0.
3. We configure the desired Port 1 pins to Analog Input mode by setting P1MDIN to 0xE3 (P1.4, P1.3, and P1.2 are Analog Inputs, so their associated P1MDIN bits are set to logic 0).
4. We enable the Crossbar by setting XBARE = 1: XBR2 = 0x42.
   - UART0 has the highest priority, so P0.0 is assigned to TX0, and P0.1 is assigned to RX0.
   - The SMBus is next in priority order, so P0.2 is assigned to SDA, and P0.3 is assigned to SCL.
   - UART1 is next in priority order, so P0.4 is assigned to TX1. Because the External Memory Interface is selected on the lower Ports, EMIFLE = 1, which causes the Crossbar to skip P0.6 (/RD) and P0.7 (/WR). Because the External Memory Interface is configured in Multiplexed mode, the Crossbar will also skip P0.5 (ALE). RX1 is assigned to the next non-skipped pin, which in this case is P1.0.
   - /INT0 is next in priority order, so it is assigned to P1.1.
   - P1MDIN is set to 0xE3, which configures P1.2, P1.3, and P1.4 as Analog Inputs, causing the Crossbar to skip these pins.
   - /INT1 is next in priority order, so it is assigned to the next non-skipped pin, which is P1.5.
   - The External Memory Interface will drive Ports 2 and 3 (denoted by red dots in Figure 17.6) during the execution of an off-chip MOVX instruction.
5. We set the UART0 TX pin (TX0, P0.0) and UART1 TX pin (TX1, P0.4) outputs to Push-Pull by setting P0MDOUT = 0x11.
6. We configure all EMIF-controlled pins to push-pull output mode by setting P0MDOUT |= 0xE0; P2MDOUT = 0xFF; P3MDOUT = 0xFF.
7. We explicitly disable the output drivers on the 3 Analog Input pins by setting P1MDOUT = 0x00 (configure outputs to Open-Drain) and P1 = 0xFF (a logic 1 selects the high-impedance state).

## SFR Definition 18.1. CAN0DATL: CAN Data Access Register Low Byte

| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | Reset Value |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 00000001 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | |

SFR Address: 0xD8
SFR Page: 1

Bit7-0: CAN0DATL: CAN Data Access Register Low Byte.
The CAN0DAT Registers are used to read/write register values and data to and from the CAN Registers pointed to with the index number in the CAN0ADR Register.
The CAN0ADR Register is used to point the [CAN0DATH:CAN0DATL] to a desired CAN Register. The desired CAN Register's index number is moved into CAN0ADR. The CAN0DAT Register can then read/write to and from the CAN Register.

## SFR Definition 18.2. CAN0ADR: CAN Address Index

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

SFR Address: 0xDA
SFR Page: 1

Bit7-0: CAN0ADR: CAN Address Index Register.
The CAN0ADR Register is used to point the [CAN0DATH:CAN0DATL] to a desired CAN Register. The desired CAN Register's index number is moved into CAN0ADR. The CAN0DAT Register can then read/write to and from the CAN Register.

Note: When the value of CAN0ADR is 0x08-0x12 and 0x20-0x2A (IF1 and IF2 registers), this register will autoincrement by 1 upon a write to CAN0DATL. See **Section "18.2.6. CAN0ADR Autoincrement Feature" on page 232**.

**All CAN registers' functions/definitions are listed and described in the Bosch CAN User's Guide.**

# 19. System Management BUS/I²C BUS (SMBUS0)

The SMBus0 I/O interface is a two-wire, bi-directional serial bus. SMBus0 is compliant with the System Management Bus Specification, version 2, and compatible with the I²C serial bus. Reads and writes to the interface by the system controller are byte oriented with the SMBus0 interface autonomously controlling the serial transfer of the data. A method of extending the clock-low duration is available to accommodate devices with different speed capabilities on the same bus.

SMBus0 may operate as a master and/or slave, and may function on a bus with multiple masters. SMBus0 provides control of SDA (serial data), SCL (serial clock) generation and synchronization, arbitration logic, and START/STOP control and generation. SMBus0 is controlled by SFRs as described in **Section 19.4 on page 245**.

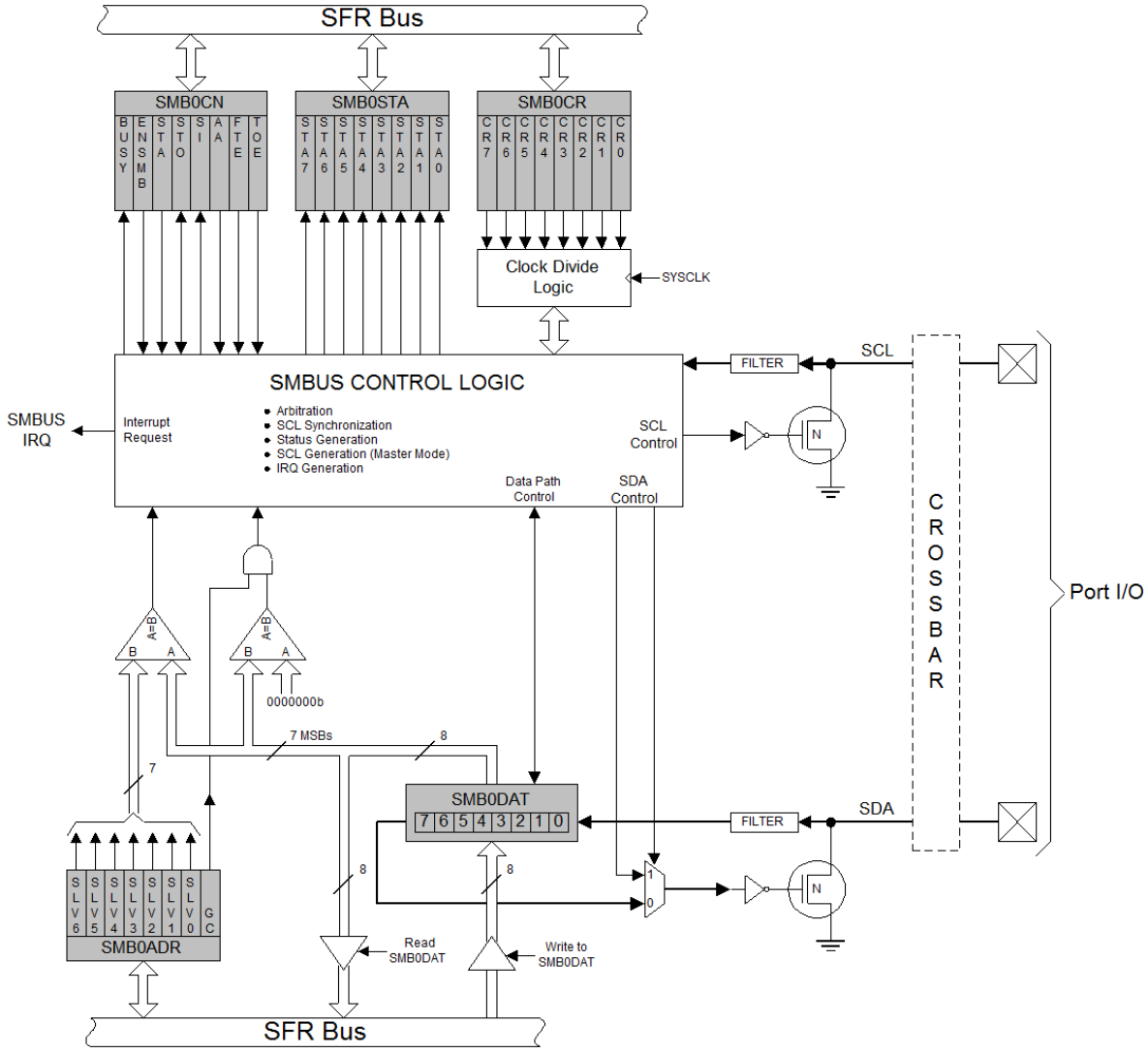


**Figure 19.1. SMBus0 Block Diagram**

Figure 19.2 shows a typical SMBus configuration. The SMBus0 interface will work at any voltage between 3.0 V and 5.0 V and 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 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 will not exceed 300 ns and 1000 ns, respectively.

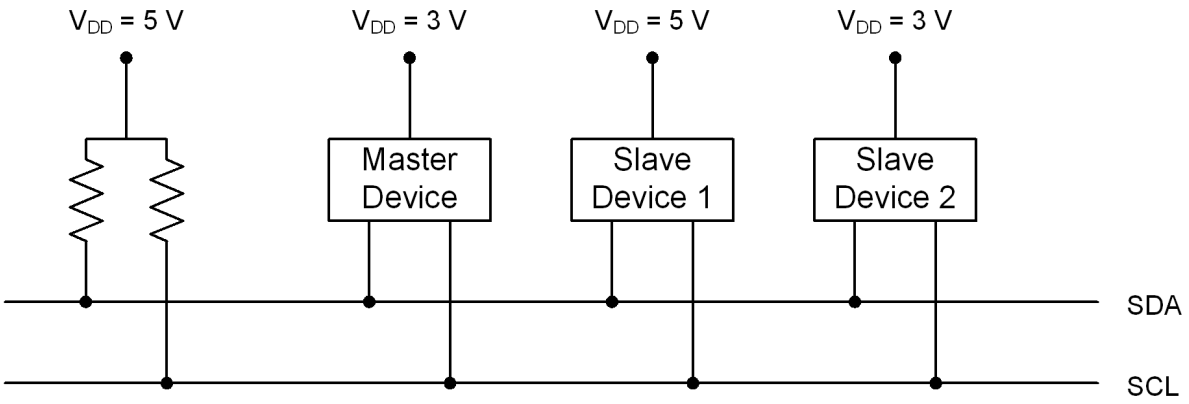


**Figure 19.2. Typical SMBus Configuration**

## 19.1. Supporting Documents

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

- $I^2C$ Manual (AN10216-01) -- March 24, 2003, Philips Semiconductor.
- System Management Bus Specification -- Version 1.1, SBS Implementers Forum.

Setting the SMBus0 Free Timer Enable bit (FTE, SMB0CN.1) to logic 1 enables the timer in SMB0CR. When SCL goes high, the timer in SMB0CR counts up. A timer overflow indicates a free bus timeout: if SMBus0 is waiting to generate a START, it will do so after this timeout. The bus free period should be less than 50 µs (see SFR Definition 19.2, SMBus0 Clock Rate Register).

When the TOE bit in SMB0CN is set to logic 1, Timer 4 is used to detect SCL low timeouts. If Timer 4 is enabled (see **Section “23.2. Timer 2, Timer 3, and Timer 4” on page 297**), Timer 4 is forced to reload when SCL is high, and forced to count when SCL is low. With Timer 4 enabled and configured to overflow after 25 ms (and TOE set), a Timer 4 overflow indicates a SCL low timeout; the Timer 4 interrupt service routine can then be used to reset SMBus0 communication in the event of an SCL low timeout.

# 20. 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.

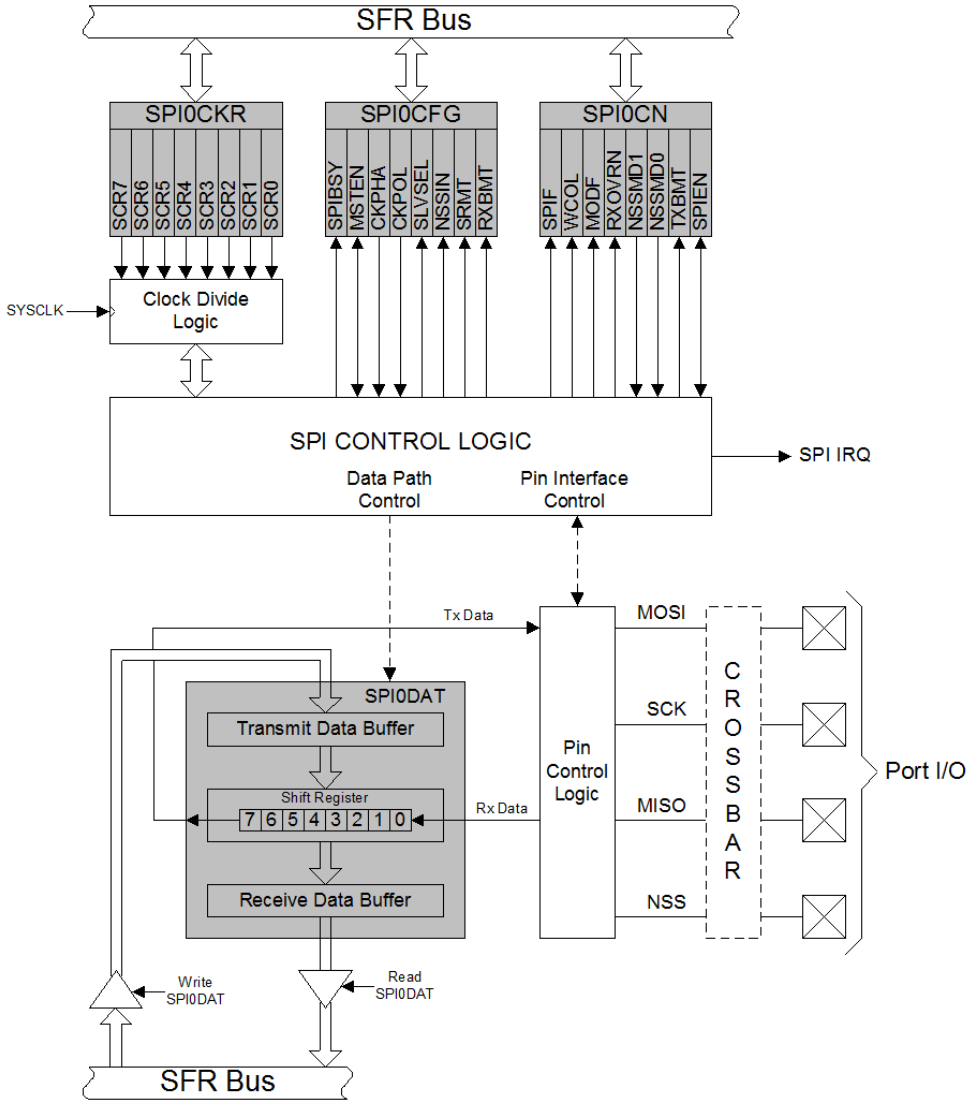


**Figure 20.1. SPI Block Diagram**

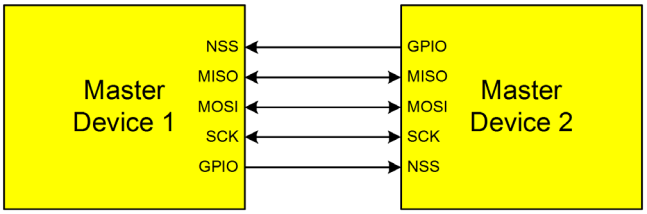


**Figure 20.2. Multiple-Master Mode Connection Diagram**

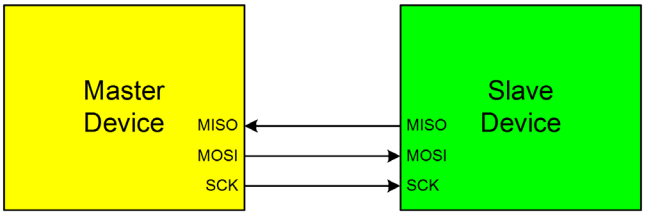


**Figure 20.3. 3-Wire Single Master and Slave Mode Connection Diagram**

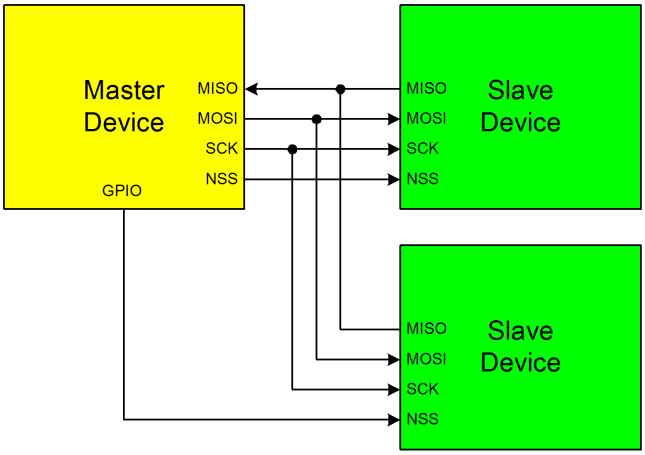


**Figure 20.4. 4-Wire Single Master and Slave Mode Connection Diagram**

## 24.1. 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 24.1. **Note that in 'External oscillator source divided by 8' mode, the external oscillator source is synchronized with the system clock, and must have a frequency less than or equal to the system clock.**

When the counter/timer overflows from 0xFFFF to 0x0000, the Counter Overflow Flag (CF) in PCA0MD is set to logic 1 and an interrupt request is generated if CF interrupts are enabled. Setting the ECF bit in PCA0MD to logic 1 enables the CF flag to generate an interrupt request. The CF bit is not automatically cleared by hardware when the CPU vectors to the interrupt service routine, and must be cleared by software (Note: PCA0 interrupts must be globally enabled before CF interrupts are recognized. PCA0 interrupts are globally enabled by setting the EA bit (IE.7) and the EPCA0 bit in EIE1 to logic 1). Clearing the CIDL bit in the PCA0MD register allows the PCA to continue normal operation while the CPU is in Idle mode.

**Table 24.1. 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[1] (max rate = system clock divided by 4) |
| 1 | 0 | 0 | System clock |
| 1 | 0 | 1 | External clock divided by 8[2] |

**Notes:**
1. The minimum high or low time for the ECI input signal is at least 2 system clock cycles.
2. External oscillator source divided by 8 is synchronized with the system clock.

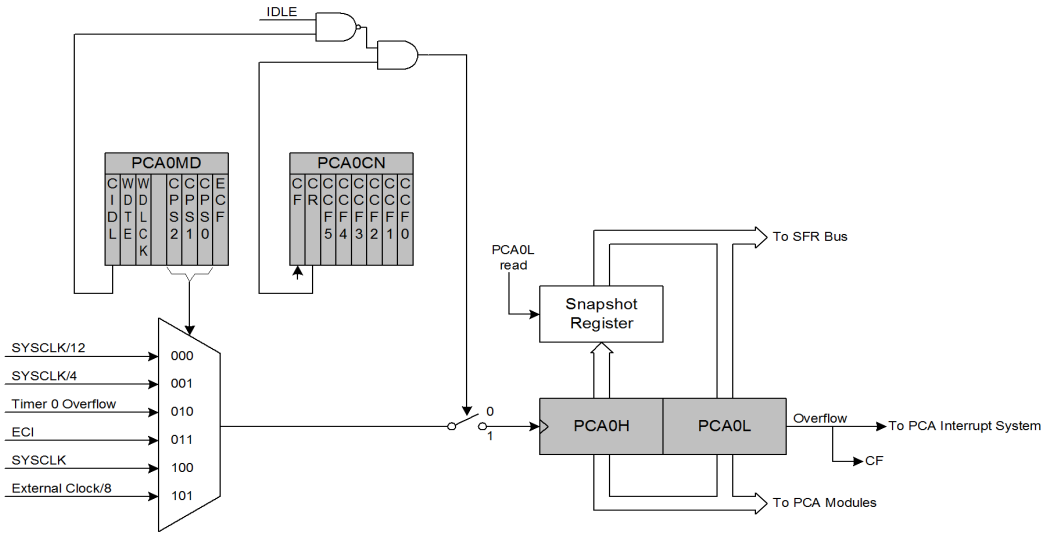


**Figure 24.2. PCA Counter/Timer Block Diagram**