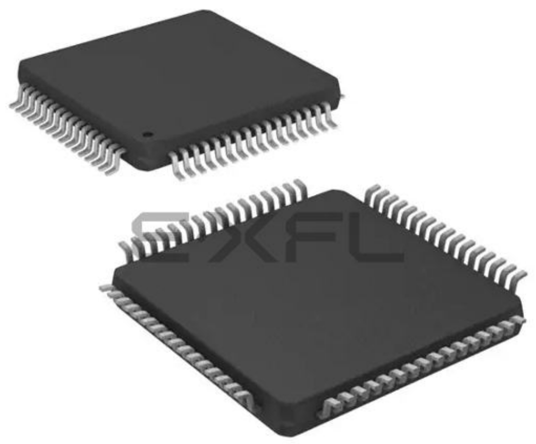
Microchip Technology - PIC17C756AT-16E/PT Datasheet



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.

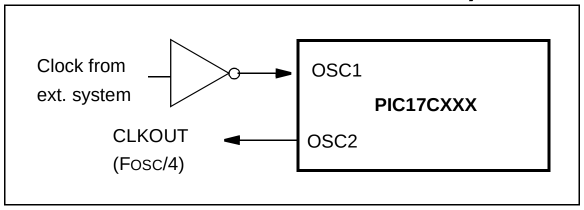
Applications of "Embedded - Microcontrollers"

Details	
Product Status	Obsolete
Core Processor	PIC
Core Size	8-Bit
Speed	16MHz
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	-40°C ~ 125°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/pic17c756at-16e-pt

4.1.4 EXTERNAL CLOCK OSCILLATOR

In the EC oscillator mode, the OSC1 input can be driven by CMOS drivers. In this mode, the OSC1/CLKIN pin is hi-impedance and the OSC2/CLKOUT pin is the CLKOUT output (4 TOSC).

FIGURE 4-4: EXTERNAL CLOCK INPUT OPERATION (EC OSC CONFIGURATION)



4.1.5 EXTERNAL CRYSTAL OSCILLATOR CIRCUIT

Either a prepackaged oscillator can be used, or a simple oscillator circuit with TTL gates can be built. Prepackaged oscillators provide a wide operating range and better stability. A well designed crystal oscillator will provide good performance with TTL gates. Two types of crystal oscillator circuits can be used: one with series resonance, or one with parallel resonance.

Figure 4-5 shows implementation of a parallel resonant oscillator circuit. The circuit is designed to use the fundamental frequency of the crystal. The 74AS04 inverter performs the 180-degree phase shift that a parallel oscillator requires. The 4.7 kΩ resistor provides the negative feedback for stability. The 10 kΩ potentiometer biases the 74AS04 in the linear region. This could be used for external oscillator designs.

FIGURE 4-5: EXTERNAL PARALLEL RESONANT CRYSTAL OSCILLATOR CIRCUIT

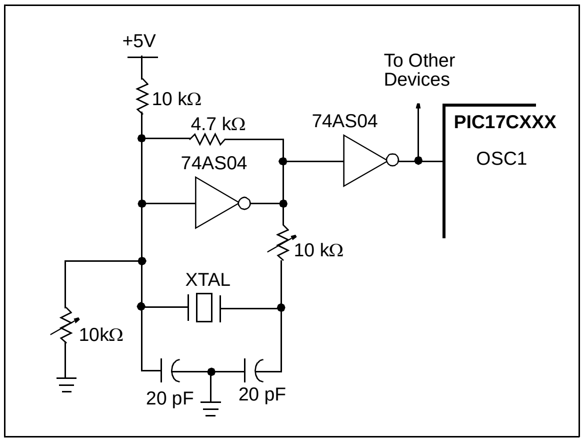
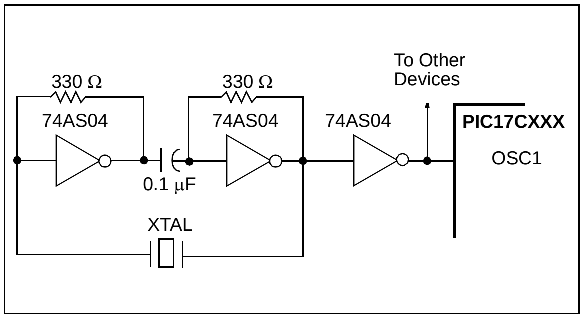


Figure 4-6 shows a series resonant oscillator circuit. This circuit is also designed to use the fundamental frequency of the crystal. The inverter performs a 180-degree phase shift in a series resonant oscillator circuit. The 330 Ω resistors provide the negative feedback to bias the inverters in their linear region.

FIGURE 4-6: EXTERNAL SERIES RESONANT CRYSTAL OSCILLATOR CIRCUIT



5.1 Power-on Reset (POR), Power-up Timer (PWRT), Oscillator Start-up Timer (OST) and Brown-out Reset (BOR)

5.1.1 POWER-ON RESET (POR)

The Power-on Reset circuit holds the device in RESET until VDD is above the trip point (in the range of 1.4V - 2.3V). The devices produce an internal RESET for both rising and falling VDD. To take advantage of the POR, just tie the $\overline{\text{MCLR}}$/VPP pin directly (or through a resistor) to VDD. This will eliminate external RC components usually needed to create Power-on Reset. A minimum rise time for VDD is required. See Electrical Specifications for details.

Figure 5-2 and Figure 5-3 show two possible POR circuits.

FIGURE 5-2: USING ON-CHIP POR

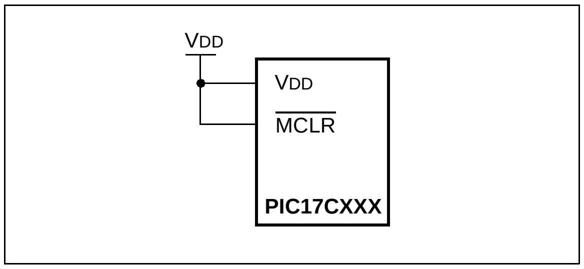
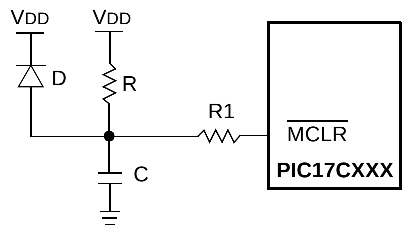


FIGURE 5-3: EXTERNAL POWER-ON RESET CIRCUIT (FOR SLOW VDD POWER-UP)



Note 1: An external Power-on Reset circuit is required only if VDD power-up time is too slow. The diode D helps discharge the capacitor quickly when VDD powers down.

2: R < 40 kΩ is recommended to ensure that the voltage drop across R does not exceed 0.2V (max. leakage current spec. on the $\overline{\text{MCLR}}$/VPP pin is 5 μA). A larger voltage drop will degrade VIH level on the $\overline{\text{MCLR}}$/VPP pin.

3: R1 = 100Ω to 1 kΩ will limit any current flowing into $\overline{\text{MCLR}}$ from external capacitor C in the event of $\overline{\text{MCLR}}$/VPP pin breakdown due to Electrostatic Discharge (ESD) or Electrical Overstress (EOS).

5.1.2 POWER-UP TIMER (PWRT)

The Power-up Timer provides a fixed 96 ms time-out (nominal) on power-up. This occurs from the rising edge of the internal POR signal if VDD and $\overline{\text{MCLR}}$ are tied, or after the first rising edge of $\overline{\text{MCLR}}$ (detected high). The Power-up Timer operates on an internal RC oscillator. The chip is kept in RESET as long as the PWRT is active. In most cases, the PWRT delay allows VDD to rise to an acceptable level.

The power-up time delay will vary from chip to chip and with VDD and temperature. See DC parameters for details.

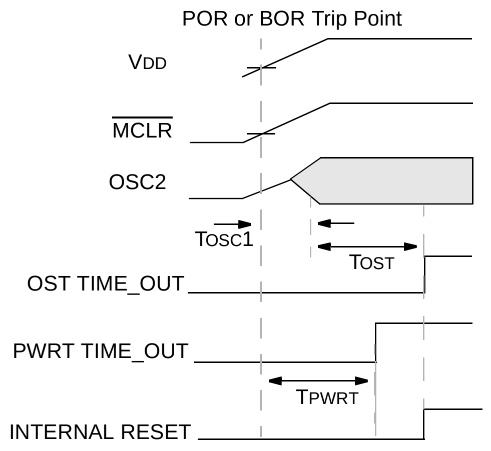
5.1.3 OSCILLATOR START-UP TIMER (OST)

The Oscillator Start-up Timer (OST) provides a 1024 oscillator cycle (1024TOSC) delay whenever the PWRT is invoked, or a wake-up from SLEEP event occurs in XT or LF mode. The PWRT and OST operate in parallel.

The OST counts the oscillator pulses on the OSC1/CLKIN pin. The counter only starts incrementing after the amplitude of the signal reaches the oscillator input thresholds. This delay allows the crystal oscillator or resonator to stabilize before the device exits RESET. The length of the time-out is a function of the crystal/resonator frequency.

Figure 5-4 shows the operation of the OST circuit. In this figure, the oscillator is of such a low frequency that although enabled simultaneously, the OST does not time-out until after the Power-up Timer time-out.

FIGURE 5-4: OSCILLATOR START-UP TIME (LOW FREQUENCY)



This figure shows in greater detail the timings involved with the oscillator start-up timer. In this example, the low frequency crystal start-up time is larger than power-up time (TPWRT).

TOSC1 = time for the crystal oscillator to react to an oscillation level detectable by the Oscillator Start-up Timer (OST).

TOST = 1024TOSC.

REGISTER 6-3: PIE2 REGISTER (ADDRESS: 11h, BANK 4)

R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
SSPIE	BCLIE	ADIE	—	CA4IE	CA3IE	TX2IE	RC2IE
bit 7							bit 0

bit 7	**SSPIE**: Synchronous Serial Port Interrupt Enable bit 1 = Enable SSP interrupt 0 = Disable SSP interrupt
bit 6	**BCLIE**: Bus Collision Interrupt Enable bit 1 = Enable bus collision interrupt 0 = Disable bus collision interrupt
bit 5	**ADIE**: A/D Module Interrupt Enable bit 1 = Enable A/D module interrupt 0 = Disable A/D module interrupt
bit 4	**Unimplemented:** Read as ‘0’
bit 3	**CA4IE**: Capture4 Interrupt Enable bit 1 = Enable Capture4 interrupt 0 = Disable Capture4 interrupt
bit 2	**CA3IE**: Capture3 Interrupt Enable bit 1 = Enable Capture3 interrupt 0 = Disable Capture3 interrupt
bit 1	**TX2IE**: USART2 Transmit Interrupt Enable bit 1 = Enable USART2 Transmit buffer empty interrupt 0 = Disable USART2 Transmit buffer empty interrupt
bit 0	**RC2IE**: USART2 Receive Interrupt Enable bit 1 = Enable USART2 Receive buffer full interrupt 0 = Disable USART2 Receive buffer full interrupt

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

8.2 Table Writes to External Memory

Table writes to external memory are always two-cycle instructions. The second cycle writes the data to the external memory location. The sequence of events for an external memory write are the same for an internal write.

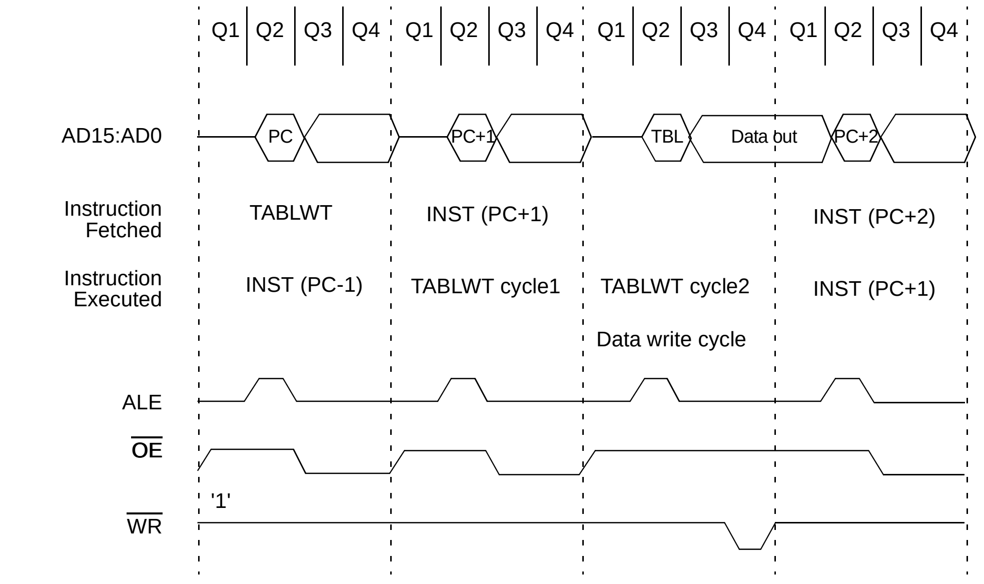
8.2.1 TABLE WRITE CODE

The "`i`" operand of the `TABLWT` instruction can specify that the value in the 16-bit TBLPTR register is automatically incremented (for the next write). In Example 8-1, the TBLPTR register is not automatically incremented.

EXAMPLE 8-1: TABLE WRITE

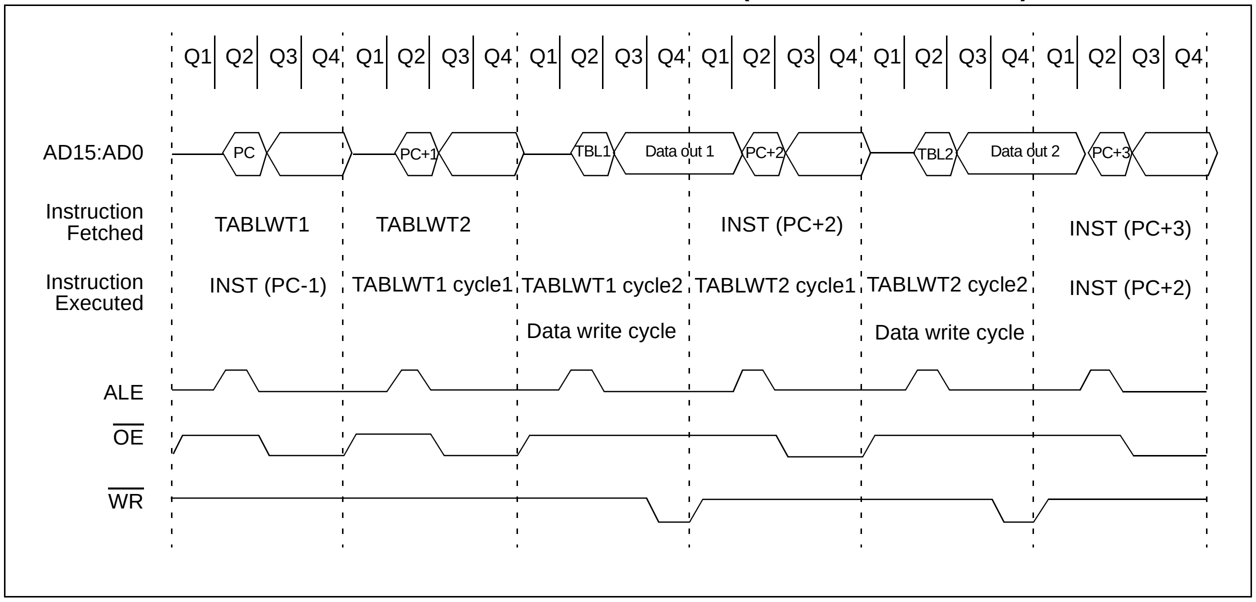
```
CLRWDT                    ; Clear WDT
MOVLW   HIGH (TBL_ADDR) ; Load the Table
MOVWF   TBLPTRH          ;   address
MOVLW   LOW (TBL_ADDR)  ;
MOVWF   TBLPTRL          ;
MOVLW   HIGH (DATA)      ; Load HI byte
TLWT    1, WREG          ;   in TABLATH
MOVLW   LOW (DATA)       ; Load LO byte
TABLWT  0,0,WREG         ;   in TABLATL
                         ;   and write to
                         ;   program memory
                         ;   (Ext. SRAM)
```

FIGURE 8-5: `TABLWT` WRITE TIMING (EXTERNAL MEMORY)



Note: If external write and GLINTD = '1' and Enable bit = '1', then when '1' → Flag bit, do table write. The highest pending interrupt is cleared.

FIGURE 8-6: CONSECUTIVE `TABLWT` WRITE TIMING (EXTERNAL MEMORY)



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	@ 16 MHz	@ 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

10.5 PORTE and DDRE Register

PORTE is a 4-bit bi-directional port. The corresponding data direction register is DDRE. A '1' in DDRE configures the corresponding port pin as an input. A '0' in the DDRE register configures the corresponding port pin as an output. Reading PORTE reads the status of the pins, whereas writing to PORTE will write to the port latch. PORTE is multiplexed with the system bus. When operating as the system bus, PORTE contains the control signals for the address/data bus (AD15:AD0). These control signals are Address Latch Enable (ALE), Output Enable ($\overline{OE}$) and Write ($\overline{WR}$). The control signals $\overline{OE}$ and $\overline{WR}$ are active low signals. The timing for the system bus is shown in the Electrical Specifications section.

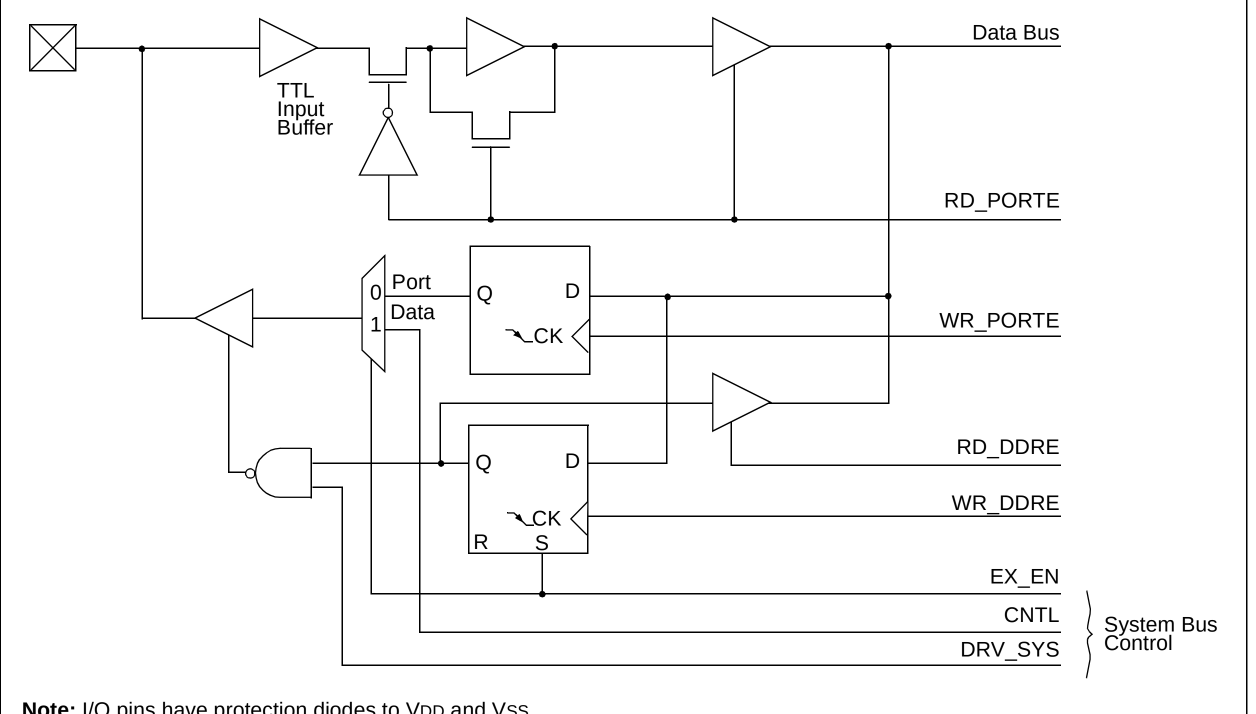
Note: Three pins of this port are configured as the system bus when the device's configuration bits are selected to Microprocessor or Extended Microcontroller modes. The other pin is a general purpose I/O or Capture4 pin. In the two other microcontroller modes, RE2:RE0 are general purpose I/O pins.

Example 10-5 shows an instruction sequence to initialize PORTE. The Bank Select Register (BSR) must be selected to Bank 1 for the port to be initialized. The following example uses the `MOVLB` instruction to load the BSR register for bank selection.

EXAMPLE 10-5: INITIALIZING PORTE

```
MOVLB  1          ; Select Bank 1
CLRF   PORTE, F   ; Initialize PORTE data
                  ; latches before setting
                  ; the data direction
                  ; register
MOVLW   0x03      ; Value used to initialize
                  ; data direction
MOVWF   DDRE      ; Set RE<1:0> as inputs
                  ; RE<3:2> as outputs
                  ; RE<7:4> are always
                  ; read as '0'
```

FIGURE 10-11: BLOCK DIAGRAM OF RE2:RE0 (IN I/O PORT MODE)



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

NOTES:

15.0 MASTER SYNCHRONOUS SERIAL PORT (MSSP) MODULE

The Master Synchronous Serial Port (MSSP) module is a serial interface useful for communicating with other peripheral or microcontroller devices. These peripheral devices may be serial EEPROMs, shift registers, display drivers, A/D converters, etc. The MSSP module can operate in one of two modes:

- Serial Peripheral Interface (SPI)
- Inter-Integrated Circuit™ (I^2C)

Figure 15-1 shows a block diagram for the SPI mode, while Figure 15-2 and Figure 15-3 show the block diagrams for the two different I^2C modes of operation.

FIGURE 15-1: SPI MODE BLOCK DIAGRAM

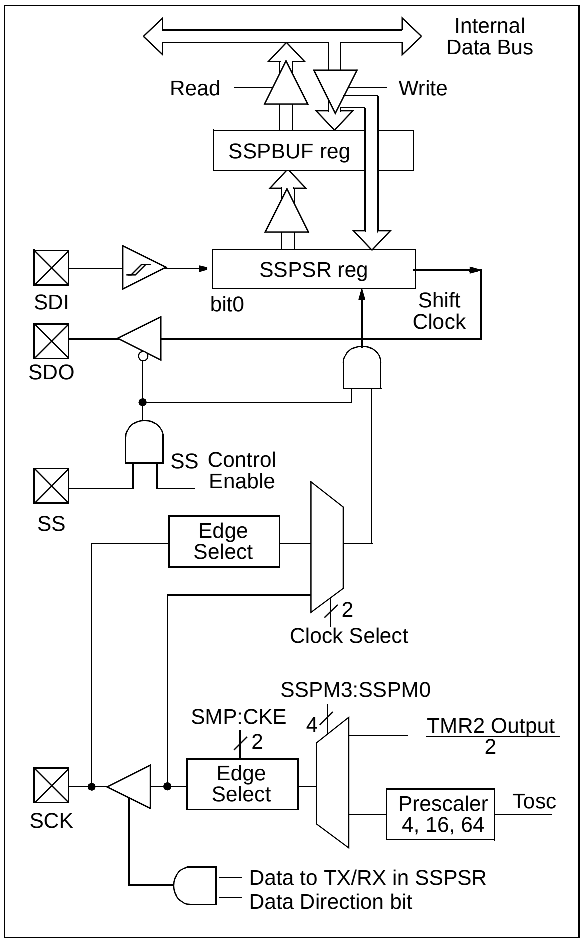


FIGURE 15-2: I^2C SLAVE MODE BLOCK DIAGRAM

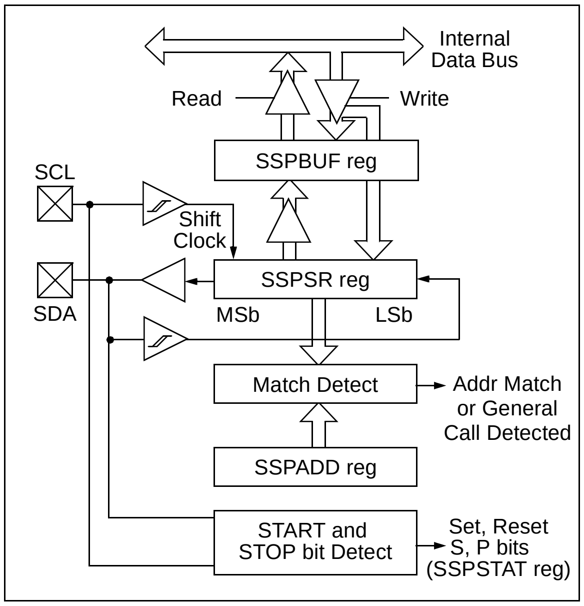

Internal Data Bus
Read
Write
SSPBUF reg
SCL
Shift Clock
SSPSR reg
SDA
MSb
LSb
Match Detect
Addr Match or General Call Detected
SSPADD reg
START and STOP bit Detect
Set, Reset S, P bits (SSPSTAT reg)

FIGURE 15-3: I^2C MASTER MODE BLOCK DIAGRAM

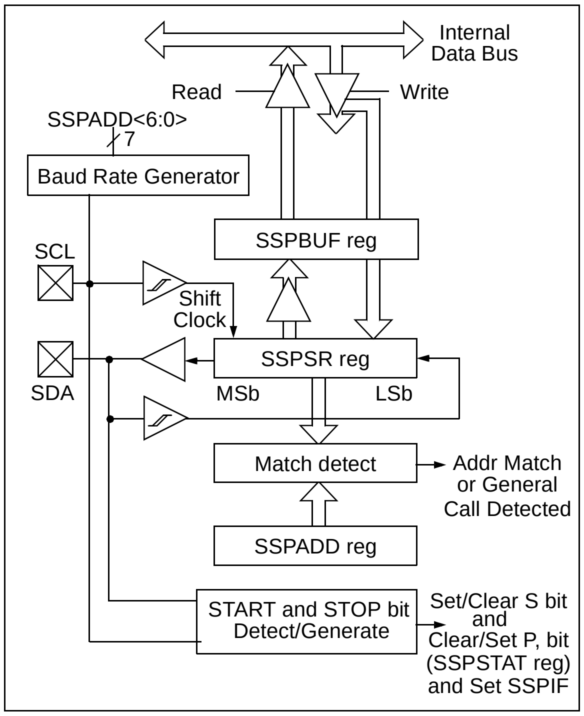
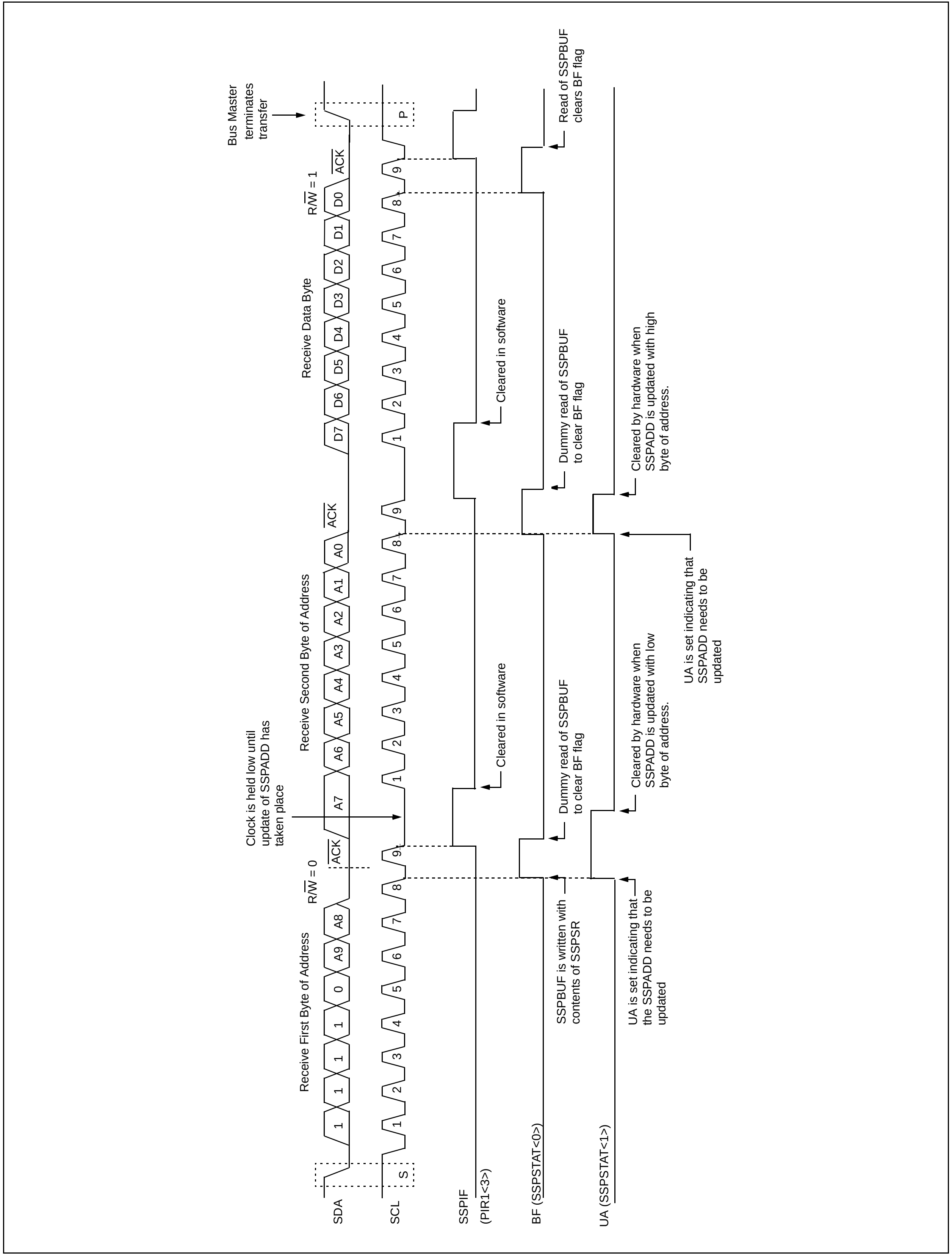

Internal Data Bus
Read
Write
SSPADD<6:0>
7
Baud Rate Generator
SSPBUF reg
SCL
Shift Clock
SSPSR reg
SDA
MSb
LSb
Match detect
Addr Match or General Call Detected
SSPADD reg
START and STOP bit Detect/Generate
Set/Clear S bit and Clear/Set P, bit (SSPSTAT reg) and Set SSPIF

FIGURE 15-15: I²C SLAVE-RECEIVER (10-BIT ADDRESS)


SDA
SCL
SSPIF
(PIR1<3>)
BF (SSPSTAT<0>)
UA (SSPSTAT<1>)
S
P
Receive First Byte of Address
1 1 1 1 0 A9 A8
R/$\overline{W}$ = 0
$\overline{ACK}$
Receive Second Byte of Address
A7 A6 A5 A4 A3 A2 A1 A0
$\overline{ACK}$
Receive Data Byte
D7 D6 D5 D4 D3 D2 D1 D0
R/$\overline{W}$ = 1
$\overline{ACK}$
1 2 3 4 5 6 7 8 9
Clock is held low until update of SSPADD has taken place
Bus Master terminates transfer
Cleared in software
SSPBUF is written with contents of SSPSR
Dummy read of SSPBUF to clear BF flag
Read of SSPBUF clears BF flag
UA is set indicating that the SSPADD needs to be updated
Cleared by hardware when SSPADD is updated with low byte of address.
UA is set indicating that SSPADD needs to be updated
Cleared by hardware when SSPADD is updated with high byte of address.

15.2.5 MASTER MODE

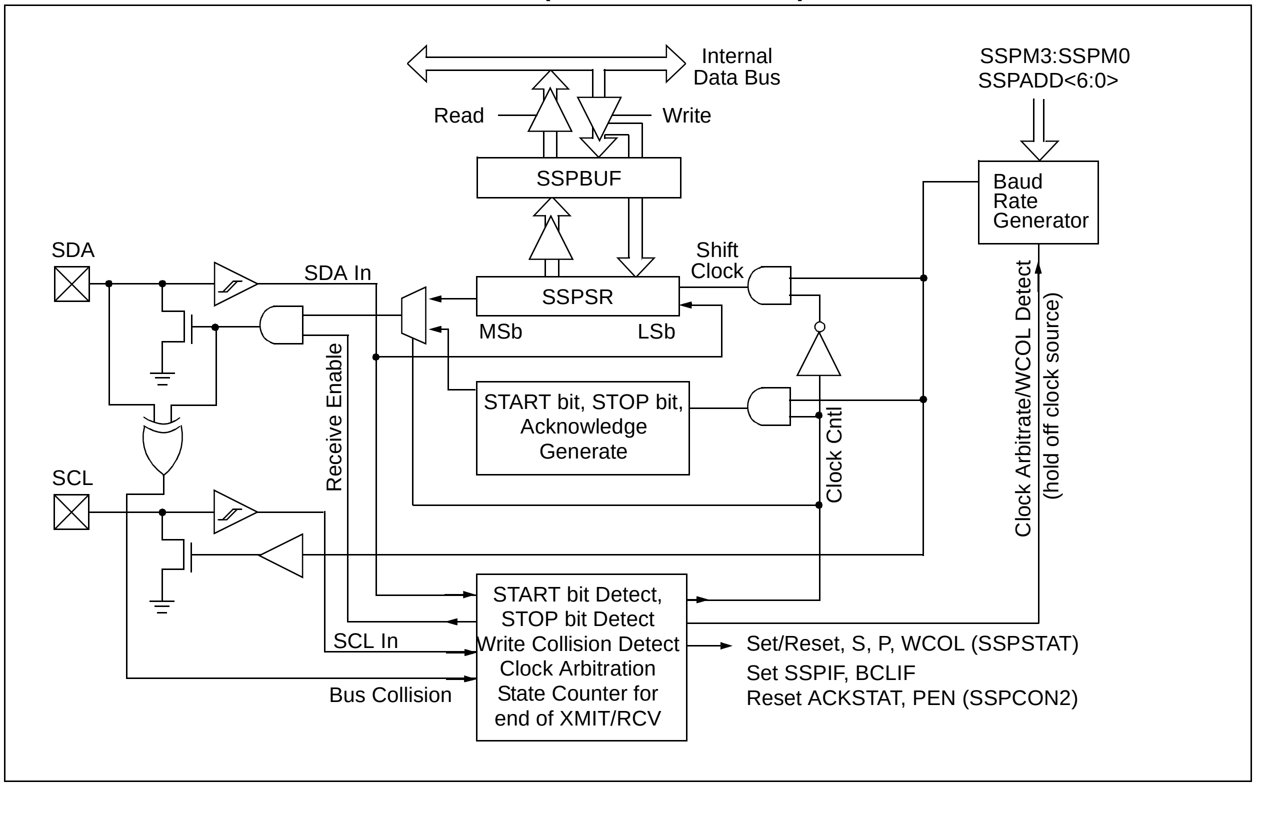
Master mode of operation is supported by interrupt generation on the detection of the START and STOP conditions. The STOP (P) and START (S) bits are cleared from a RESET, or when the MSSP module is disabled. Control of the I^{2}C bus may be taken when the P bit is set, or the bus is idle, with both the S and P bits clear.

In Master mode, the SCL and SDA lines are manipulated by the MSSP hardware.

The following events will cause SSP Interrupt Flag bit, SSPIF, to be set (SSP Interrupt if enabled):

- START condition
- STOP condition
- Data transfer byte transmitted/received
- Acknowledge transmit
- Repeated Start

FIGURE 15-17: SSP BLOCK DIAGRAM (I^{2}C MASTER MODE)


Internal Data Bus
SSPM3:SSPM0
SSPADD<6:0>
Read
Write
SSPBUF
Baud Rate Generator
SDA
SDA In
Shift Clock
SSPSR
MSb
LSb
Receive Enable
START bit, STOP bit, Acknowledge Generate
Clock Cntl
Clock Arbitrate/WCOL Detect (hold off clock source)
SCL
START bit Detect, STOP bit Detect Write Collision Detect Clock Arbitration State Counter for end of XMIT/RCV
SCL In
Bus Collision
Set/Reset, S, P, WCOL (SSPSTAT)
Set SSPIF, BCLIF
Reset ACKSTAT, PEN (SSPCON2)

15.2.10 I^{2}C MASTER MODE REPEATED START CONDITION TIMING

A Repeated Start condition occurs when the RSEN bit (SSPCON2<1>) is programmed high and the I^{2}C module is in the idle state. When the RSEN bit is set, the SCL pin is asserted low. When the SCL pin is sampled low, the baud rate generator is loaded with the contents of SSPADD<6:0> and begins counting. The SDA pin is released (brought high) for one baud rate generator count (TBRG). When the baud rate generator times out, if SDA is sampled high, the SCL pin will be de-asserted (brought high). When SCL is sampled high the baud rate generator is reloaded with the contents of SSPADD<6:0> and begins counting. SDA and SCL must be sampled high for one TBRG. This action is then followed by assertion of the SDA pin (SDA is low) for one TBRG while SCL is high. Following this, the RSEN bit in the SSPCON2 register will be automatically cleared and the baud rate generator is not reloaded, leaving the SDA pin held low. As soon as a START condition is detected on the SDA and SCL pins, the S bit (SSPSTAT<3>) will be set. The SSPIF bit will not be set until the baud rate generator has timed out.

Note 1: If the RSEN is programmed while any other event is in progress, it will not take effect.

2: A bus collision during the Repeated Start condition occurs if:

- SDA is sampled low when SCL goes from low to high.
- SCL goes low before SDA is asserted low. This may indicate that another master is attempting to transmit a data “1”.

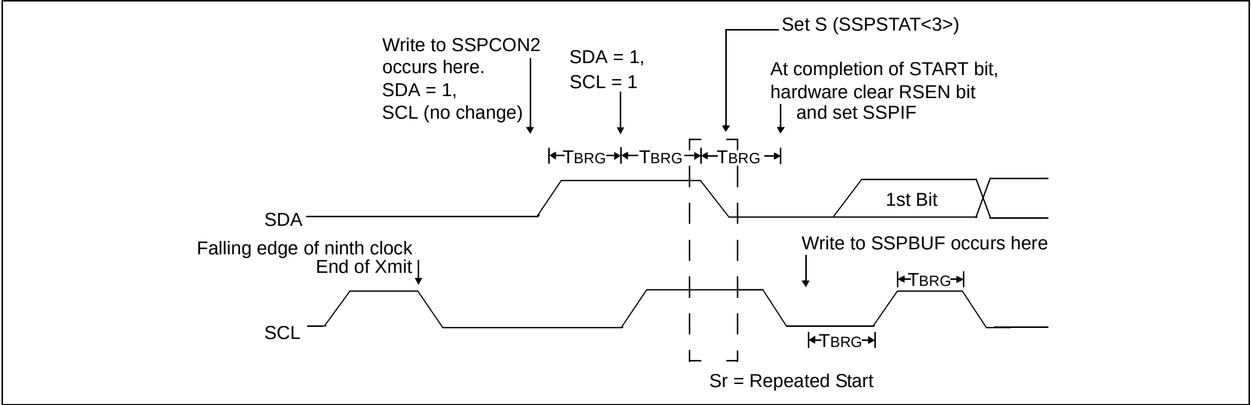
Immediately following the SSPIF bit getting set, the user may write the SSPBUF with the 7-bit address in 7-bit mode, or the default first address in 10-bit mode. After the first eight bits are transmitted and an ACK is received, the user may then transmit an additional eight bits of address (10-bit mode), or eight bits of data (7-bit mode).

15.2.10.1 WCOL status flag

If the user writes the SSPBUF when a Repeated Start sequence is in progress, then WCOL is set and the contents of the buffer are unchanged (the write doesn’t occur).

Note: Because queueing of events is not allowed, writing of the lower 5 bits of SSPCON2 is disabled until the Repeated Start condition is complete.

FIGURE 15-22: REPEAT START CONDITION WAVEFORM


Write to SSPCON2 occurs here. SDA = 1, SCL (no change)
SDA = 1, SCL = 1
Set S (SSPSTAT<3>)
At completion of START bit, hardware clear RSEN bit and set SSPIF
TBRG
TBRG
TBRG
SDA
1st Bit
Falling edge of ninth clock End of Xmit
Write to SSPBUF occurs here
TBRG
SCL
TBRG
Sr = Repeated Start

15.2.14 STOP CONDITION TIMING

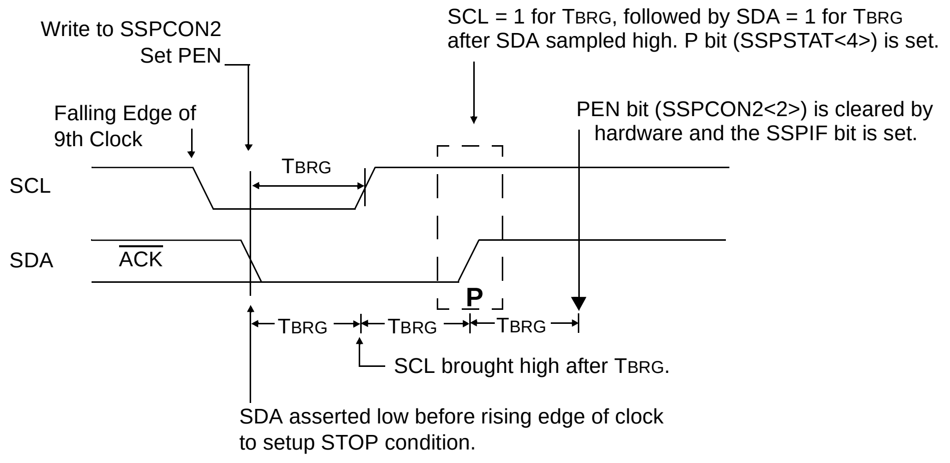
A STOP bit is asserted on the SDA pin at the end of a receive/transmit by setting the Stop Sequence Enable bit PEN (SSPCON2<2>). At the end of a receive/transmit the SCL line is held low after the falling edge of the ninth clock. When the PEN bit is set, the master will assert the SDA line low. When the SDA line is sampled low, the baud rate generator is reloaded and counts down to '0'. When the baud rate generator times out, the SCL pin will be brought high and one TBRG (baud rate generator rollover count) later, the SDA pin will be de-asserted. When the SDA pin is sampled high while SCL is high, the P bit (SSPSTAT<4>) is set. A TBRG later, the PEN bit is cleared and the SSPIF bit is set (Figure 15-31).

Whenever the firmware decides to take control of the bus, it will first determine if the bus is busy by checking the S and P bits in the SSPSTAT register. If the bus is busy, then the CPU can be interrupted (notified) when a STOP bit is detected (i.e., bus is free).

15.2.14.1 WCOL Status Flag

If the user writes the SSPBUF when a STOP sequence is in progress, then WCOL is set and the contents of the buffer are unchanged (the write doesn't occur).

FIGURE 15-31: STOP CONDITION RECEIVE OR TRANSMIT MODE


Write to SSPCON2
Set PEN
SCL = 1 for TBRG, followed by SDA = 1 for TBRG after SDA sampled high. P bit (SSPSTAT<4>) is set.
PEN bit (SSPCON2<2>) is cleared by hardware and the SSPIF bit is set.
Falling Edge of 9th Clock
TBRG
SCL
SDA
ACK
P
TBRG
TBRG
TBRG
SCL brought high after TBRG.
SDA asserted low before rising edge of clock to setup STOP condition.

Note: TBRG = one baud rate generator period.

EXAMPLE 15-2: INTERFACING TO A 24LC01B SERIAL EEPROM (USING MPLAB C17)

```
// Writes the byte data to 24LC01B at the specified address
void ByteWrite(static unsigned char address, static unsigned char data)
{
    StartI2C();                      // Send start bit
    IdleI2C();                       // Wait for idle condition
    WriteI2C(CONTROL);               // Send control byte
    IdleI2C();                       // Wait for idle condition
    if (!SSPCON2bits.ACKSTAT)        // If 24LC01B ACKs
    {
        WriteI2C(address);           // Send control byte
        IdleI2C();                   // Wait for idle condition

        if (!SSPCON2bits.ACKSTAT)    // If 24LC01B ACKs
            WriteI2C(data);          // Send data
    }
    IdleI2C();                       // Wait for idle condition
    StopI2C();                       // Send stop bit
    IdleI2C();                       // Wait for idle condition
    return;
}


// Reads a byte of data from 24LC01B at the specified address
unsigned char ByteRead(static unsigned char address)
{
    StartI2C();                      // Send start bit
    IdleI2C();                       // Wait for idle condition
    WriteI2C(CONTROL);               // Send control byte
    IdleI2C();                       // Wait for idle condition
    if (!SSPCON2bits.ACKSTAT)        // If the 24LC01B ACKs
    {
        WriteI2C(address);           // Send address
        IdleI2C();                   // Wait for idle condition
        if (!SSPCON2bits.ACKSTAT)    // If the 24LC01B ACKs
        {
            RestartI2C();            // Send restart
            IdleI2C();               // Wait for idle condition
            WriteI2C(CONTROL+1);     // Send control byte with R/W set
            IdleI2C();               // Wait for idle condition
            if (!SSPCON2bits.ACKSTAT)    // If the 24LC01B ACKs
            {
                getcI2C();               // Read a byte of data from 24LC01B
                IdleI2C();               // Wait for idle condition
                NotAckI2C();             // Send a NACK to 24LC01B
                IdleI2C();               // Wait for idle condition
                StopI2C();               // Send stop bit
                IdleI2C();               // Wait for idle condition
             }
        }
    }
    return(SSPBUF);
}
```

17.3 Watchdog Timer (WDT)

The Watchdog Timer's function is to recover from software malfunction, or to reset the device while in SLEEP mode. The WDT uses an internal free running on-chip RC oscillator for its clock source. This does not require any external components. This RC oscillator is separate from the RC oscillator of the OSC1/CLKIN pin. That means that the WDT will run even if the clock on the OSC1/CLKIN and OSC2/CLKOUT pins has been stopped, for example, by execution of a `SLEEP` instruction. During normal operation, a WDT time-out generates a device RESET. The WDT can be permanently disabled by programming the configuration bits WDTPS1:WDTPS0 as '`00`' (Section 17.1).

Under normal operation, the WDT must be cleared on a regular interval. This time must be less than the minimum WDT overflow time. Not clearing the WDT in this time frame will cause the WDT to overflow and reset the device.

17.3.1 WDT PERIOD

The WDT has a nominal time-out period of 12 ms (with postscaler = 1). The time-out periods vary with temperature, V_{DD} and process variations from part to part (see DC specs). If longer time-out periods are desired, configuration bits should be used to enable the WDT with a greater prescale. Thus, typical time-out periods up to 3.0 seconds can be realized.

The `CLRWDT` and `SLEEP` instructions clear the WDT and its postscale setting and prevent it from timing out, thus generating a device RESET condition.

The $\overline{TO}$ bit in the CPUSTA register will be cleared upon a WDT time-out.

17.3.2 CLEARING THE WDT AND POSTSCALER

The WDT and postscaler are cleared when:

- The device is in the RESET state
- A `SLEEP` instruction is executed
- A `CLRWDT` instruction is executed
- Wake-up from SLEEP by an interrupt

The WDT counter/postscaler will start counting on the first edge after the device exits the RESET state.

17.3.3 WDT PROGRAMMING CONSIDERATIONS

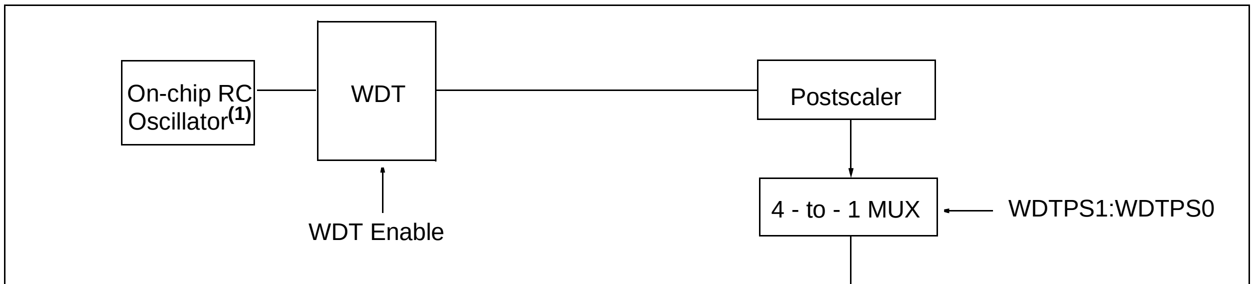
It should also be taken in account that under worst case conditions (V_{DD} = Min., Temperature = Max., Max. WDT postscaler), it may take several seconds before a WDT time-out occurs.

The WDT and postscaler become the Power-up Timer whenever the PWRT is invoked.

17.3.4 WDT AS NORMAL TIMER

When the WDT is selected as a normal timer, the clock source is the device clock. Neither the WDT nor the postscaler are directly readable or writable. The overflow time is 65536 T_{OSC} cycles. On overflow, the $\overline{TO}$ bit is cleared (device is not RESET). The `CLRWDT` instruction can be used to set the $\overline{TO}$ bit. This allows the WDT to be a simple overflow timer. The simple timer does not increment when in SLEEP.

FIGURE 17-1: WATCHDOG TIMER BLOCK DIAGRAM



Note 1: This oscillator is separate from the external RC oscillator on the OSC1 pin.

TABLE 17-2: REGISTERS/BITS ASSOCIATED WITH THE WATCHDOG TIMER

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	$\overline{MCLR}$, WDT
—	Config	See Figure 17-1 for location of WDTPSx bits in Configuration Word.								**(Note 1)**	**(Note 1)**
06h, Unbanked	CPUSTA	—	—	STKAV	GLINTD	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	`--11 11qq`	`--11 qquu`

Legend: - = unimplemented, read as '0', `q` = value depends on condition. Shaded cells are not used by the WDT.
Note 1: This value will be as the device was programmed, or if unprogrammed, will read as all '1's.

17.4.2 MINIMIZING CURRENT CONSUMPTION

To minimize current consumption, all I/O pins should be either at V_{DD}, or V_{SS}, with no external circuitry drawing current from the I/O pin. I/O pins that are hi-impedance inputs should be pulled high or low externally to avoid switching currents caused by floating inputs. The T0CKI input should be at V_{DD} or V_{SS}. The contributions from on-chip pull-ups on PORTB should also be considered and disabled, when possible.

17.5 Code Protection

The code in the program memory can be protected by selecting the microcontroller in Code Protected mode (PM2:PM0 = '000').

In this mode, instructions that are in the on-chip program memory space, can continue to read or write the program memory. An instruction that is executed outside of the internal program memory range will be inhibited from writing to, or reading from, program memory.

Note:	Microchip does not recommend code protecting windowed devices.

If the code protection bit(s) have not been programmed, the on-chip program memory can be read out for verification purposes.

IORWF Inclusive OR WREG with f

Syntax:	[*label*] IORWF f,d
Operands:	$0 \le f \le 255$ $d \in [0,1]$
Operation:	(WREG) .OR. (f) $\rightarrow$ (dest)
Status Affected:	$\overline{Z}$
Encoding:	`0000` `100d` `ffff` `ffff`
Description:	Inclusive OR WREG with register 'f'. If 'd' is 0, the result is placed in WREG. If 'd' is 1, the result is placed back in register 'f'.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: `IORWF RESULT, 0`

Before Instruction

RESULT	=	0x13
WREG	=	0x91

After Instruction

RESULT	=	0x13
WREG	=	0x93

LCALL Long Call

Syntax:	[*label*] LCALL k
Operands:	$0 \le k \le 255$
Operation:	PC + 1 $\rightarrow$ TOS; k $\rightarrow$ PCL, (PCLATH) $\rightarrow$ PCH
Status Affected:	None
Encoding:	`1011` `0111` `kkkk` `kkkk`
Description:	`LCALL` allows an unconditional subroutine call to anywhere within the 64K program memory space. First, the return address (PC + 1) is pushed onto the stack. A 16-bit destination address is then loaded into the program counter. The lower 8-bits of the destination address are embedded in the instruction. The upper 8-bits of PC are loaded from PC high holding latch, PCLATH.
Words:	1
Cycles:	2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write register PCL
No operation	No operation	No operation	No operation

Example:

```
MOVLW  HIGH(SUBROUTINE)
MOVPF  WREG, PCLATH
LCALL  LOW(SUBROUTINE)
```

Before Instruction

`SUBROUTINE`	=	16-bit Address
PC	=	?

After Instruction

PC	=	Address `(SUBROUTINE)`

TABLWT Table Write

Example1: `TABLWT 1, 1, REG`

Before Instruction

REG	=	0x53
TBLATH	=	0xAA
TBLATL	=	0x55
TBLPTR	=	0xA356
MEMORY(TBLPTR)	=	0xFFFF

After Instruction (table write completion)

REG	=	0x53
TBLATH	=	0x53
TBLATL	=	0x55
TBLPTR	=	0xA357
MEMORY(TBLPTR - 1)	=	0x5355

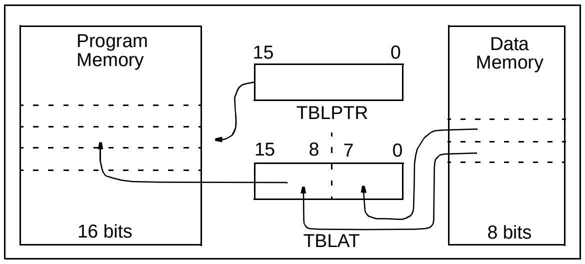
Example 2: `TABLWT 0, 0, REG`

Before Instruction

REG	=	0x53
TBLATH	=	0xAA
TBLATL	=	0x55
TBLPTR	=	0xA356
MEMORY(TBLPTR)	=	0xFFFF

After Instruction (table write completion)

REG	=	0x53
TBLATH	=	0xAA
TBLATL	=	0x53
TBLPTR	=	0xA356
MEMORY(TBLPTR)	=	0xAA53



TLRD Table Latch Read

Syntax:	[*label*] TLRD t,f
Operands:	0 ≤ f ≤ 255 t ∈ [0,1]
Operation:	If t = 0, TBLATL → f; If t = 1, TBLATH → f
Status Affected:	None
Encoding:	`1010` `00tx` `ffff` `ffff`
Description:	Read data from 16-bit table latch (TBLAT) into file register 'f'. Table Latch is unaffected. If t = 1; high byte is read If t = 0; low byte is read This instruction is used in conjunction with `TABLRD` to transfer data from program memory to data memory.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register TBLATH or TBLATL	Process Data	Write register 'f'

Example: `TLRD t, RAM`

Before Instruction

t	=	0	
RAM	=	?	
TBLAT	=	0x00AF	(TBLATH = 0x00)
			(TBLATL = 0xAF)

After Instruction

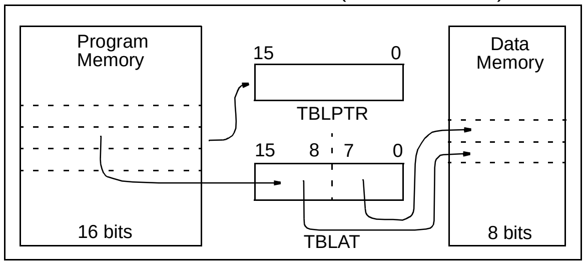
RAM	=	0xAF	
TBLAT	=	0x00AF	(TBLATH = 0x00)
			(TBLATL = 0xAF)

Before Instruction

t	=	1	
RAM	=	?	
TBLAT	=	0x00AF	(TBLATH = 0x00)
			(TBLATL = 0xAF)

After Instruction

RAM	=	0x00	
TBLAT	=	0x00AF	(TBLATH = 0x00)
			(TBLATL = 0xAF)



DC CHARACTERISTICS			**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +125°C for extended -40°C ≤ TA ≤ +85°C for industrial 0°C ≤ TA ≤ +70°C for commercial Operating voltage VDD range as described in Section 20.1				
Param. No.	**Sym**	**Characteristic**	**Min**	**Typ†**	**Max**	**Units**	**Conditions**
		Input Leakage Current (Notes 2, 3)					
D060	IIL	I/O ports (except RA2, RA3)	–	–	±1	μA	Vss ≤ VPIN ≤ VDD, I/O Pin (in digital mode) at hi-impedance PORTB weak pull-ups disabled
D061		$\overline{\text{MCLR}}$, TEST	–	–	±2	μA	VPIN = Vss or VPIN = VDD
D062		RA2, RA3			±2	μA	Vss ≤ VRA2, VRA3 ≤ 12V
D063		OSC1 (EC, RC modes)	–	–	±1	μA	Vss ≤ VPIN ≤ VDD
D063B		OSC1 (XT, LF modes)	–	–	VPIN	μA	RF ≥ 1 MΩ
D064		$\overline{\text{MCLR}}$, TEST	–	–	25	μA	VMCLR = VPP = 12V (when not programming)
D070	IPURB	**PORTB Weak Pull-up Current**	85	130	260	μA	VPIN = VSS, $\overline{\text{RBPU}}$ = 0 4.5V ≤ VDD ≤ 5.5V
		Output Low Voltage					
D080	VOL	I/O ports					IOL = VDD/1.250 mA
			–	–	0.1VDD	V	4.5V ≤ VDD ≤ 5.5V
			–	–	0.1VDD	V	VDD = 3.0V
D081		with TTL buffer	–	–	0.4	V	IOL = 6 mA, VDD = 4.5V **(Note 6)**
D082		RA2 and RA3	–	–	3.0	V	IOL = 60.0 mA, VDD = 5.5V
			–	–	0.6	V	IOL = 60.0 mA, VDD = 4.5V
D083		OSC2/CLKOUT	–	–	0.4	V	IOL = 1 mA, VDD = 4.5V
D084		(RC and EC osc modes)	–	–	0.1VDD	V	IOL = VDD/5 mA (PIC17LC7XX only)
		Output High Voltage (Note 3)					
D090	VOH	I/O ports (except RA2 and RA3)					IOH = -VDD/2.5 mA
			0.9VDD	–	–	V	4.5V ≤ VDD ≤ 5.5V
			0.9VDD	–	–	V	VDD = 3.0V
D091		with TTL buffer	2.4	–	–	V	IOH = -6.0 mA, VDD = 4.5V **(Note 6)**
D093		OSC2/CLKOUT	2.4	–	–	V	IOH = -5 mA, VDD = 4.5V
D094		(RC and EC osc modes)	0.9VDD	–	–	V	IOH = -VDD/5 mA (PIC17LC7XX only)

† Data in "Typ" column is at 5V, 25°C unless otherwise stated.

Note 1: In RC oscillator configuration, the OSC1/CLKIN pin is a Schmitt Trigger input. It is not recommended that the PIC17CXXX devices be driven with external clock in RC mode.

2: The leakage current on the $\overline{\text{MCLR}}$ pin is strongly dependent on the applied voltage level. Higher leakage current may be measured at different input voltages.

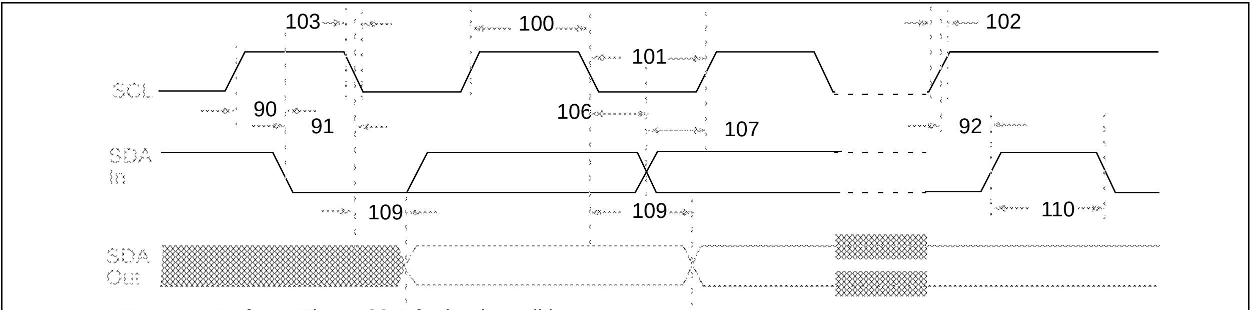
3: Negative current is defined as current sourced by the pin.

4: These specifications are for the programming of the on-chip program memory EPROM through the use of the table write instructions. The complete programming specifications can be found in: PIC17C7XX Programming Specifications (Literature number DS TBD).

5: The $\overline{\text{MCLR}}$/VPP pin may be kept in this range at times other than programming, but is not recommended.

6: For TTL buffers, the better of the two specifications may be used.

FIGURE 20-18: I²C BUS DATA TIMING



Note: Refer to Figure 20-5 for load conditions.

TABLE 20-13: I²C BUS DATA REQUIREMENTS

Param No.	Sym	Characteristic		Min	Max	Units	Conditions
100	Thigh	Clock high time	100 kHz mode	2(Tosc)(BRG + 1)	—	ms	
			400 kHz mode	2(Tosc)(BRG + 1)	—	ms	
			1 MHz mode(1)	2(Tosc)(BRG + 1)	—	ms	
101	Tlow	Clock low time	100 kHz mode	2(Tosc)(BRG + 1)	—	ms	
			400 kHz mode	2(Tosc)(BRG + 1)	—	ms	
			1 MHz mode(1)	2(Tosc)(BRG + 1)	—	ms	
102	Tr	SDA and SCL rise time	100 kHz mode	—	1000	ns	Cb is specified to be from 10 to 400 pF
			400 kHz mode	20 + 0.1Cb	300	ns	
			1 MHz mode(1)	—	300	ns	
103	Tf	SDA and SCL fall time	100 kHz mode	—	300	ns	Cb is specified to be from 10 to 400 pF
			400 kHz mode	20 + 0.1Cb	300	ns	
			1 MHz mode(1)	—	10	ns	
90	Tsu:sta	START condition setup time	100 kHz mode	2(Tosc)(BRG + 1)	—	ms	Only relevant for Repeated Start condition
			400 kHz mode	2(Tosc)(BRG + 1)	—	ms	
			1 MHz mode(1)	2(Tosc)(BRG + 1)	—	ms	
91	Thd:sta	START condition hold time	100 kHz mode	2(Tosc)(BRG + 1)	—	ms	After this period, the first clock pulse is generated
			400 kHz mode	2(Tosc)(BRG + 1)	—	ms	
			1 MHz mode(1)	2(Tosc)(BRG + 1)	—	ms	
106	Thd:dat	Data input hold time	100 kHz mode	0	—	ns	
			400 kHz mode	0	0.9	ms	
			1 MHz mode(1)	0	—	ns	
107	Tsu:dat	Data input setup time	100 kHz mode	250	—	ns	**(Note 2)**
			400 kHz mode	100	—	ns	
			1 MHz mode(1)	100	—	ns	
92	Tsu:sto	STOP condition setup time	100 kHz mode	2(Tosc)(BRG + 1)	—	ms	
			400 kHz mode	2(Tosc)(BRG + 1)	—	ms	
			1 MHz mode(1)	2(Tosc)(BRG + 1)	—	ms	
109	Taa	Output valid from clock	100 kHz mode	—	3500	ns	
			400 kHz mode	—	1000	ns	
			1 MHz mode(1)	—	400	ns	

Note 1: Maximum pin capacitance = 10 pF for all I²C pins.

2: A fast mode (400 KHz) I²C bus device can be used in a standard mode I²C bus system, but the parameter # 107 ≥ 250 ns must then be met. This will automatically be the case if the device does not stretch the LOW period of the SCL signal. If such a device does stretch the LOW period of the SCL signal, it must output the next data bit to the SDA line. Parameter #102 + #107 = 1000 + 250 = 1250 ns (for 100 kHz mode) before the SCL line is released.

3: C_b is specified to be from 10-400pF. The minimum specifications are characterized with C_b=10pF. The rise time spec (t_r) is characterized with R_p=R_p min. The minimum fall time specification (t_f) is characterized with C_b=10pF,and R_p=R_p max. These are only valid for fast mode operation (VDD=4.5-5.5V) and where the SPM bit (SSPSTAT<7>) =1.)

4: Max specifications for these parameters are valid for falling edge only. Specs are characterized with R_p=R_p min and C_b=400pF for standard mode, 200pF for fast mode, and 10pF for 1MHz mode.