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	10MHz
Connectivity	I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	36
Program Memory Size	7KB (4K x 14)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	368 x 8
Voltage - Supply (Vcc/Vdd)	2V ~ 5.5V
Data Converters	A/D 14x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	44-TQFP
Supplier Device Package	44-TQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic16lf747-i-pt

PIC16F7X7

Table of Contents

TO OUR VALUED CUSTOMERS

It is our intention to provide our valued customers with the best documentation possible to ensure successful use of your Microchip products. To this end, we will continue to improve our publications to better suit your needs. Our publications will be refined and enhanced as new volumes and updates are introduced.

If you have any questions or comments regarding this publication, please contact the Marketing Communications Department via E-mail at **docerrors@microchip.com** or fax the **Reader Response Form** in the back of this data sheet to (480) 792-4150. We welcome your feedback.

Most Current Data Sheet

To obtain the most up-to-date version of this data sheet, please register at our Worldwide Web site at:

   http://www.microchip.com

You can determine the version of a data sheet by examining its literature number found on the bottom outside corner of any page. The last character of the literature number is the version number, (e.g., DS30000A is version A of document DS30000).

Errata

An errata sheet, describing minor operational differences from the data sheet and recommended workarounds, may exist for current devices. As device/documentation issues become known to us, we will publish an errata sheet. The errata will specify the revision of silicon and revision of document to which it applies.

To determine if an errata sheet exists for a particular device, please check with one of the following:

* Microchip's Worldwide Web site; http://www.microchip.com
* Your local Microchip sales office (see last page)

When contacting a sales office, please specify which device, revision of silicon and data sheet (include literature number) you are using.

Customer Notification System

Register on our web site at **www.microchip.com** to receive the most current information on all of our products.

2.2.2.1 Status Register

The Status register contains the arithmetic status of the ALU, the Reset status and the bank select bits for data memory.

The Status register can be the destination for any instruction, as with any other register. If the Status register is the destination for an instruction that affects the Z, DC or C bits, then the write to these three bits is disabled. These bits are set or cleared according to the device logic. Furthermore, the TO and PD bits are not writable, therefore, the result of an instruction with the Status register as destination may be different than intended.

For example, CLRF STATUS, will clear the upper three bits and set the Z bit. This leaves the Status register as 000u u1uu (where u = unchanged).

It is recommended, therefore, that only BCF, BSF, SWAPF and MOVWF instructions are used to alter the Status register because these instructions do not affect the Z, C or DC bits from the Status register. For other instructions not affecting any Status bits, see **Section 16.0 "Instruction Set Summary"**.

> **Note 1:** The C and DC bits operate as a borrow and digit borrow bit, respectively, in subtraction. See the SUBLW and SUBWF instructions for examples.

REGISTER 2-1:   STATUS: ARITHMETIC STATUS REGISTER (ADDRESS 03h, 83h, 103h, 183h)

R/W-0	R/W-0	R/W-0	R-1	R-1	R/W-x	R/W-x	R/W-x
IRP	RP1	RP0	$\overline{\text{TO}}$	$\overline{\text{PD}}$	Z	DC	C
bit 7							bit 0

bit 7   **IRP:** Register Bank Select bit (used for indirect addressing)

1 = Bank 2, 3 (100h-1FFh)
0 = Bank 0, 1 (00h-FFh)

bit 6-5   **RP1:RP0**: Register Bank Select bits (used for direct addressing)

11 = Bank 3 (180h-1FFh)
10 = Bank 2 (100h-17Fh)
01 = Bank 1 (80h-FFh)
00 = Bank 0 (00h-7Fh)
Each bank is 128 bytes.

bit 4   **$\overline{\text{TO}}$**: Time-out bit

1 = After power-up, CLRWDT instruction or SLEEP instruction
0 = A WDT time-out occurred

bit 3   **$\overline{\text{PD}}$**: Power-Down bit

1 = After power-up or by the CLRWDT instruction
0 = By execution of the SLEEP instruction

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 (ADDWF, ADDLW, SUBLW, SUBWF 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

bit 0   **C:** Carry/$\overline{\text{borrow}}$ bit (ADDWF, ADDLW, SUBLW, SUBWF instructions)

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 occurred

> **Note:** For $\overline{\text{borrow}}$, the polarity is reversed. A subtraction is executed by adding the two's complement of the second operand. For rotate (RRF, RLF) instructions, this bit is loaded with either the high or low-order bit of the source register.

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

6.0   TIMER0 MODULE

The Timer0 module timer/counter has the following features:

• 8-bit timer/counter
• Readable and writable
• 8-bit software programmable prescaler
• Internal or external clock select
• Interrupt on overflow from FFh to 00h
• Edge select for external clock

Additional information on the Timer0 module is available in the *"PIC® Mid-Range MCU Family Reference Manual"* (DS33023).

Figure 6-1 is a block diagram of the Timer0 module and the prescaler shared with the WDT.

6.1   Timer0 Operation

Timer0 operation is controlled through the OPTION_REG register (see Register 2-2). Timer mode is selected by clearing bit T0CS (OPTION_REG<5>). In Timer mode, the Timer0 module will increment every instruction cycle (without prescaler). If the TMR0 register is written, the increment is inhibited for the following two instruction cycles. The user can work around this by writing an adjusted value to the TMR0 register.

Counter mode is selected by setting bit, T0CS (OPTION_REG<5>). In Counter mode, Timer0 will increment, either on every rising or falling edge of pin RA4/T0CKI/C1OUT. The incrementing edge is determined by the Timer0 Source Edge Select bit, T0SE (OPTION_REG<4>). Clearing bit T0SE selects the rising edge. Restrictions on the external clock input are discussed in detail in **Section 6.3 "Using Timer0 With an External Clock"**.

The prescaler is mutually, exclusively shared between the Timer0 module and the Watchdog Timer. The prescaler is not readable or writable. **Section 6.4 "Prescaler"** details the operation of the prescaler.

6.2   Timer0 Interrupt

The TMR0 interrupt is generated when the TMR0 register overflows from FFh to 00h. This overflow sets bit TMR0IF (INTCON<2>). The interrupt can be masked by clearing bit TMR0IE (INTCON<5>). Bit TMR0IF must be cleared in software by the Timer0 module Interrupt Service Routine before re-enabling this interrupt. The TMR0 interrupt cannot awaken the processor from Sleep since the timer is shut-off during Sleep.

FIGURE 6-1:   BLOCK DIAGRAM OF THE TIMER0/WDT PRESCALER

Note: T0CS, T0SE, PSA and PS2:PS0 are (OPTION_REG<5:0>).
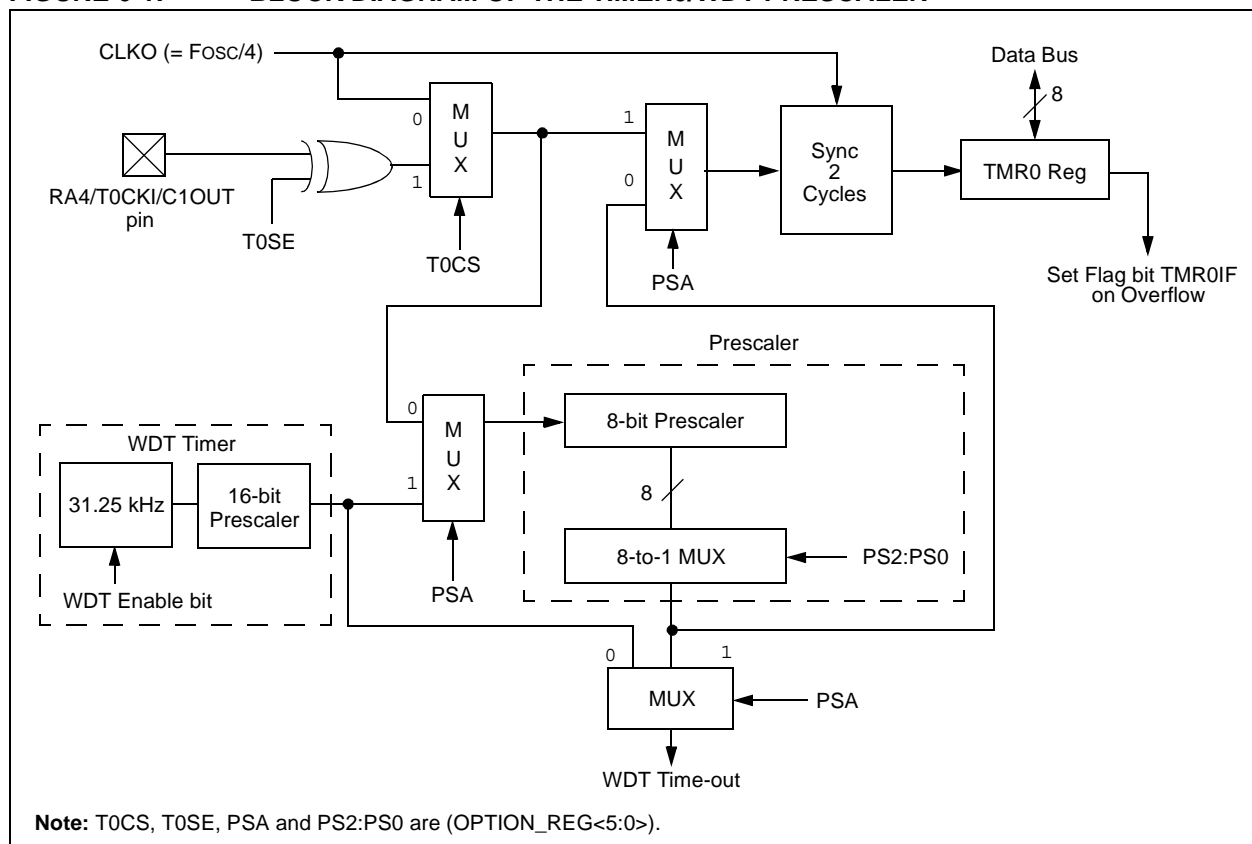
PIC16F7X7

7.5 Timer1 Operation in Asynchronous Counter Mode

If control bit, $\overline{\text{T1SYNC}}$ (T1CON<2>), is set, the external clock input is not synchronized. The timer continues to increment asynchronous to the internal phase clocks. The timer will continue to run during Sleep and can generate an interrupt on overflow that will wake-up the processor. However, special precautions in software are needed to read/write the timer (**Section 7.5.1 "Reading and Writing Timer1 in Asynchronous Counter Mode"**).

In Asynchronous Counter mode, Timer1 cannot be used as a time base for capture or compare operations.

7.5.1 READING AND WRITING TIMER1 IN ASYNCHRONOUS COUNTER MODE

Reading TMR1H or TMR1L while the timer is running from an external asynchronous clock will ensure a valid read (taken care of in hardware). However, the user should keep in mind that reading the 16-bit timer in two 8-bit values itself, poses certain problems, since the timer may overflow between the reads.

For writes, it is recommended that the user simply stop the timer and write the desired values. A write contention may occur by writing to the Timer registers while the register is incrementing. This may produce an unpredictable value in the Timer register.

Reading the 16-bit value requires some care. The example codes provided in Example 7-1 and Example 7-2 demonstrate how to write to and read Timer1 while it is running in Asynchronous mode.

EXAMPLE 7-1: WRITING A 16-BIT FREE RUNNING TIMER

```
; All interrupts are disabled
CLRF       TMR1L           ; Clear Low byte, Ensures no rollover into TMR1H
MOVLW      HI_BYTE         ; Value to load into TMR1H
MOVWF      TMR1H, F        ; Write High byte
MOVLW      LO_BYTE         ; Value to load into TMR1L
MOVWF      TMR1H, F        ; Write Low byte
; Re-enable the Interrupt (if required)
CONTINUE                   ; Continue with your code
```

EXAMPLE 7-2: READING A 16-BIT FREE RUNNING TIMER

```
; All interrupts are disabled
MOVF       TMR1H, W        ; Read high byte
MOVWF      TMPH
MOVF       TMR1L, W        ; Read low byte
MOVWF      TMPL
MOVF       TMR1H, W        ; Read high byte
SUBWF      TMPH, W         ; Sub 1st read with 2nd read
BTFSC      STATUS, Z       ; Is result = 0
GOTO       CONTINUE        ; Good 16-bit read
; TMR1L may have rolled over between the read of the high and low bytes.
; Reading the high and low bytes now will read a good value.
MOVF       TMR1H, W        ; Read high byte
MOVWF      TMPH
MOVF       TMR1L, W        ; Read low byte
MOVWF      TMPL            ; Re-enable the Interrupt (if required)
CONTINUE                   ; Continue with your code
```

PIC16F7X7

REGISTER 9-1: **CCPxCON: CCPx CONTROL REGISTER (ADDRESS 17h, 1Dh, 97h)**

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	CCPxX	CCPxY	CCPxM3	CCPxM2	CCPxM1	CCPxM0

bit 7   bit 0

bit 7-6   **Unimplemented:** Read as '0'

bit 5-4   **CCPxX:CCPxY**: PWM Least Significant bits

Capture mode:
Unused.

Compare mode:
Unused.

PWM mode:
These bits are the two LSbs of the PWM duty cycle. The eight MSbs are found in CCPRxL.

bit 3-0   **CCPxM3:CCPxM0**: CCPx Mode Select bits

0000 = Capture/Compare/PWM disabled (resets CCPx module)
0100 = Capture mode, every falling edge
0101 = Capture mode, every rising edge
0110 = Capture mode, every 4th rising edge
0111 = Capture mode, every 16th rising edge
1000 = Compare mode, set output on match (CCPxIF bit is set)
1001 = Compare mode, clear output on match (CCPxIF bit is set)
1010 = Compare mode, generate software interrupt on match (CCPxIF bit is set, CCPx pin is unaffected)
1011 = Compare mode, trigger special event (CCPxIF bit is set, CCPx pin is unaffected); CCP1 clears Timer1; CCP2 clears Timer1 and starts an A/D conversion (if A/D module is enabled)
11xx = PWM mode

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

PIC16F7X7

FIGURE 10-5: SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 0)
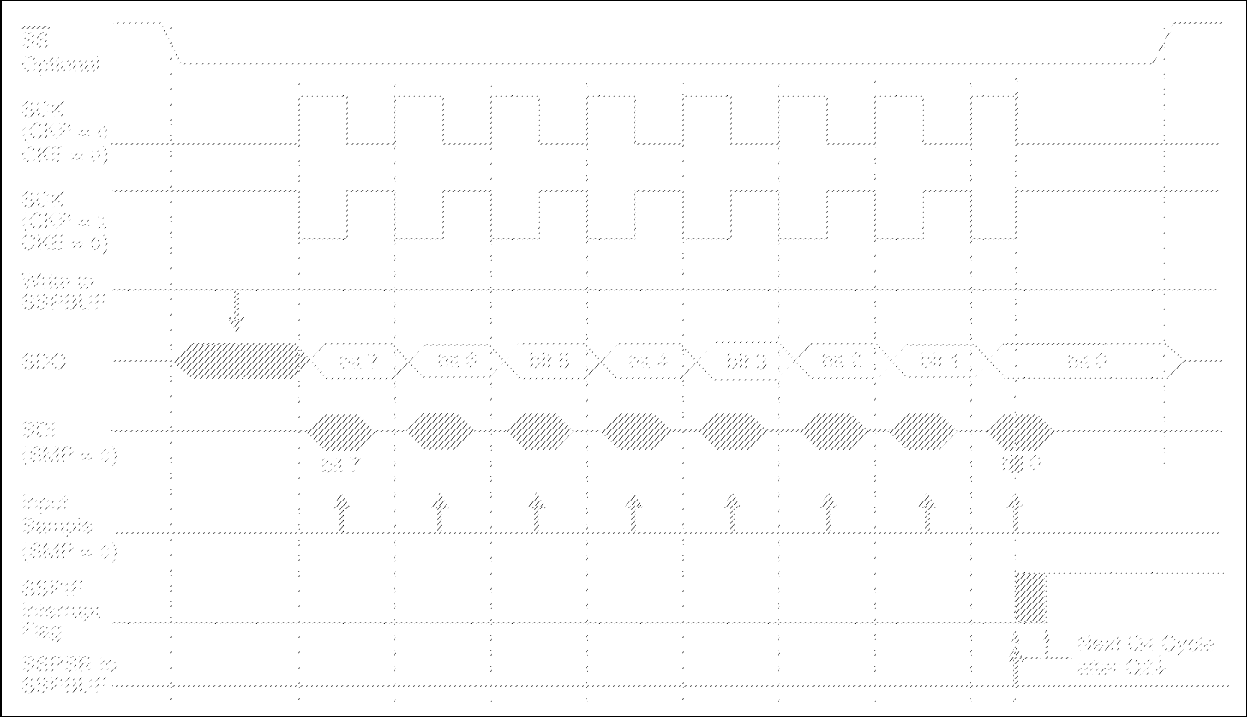

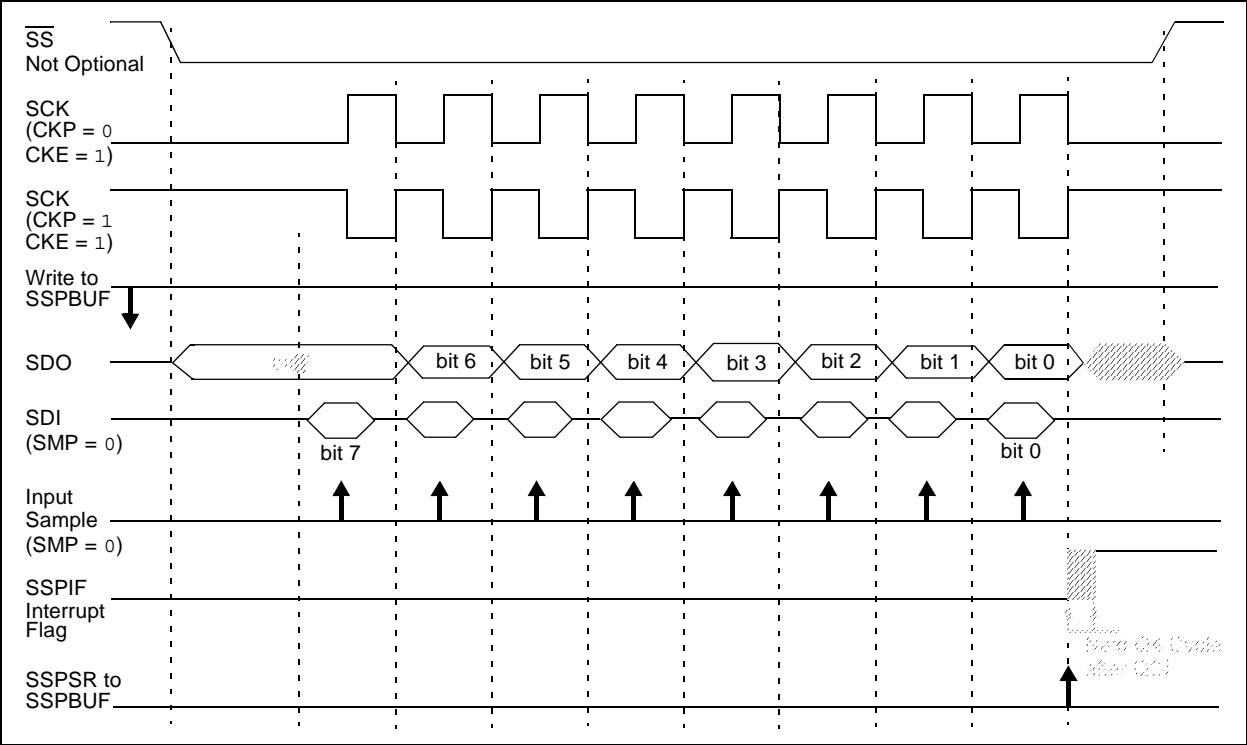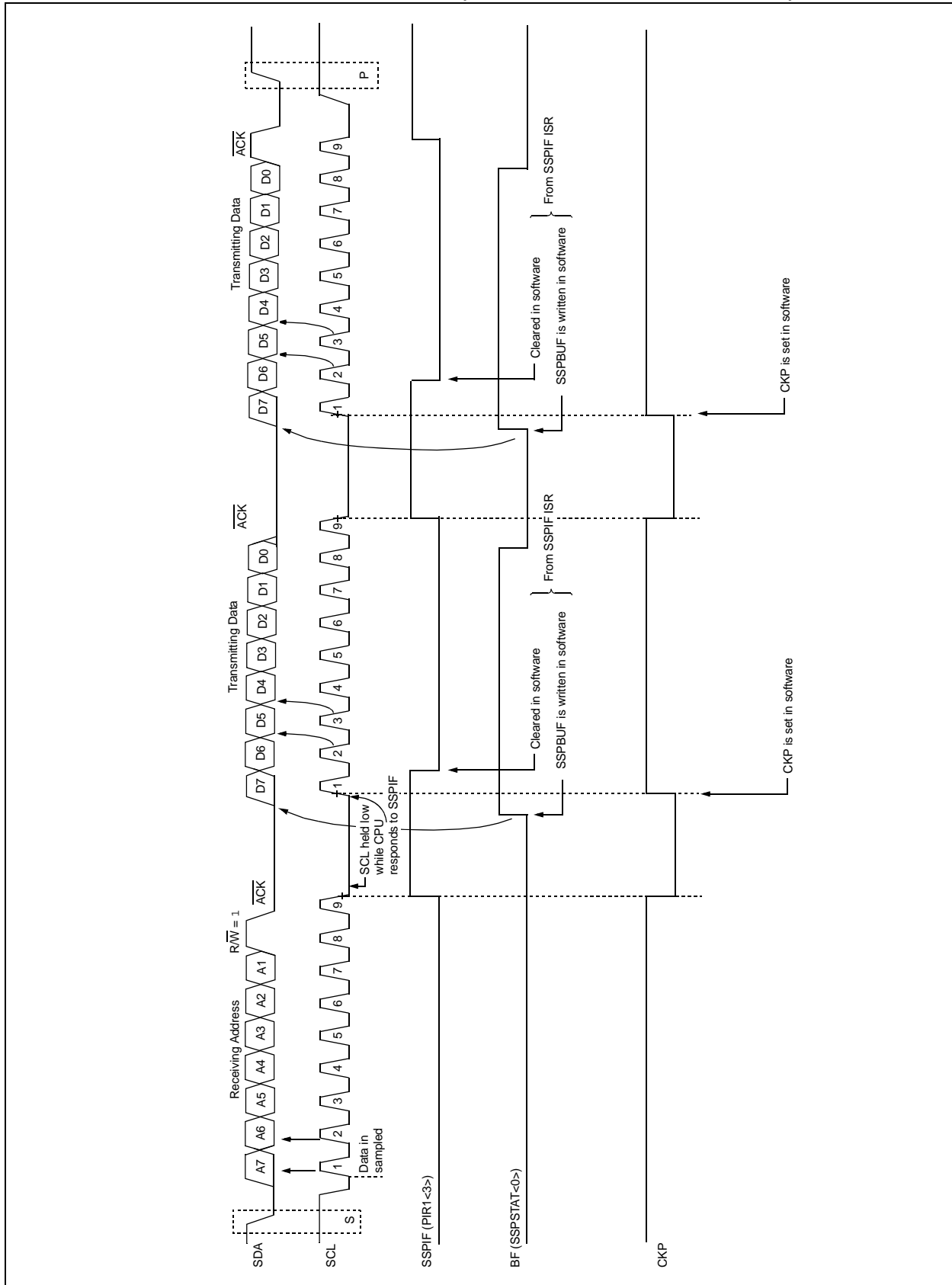**FIGURE 10-6:** SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 1)

FIGURE 10-9: **I²C™ SLAVE MODE TIMING (TRANSMISSION, 7-BIT ADDRESS)**

PIC16F7X7

10.4.5 GENERAL CALL ADDRESS SUPPORT

The addressing procedure for the I²C bus is such that the first byte after the Start condition usually determines which device will be the slave addressed by the master. The exception is the general call address which can address all devices. When this address is used, all devices should, in theory, respond with an Acknowledge.

The general call address is one of eight addresses reserved for specific purposes by the I²C protocol. It consists of all '0's with R/$\overline{W}$ = 0.

The general call address is recognized when the General Call Enable bit (GCEN) is enabled (SSPCON2<7> set). Following a Start bit detect, 8 bits are shifted into the SSPSR and the address is compared against the SSPADD. It is also compared to the general call address and fixed in hardware.

If the general call address matches, the SSPSR is transferred to the SSPBUF, the BF flag bit is set (eighth bit) and on the falling edge of the ninth bit ($\overline{ACK}$ bit), the SSPIF interrupt flag bit is set.

When the interrupt is serviced, the source for the interrupt can be checked by reading the contents of the SSPBUF. The value can be used to determine if the address was device specific or a general call address.

In 10-bit mode, the SSPADD is required to be updated for the second half of the address to match and the UA bit is set (SSPSTAT<1>). If the general call address is sampled when the GCEN bit is set and while the slave is configured in 10-bit Address mode, then the second half of the address is not necessary, the UA bit will not be set and the slave will begin receiving data after the Acknowledge (Figure 10-15).

FIGURE 10-15: SLAVE MODE GENERAL CALL ADDRESS SEQUENCE (7 OR 10-BIT ADDRESS MODE)
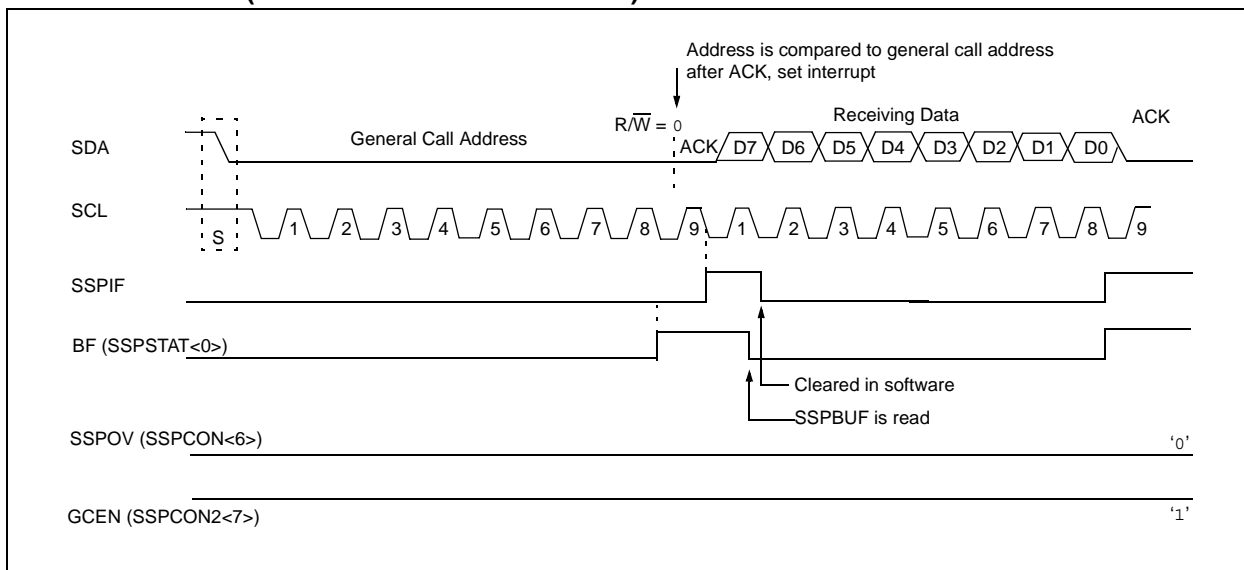
10.4.14 SLEEP OPERATION

While in Sleep mode, the I^{2}C module can receive addresses or data and when an address match or complete byte transfer occurs, wake the processor from Sleep (if the MSSP interrupt is enabled).

10.4.15 EFFECT OF A RESET

A Reset disables the MSSP module and terminates the current transfer.

10.4.16 MULTI-MASTER MODE

In Multi-Master mode, the interrupt generation on the detection of the Start and Stop conditions allows the determination of when the bus is free. 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 (SSPSTAT<4>) is set or the bus is Idle, with both the S and P bits clear. When the bus is busy, enabling the SSP interrupt will generate the interrupt when the Stop condition occurs.

In multi-master operation, the SDA line must be monitored for arbitration to see if the signal level is at the expected output level. This check is performed in hardware with the result placed in the BCLIF bit.

The states where arbitration can be lost are:

• Address Transfer
• Data Transfer
• A Start Condition
• A Repeated Start Condition
• An Acknowledge Condition

10.4.17 MULTI-MASTER COMMUNICATION, BUS COLLISION AND BUS ARBITRATION

Multi-Master mode support is achieved by bus arbitration. When the master outputs address/data bits onto the SDA pin, arbitration takes place when the master outputs a '1' on SDA by letting SDA float high and another master asserts a '0'. When the SCL pin floats high, data should be stable. If the expected data on SDA is a '1' and the data sampled on the SDA pin = 0, then a bus collision has taken place. The master will set the Bus Collision Interrupt Flag, BCLIF and reset the I^{2}C port to its Idle state (Figure 10-25).

If a transmit was in progress when the bus collision occurred, the transmission is halted, the BF flag is cleared, the SDA and SCL lines are deasserted and the SSPBUF can be written to. When the user services the bus collision Interrupt Service Routine and if the I^{2}C bus is free, the user can resume communication by asserting a Start condition.

If a Start, Repeated Start, Stop or Acknowledge condition was in progress when the bus collision occurred, the condition is aborted, the SDA and SCL lines are deasserted and the respective control bits in the SSPCON2 register are cleared. When the user services the bus collision Interrupt Service Routine and if the I^{2}C bus is free, the user can resume communication by asserting a Start condition.

The master will continue to monitor the SDA and SCL pins. If a Stop condition occurs, the SSPIF bit will be set.

A write to the SSPBUF will start the transmission of data at the first data bit, regardless of where the transmitter left off when the bus collision occurred.

In Multi-Master mode, the interrupt generation on the detection of Start and Stop conditions allows the determination of when the bus is free. Control of the I^{2}C bus can be taken when the P bit is set in the SSPSTAT register or the bus is Idle and the S and P bits are cleared.

FIGURE 10-25: BUS COLLISION TIMING FOR TRANSMIT AND ACKNOWLEDGE

PIC16F7X7
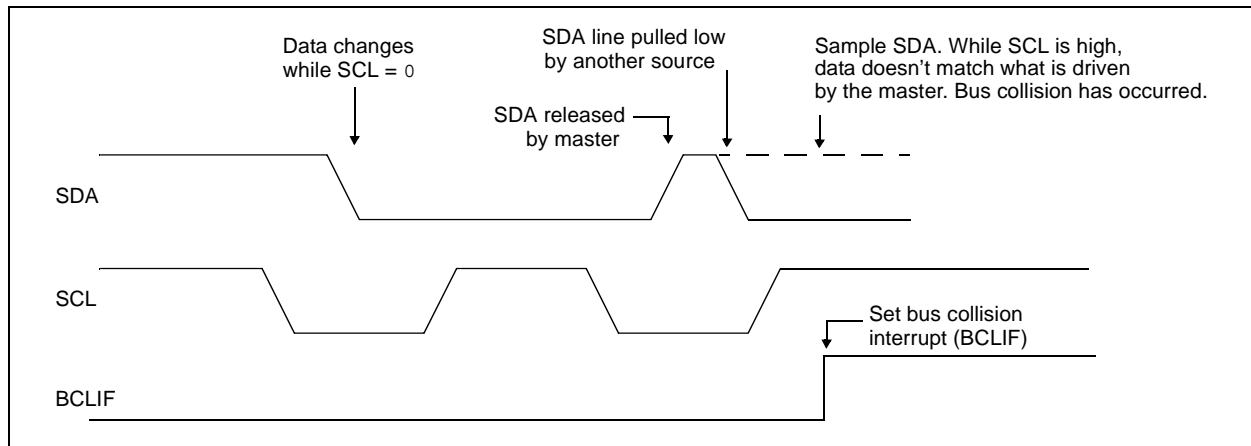
11.2 AUSART Asynchronous Mode

In this mode, the AUSART uses standard Non-Return-to-Zero (NRZ) format (one Start bit, eight or nine data bits and one Stop bit). The most common data format is 8 bits. An on-chip, dedicated, 8-bit Baud Rate Generator can be used to derive standard baud rate frequencies from the oscillator. The AUSART transmits and receives the LSb first. The transmitter and receiver are functionally independent but use the same data format and baud rate. The Baud Rate Generator produces a clock, either x16 or x64 of the bit shift rate, depending on bit BRGH (TXSTA<2>). Parity is not supported by the hardware but can be implemented in software (and stored as the ninth data bit). Asynchronous mode is stopped during Sleep.

Asynchronous mode is selected by clearing bit, SYNC (TXSTA<4>).

The AUSART asynchronous module consists of the following important elements:

- Baud Rate Generator
- Sampling Circuit
- Asynchronous Transmitter
- Asynchronous Receiver

11.2.1 AUSART ASYNCHRONOUS TRANSMITTER

The AUSART transmitter block diagram is shown in Figure 11-1. The heart of the transmitter is the Transmit (Serial) Shift Register (TSR). The Shift register obtains its data from the Read/Write Transmit Buffer register, TXREG. The TXREG register is loaded with data in software. The TSR register is not loaded until the Stop bit has been transmitted from the previous load. As soon as the Stop bit is transmitted, the TSR is loaded with new data from the TXREG register (if available). Once the TXREG register transfers the data to the TSR register (occurs in one T_{CY}), the TXREG register is empty and flag bit, TXIF (PIR1<4>), is set. This

interrupt can be enabled/disabled by setting/clearing enable bit, TXIE (PIE1<4>). Flag bit TXIF will be set regardless of the state of enable bit TXIE and cannot be cleared in software. It will reset only when new data is loaded into the TXREG register. While flag bit TXIF indicates the status of the TXREG register, another bit, TRMT (TXSTA<1>), shows the status of the TSR register. Status bit TRMT is a read-only bit which is set when the TSR register is empty. No interrupt logic is tied to this bit, so the user has to poll this bit in order to determine if the TSR register is empty.

> **Note 1:** The TSR register is not mapped in data memory so it is not available to the user.
>
> **2:** Flag bit TXIF is set when enable bit TXEN is set. TXIF is cleared by loading TXREG.

Transmission is enabled by setting enable bit, TXEN (TXSTA<5>). The actual transmission will not occur until the TXREG register has been loaded with data and the Baud Rate Generator (BRG) has produced a shift clock (Figure 11-2). The transmission can also be started by first loading the TXREG register and then setting enable bit TXEN. Normally, when transmission is first started, the TSR register is empty. At that point, transfer to the TXREG register will result in an immediate transfer to TSR, resulting in an empty TXREG. A back-to-back transfer is thus possible (Figure 11-3). Clearing enable bit TXEN during a transmission will cause the transmission to be aborted and will reset the transmitter. As a result, the RC6/TX/CK pin will revert to high-impedance.

In order to select 9-bit transmission, transmit bit, TX9 (TXSTA<6>), should be set and the ninth bit should be written to TX9D (TXSTA<0>). The ninth bit must be written before writing the 8-bit data to the TXREG register. This is because a data write to the TXREG register can result in an immediate transfer of the data to the TSR register (if the TSR is empty). In such a case, an incorrect ninth data bit may be loaded in the TSR register.

FIGURE 11-1:   AUSART TRANSMIT BLOCK DIAGRAM

© 2003-2013 Microchip Technology Inc.
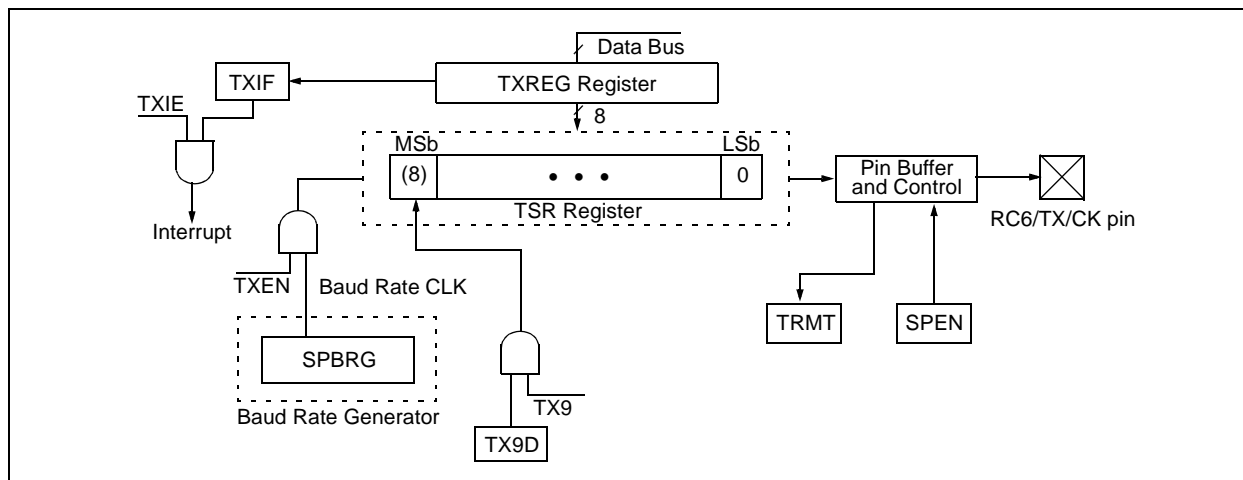
When setting up an Asynchronous Transmission, follow these steps:

1. Initialize the SPBRG register for the appropriate baud rate. If a high-speed baud rate is desired, set bit BRGH (see **Section 11.1 "AUSART Baud Rate Generator (BRG)"**).
2. Enable the asynchronous serial port by clearing bit SYNC and setting bit SPEN.
3. If interrupts are desired, then set enable bit TXIE.
4. If 9-bit transmission is desired, then set transmit bit TX9.
5. Enable the transmission by setting bit TXEN which will also set bit TXIF.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit TX9D.
7. Load data to the TXREG register (starts transmission).
8. If using interrupts, ensure that GIE and PEIE (bits 7 and 6) of the INTCON register are set.

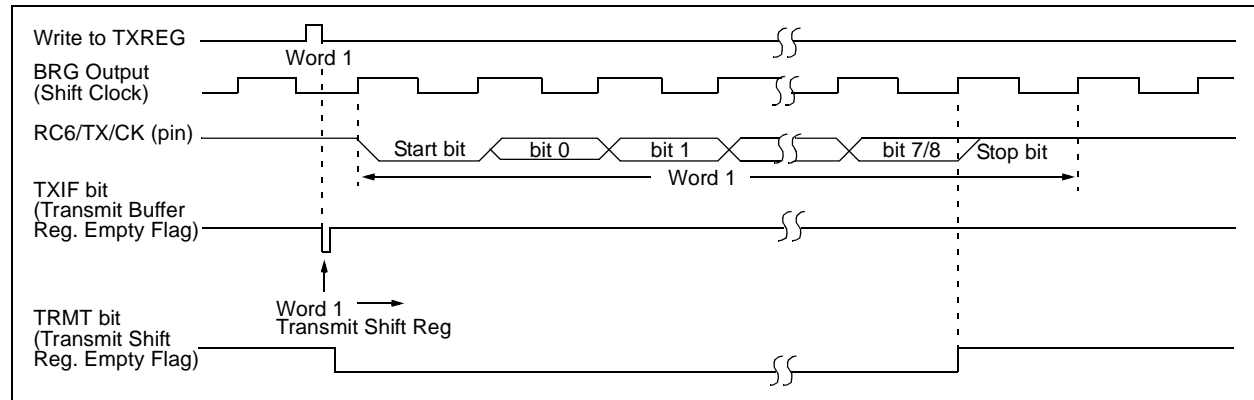**FIGURE 11-2: ASYNCHRONOUS MASTER TRANSMISSION**

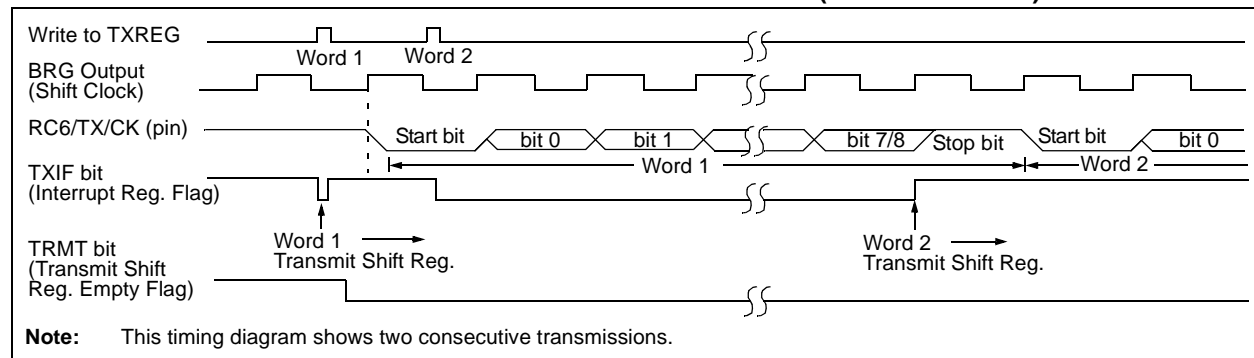**FIGURE 11-3: ASYNCHRONOUS MASTER TRANSMISSION (BACK TO BACK)**

Note: This timing diagram shows two consecutive transmissions.

TABLE 11-7: REGISTERS ASSOCIATED WITH ASYNCHRONOUS TRANSMISSION

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on all other Resets
0Bh, 8Bh, 10Bh,18Bh	INTCON	GIE	PEIE	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	0000 000u
0Ch	PIR1	PSPIF[1]	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
18h	RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	0000 000x
19h	TXREG	AUSART Transmit Data Register								0000 0000	0000 0000
8Ch	PIE1	PSPIE[1]	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
98h	TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
99h	SPBRG	Baud Rate Generator Register								0000 0000	0000 0000

Legend: x = unknown, — = unimplemented locations read as '0'. Shaded cells are not used for asynchronous transmission.
Note 1: Bits PSPIE and PSPIF are reserved on 28-pin devices; always maintain these bits clear.

PIC16F7X7

12.1 A/D Acquisition Requirements

For the A/D converter to meet its specified accuracy, the charge holding capacitor (C_{HOLD}) must be allowed to fully charge to the input channel voltage level. The analog input model is shown in Figure 12-2. The source impedance (R_S) and the internal sampling switch (R_{SS}) impedance directly affect the time required to charge the capacitor C_{HOLD}. The sampling switch (R_{SS}) impedance varies over the device voltage (V_{DD}), see Figure 12-2. **The maximum recommended impedance for analog sources is 2.5 kΩ.** As the impedance is decreased, the acquisition time may be decreased. After the analog input channel is selected (changed), this acquisition must be done before the conversion can be started.

To calculate the minimum acquisition time, Equation 12-1 may be used. This equation assumes that 1/2 LSb error is used (1024 steps for the A/D). The 1/2 LSb error is the maximum error allowed for the A/D to meet its specified resolution.

To calculate the minimum acquisition time, T_{ACQ}, see the *"PIC® Mid-Range MCU Family Reference Manual"* (DS33023).

EQUATION 12-1: ACQUISITION TIME

$$
\begin{aligned}
T_{ACQ} &= \text{Amplifier Settling Time + Hold Capacitor Charging Time + Temperature Coefficient} \\[4pt]
&= T_{AMP} + T_C + T_{COFF} \\
&= 2\ \mu s + T_C + [(\text{Temperature} - 25°C)(0.05\ \mu s/°C)] \\
T_C &= C_{HOLD}\ (R_{IC} + R_{SS} + R_S)\ \ln(1/2047) \\
&= -120\ pF\ (1\ k\Omega + 7\ k\Omega + 10\ k\Omega)\ \ln(0.0004885) \\
&= 16.47\ \mu s \\
T_{ACQ} &= 2\ \mu s + 16.47\ \mu s + [(50°C - 25°C)(0.05\ \mu s/°C)] \\
&= 19.72\ \mu s
\end{aligned}
$$

Note 1: The reference voltage (V_{REF}) has no effect on the equation since it cancels itself out.

2: The charge holding capacitor (C_{HOLD}) is not discharged after each conversion.

3: The maximum recommended impedance for analog sources is 10 kΩ. This is required to meet the pin leakage specification.

4: After a conversion has completed, a 2.0 T_{AD} delay must complete before acquisition can begin again. During this time, the holding capacitor is not connected to the selected A/D input channel.

FIGURE 12-2: ANALOG INPUT MODEL
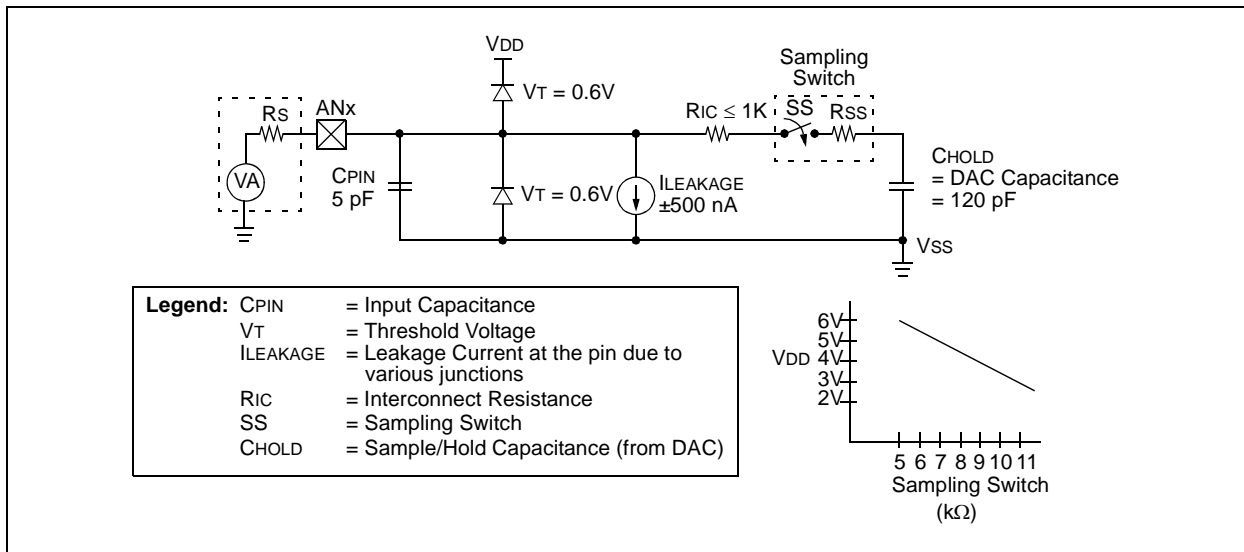
12.7 A/D Operation During Sleep

The A/D module can operate during Sleep mode. This requires that the A/D clock source be set to RC (ADCS1:ADCS0 = 11). When the RC clock source is selected, the A/D module waits one instruction cycle before starting the conversion. This allows the SLEEP instruction to be executed which eliminates all digital switching noise from the conversion. When the conversion is completed, the GO/DONE bit will be cleared and the result loaded into the ADRESH register. If the A/D interrupt is enabled, the device will wake-up from Sleep. If the A/D interrupt is not enabled, the A/D module will then be turned off, although the ADON bit will remain set.

When the A/D clock source is another clock option (not RC), a SLEEP instruction will cause the present conversion to be aborted and the A/D module to be turned off, though the ADON bit will remain set.

Turning off the A/D places the A/D module in its lowest current consumption state.

Note:	For the A/D module to operate in Sleep, the A/D clock source must be set to RC (ADCS1:ADCS0 = 11). To perform an A/D conversion in Sleep, ensure the SLEEP instruction immediately follows the instruction that sets the GO/DONE bit.

12.8 Effects of a Reset

A device Reset forces all registers to their Reset state. The A/D module is disabled and any conversion in progress is aborted. All A/D input pins are configured as analog inputs.

The ADRESH register will contain unknown data after a Power-on Reset.

12.9 Use of the CCP Trigger

An A/D conversion can be started by the "special event trigger" of the CCP2 module. This requires that the CCP2M3:CCP2M0 bits (CCP2CON<3:0>) be programmed as '1011' and that the A/D module is enabled (ADON bit is set). When the trigger occurs, the GO/DONE bit will be set, starting the A/D conversion and the Timer1 counter will be reset to zero. Timer1 is reset to automatically repeat the A/D acquisition period with minimal software overhead (moving the ADRESH to the desired location). The appropriate analog input channel must be selected and an appropriate acquisition time should pass before the "special event trigger" sets the GO/DONE bit (starts a conversion).

If the A/D module is not enabled (ADON is cleared), then the "special event trigger" will be ignored by the A/D module but will still reset the Timer1 counter.

TABLE 12-2: SUMMARY OF A/D REGISTERS

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on all other Resets
0Bh,8Bh, 10Bh, 18Bh	INTCON	GIE	PEIE	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	0000 000u
0Ch	PIR1	PSPIF[1]	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
0Dh	PIR2	OSFIF	CMIF	LVDIF	—	BCLIF	—	CCP3IF	CCP2IF	000- 0-00	000- 0-00
8Ch	PIE1	PSPIE[1]	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
8Dh	PIE2	OSFIE	CMIE	LVDIE	—	BCLIE	—	CCP3IE	CCP2IE	000- 0-00	000- 0-00
1Eh	ADRESH	A/D Result Register High Byte								xxxx xxxx	uuuu uuuu
1Fh	ADCON0	ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/DONE	CHS3	ADON	0000 0000	0000 0000
9Fh	ADCON1	ADFM	ADCS2	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	0000 000	0000 0000
05h	PORTA	RA7	RA6	RA5	RA4	RA3	RA2	RA1	RA0	xx0x 0000	uu0u 0000
85h	TRISA	PORTA Data Direction Register								1111 1111	1111 1111
09h	PORTE[2]	—	—	—	—	RE3[3]	RE2	RE1	RE0	---- x000	---- x000
89h	TRISE[2]	IBF	OBF	IBOV	PSPMODE	—[3]	PORTE Data Direction bits			0000 1111	0000 1111

Legend: x = unknown, u = unchanged, — = unimplemented, read as '0'. Shaded cells are not used for A/D conversion.

Note 1: Bits PSPIE and PSPIF are reserved on the PIC16F737/767 devices; always maintain these bits clear.

2: These registers are reserved on the PIC16F737/767 devices.

3: RE3 is an input only. The state of the TRISE3 bit has no effect and will always read '1'.

PIC16F7X7

TABLE 15-4: INITIALIZATION CONDITIONS FOR ALL REGISTERS

Register	Power-on Reset, Brown-out Reset	MCLR Reset, WDT Reset	Wake-up via WDT or Interrupt
W	xxxx xxxx	uuuu uuuu	uuuu uuuu
INDF	N/A	N/A	N/A
TMR0	xxxx xxxx	uuuu uuuu	uuuu uuuu
PCL	0000h	0000h	PC + 1[2]
STATUS	0001 1xxx	000q quuu[3]	uuuq quuu[3]
FSR	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTA	xx0x 0000	uu0u 0000	uuuu uuuu
PORTB	xx00 0000	uu00 0000	uuuu uuuu
PORTC	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTD	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTE (PIC16F737/767)	---- x---	---- u---	---- u---
PORTE (PIC16F747/777)	---- x000	---- u000	---- uuuu
PCLATH	---0 0000	---0 0000	---u uuuu
INTCON	0000 000x	0000 000u	uuuu uuuu[1]
PIR1	0000 0000	0000 0000	uuuu uuuu[1]
PIR2	000- 0-00	000- 0-00	uuu- u-uu
TMR1L	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR1H	xxxx xxxx	uuuu uuuu	uuuu uuuu
T1CON	-000 0000	-uuu uuuu	-uuu uuuu
TMR2	0000 0000	0000 0000	uuuu uuuu
T2CON	-000 0000	-000 0000	-uuu uuuu
SSPBUF	xxxx xxxx	uuuu uuuu	uuuu uuuu
SSPCON	0000 0000	0000 0000	uuuu uuuu
SSPCON2	0000 0000	0000 0000	uuuu uuuu
CCPR1L	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR1H	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP1CON	--00 0000	--00 0000	--uu uuuu
CCP2CON	--00 0000	--00 0000	--uu uuuu
CCP3CON	--00 0000	--00 0000	uuuu uuuu
CCPR2L	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR2H	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR3L	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR3H	xxxx xxxx	uuuu uuuu	uuuu uuuu
RCSTA	0000 000x	0000 000x	uuuu uuuu
TXREG	0000 0000	0000 0000	uuuu uuuu
RCREG	0000 0000	0000 0000	uuuu uuuu
ADRESH	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADCON0	0000 0000	0000 0000	uuuu uuuu
OPTION_REG	1111 1111	1111 1111	uuuu uuuu

Legend: u = unchanged, x = unknown, — = unimplemented bit, read as '0', q = value depends on condition.

Note 1: One or more bits in INTCON, PIR1 and PR2 will be affected (to cause wake-up).

2: When the wake-up is due to an interrupt and the GIE bit is set, the PC is loaded with the interrupt vector (0004h).

3: See Table 15-3 for Reset value for specific condition.

PIC16F7X7

15.15 Interrupts

The PIC16F7X7 has up to 17 sources of interrupt. The Interrupt Control register (INTCON) records individual interrupt requests in flag bits. It also has individual and global interrupt enable bits.

> **Note:** Individual interrupt flag bits are set regardless of the status of their corresponding mask bit or the GIE bit.

A Global Interrupt Enable bit, GIE (INTCON<7>), enables (if set) all unmasked interrupts or disables (if cleared) all interrupts. When bit GIE is enabled and an interrupt's flag bit and mask bit are set, the interrupt will vector immediately. Individual interrupts can be disabled through their corresponding enable bits in various registers. Individual interrupt bits are set regardless of the status of the GIE bit. The GIE bit is cleared on Reset.

The "return from interrupt" instruction, `RETFIE`, exits the interrupt routine as well as sets the GIE bit which re-enables interrupts.

The RB0/INT pin interrupt, the RB port change interrupt and the TMR0 overflow interrupt flags are contained in the INTCON register.

The peripheral interrupt flags are contained in the Special Function Register, PIR1. The corresponding interrupt enable bits are contained in Special Function Register, PIE1 and the peripheral interrupt enable bit is contained in Special Function Register, INTCON.

When an interrupt is serviced, the GIE bit is cleared to disable any further interrupt, the return address is pushed onto the stack and the PC is loaded with 0004h. Once in the Interrupt Service Routine, the source(s) of the interrupt can be determined by polling the interrupt flag bits. The interrupt flag bit(s) must be cleared in software before re-enabling interrupts to avoid recursive interrupts.

For external interrupt events, such as the INT pin or PORTB change interrupt, the interrupt latency will be three or four instruction cycles. The exact latency depends on when the interrupt event occurs relative to the current Q cycle. The latency is the same for one or two-cycle instructions. Individual interrupt flag bits are set regardless of the status of their corresponding mask bit, PEIE bit or the GIE bit.

FIGURE 15-11: INTERRUPT LOGIC

Note 1: PSP interrupt is implemented only on PIC16F747/777 devices.
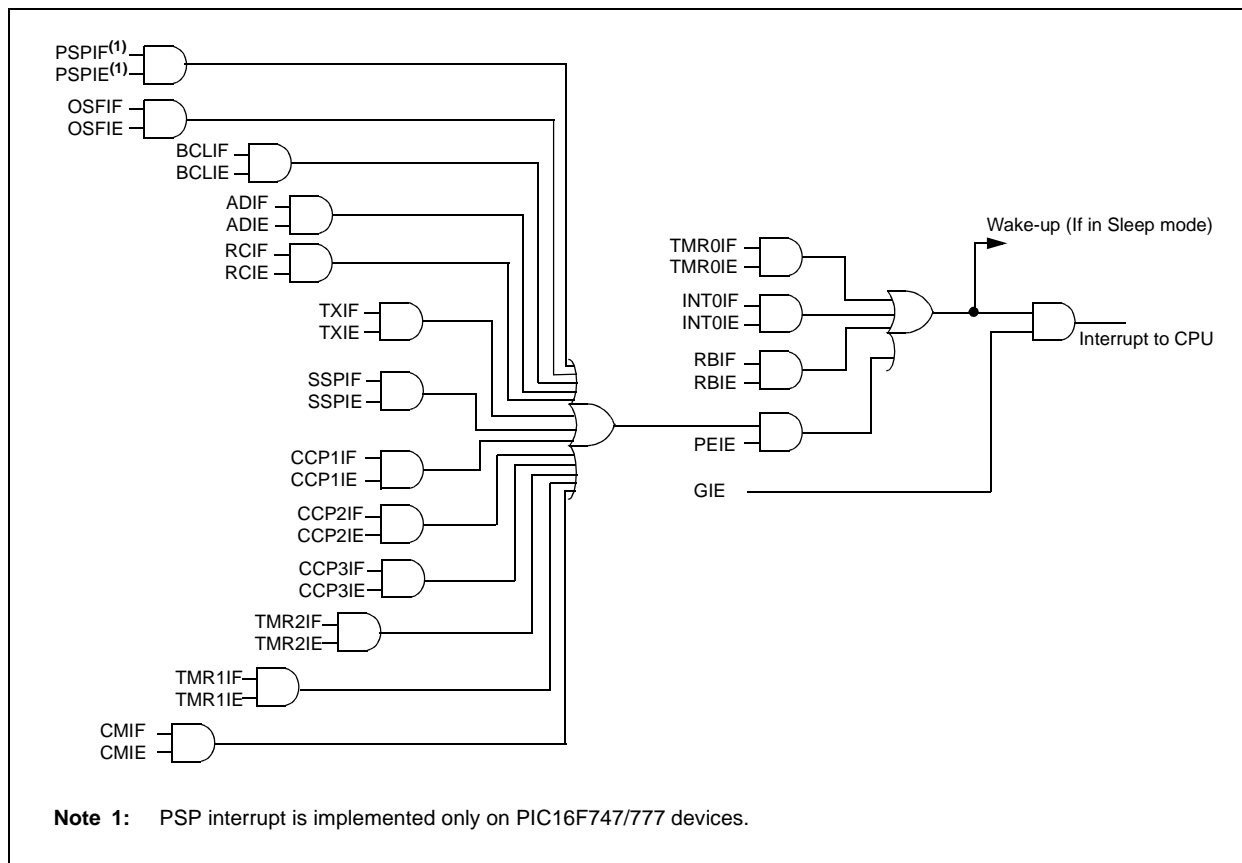
17.2 MPLAB C Compilers for Various Device Families

The MPLAB C Compiler code development systems are complete ANSI C compilers for Microchip's PIC18, PIC24 and PIC32 families of microcontrollers and the dsPIC30 and dsPIC33 families of digital signal controllers. These compilers provide powerful integration capabilities, superior code optimization and ease of use.

For easy source level debugging, the compilers provide symbol information that is optimized to the MPLAB IDE debugger.

17.3 HI-TECH C for Various Device Families

The HI-TECH C Compiler code development systems are complete ANSI C compilers for Microchip's PIC family of microcontrollers and the dsPIC family of digital signal controllers. These compilers provide powerful integration capabilities, omniscient code generation and ease of use.

For easy source level debugging, the compilers provide symbol information that is optimized to the MPLAB IDE debugger.

The compilers include a macro assembler, linker, preprocessor, and one-step driver, and can run on multiple platforms.

17.4 MPASM Assembler

The MPASM Assembler is a full-featured, universal macro assembler for PIC10/12/16/18 MCUs.

The MPASM Assembler generates relocatable object files for the MPLINK Object Linker, Intel® standard HEX files, MAP files to detail memory usage and symbol reference, absolute LST files that contain source lines and generated machine code and COFF files for debugging.

The MPASM Assembler features include:

* Integration into MPLAB IDE projects
* User-defined macros to streamline assembly code
* Conditional assembly for multi-purpose source files
* Directives that allow complete control over the assembly process

17.5 MPLINK Object Linker/ MPLIB Object Librarian

The MPLINK Object Linker combines relocatable objects created by the MPASM Assembler and the MPLAB C18 C Compiler. It can link relocatable objects from precompiled libraries, using directives from a linker script.

The MPLIB Object Librarian manages the creation and modification of library files of precompiled code. When a routine from a library is called from a source file, only the modules that contain that routine will be linked in with the application. This allows large libraries to be used efficiently in many different applications.

The object linker/library features include:

* Efficient linking of single libraries instead of many smaller files
* Enhanced code maintainability by grouping related modules together
* Flexible creation of libraries with easy module listing, replacement, deletion and extraction

17.6 MPLAB Assembler, Linker and Librarian for Various Device Families

MPLAB Assembler produces relocatable machine code from symbolic assembly language for PIC24, PIC32 and dsPIC devices. MPLAB C Compiler uses the assembler to produce its object file. The assembler generates relocatable object files that can then be archived or linked with other relocatable object files and archives to create an executable file. Notable features of the assembler include:

* Support for the entire device instruction set
* Support for fixed-point and floating-point data
* Command line interface
* Rich directive set
* Flexible macro language
* MPLAB IDE compatibility

PIC16F7X7

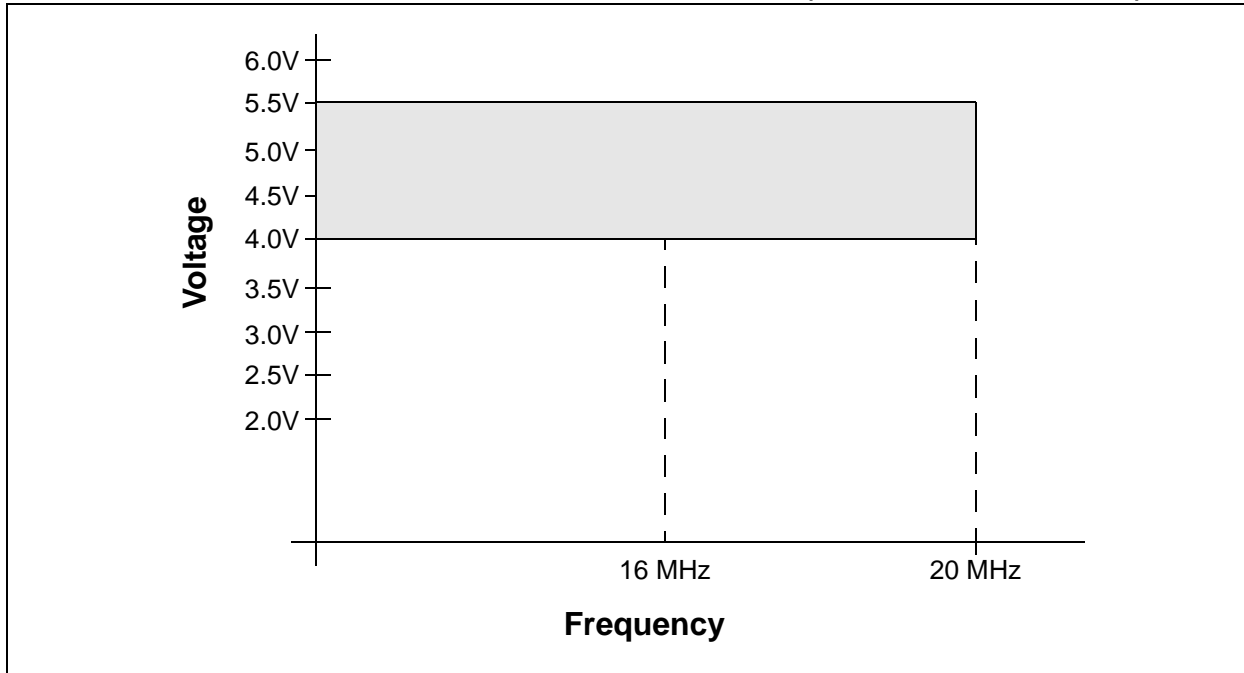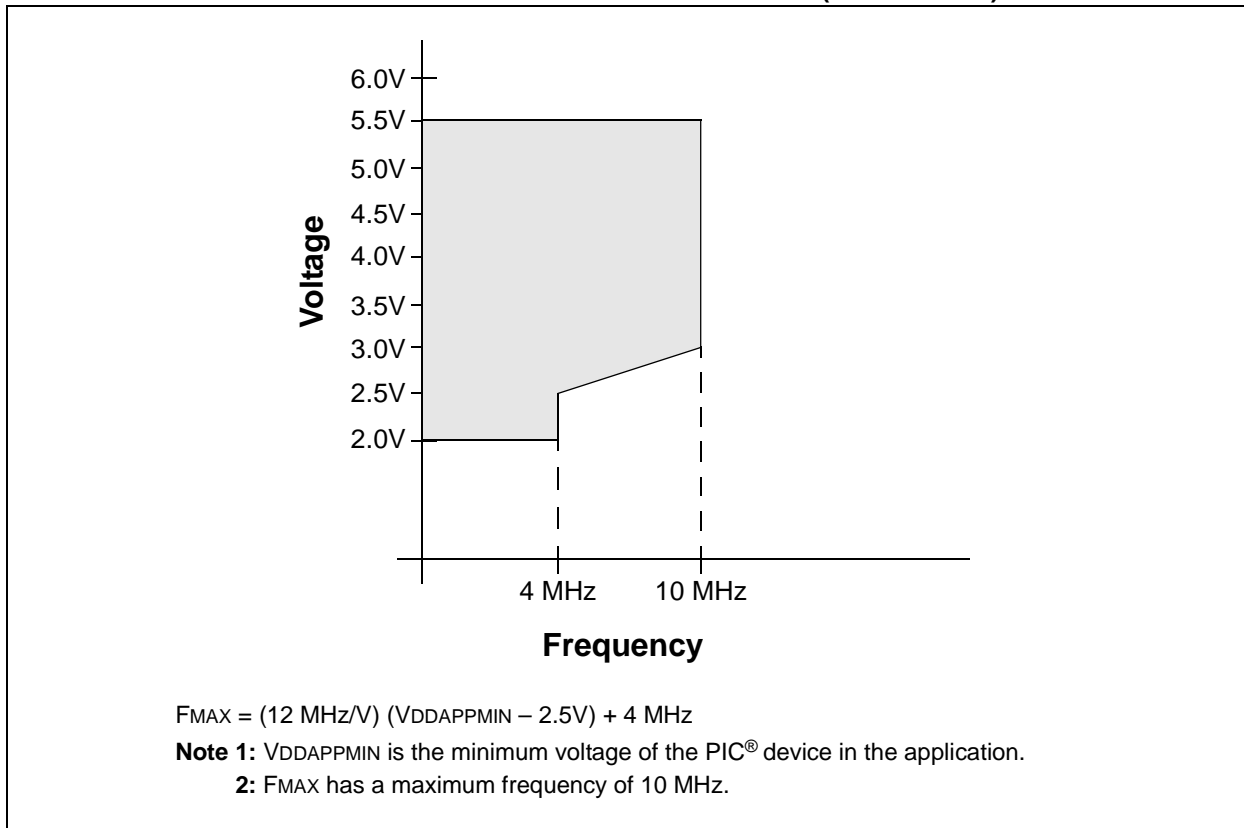**FIGURE 18-1:** **PIC16F7X7 VOLTAGE-FREQUENCY GRAPH (INDUSTRIAL, EXTENDED)**

FIGURE 18-2: **PIC16LF7X7 VOLTAGE-FREQUENCY GRAPH (INDUSTRIAL)**

$F_{MAX} = (12\ MHz/V)\ (V_{DDAPPMIN} - 2.5V) + 4\ MHz$

Note 1: $V_{DDAPPMIN}$ is the minimum voltage of the PIC® device in the application.

2: F_{MAX} has a maximum frequency of 10 MHz.

PIC16F7X7

FIGURE 18-20: A/D CONVERSION TIMING

Note: If the A/D clock source is selected as RC, a time of T_{CY} is added before the A/D clock starts. This allows the SLEEP instruction to be executed.
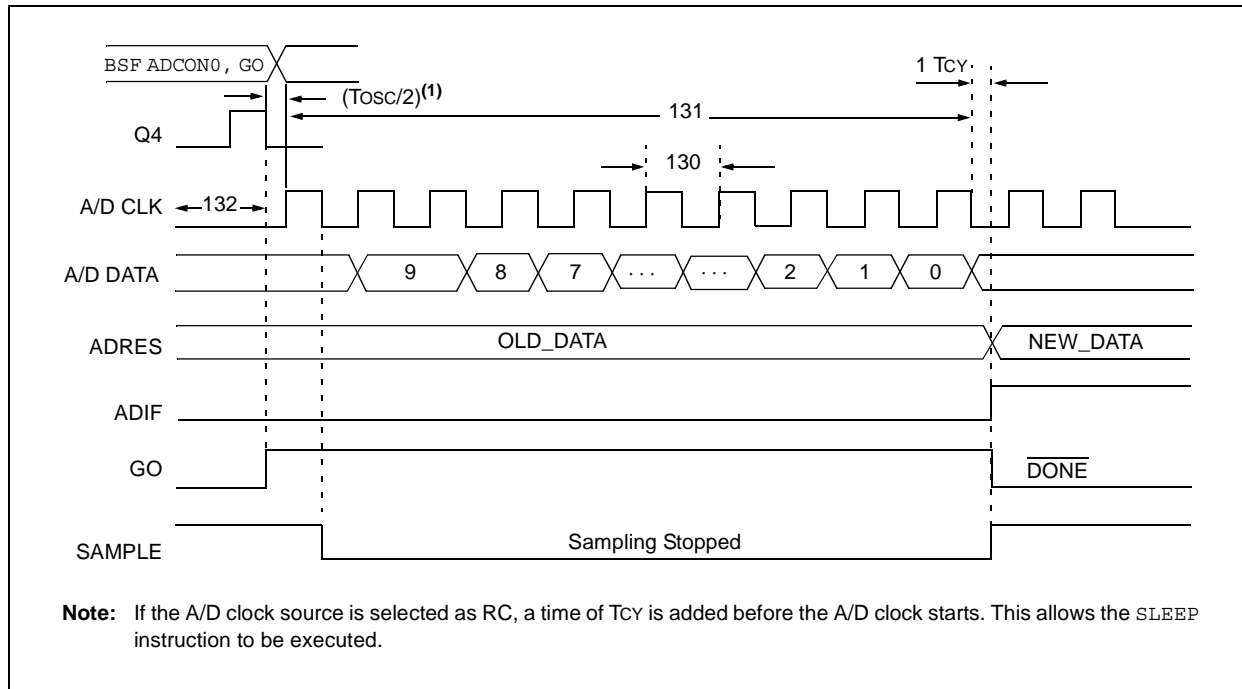
TABLE 18-16: A/D CONVERSION REQUIREMENTS

Param No.	Symbol	Characteristic		Min	Typ†	Max	Units	Conditions
130	T_{AD}	A/D Clock Period	PIC16F7X7	1.6	—	—	μs	T_{OSC} based, $V_{REF} \geq 3.0V$
			PIC16LF7X7	3.0	—	—	μs	T_{OSC} based, $V_{REF} \geq 2.0V$
			PIC16F7X7	2.0	4.0	6.0	μs	A/D RC mode
			PIC16LF7X7	3.0	6.0	9.0	μs	A/D RC mode
131	T_{CNV}	Conversion Time (not including S/H time) **(Note 1)**			—	12	T_{AD}	
132	T_{ACQ}	Acquisition Time		**(Note 2)**	40	—	μs	
				10*	—	—	μs	The minimum time is the amplifier settling time. This may be used if the "new" input voltage has not changed by more than 1 LSb (i.e., 5.0 mV @ 5.12V) from the last sampled voltage (as stated on C_{HOLD}).
134	T_{GO}	Q4 to A/D Clock Start		—	$T_{OSC}/2$ §	—	—	If the A/D clock source is selected as RC, a time of T_{CY} is added before the A/D clock starts. This allows the SLEEP instruction to be executed.

*   These parameters are characterized but not tested.

†   Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

§   This specification ensured by design.

Note 1: ADRES register may be read on the following T_{CY} cycle.

2: See **Section 12.1 "A/D Acquisition Requirements"** for minimum conditions.

© 2003-2013 Microchip Technology Inc.

PIC16F7X7

FIGURE 19-7: TYPICAL I_{DD} vs. V_{DD}, -40°C TO +125°C, 1 MHz TO 8 MHz (RC_RUN MODE, ALL PERIPHERALS DISABLED)
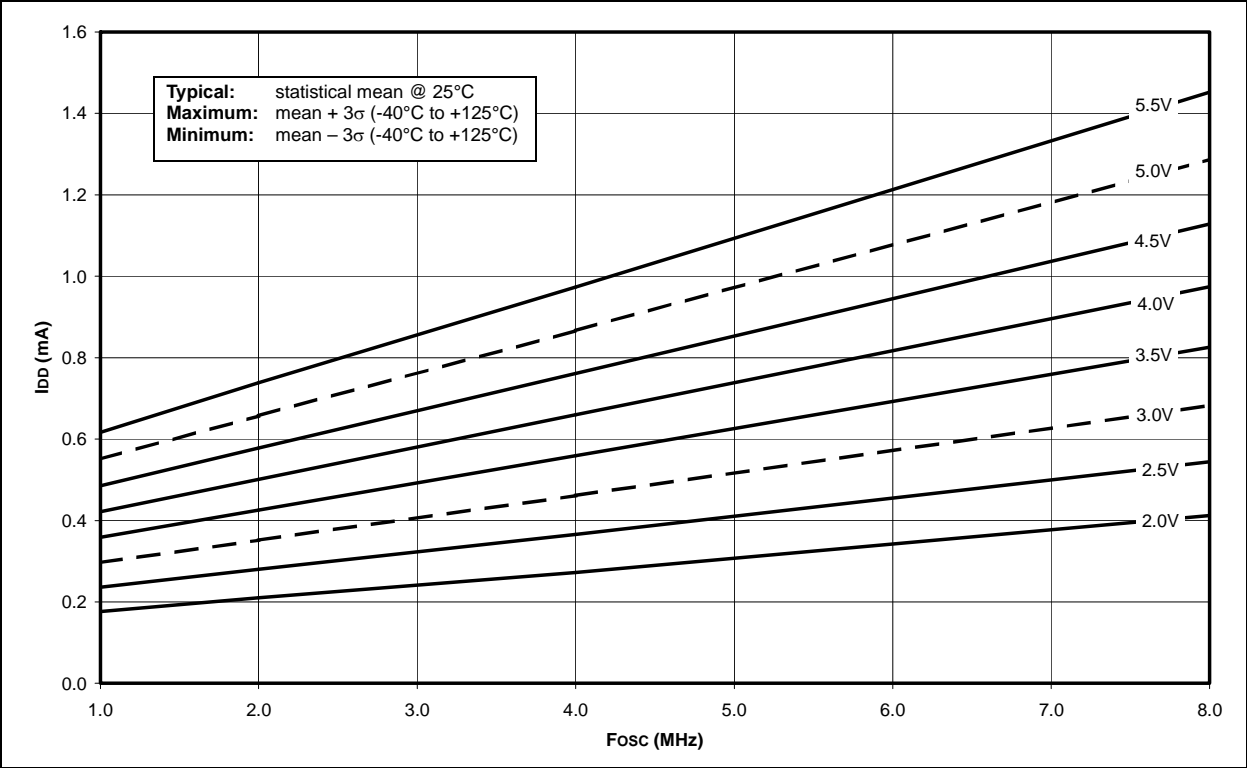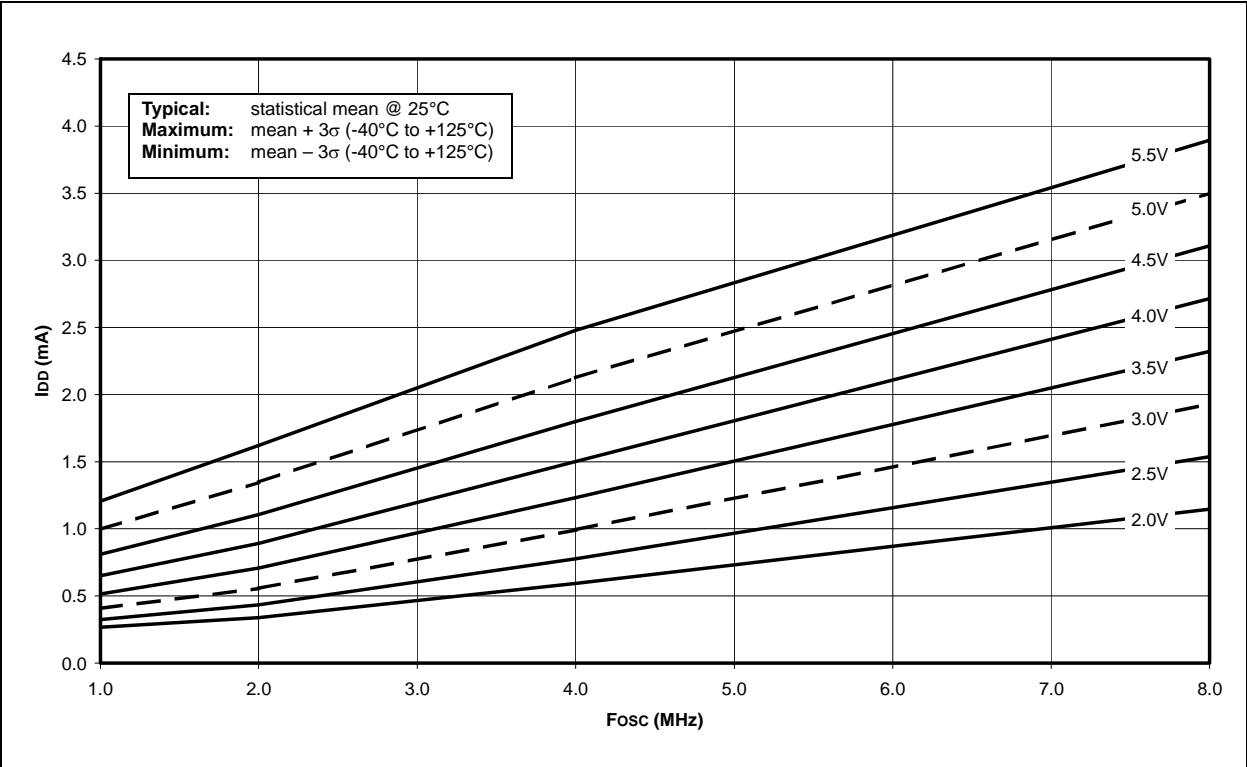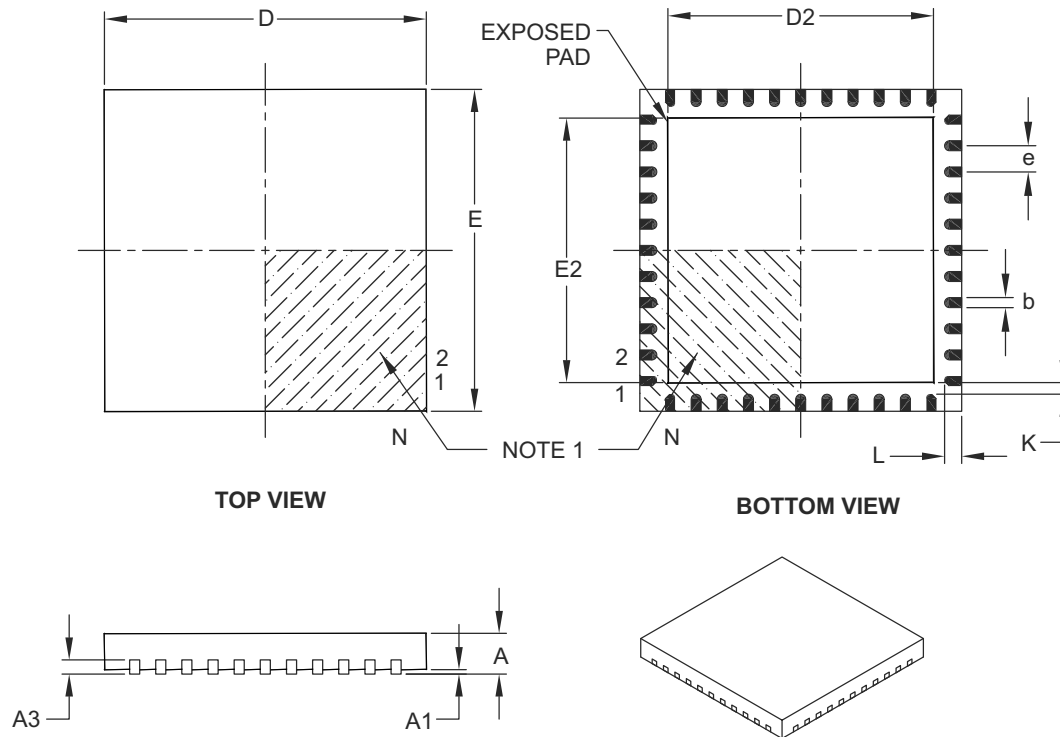

FIGURE 19-8: MAXIMUM I_{DD} vs. V_{DD}, -40°C TO +125°C, 1 MHz TO 8 MHz (RC_RUN MODE, ALL PERIPHERALS DISABLED)

PIC16F7X7

44-Lead Plastic Quad Flat, No Lead Package (ML) – 8x8 mm Body [QFN]

> **Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging

TOP VIEW

BOTTOM VIEW

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N		44	
Pitch	e		0.65 BSC	
Overall Height	A	0.80	0.90	1.00
Standoff	A1	0.00	0.02	0.05
Contact Thickness	A3		0.20 REF	
Overall Width	E		8.00 BSC	
Exposed Pad Width	E2	6.30	6.45	6.80
Overall Length	D		8.00 BSC	
Exposed Pad Length	D2	6.30	6.45	6.80
Contact Width	b	0.25	0.30	0.38
Contact Length	L	0.30	0.40	0.50
Contact-to-Exposed Pad	K	0.20	–	–

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Package is saw singulated.
3. Dimensioning and tolerancing per ASME Y14.5M.

    BSC: Basic Dimension. Theoretically exact value shown without tolerances.

    REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-103B