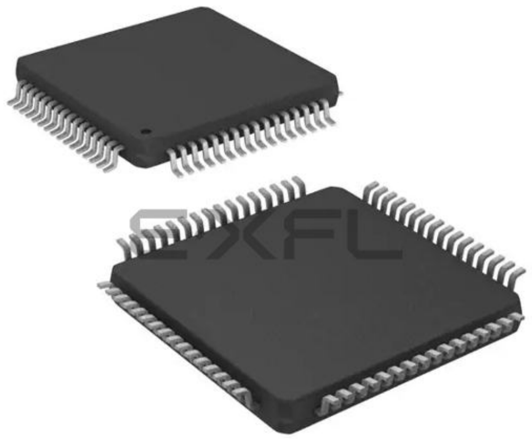


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

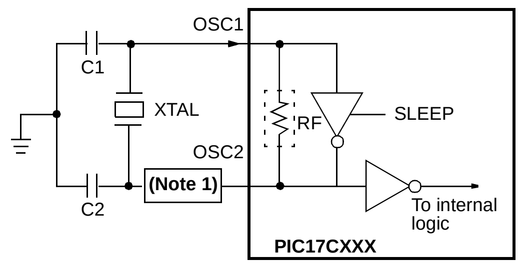
"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" 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	Active
Core Processor	PIC
Core Size	8-Bit
Speed	33MHz
Connectivity	I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	50
Program Memory Size	32KB (16K x 16)
Program Memory Type	OTP
EEPROM Size	-
RAM Size	902 x 8
Voltage - Supply (Vcc/Vdd)	4.5V ~ 5.5V
Data Converters	A/D 12x10b
Oscillator Type	External
Operating Temperature	0°C ~ 70°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/microchip-technology/pic17c756a-33-pt

NOTES:

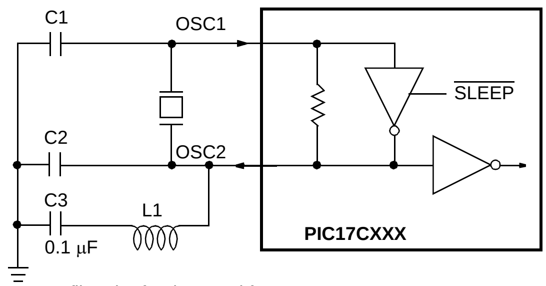
FIGURE 4-2: CRYSTAL OR CERAMIC RESONATOR OPERATION (XT OR LF OSC CONFIGURATION)



See Table 4-1 and Table 4-2 for recommended values of C1 and C2.

Note 1: A series resistor (Rs) may be required for AT strip cut crystals.

FIGURE 4-3: CRYSTAL OPERATION, OVERTONE CRYSTALS (XT OSC CONFIGURATION)



To filter the fundamental frequency:

$$\frac{1}{L1 * C2} = (2\pi f)^2$$

Where f = tank circuit resonant frequency. This should be midway between the fundamental and the 3rd overtone frequencies of the crystal.
C3 blocks DC current to ground.

TABLE 4-1: CAPACITOR SELECTION FOR CERAMIC RESONATORS

Oscillator Type	Resonator Frequency	Capacitor Range C1 = C2[1]
LF	455 kHz 2.0 MHz	15 - 68 pF 10 - 33 pF
XT	4.0 MHz 8.0 MHz 16.0 MHz	22 - 68 pF 33 - 100 pF 33 - 100 pF

Higher capacitance increases the stability of the oscillator, but also increases the start-up time. These values are for design guidance only. Since each resonator has its own characteristics, the user should consult the resonator manufacturer for appropriate values of external components.

Note 1: These values include all board capacitances on this pin. Actual capacitor value depends on board capacitance.

Resonators Used:

455 kHz	Panasonic EFO-A455K04B	± 0.3%
2.0 MHz	Murata Erie CSA2.00MG	± 0.5%
4.0 MHz	Murata Erie CSA4.00MG	± 0.5%
8.0 MHz	Murata Erie CSA8.00MT	± 0.5%
16.0 MHz	Murata Erie CSA16.00MX	± 0.5%

Resonators used did not have built-in capacitors.

TABLE 4-2: CAPACITOR SELECTION FOR CRYSTAL OSCILLATOR

Osc Type	Freq	C1[2]	C2[2]
LF	32 kHz 1 MHz 2 MHz	100-150 pF 10-68 pF 10-68 pF	100-150 pF 10-68 pF 10-68 pF
XT	2 MHz 4 MHz 8 MHz 16 MHz 24 MHz[1] 32 MHz[1]	47-100 pF 15-68 pF 15-47 pF 15-47 pF 15-47 pF 10-47 pF	47-100 pF 15-68 pF 15-47 pF 15-47 pF 15-47 pF 10-47 pF

Higher capacitance increases the stability of the oscillator, but also increases the start-up time and the oscillator current. These values are for design guidance only. Rs may be required in XT mode to avoid overdriving the crystals with low drive level specification. Since each crystal has its own characteristics, the user should consult the crystal manufacturer for appropriate values for external components.

Note 1: Overtone crystals are used at 24 MHz and higher. The circuit in Figure 4-3 should be used to select the desired harmonic frequency.

2: These values include all board capacitances on this pin. Actual capacitor value depends on board capacitance.

Crystals Used:

32.768 kHz	Epson C-001R32.768K-A	± 20 PPM
1.0 MHz	ECS-10-13-1	± 50 PPM
2.0 MHz	ECS-20-20-1	± 50 PPM
4.0 MHz	ECS-40-20-1	± 50 PPM
8.0 MHz	ECS ECS-80-S-4 ECS-80-18-1	± 50 PPM
16.0 MHz	ECS-160-20-1	± 50 PPM
25 MHz	CTS CTS25M	± 50 PPM
32 MHz	CRYSTEK HF-2	± 50 PPM

TABLE 7-3: SPECIAL FUNCTION REGISTERS (CONTINUED)

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	$\overline{\text{MCLR}}$, WDT
Bank 2											
10h	TMR1	Timer1's Register								xxxx xxxx	uuuu uuuu
11h	TMR2	Timer2's Register								xxxx xxxx	uuuu uuuu
12h	TMR3L	Timer3's Register; Low Byte								xxxx xxxx	uuuu uuuu
13h	TMR3H	Timer3's Register; High Byte								xxxx xxxx	uuuu uuuu
14h	PR1	Timer1's Period Register								xxxx xxxx	uuuu uuuu
15h	PR2	Timer2's Period Register								xxxx xxxx	uuuu uuuu
16h	PR3L/CA1L	Timer3's Period Register - Low Byte/Capture1 Register; Low Byte								xxxx xxxx	uuuu uuuu
17h	PR3H/CA1H	Timer3's Period Register - High Byte/Capture1 Register; High Byte								xxxx xxxx	uuuu uuuu
Bank 3											
10h	PW1DCL	DC1	DC0	—	—	—	—	—	—	xx-- ----	uu-- ----
11h	PW2DCL	DC1	DC0	TM2PW2	—	—	—	—	—	xx0- ----	uu0- ----
12h	PW1DCH	DC9	DC8	DC7	DC6	DC5	DC4	DC3	DC2	xxxx xxxx	uuuu uuuu
13h	PW2DCH	DC9	DC8	DC7	DC6	DC5	DC4	DC3	DC2	xxxx xxxx	uuuu uuuu
14h	CA2L	Capture2 Low Byte								xxxx xxxx	uuuu uuuu
15h	CA2H	Capture2 High Byte								xxxx xxxx	uuuu uuuu
16h	TCON1	CA2ED1	CA2ED0	CA1ED1	CA1ED0	T16	TMR3CS	TMR2CS	TMR1CS	0000 0000	0000 0000
17h	TCON2	CA2OVF	CA1OVF	PWM2ON	PWM1ON	CA1/$\overline{\text{PR3}}$	TMR3ON	TMR2ON	TMR1ON	0000 0000	0000 0000
Bank 4											
10h	PIR2	SSPIF	BCLIF	ADIF	—	CA4IF	CA3IF	TX2IF	RC2IF	000- 0010	000- 0010
11h	PIE2	SSPIE	BCLIE	ADIE	—	CA4IE	CA3IE	TX2IE	RC2IE	000- 0000	000- 0000
12h	Unimplemented	—	—	—	—	—	—	—	—	---- ----	---- ----
13h	RCSTA2	SPEN	RX9	SREN	CREN	—	FERR	OERR	RX9D	0000 -00x	0000 -00u
14h	RCREG2	Serial Port Receive Register for USART2								xxxx xxxx	uuuu uuuu
15h	TXSTA2	CSRC	TX9	TXEN	SYNC	—	—	TRMT	TX9D	0000 --1x	0000 --1u
16h	TXREG2	Serial Port Transmit Register for USART2								xxxx xxxx	uuuu uuuu
17h	SPBRG2	Baud Rate Generator for USART2								0000 0000	0000 0000
Bank 5:											
10h	DDRF	Data Direction Register for PORTF								1111 1111	1111 1111
11h	PORTF(4)	RF7/ AN11	RF6/ AN10	RF5/ AN9	RF4/ AN8	RF3/ AN7	RF2/ AN6	RF1/ AN5	RF0/ AN4	0000 0000	0000 0000
12h	DDRG	Data Direction Register for PORTG								1111 1111	1111 1111
13h	PORTG(4)	RG7/ TX2/CK2	RG6/ RX2/DT2	RG5/ PWM3	RG4/ CAP3	RG3/ AN0	RG2/ AN1	RG1/ AN2	RG0/ AN3	xxxx 0000	uuuu 0000
14h	ADCON0	CHS3	CHS2	CHS1	CHS0	—	GO/$\overline{\text{DONE}}$	—	ADON	0000 -0-0	0000 -0-0
15h	ADCON1	ADCS1	ADCS0	ADFM	—	PCFG3	PCFG2	PCFG1	PCFG0	000- 0000	000- 0000
16h	ADRESL	A/D Result Register Low Byte								xxxx xxxx	uuuu uuuu
17h	ADRESH	A/D Result Register High Byte								xxxx xxxx	uuuu uuuu

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0', q = value depends on condition.
Shaded cells are unimplemented, read as '0'.

Note 1: The upper byte of the program counter is not directly accessible. PCLATH is a holding register for PC<15:8> whose contents are updated from, or transferred to, the upper byte of the program counter.

2: The $\overline{\text{TO}}$ and $\overline{\text{PD}}$ status bits in CPUSTA are not affected by a $\overline{\text{MCLR}}$ Reset.

3: Bank 8 and associated registers are only implemented on the PIC17C76X devices.

4: This is the value that will be in the port output latch.

5: When the device is configured for Microprocessor or Extended Microcontroller mode, the operation of this port does not rely on these registers.

6: On any device RESET, these pins are configured as inputs.

7.2.2.1 ALU Status Register (ALUSTA)

The ALUSTA register contains the status bits of the Arithmetic and Logic Unit and the mode control bits for the indirect addressing register.

As with all the other registers, the ALUSTA register can be the destination for any instruction. If the ALUSTA register is the destination for an instruction that affects the Z, DC, C, or OV bits, then the write to these three bits is disabled. These bits are set or cleared according to the device logic. Therefore, the result of an instruction with the ALUSTA register as destination may be different than intended.

For example, the `CLRF ALUSTA, F` instruction will clear the upper four bits and set the Z bit. This leaves the ALUSTA register as `0000u1uu` (where `u` = unchanged).

It is recommended, therefore, that only `BCF`, `BSF`, `SWAPF` and `MOVWF` instructions be used to alter the ALUSTA register, because these instructions do not affect any status bits. To see how other instructions affect the status bits, see the "Instruction Set Summary."

> **Note 1:** The C and DC bits operate as a $\overline{\text{borrow}}$ and $\overline{\text{digit borrow}}$ bit, respectively, in subtraction. See the `SUBLW` and `SUBWF` instructions for examples.
>
> **2:** The overflow bit will be set if the 2's complement result exceeds +127, or is less than -128.

The Arithmetic and Logic Unit (ALU) is capable of carrying out arithmetic or logical operations on two operands, or a single operand. All single operand instructions operate either on the WREG register, or the given file register. For two operand instructions, one of the operands is the WREG register and the other is either a file register, or an 8-bit immediate constant.

REGISTER 7-1: ALUSTA REGISTER (ADDRESS: 04h, UNBANKED)

R/W-1	R/W-1	R/W-1	R/W-1	R/W-x	R/W-x	R/W-x	R/W-x
FS3	FS2	FS1	FS0	OV	Z	DC	C
bit 7							bit 0

bit 7-6 **FS3:FS2**: FSR1 Mode Select bits
`00` = Post auto-decrement FSR1 value
`01` = Post auto-increment FSR1 value
`1x` = FSR1 value does not change

bit 5-4 **FS1:FS0**: FSR0 Mode Select bits
`00` = Post auto-decrement FSR0 value
`01` = Post auto-increment FSR0 value
`1x` = FSR0 value does not change

bit 3 **OV**: Overflow bit
This bit is used for signed arithmetic (2's complement). It indicates an overflow of the 7-bit magnitude, which causes the sign bit (bit7) to change state.
`1` = Overflow occurred for signed arithmetic (in this arithmetic operation)
`0` = No overflow occurred

bit 2 **Z**: Zero bit
`1` = The result of an arithmetic or logic operation is zero
`0` = The result of an arithmetic or logic operation is not zero

bit 1 **DC**: Digit carry/$\overline{\text{borrow}}$ bit
For `ADDWF` and `ADDLW` instructions.
`1` = A carry-out from the 4th low order bit of the result occurred
`0` = No carry-out from the 4th low order bit of the result

> **Note:** For borrow, the polarity is reversed.

bit 0 **C**: Carry/$\overline{\text{borrow}}$ bit
For `ADDWF` and `ADDLW` instructions. Note that a subtraction is executed by adding the two's complement of the second operand.
For rotate (`RRCF`, `RLCF`) instructions, this bit is loaded with either the high or low order bit of the source register.
`1` = A carry-out from the Most Significant bit of the result occurred
`0` = No carry-out from the Most Significant bit of the result

> **Note:** For borrow, the polarity is reversed.

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR Reset	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

NOTES:

9.0 HARDWARE MULTIPLIER

All PIC17C7XX devices have an 8 x 8 hardware multiplier included in the ALU of the device. By making the multiply a hardware operation, it completes in a single instruction cycle. This is an unsigned multiply that gives a 16-bit result. The result is stored into the 16-bit Product register (PRODH:PRODL). The multiplier does not affect any flags in the ALUSTA register.

Making the 8 x 8 multiplier execute in a single cycle gives the following advantages:

- Higher computational throughput
- Reduces code size requirements for multiply algorithms

The performance increase allows the device to be used in applications previously reserved for Digital Signal Processors.

Table 9-1 shows a performance comparison between PIC17CXXX devices using the single cycle hardware multiply and performing the same function without the hardware multiply.

Example 9-1 shows the sequence to do an 8 x 8 unsigned multiply. Only one instruction is required when one argument of the multiply is already loaded in the WREG register.

Example 9-2 shows the sequence to do an 8 x 8 signed multiply. To account for the sign bits of the arguments, each argument's most significant bit (MSb) is tested and the appropriate subtractions are done.

EXAMPLE 9-1: 8 x 8 UNSIGNED MULTIPLY ROUTINE

```
MOVFP    ARG1, WREG  ;
MULWF    ARG2        ; ARG1 * ARG2 ->
                     ;   PRODH:PRODL
```

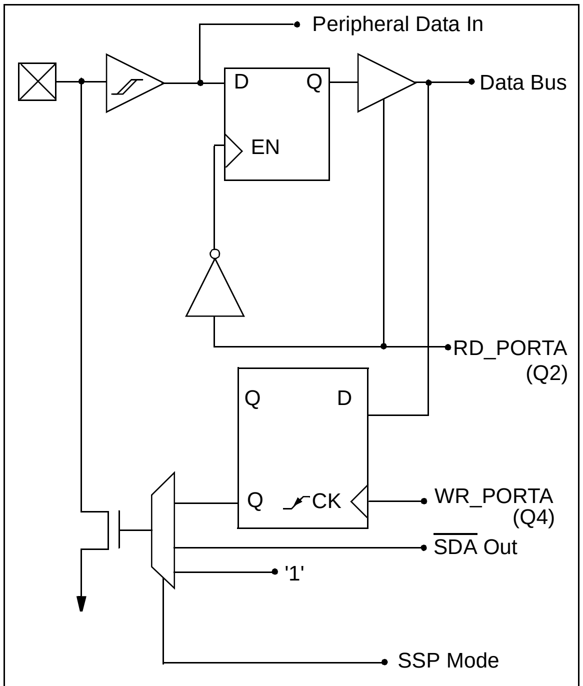
EXAMPLE 9-2: 8 x 8 SIGNED MULTIPLY ROUTINE

```
MOVFP    ARG1, WREG
MULWF    ARG2        ; ARG1 * ARG2 ->
                     ;   PRODH:PRODL
BTFSC    ARG2, SB    ; Test Sign Bit
SUBWF    PRODH, F    ; PRODH = PRODH
                     ;         - ARG1
MOVFP    ARG2, WREG
BTFSC    ARG1, SB    ; Test Sign Bit
SUBWF    PRODH, F    ; PRODH = PRODH
                     ;         - ARG2
```

TABLE 9-1: PERFORMANCE COMPARISON

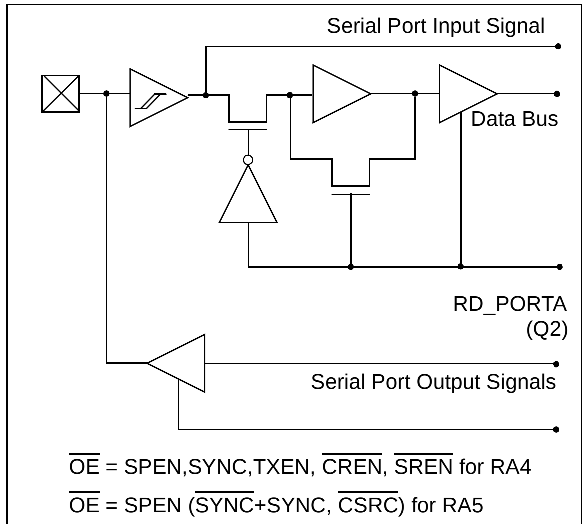
Routine	Multiply Method	Program Memory (Words)	Cycles (Max)	Time @ 33 MHz	Time @ 16 MHz	Time @ 8 MHz
8 x 8 unsigned	Without hardware multiply	13	69	8.364 μs	17.25 μs	34.50 μs
	Hardware multiply	1	1	0.121 μs	0.25 μs	0.50 μs
8 x 8 signed	Without hardware multiply	—	—	—	—	—
	Hardware multiply	6	6	0.727 μs	1.50 μs	3.0 μs
16 x 16 unsigned	Without hardware multiply	21	242	29.333 μs	60.50 μs	121.0 μs
	Hardware multiply	24	24	2.91 μs	6.0 μs	12.0 μs
16 x 16 signed	Without hardware multiply	52	254	30.788 μs	63.50 μs	127.0 μs
	Hardware multiply	36	36	4.36 μs	9.0 μs	18.0 μs

FIGURE 10-3: RA3 BLOCK DIAGRAM



Note: I/O pin has protection diodes to VSS.

FIGURE 10-4: RA4 AND RA5 BLOCK DIAGRAM



Note: I/O pins have protection diodes to VDD and VSS.

TABLE 10-1: PORTA FUNCTIONS

Name	Bit0	Buffer Type	Function
RA0/INT	bit0	ST	Input or external interrupt input.
RA1/T0CKI	bit1	ST	Input or clock input to the TMR0 timer/counter and/or an external interrupt input.
RA2/SS/SCL	bit2	ST	Input/output or slave select input for the SPI, or clock input for the I^{2}C bus. Output is open drain type.
RA3/SDI/SDA	bit3	ST	Input/output or data input for the SPI, or data for the I^{2}C bus. Output is open drain type.
RA4/RX1/DT1	bit4	ST	Input or USART1 Asynchronous Receive input, or USART1 Synchronous Data input/output.
RA5/TX1/CK1	bit5	ST	Input or USART1 Asynchronous Transmit output, or USART1 Synchronous Clock input/output.
RBPU	bit7	—	Control bit for PORTB weak pull-ups.

Legend: ST = Schmitt Trigger input

TABLE 10-2: REGISTERS/BITS ASSOCIATED WITH PORTA

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	MCLR, WDT
10h, Bank 0	PORTA(1)	RBPU	—	RA5/TX1/CK1	RA4/RX1/DT1	RA3/SDI/SDA	RA2/SS/SCL	RA1/T0CKI	RA0/INT	0-xx 11xx	0-uu 11uu
05h, Unbanked	T0STA	INTEDG	T0SE	T0CS	T0PS3	T0PS2	T0PS1	T0PS0	—	0000 000-	0000 000-
13h, Bank 0	RCSTA1	SPEN	RX9	SREN	CREN	—	FERR	OERR	RX9D	0000 -00x	0000 -00u
15h, Bank 0	TXSTA1	CSRC	TX9	TXEN	SYNC	—	—	TRMT	TX9D	0000 --1x	0000 --1u

Legend: x = unknown, u = unchanged, - = unimplemented, reads as '0'. Shaded cells are not used by PORTA.
Note 1: On any device RESET, these pins are configured as inputs.

13.1.3 USING PULSE WIDTH MODULATION (PWM) OUTPUTS WITH TIMER1 AND TIMER2

Three high speed pulse width modulation (PWM) outputs are provided. The PWM1 output uses Timer1 as its time base, while PWM2 and PWM3 may independently be software configured to use either Timer1 or Timer2 as the time base. The PWM outputs are on the RB2/PWM1, RB3/PWM2 and RG5/PWM3 pins.

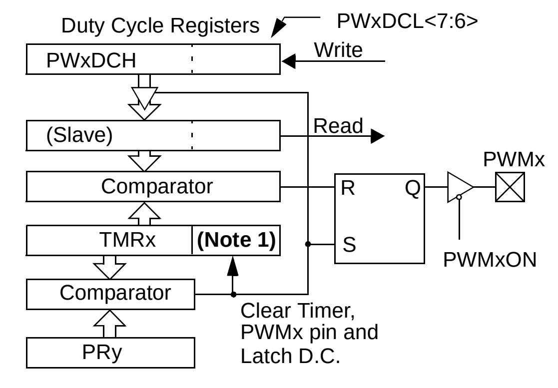
Each PWM output has a maximum resolution of 10-bits. At 10-bit resolution, the PWM output frequency is 32.2 kHz (@ 32 MHz clock) and at 8-bit resolution the PWM output frequency is 128.9 kHz. The duty cycle of the output can vary from 0% to 100%.

Figure 13-3 shows a simplified block diagram of a PWM module.

The duty cycle registers are double buffered for glitch free operation. Figure 13-4 shows how a glitch could occur if the duty cycle registers were not double buffered.

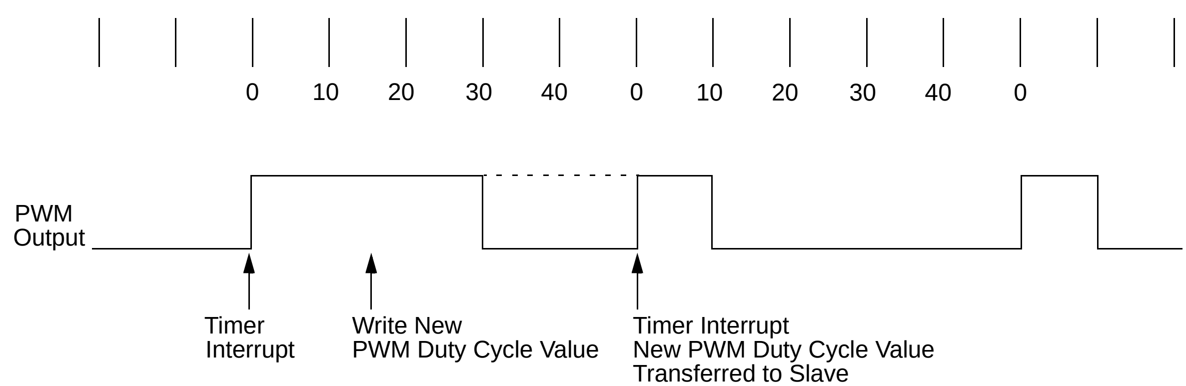
The user needs to set the PWM1ON bit (TCON2<4>) to enable the PWM1 output. When the PWM1ON bit is set, the RB2/PWM1 pin is configured as PWM1 output and forced as an output, irrespective of the data direction bit (DDRB<2>). When the PWM1ON bit is clear, the pin behaves as a port pin and its direction is controlled by its data direction bit (DDRB<2>). Similarly, the PWM2ON (TCON2<5>) bit controls the configuration of the RB3/PWM2 pin and the PWM3ON (TCON3<0>) bit controls the configuration of the RG5/PWM3 pin.

FIGURE 13-3: SIMPLIFIED PWM BLOCK DIAGRAM



Note 1: 8-bit timer is concatenated with 2-bit internal Q clock or 2 bits of the prescaler to create 10-bit time base.

FIGURE 13-4: PWM OUTPUT (NOT BUFFERED)



Note: The dotted line shows PWM output if duty cycle registers were not double buffered. If the new duty cycle is written after the timer has passed that value, then the PWM does not reset at all during the current cycle, causing a “glitch”.

In this example, PWM period = 50. Old duty cycle is 30. New duty cycle value is 10.

FIGURE 13-8: TIMER1, TIMER2 AND TIMER3 OPERATION (IN TIMER MODE)

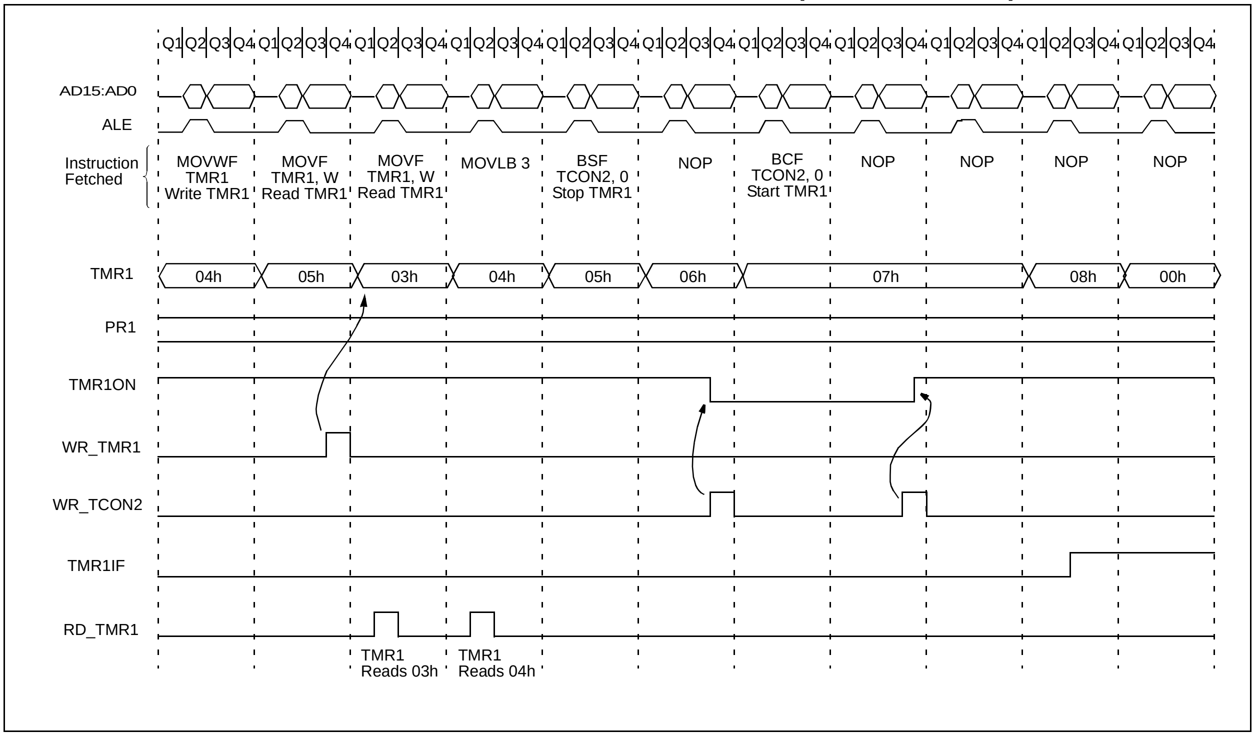


TABLE 14-10: REGISTERS ASSOCIATED WITH SYNCHRONOUS SLAVE TRANSMISSION

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	MCLR, WDT
16h, Bank 1	PIR1	RBIF	TMR3IF	TMR2IF	TMR1IF	CA2IF	CA1IF	TX1IF	RC1IF	x000 0010	u000 0010
17h, Bank 1	PIE1	RBIE	TMR3IE	TMR2IE	TMR1IE	CA2IE	CA1IE	TX1IE	RC1IE	0000 0000	0000 0000
13h, Bank 0	RCSTA1	SPEN	RX9	SREN	CREN	—	FERR	OERR	RX9D	0000 -00x	0000 -00u
15h, Bank 0	TXSTA1	CSRC	TX9	TXEN	SYNC	—	—	TRMT	TX9D	0000 --1x	0000 --1u
16h, Bank 0	TXREG1	TX7	TX6	TX5	TX4	TX3	TX2	TX1	TX0	xxxx xxxx	uuuu uuuu
17h, Bank 0	SPBRG1	Baud Rate Generator Register								0000 0000	0000 0000
10h, Bank 4	PIR2	SSPIF	BCLIF	ADIF	—	CA4IF	CA3IF	TX2IF	RC2IF	000- 0010	000- 0010
11h, Bank 4	PIE2	SSPIE	BCLIE	ADIE	—	CA4IE	CA3IE	TX2IE	RC2IE	000- 0000	000- 0000
13h, Bank 4	RCSTA2	SPEN	RX9	SREN	CREN	—	FERR	OERR	RX9D	0000 -00x	0000 -00u
16h, Bank 4	TXREG2	TX7	TX6	TX5	TX4	TX3	TX2	TX1	TX0	xxxx xxxx	uuuu uuuu
15h, Bank 4	TXSTA2	CSRC	TX9	TXEN	SYNC	—	—	TRMT	TX9D	0000 --1x	0000 --1u
17h, Bank 4	SPBRG2	Baud Rate Generator Register								0000 0000	0000 0000

Legend: x = unknown, u = unchanged, - = unimplemented, read as a '0'. Shaded cells are not used for synchronous slave transmission.

TABLE 14-11: REGISTERS ASSOCIATED WITH SYNCHRONOUS SLAVE RECEPTION

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	MCLR, WDT
16h, Bank1	PIR1	RBIF	TMR3IF	TMR2IF	TMR1IF	CA2IF	CA1IF	TX1IF	RC1IF	x000 0010	u000 0010
17h, Bank1	PIE1	RBIE	TMR3IE	TMR2IE	TMR1IE	CA2IE	CA1IE	TX1IE	RC1IE	0000 0000	0000 0000
13h, Bank0	RCSTA1	SPEN	RX9	SREN	CREN	—	FERR	OERR	RX9D	0000 -00x	0000 -00u
14h, Bank0	RCREG1	RX7	RX6	RX5	RX4	RX3	RX2	RX1	RX0	xxxx xxxx	uuuu uuuu
15h, Bank 0	TXSTA1	CSRC	TX9	TXEN	SYNC	—	—	TRMT	TX9D	0000 --1x	0000 --1u
17h, Bank 0	SPBRG1	Baud Rate Generator Register								0000 0000	0000 0000
10h, Bank 4	PIR2	SSPIF	BCLIF	ADIF	—	CA4IF	CA3IF	TX2IF	RC2IF	000- 0010	000- 0010
11h, Bank 4	PIE2	SSPIE	BCLIE	ADIE	—	CA4IE	CA3IE	TX2IE	RC2IE	000- 0000	000- 0000
13h, Bank 4	RCSTA2	SPEN	RX9	SREN	CREN	—	FERR	OERR	RX9D	0000 -00x	0000 -00u
14h, Bank 4	RCREG2	RX7	RX6	RX5	RX4	RX3	RX2	RX1	RX0	xxxx xxxx	uuuu uuuu
15h, Bank 4	TXSTA2	CSRC	TX9	TXEN	SYNC	—	—	TRMT	TX9D	0000 --1x	0000 --1u
17h, Bank 4	SPBRG2	Baud Rate Generator Register								0000 0000	0000 0000

Legend: x = unknown, u = unchanged, - = unimplemented, read as a '0'. Shaded cells are not used for synchronous slave reception.

15.1.5 SLAVE MODE

In Slave mode, the data is transmitted and received as the external clock pulses appear on SCK. When the last bit is latched, the interrupt flag bit SSPIF (PIR2<7>) is set.

While in Slave mode, the external clock is supplied by the external clock source on the SCK pin. This external clock must meet the minimum high and low times as specified in the electrical specifications.

While in SLEEP mode, the slave can transmit/receive data. When a byte is received, the device will wake-up from SLEEP.

15.1.6 SLAVE SELECT SYNCHRONIZATION

The $\overline{SS}$ pin allows a Synchronous Slave mode. The SPI must be in Slave mode with $\overline{SS}$ pin control enabled (SSPCON1<3:0> = 04h). The pin must not be driven low for the $\overline{SS}$ pin to function as an input. The RA2 Data Latch must be high. When the $\overline{SS}$ pin is low, transmission and reception are enabled and the SDO pin is driven. When the $\overline{SS}$ pin goes high, the SDO pin is no longer driven, even if in the middle of a transmitted byte and becomes a floating output. External pull-up/pull-down resistors may be desirable, depending on the application.

> **Note 1:** When the SPI is in Slave mode with $\overline{SS}$ pin control enabled (SSPCON<3:0> = `0100`), the SPI module will reset if the $\overline{SS}$ pin is set to VDD.
>
> **2:** If the SPI is used in Slave mode with CKE = '1', then the $\overline{SS}$ pin control must be enabled.

When the SPI module resets, the bit counter is forced to 0. This can be done by either forcing the $\overline{SS}$ pin to a high level, or clearing the SSPEN bit.

To emulate two-wire communication, the SDO pin can be connected to the SDI pin. When the SPI needs to operate as a receiver, the SDO pin can be configured as an input. This disables transmissions from the SDO. The SDI can always be left as an input (SDI function), since it cannot create a bus conflict.

FIGURE 15-7: SLAVE SYNCHRONIZATION WAVEFORM

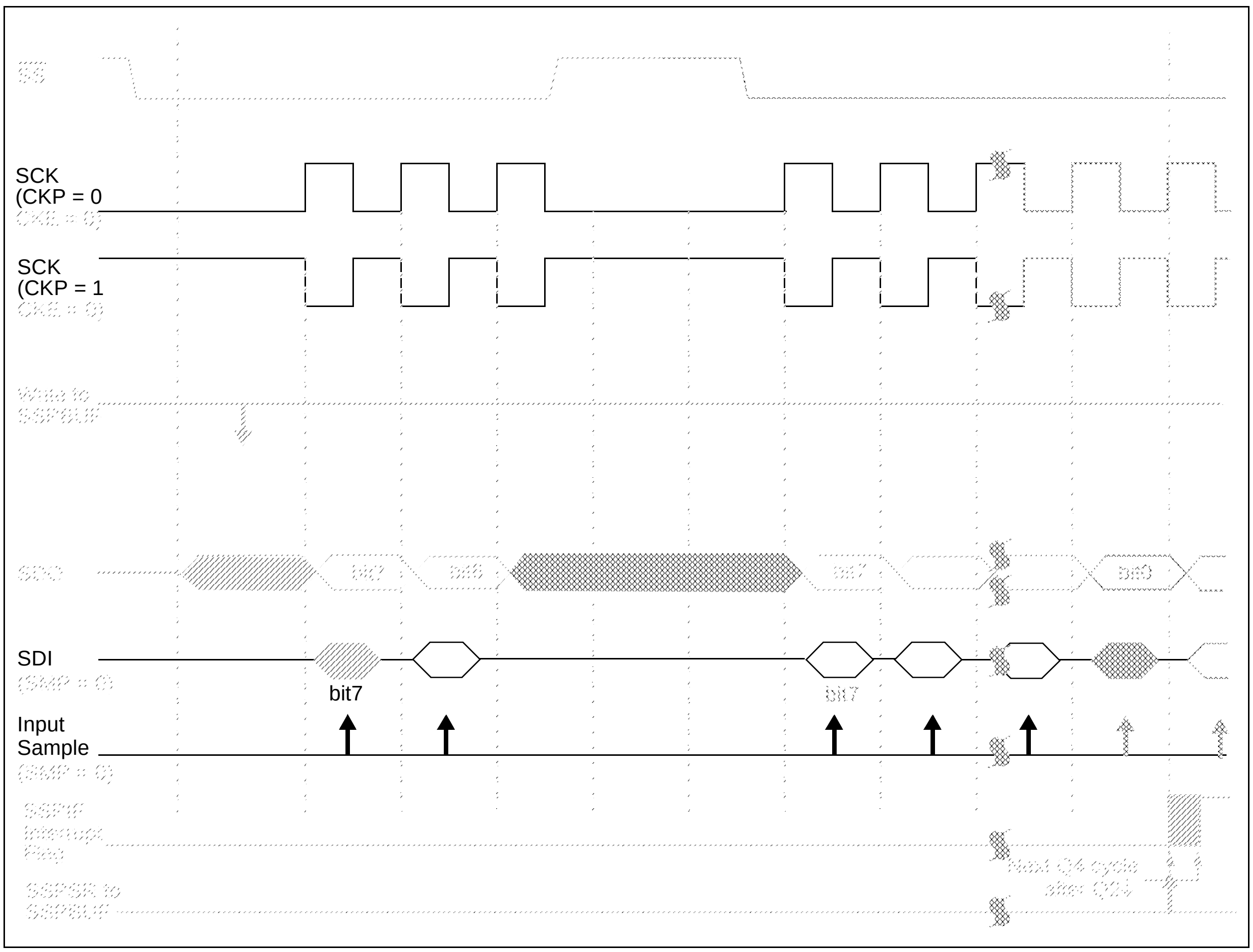


FIGURE 15-8: SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 0)

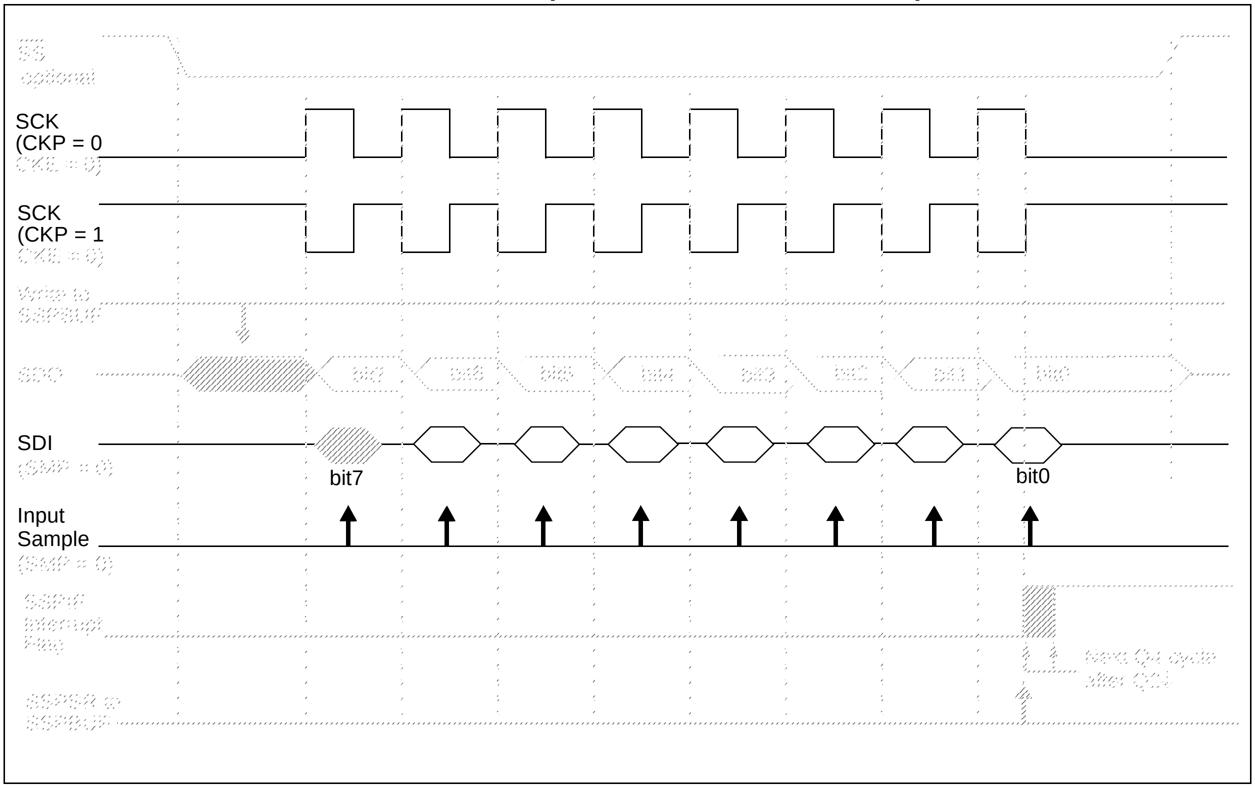
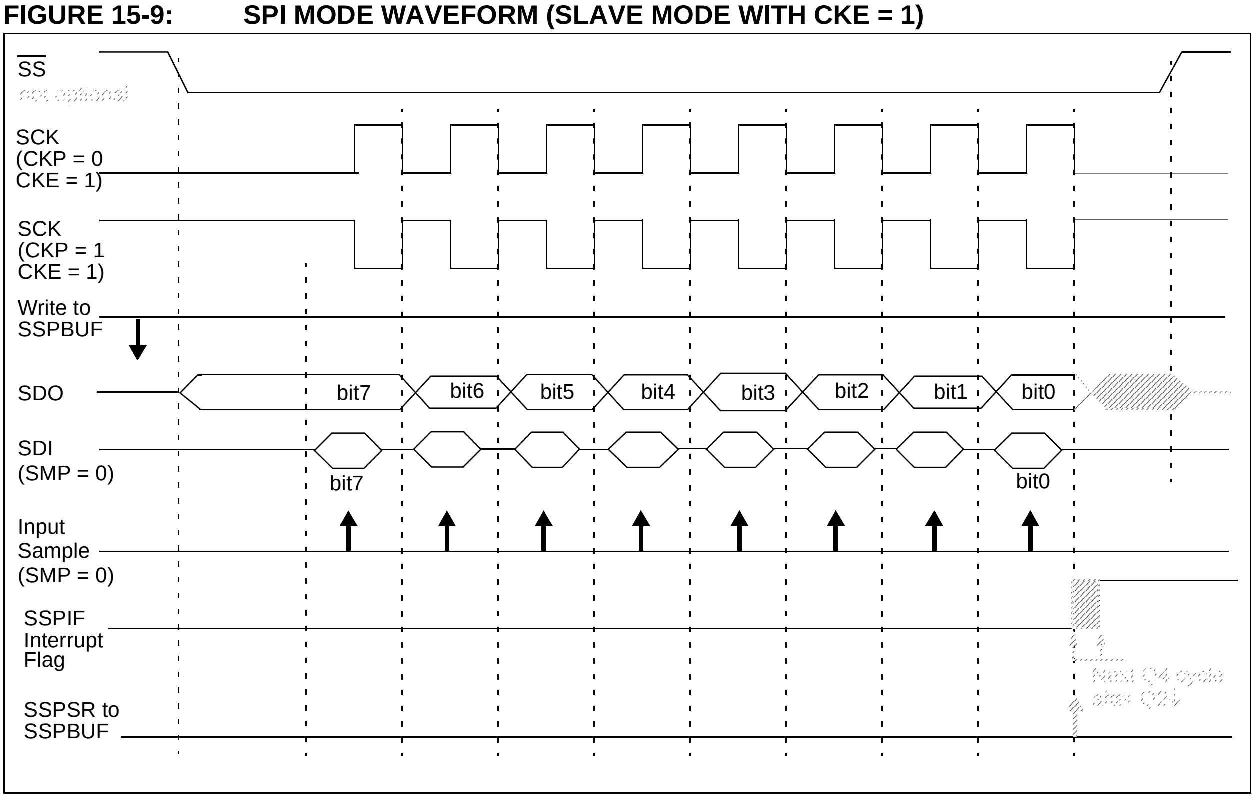


FIGURE 15-9: SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 1)



15.3 Connection Considerations for I^2C Bus

For standard mode I^2C bus devices, the values of resistors R_p R_s in Figure 15-42 depends on the following parameters:

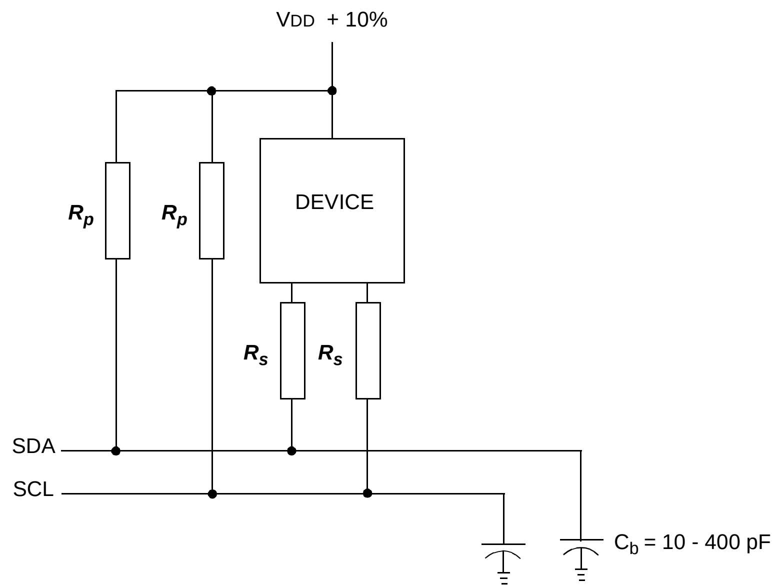
- Supply voltage
- Bus capacitance
- Number of connected devices (input current + leakage current)

The supply voltage limits the minimum value of resistor R_p due to the specified minimum sink current of 3 mA at VOL max = 0.4V for the specified output stages. For example, with a supply voltage of VDD = 5V ±10% and VOL max = 0.4V at 3 mA, R_p min = (5.5-0.4)/0.003 = 1.7 kΩ. VDD as a function of R_p is shown in Figure 15-42. The desired noise margin of 0.1 VDD for the low level, limits the maximum value of R_s. Series resistors are optional and used to improve ESD susceptibility.

The bus capacitance is the total capacitance of wire, connections and pins. This capacitance limits the maximum value of R_p due to the specified rise time (Figure 15-42).

The SMP bit is the slew rate control enabled bit. This bit is in the SSPSTAT register and controls the slew rate of the I/O pins when in I^2C mode (master or slave).

FIGURE 15-42: SAMPLE DEVICE CONFIGURATION FOR I^2C BUS



Note: I^2C devices with input levels related to VDD must have one common supply line to which the pull-up resistor is also connected.

The ADRESH:ADRESL registers contain the 10-bit result of the A/D conversion. When the A/D conversion is complete, the result is loaded into this A/D result register pair, the GO/$\overline{\text{DONE}}$ bit (ADCON0<2>) is cleared and A/D interrupt flag bit, ADIF is set. The block diagrams of the A/D module are shown in Figure 16-1.

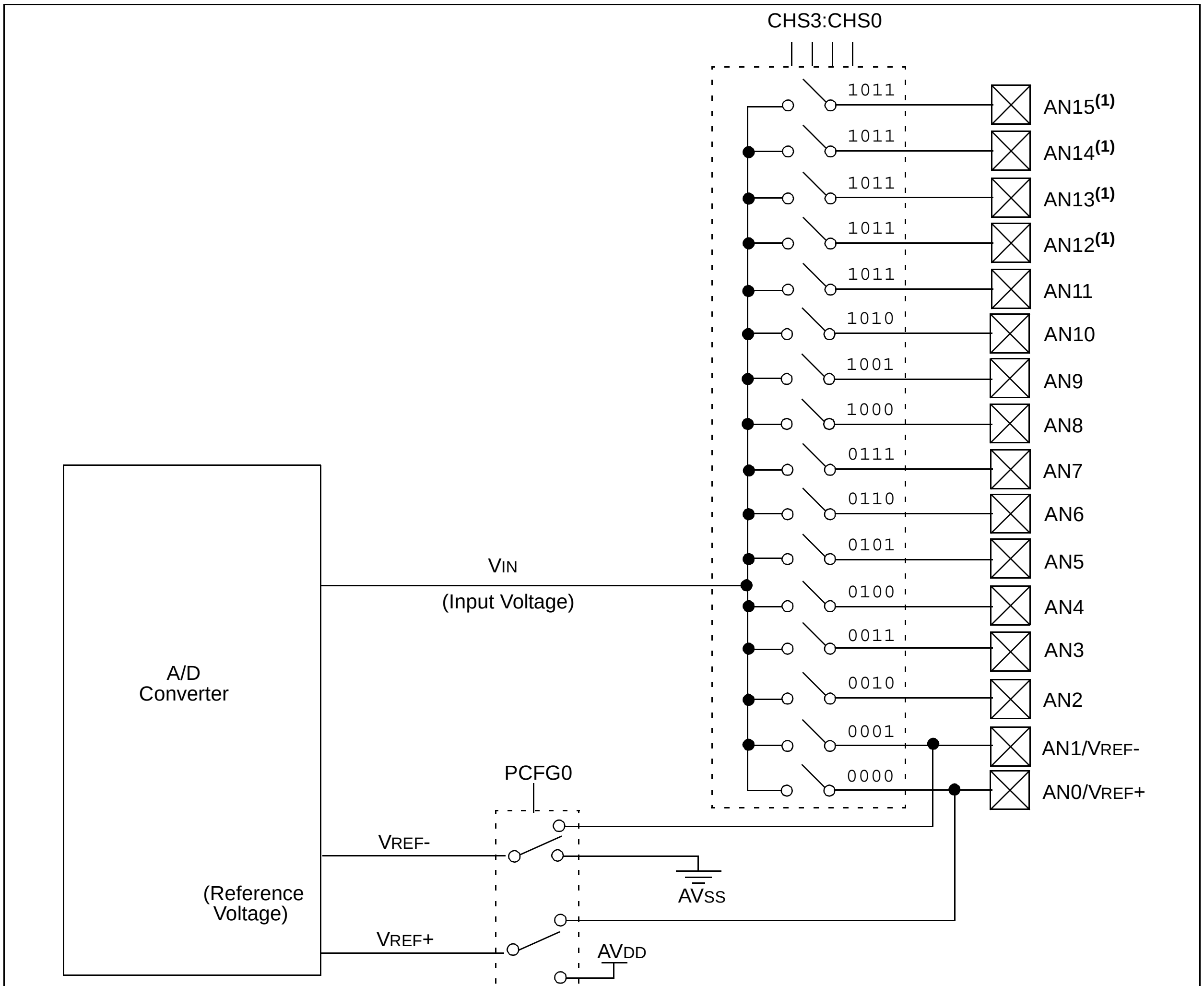
After the A/D module has been configured as desired, the selected channel must be acquired before the conversion is started. The analog input channels must have their corresponding DDR bits selected as inputs. To determine sample time, see Section 16.1. After this acquisition time has elapsed, the A/D conversion can be started. The following steps should be followed for doing an A/D conversion:

1. Configure the A/D module:
   a) Configure analog pins/voltage reference/ and digital I/O (ADCON1)
   b) Select A/D input channel (ADCON0)
   c) Select A/D conversion clock (ADCON0)
   d) Turn on A/D module (ADCON0)
2. Configure A/D interrupt (if desired):
   a) Clear ADIF bit
   b) Set ADIE bit
   c) Clear GLINTD bit
3. Wait the required acquisition time.
4. Start conversion:
   a) Set GO/$\overline{\text{DONE}}$ bit (ADCON0)
5. Wait for A/D conversion to complete, by either:
   a) Polling for the GO/$\overline{\text{DONE}}$ bit to be cleared

   OR

   b) Waiting for the A/D interrupt
6. Read A/D Result register pair (ADRESH:ADRESL), clear bit ADIF, if required.
7. For next conversion, go to step 1 or step 2, as required. The A/D conversion time per bit is defined as T_{AD}. A minimum wait of $2T_{AD}$ is required before next acquisition starts.

FIGURE 16-1: A/D BLOCK DIAGRAM



Note 1: These channels are only available on PIC16C76X devices.

17.4 Power-down Mode (SLEEP)

The Power-down mode is entered by executing a SLEEP instruction. This clears the Watchdog Timer and postscaler (if enabled). The $\overline{\text{PD}}$ bit is cleared and the $\overline{\text{TO}}$ bit is set (in the CPUSTA register). In SLEEP mode, the oscillator driver is turned off. The I/O ports maintain their status (driving high,low, or hi-impedance input).

The $\overline{\text{MCLR}}$/VPP pin must be at a logic high level (VIHMC). A WDT time-out RESET does not drive the $\overline{\text{MCLR}}$/VPP pin low.

17.4.1 WAKE-UP FROM SLEEP

The device can wake-up from SLEEP through one of the following events:

- Power-on Reset
- Brown-out Reset
- External RESET input on $\overline{\text{MCLR}}$/VPP pin
- WDT Reset (if WDT was enabled)
- Interrupt from RA0/INT pin, RB port change, T0CKI interrupt, or some peripheral interrupts

The following peripheral interrupts can wake the device from SLEEP:

- Capture interrupts
- USART synchronous slave transmit interrupts
- USART synchronous slave receive interrupts
- A/D conversion complete
- SPI slave transmit/receive complete
- I^2C slave receive

Other peripherals cannot generate interrupts since during SLEEP, no on-chip Q clocks are present.

Any RESET event will cause a device RESET. Any interrupt event is considered a continuation of program execution. The $\overline{\text{TO}}$ and $\overline{\text{PD}}$ bits in the CPUSTA register can be used to determine the cause of a device RESET. The $\overline{\text{PD}}$ bit, which is set on power-up, is cleared when SLEEP is invoked. The $\overline{\text{TO}}$ bit is cleared if WDT time-out occurred (and caused a RESET).

When the SLEEP instruction is being executed, the next instruction (PC + 1) is pre-fetched. For the device to wake-up through an interrupt event, the corresponding interrupt enable bit must be set (enabled). Wake-up is regardless of the state of the GLINTD bit. If the GLINTD bit is set (disabled), the device continues execution at the instruction after the SLEEP instruction. If the GLINTD bit is clear (enabled), the device executes the instruction after the SLEEP instruction and then branches to the interrupt vector address. In cases where the execution of the instruction following SLEEP is not desirable, the user should have a NOP after the SLEEP instruction.

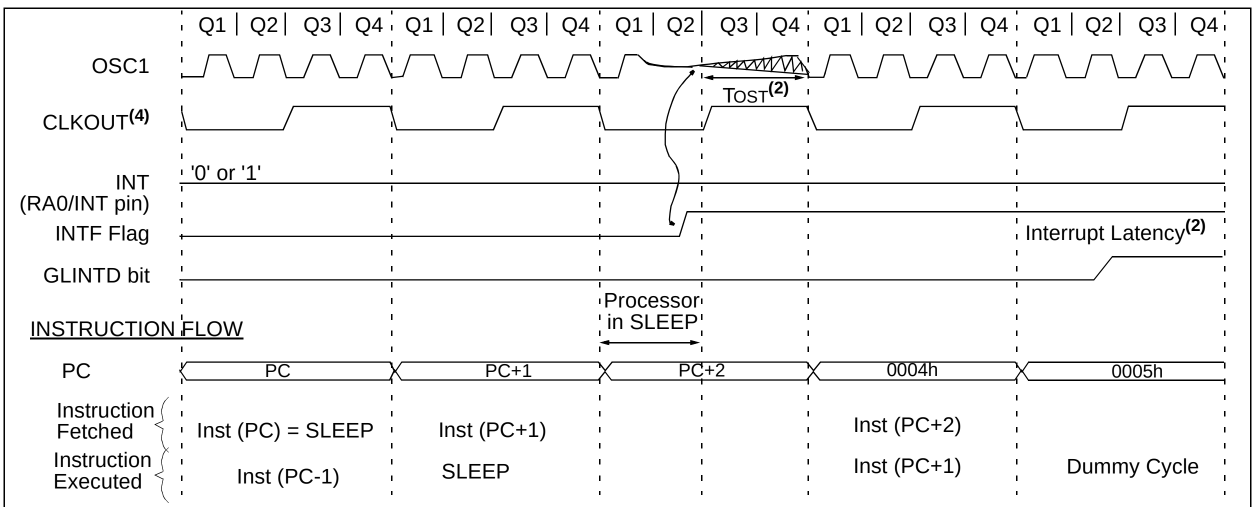
Note:	If the global interrupt is disabled (GLINTD is set), but any interrupt source has both its interrupt enable bit and the corresponding interrupt flag bit set, the device will immediately wake-up from SLEEP. The $\overline{\text{TO}}$ bit is set and the $\overline{\text{PD}}$ bit is cleared.

The WDT is cleared when the device wakes from SLEEP, regardless of the source of wake-up.

17.4.1.1 Wake-up Delay

When the oscillator type is configured in XT or LF mode, the Oscillator Start-up Timer (OST) is activated on wake-up. The OST will keep the device in RESET for 1024TOSC. This needs to be taken into account when considering the interrupt response time when coming out of SLEEP.

FIGURE 17-2: WAKE-UP FROM SLEEP THROUGH INTERRUPT



Note 1: XT or LF oscillator mode assumed.
2: TOST = 1024TOSC (drawing not to scale). This delay will not be there for RC osc mode.
3: When GLINTD = 0, processor jumps to interrupt routine after wake-up. If GLINTD = 1, execution will continue in line.
4: CLKOUT is not available in these osc modes, but shown here for timing reference.

17.6 In-Circuit Serial Programming

The PIC17C7XX group of the high-end family (PIC17CXXX) has an added feature that allows serial programming while in the end application circuit. This is simply done with two lines for clock and data and three other lines for power, ground, and the programming voltage. This allows customers to manufacture boards with unprogrammed devices and then program the microcontroller just before shipping the product. This also allows the most recent firmware, or a custom firmware to be programmed.

Devices may be serialized to make the product unique; “special” variants of the product may be offered and code updates are possible. This allows for increased design flexibility.

To place the device into the Serial Programming Test mode, two pins will need to be placed at V_{IHH}. These are the TEST pin and the $\overline{MCLR}/V_{PP}$ pin. Also, a sequence of events must occur as follows:

1. The TEST pin is placed at V_{IHH}.
2. The $\overline{MCLR}/V_{PP}$ pin is placed at V_{IHH}.

There is a setup time between step 1 and step 2 that must be met.

After this sequence, the Program Counter is pointing to program memory address 0xFF60. This location is in the Boot ROM. The code initializes the USART/SCI so that it can receive commands. For this, the device must be clocked. The device clock source in this mode is the RA1/T0CKI pin. After delaying to allow the USART/SCI to initialize, commands can be received. The flow is shown in these 3 steps:

1. The device clock source starts.
2. Wait 80 device clocks for Boot ROM code to configure the USART/SCI.
3. Commands may now be sent.

For complete details of serial programming, please refer to the PIC17C7XX Programming Specification. (Contact your local Microchip Technology Sales Office for availability.)

FIGURE 17-3: TYPICAL IN-CIRCUIT SERIAL PROGRAMMING CONNECTION

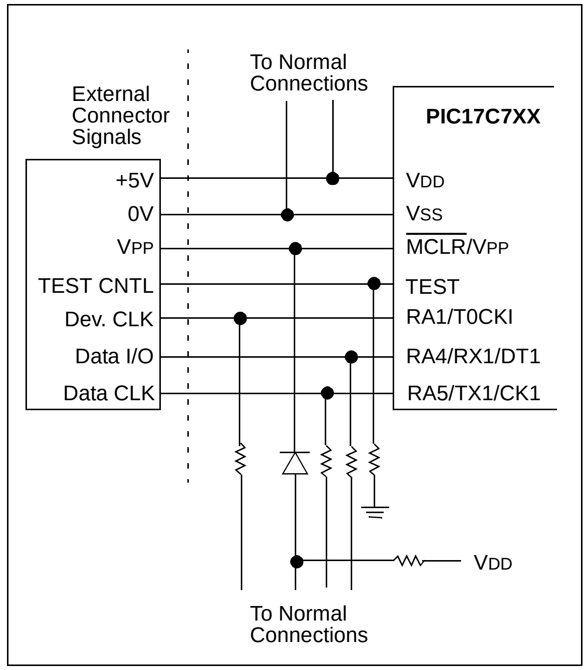


TABLE 17-3: ICSP INTERFACE PINS

Name	During Programming		
	Function	**Type**	**Description**
RA4/RX1/DT1	DT	I/O	Serial Data
RA5/TX1/CK1	CK	I	Serial Clock
RA1/T0CKI	OSCI	I	Device Clock Source
TEST	TEST	I	Test mode selection control input, force to V_{IHH}
$\overline{MCLR}/V_{PP}$	$\overline{MCLR}/V_{PP}$	P	Master Clear Reset and Device Programming Voltage
V_{DD}	V_{DD}	P	Positive supply for logic and I/O pins
V_{SS}	V_{SS}	P	Ground reference for logic and I/O pins

18.0 INSTRUCTION SET SUMMARY

The PIC17CXXX instruction set consists of 58 instructions. Each instruction is a 16-bit word divided into an OPCODE and one or more operands. The opcode specifies the instruction type, while the operand(s) further specify the operation of the instruction. The PIC17CXXX instruction set can be grouped into three types:

- byte-oriented
- bit-oriented
- literal and control operations

These formats are shown in Figure 18-1.

Table 18-1 shows the field descriptions for the opcodes. These descriptions are useful for understanding the opcodes in Table 18-2 and in each specific instruction descriptions.

For **byte-oriented instructions**, 'f' represents a file register designator and 'd' represents a destination designator. The file register designator specifies which file register is to be used by the instruction.

The destination designator specifies where the result of the operation is to be placed. If 'd' = '0', the result is placed in the WREG register. If 'd' = '1', the result is placed in the file register specified by the instruction.

For **bit-oriented instructions**, 'b' represents a bit field designator which selects the number of the bit affected by the operation, while 'f' represents the number of the file in which the bit is located.

For **literal and control operations**, 'k' represents an 8- or 13-bit constant or literal value.

The instruction set is highly orthogonal and is grouped into:

- byte-oriented operations
- bit-oriented operations
- literal and control operations

All instructions are executed within one single instruction cycle, unless:

- a conditional test is true
- the program counter is changed as a result of an instruction
- a table read or a table write instruction is executed (in this case, the execution takes two instruction cycles with the second cycle executed as a NOP)

One instruction cycle consists of four oscillator periods. Thus, for an oscillator frequency of 25 MHz, the normal instruction execution time is 160 ns. If a conditional test is true or the program counter is changed as a result of an instruction, the instruction execution time is 320 ns.

TABLE 18-1: OPCODE FIELD DESCRIPTIONS

Field	Description
f	Register file address (00h to FFh)
p	Peripheral register file address (00h to 1Fh)
i	Table pointer control i = '0' (do not change) i = '1' (increment after instruction execution)
t	Table byte select t = '0' (perform operation on lower byte) t = '1' (perform operation on upper byte literal field, constant data)
WREG	Working register (accumulator)
b	Bit address within an 8-bit file register
k	Literal field, constant data or label
x	Don't care location (= '0' or '1') The assembler will generate code with x = '0'. It is the recommended form of use for compatibility with all Microchip software tools.
d	Destination select 0 = store result in WREG 1 = store result in file register f Default is d = '1'
u	Unused, encoded as '0'
s	Destination select 0 = store result in file register f and in the WREG 1 = store result in file register f Default is s = '1'
label	Label name
C,DC, Z,OV	ALU status bits Carry, Digit Carry, Zero, Overflow
GLINTD	Global Interrupt Disable bit (CPUSTA<4>)
TBLPTR	Table Pointer (16-bit)
TBLAT	Table Latch (16-bit) consists of high byte (TBLATH) and low byte (TBLATL)
TBLATL	Table Latch low byte
TBLATH	Table Latch high byte
TOS	Top-of-Stack
PC	Program Counter
BSR	Bank Select Register
WDT	Watchdog Timer Counter
$\overline{TO}$	Time-out bit
$\overline{PD}$	Power-down bit
dest	Destination either the WREG register or the specified register file location
[ ]	Options
( )	Contents
$\rightarrow$	Assigned to
< >	Register bit field
$\in$	In the set of
italics	User defined term (font is courier)

FIGURE 21-17: TYPICAL, MINIMUM AND MAXIMUM VOH vs. IOH (VDD = 5V, -40°C TO +125°C)

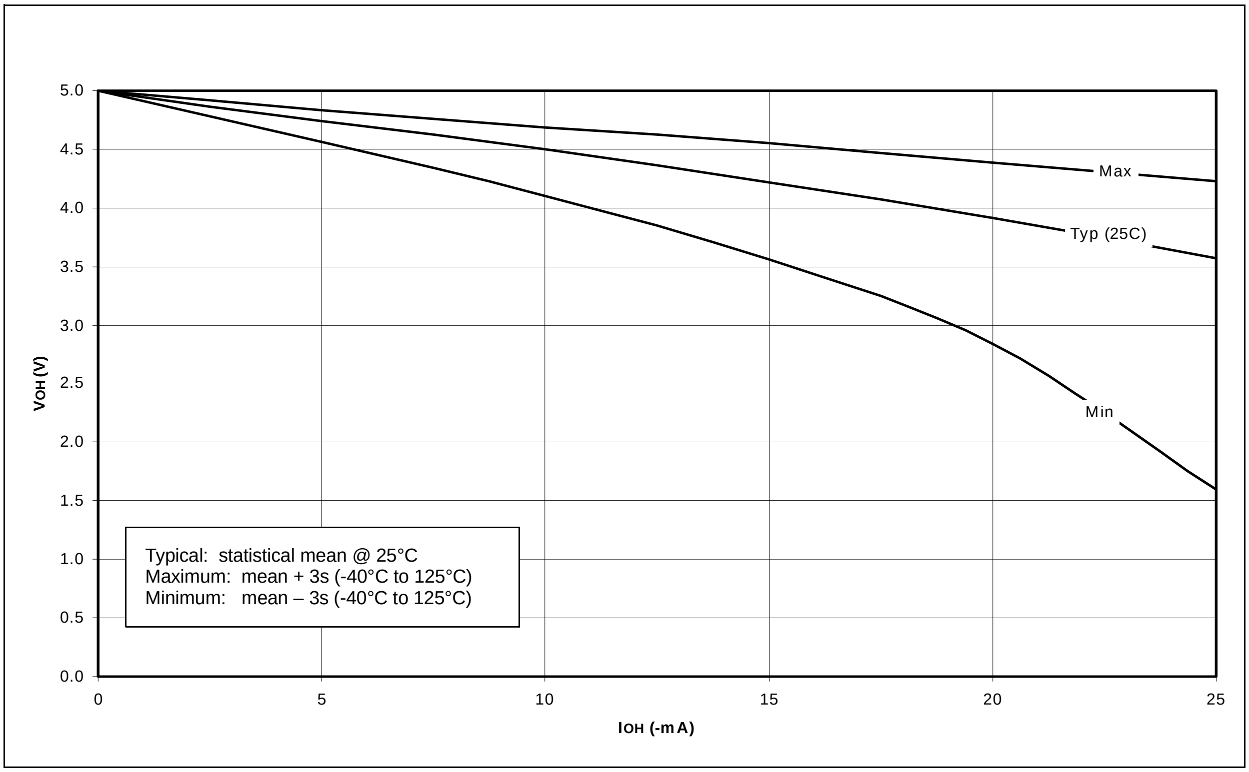


FIGURE 21-18: TYPICAL, MINIMUM AND MAXIMUM VOL vs. IOL (VDD = 5V, -40°C TO +125°C)

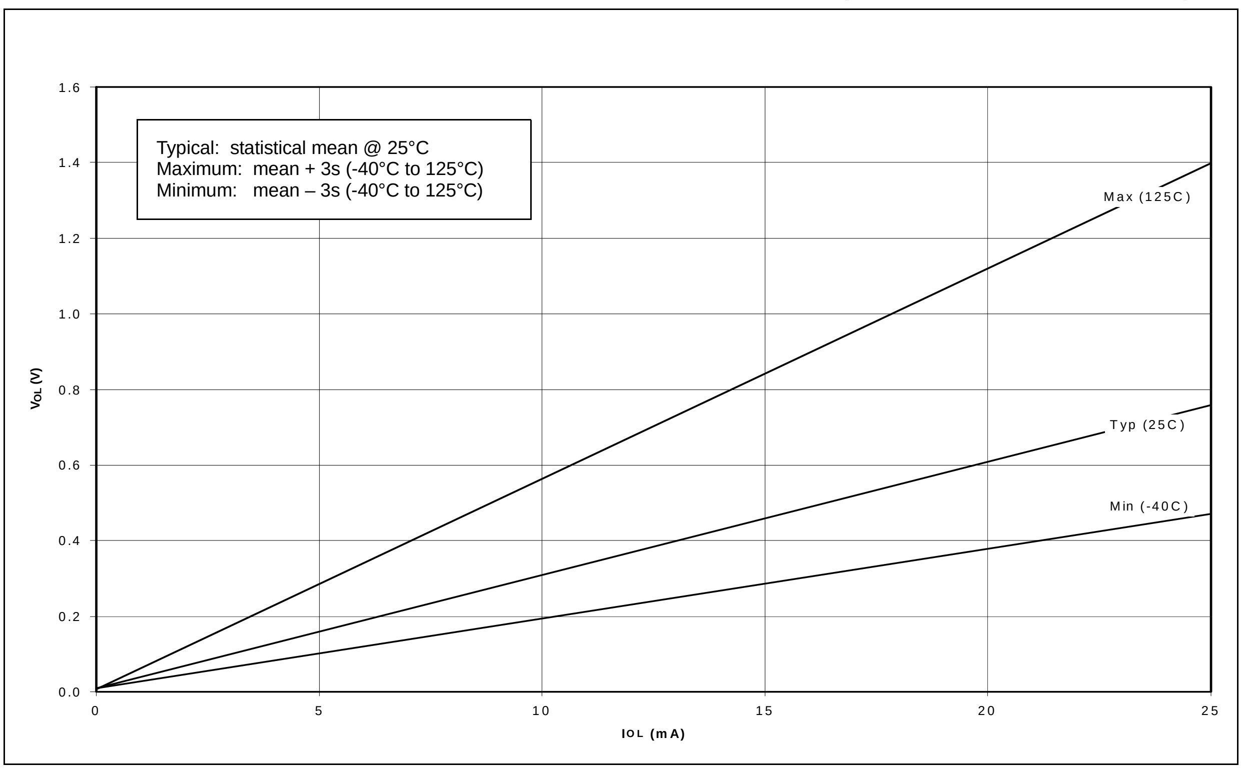


FIGURE 21-21: TYPICAL, MAXIMUM AND MINIMUM VIN vs. VDD (TTL INPUT, -40°C to 125°C)

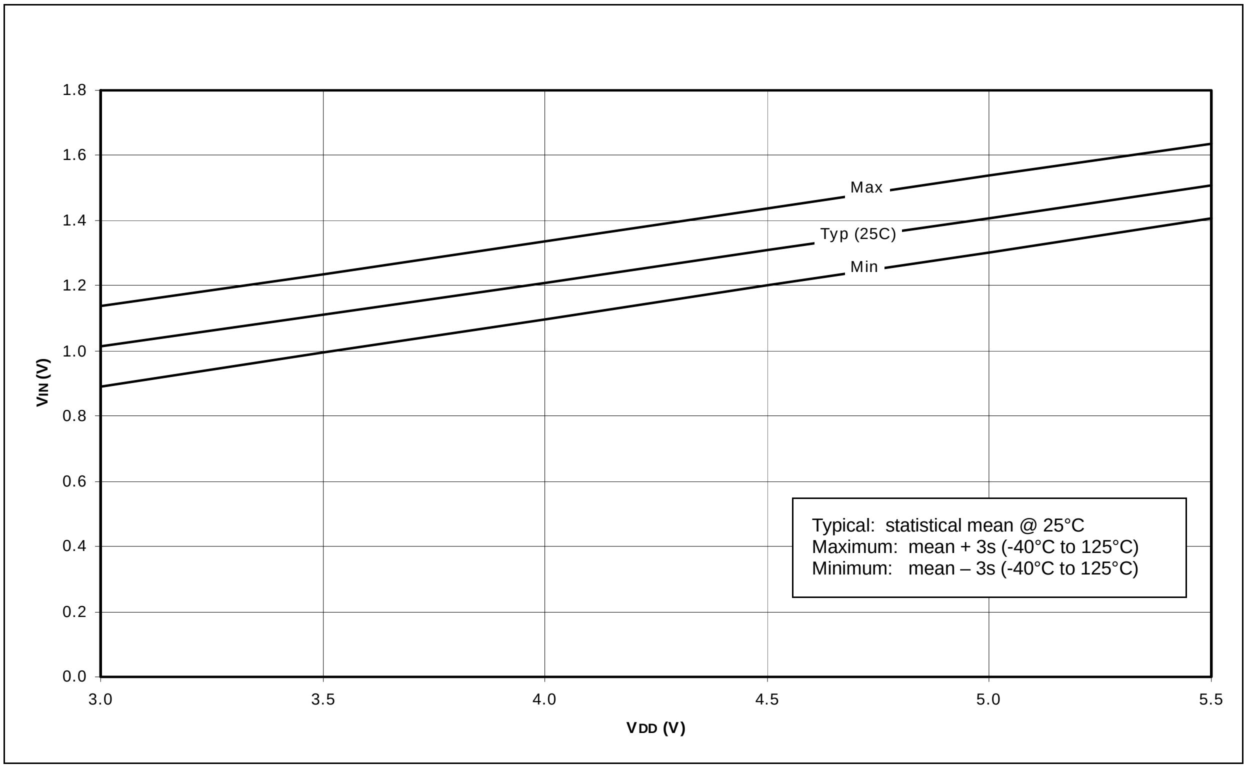
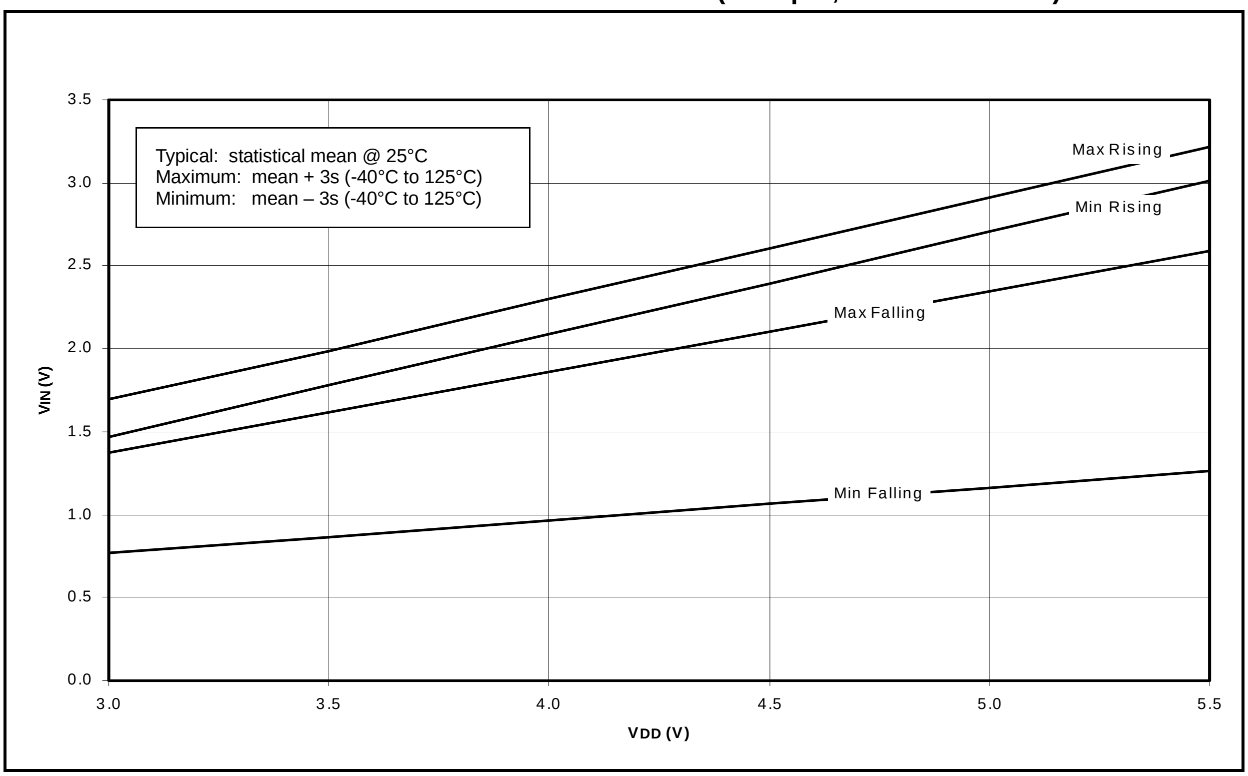


FIGURE 21-22: MAXIMUM AND MINIMUM VIN vs. VDD (ST Input, -40° C to +125°C)



APPENDIX A: MODIFICATIONS

The following is the list of modifications over the PIC16CXX microcontroller family:

1. Instruction word length is increased to 16-bit. This allows larger page sizes, both in program memory (8 Kwords verses 2 Kwords) and register file (256 bytes versus 128 bytes).
2. Four modes of operation: Microcontroller, Protected Microcontroller, Extended Microcontroller, and Microprocessor.
3. 22 new instructions. The `MOVF`, `TRIS` and `OPTION` instructions are no longer supported.
4. Four new instructions (`TLRD`, `TLWT`, `TABLRD`, `TABLWT`) for transferring data between data memory and program memory. They can be used to "self program" the EPROM program memory.
5. Single cycle data memory to data memory transfers possible (`MOVPF` and `MOVFP` instructions). These instructions do not affect the Working register (WREG).
6. W register (WREG) is now directly addressable.
7. A PC high latch register (PCLATH) is extended to 8-bits. The PCLATCH register is now both readable and writable.
8. Data memory paging is redefined slightly.
9. DDR registers replace function of TRIS registers.
10. Multiple Interrupt vectors added. This can decrease the latency for servicing interrupts.
11. Stack size is increased to 16 deep.
12. BSR register for data memory paging.
13. Wake-up from SLEEP operates slightly differently.
14. The Oscillator Start-Up Timer (OST) and Power-Up Timer (PWRT) operate in parallel and not in series.
15. PORTB interrupt-on-change feature works on all eight port pins.
16. TMR0 is 16-bit, plus 8-bit prescaler.
17. Second indirect addressing register added (FSR1 and FSR2). Control bits can select the FSR registers to auto-increment, auto-decrement, remain unchanged after an indirect address.
18. Hardware multiplier added (8 x 8 $\rightarrow$ 16-bit).
19. Peripheral modules operate slightly differently.
20. A/D has both VREF+ and VREF- inputs.
21. USARTs do not implement BRGH feature.
22. Oscillator modes slightly redefined.
23. Control/Status bits and registers have been placed in different registers and the control bit for globally enabling interrupts has inverse polarity.
24. In-circuit serial programming is implemented differently.

APPENDIX B: COMPATIBILITY

To convert code written for PIC16CXXX to PIC17CXXX, the user should take the following steps:

1. Remove any `TRIS` and `OPTION` instructions, and implement the equivalent code.
2. Separate the Interrupt Service Routine into its four vectors.
3. Replace:
   ```
   MOVF    REG1, W
   ```
   with:
   ```
   MOVFP   REG1, WREG
   ```
4. Replace:
   ```
   MOVF    REG1, W
   MOVWF   REG2
   ```
   with:
   ```
   MOVPF   REG1, REG2 ; Addr(REG1)<20h
   ```
   or
   ```
   MOVFP  REG1, REG2 ; Addr(REG2)<20h
   ```

Note: If REG1 and REG2 are both at addresses greater then 20h, two instructions are required.
```
MOVFP   REG1, WREG ;
MOVPF   WREG, REG2 ;
```

5. Ensure that all bit names and register names are updated to new data memory map locations.
6. Verify data memory banking.
7. Verify mode of operation for indirect addressing.
8. Verify peripheral routines for compatibility.
9. Weak pull-ups are enabled on RESET.
10. WDT time-outs always reset the device (in run or SLEEP mode).

B.1 Upgrading from PIC17C42 Devices

To convert code from the PIC17C42 to all the other PIC17CXXX devices, the user should take the following steps.

1. If the hardware multiply is to be used, ensure that any variables at address 18h and 19h are moved to another address.
2. Ensure that the upper nibble of the BSR was not written with a non-zero value. This may cause unexpected operation since the RAM bank is no longer 0.
3. The disabling of global interrupts has been enhanced, so there is no additional testing of the GLINTD bit after a `BSF  CPUSTA,  GLINTD` instruction.

