



Welcome to [E-XFL.COM](https://www.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	20MHz
Connectivity	I <sup>2</sup> C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	22
Program Memory Size	7KB (4K x 14)
Program Memory Type	FLASH
EEPROM Size	128 x 8
RAM Size	192 x 8
Voltage - Supply (Vcc/Vdd)	4V ~ 5.5V
Data Converters	A/D 5x10b
Oscillator Type	External
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	28-SOIC (0.295", 7.50mm Width)
Supplier Device Package	28-SOIC
Purchase URL	<a href="https://www.e-xfl.com/product-detail/microchip-technology/pic16f873t-20i-so">https://www.e-xfl.com/product-detail/microchip-technology/pic16f873t-20i-so</a>

# PIC16F87X

## 2.2.2.3 INTCON Register

The INTCON Register is a readable and writable register, which contains various enable and flag bits for the TMR0 register overflow, RB Port change and External RB0/INT pin interrupts.

**Note:** Interrupt flag bits are set when an interrupt condition occurs, regardless of the state of its corresponding enable bit or the global enable bit, GIE (INTCON<7>). User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt.

### REGISTER 2-3: INTCON REGISTER (ADDRESS 0Bh, 8Bh, 10Bh, 18Bh)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-x
GIE	PEIE	TOIE	INTE	RBIE	TOIF	INTF	RBIF

bit 7

bit 0

- bit 7     **GIE:** Global Interrupt Enable bit  
1 = Enables all unmasked interrupts  
0 = Disables all interrupts
- bit 6     **PEIE:** Peripheral Interrupt Enable bit  
1 = Enables all unmasked peripheral interrupts  
0 = Disables all peripheral interrupts
- bit 5     **TOIE:** TMR0 Overflow Interrupt Enable bit  
1 = Enables the TMR0 interrupt  
0 = Disables the TMR0 interrupt
- bit 4     **INTE:** RB0/INT External Interrupt Enable bit  
1 = Enables the RB0/INT external interrupt  
0 = Disables the RB0/INT external interrupt
- bit 3     **RBIE:** RB Port Change Interrupt Enable bit  
1 = Enables the RB port change interrupt  
0 = Disables the RB port change interrupt
- bit 2     **TOIF:** TMR0 Overflow Interrupt Flag bit  
1 = TMR0 register has overflowed (must be cleared in software)  
0 = TMR0 register did not overflow
- bit 1     **INTF:** RB0/INT External Interrupt Flag bit  
1 = The RB0/INT external interrupt occurred (must be cleared in software)  
0 = The RB0/INT external interrupt did not occur
- bit 0     **RBIF:** RB Port Change Interrupt Flag bit  
1 = At least one of the RB7:RB4 pins changed state; a mismatch condition will continue to set the bit. Reading PORTB will end the mismatch condition and allow the bit to be cleared (must be cleared in software).  
0 = None of the RB7:RB4 pins have changed state

#### 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

# PIC16F87X

## 2.2.2.5 PIR1 Register

The PIR1 register contains the individual flag bits for the peripheral interrupts.

**Note:** Interrupt flag bits are set when an interrupt condition occurs, regardless of the state of its corresponding enable bit or the global enable bit, GIE (INTCON<7>). User software should ensure the appropriate interrupt bits are clear prior to enabling an interrupt.

### REGISTER 2-5: PIR1 REGISTER (ADDRESS 0Ch)

R/W-0	R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
PSPIF <sup>(1)</sup>	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF
bit 7							bit 0

- bit 7 **PSPIF<sup>(1)</sup>**: Parallel Slave Port Read/Write Interrupt Flag bit  
 1 = A read or a write operation has taken place (must be cleared in software)  
 0 = No read or write has occurred
- bit 6 **ADIF**: A/D Converter Interrupt Flag bit  
 1 = An A/D conversion completed  
 0 = The A/D conversion is not complete
- bit 5 **RCIF**: USART Receive Interrupt Flag bit  
 1 = The USART receive buffer is full  
 0 = The USART receive buffer is empty
- bit 4 **TXIF**: USART Transmit Interrupt Flag bit  
 1 = The USART transmit buffer is empty  
 0 = The USART transmit buffer is full
- bit 3 **SSPIF**: Synchronous Serial Port (SSP) Interrupt Flag  
 1 = The SSP interrupt condition has occurred, and must be cleared in software before returning from the Interrupt Service Routine. The conditions that will set this bit are:
- SPI
    - A transmission/reception has taken place.
  - I<sup>2</sup>C Slave
    - A transmission/reception has taken place.
  - I<sup>2</sup>C Master
    - A transmission/reception has taken place.
    - The initiated START condition was completed by the SSP module.
    - The initiated STOP condition was completed by the SSP module.
    - The initiated Restart condition was completed by the SSP module.
    - The initiated Acknowledge condition was completed by the SSP module.
    - A START condition occurred while the SSP module was idle (Multi-Master system).
    - A STOP condition occurred while the SSP module was idle (Multi-Master system).
- 0 = No SSP interrupt condition has occurred.
- bit 2 **CCP1IF**: CCP1 Interrupt Flag bit  
Capture mode:  
 1 = A TMR1 register capture occurred (must be cleared in software)  
 0 = No TMR1 register capture occurred  
Compare mode:  
 1 = A TMR1 register compare match occurred (must be cleared in software)  
 0 = No TMR1 register compare match occurred  
PWM mode:  
 Unused in this mode
- bit 1 **TMR2IF**: TMR2 to PR2 Match Interrupt Flag bit  
 1 = TMR2 to PR2 match occurred (must be cleared in software)  
 0 = No TMR2 to PR2 match occurred
- bit 0 **TMR1IF**: TMR1 Overflow Interrupt Flag bit  
 1 = TMR1 register overflowed (must be cleared in software)  
 0 = TMR1 register did not overflow

**Note 1:** PSPIF is reserved on PIC16F873/876 devices; always maintain this bit clear.

#### 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

## 2.5 Indirect Addressing, INDF and FSR Registers

The INDF register is not a physical register. Addressing the INDF register will cause indirect addressing.

Indirect addressing is possible by using the INDF register. Any instruction using the INDF register actually accesses the register pointed to by the File Select Register, FSR. Reading the INDF register itself, indirectly (FSR = '0') will read 00h. Writing to the INDF register indirectly results in a no operation (although status bits may be affected). An effective 9-bit address is obtained by concatenating the 8-bit FSR register and the IRP bit (STATUS<7>), as shown in Figure 2-6.

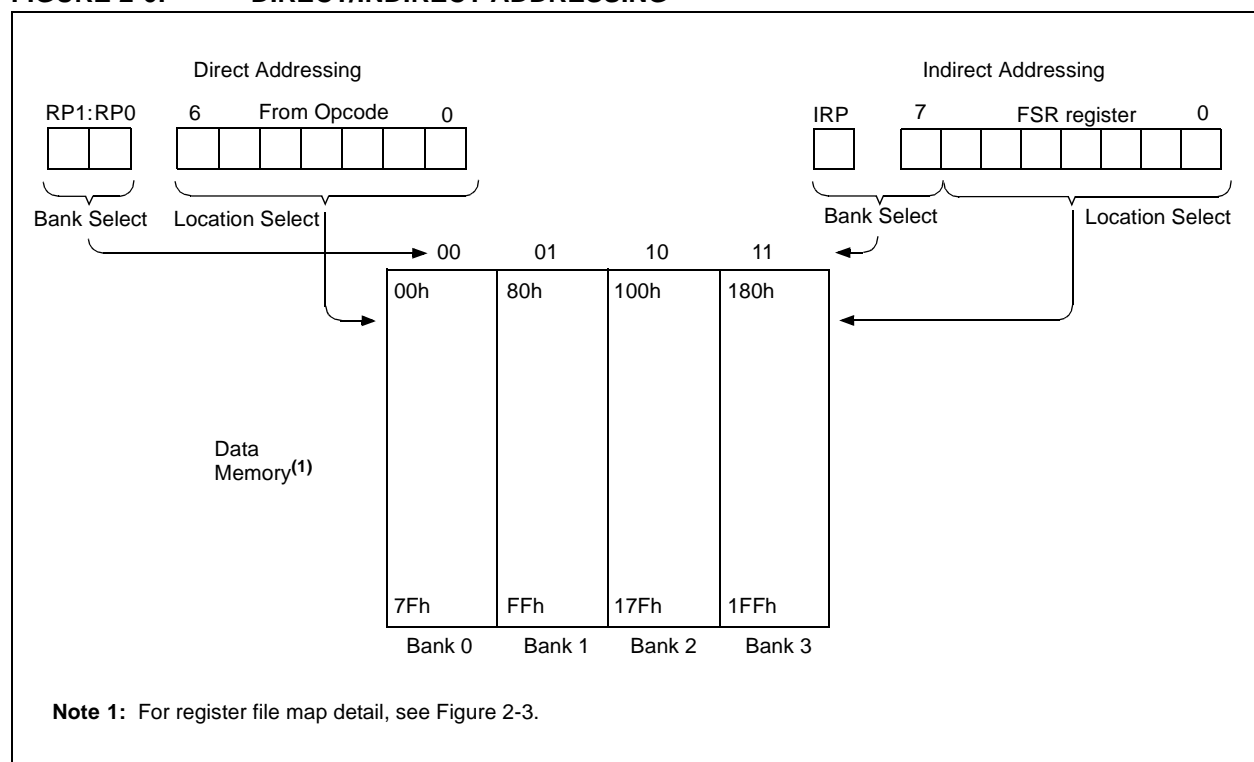
A simple program to clear RAM locations 20h-2Fh using indirect addressing is shown in Example 2-2.

### EXAMPLE 2-2: INDIRECT ADDRESSING

```

        MOVLW 0x20    ;initialize pointer
        MOVWF FSR     ;to RAM
NEXT    CLRF INDF     ;clear INDF register
        INCF FSR,F    ;inc pointer
        BTFSS FSR,4   ;all done?
        GOTO NEXT     ;no clear next
CONTINUE
        :             ;yes continue
    
```

**FIGURE 2-6: DIRECT/INDIRECT ADDRESSING**



# PIC16F87X

**TABLE 3-3: PORTB FUNCTIONS**

Name	Bit#	Buffer	Function
RB0/INT	bit0	TTL/ST <sup>(1)</sup>	Input/output pin or external interrupt input. Internal software programmable weak pull-up.
RB1	bit1	TTL	Input/output pin. Internal software programmable weak pull-up.
RB2	bit2	TTL	Input/output pin. Internal software programmable weak pull-up.
RB3/PGM <sup>(3)</sup>	bit3	TTL	Input/output pin or programming pin in LVP mode. Internal software programmable weak pull-up.
RB4	bit4	TTL	Input/output pin (with interrupt-on-change). Internal software programmable weak pull-up.
RB5	bit5	TTL	Input/output pin (with interrupt-on-change). Internal software programmable weak pull-up.
RB6/PGC	bit6	TTL/ST <sup>(2)</sup>	Input/output pin (with interrupt-on-change) or In-Circuit Debugger pin. Internal software programmable weak pull-up. Serial programming clock.
RB7/PGD	bit7	TTL/ST <sup>(2)</sup>	Input/output pin (with interrupt-on-change) or In-Circuit Debugger pin. Internal software programmable weak pull-up. Serial programming data.

Legend: TTL = TTL input, ST = Schmitt Trigger input

**Note 1:** This buffer is a Schmitt Trigger input when configured as the external interrupt.

**2:** This buffer is a Schmitt Trigger input when used in Serial Programming mode.

**3:** Low Voltage ICSP Programming (LVP) is enabled by default, which disables the RB3 I/O function. LVP must be disabled to enable RB3 as an I/O pin and allow maximum compatibility to the other 28-pin and 40-pin mid-range devices.

**TABLE 3-4: SUMMARY OF REGISTERS ASSOCIATED WITH PORTB**

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
06h, 106h	PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	xxxx xxxx	uuuu uuuu
86h, 186h	TRISB	PORTB Data Direction Register								1111 1111	1111 1111
81h, 181h	OPTION_REG	RBPu	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	1111 1111

Legend: x = unknown, u = unchanged. Shaded cells are not used by PORTB.

---

# PIC16F87X

## REGISTER 8-1: CCP1CON REGISTER/CCP2CON REGISTER (ADDRESS: 17h/1Dh)

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 CCPxL.

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 resets TMR1; CCP2 resets TMR1 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

## 8.3 PWM Mode (PWM)

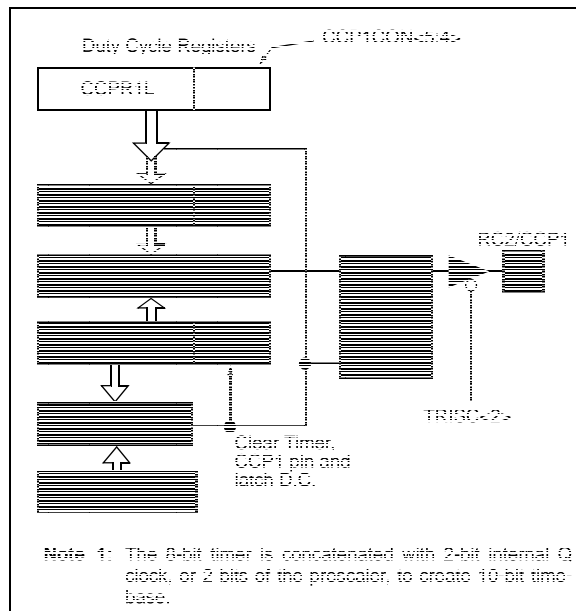
In Pulse Width Modulation mode, the CCPx pin produces up to a 10-bit resolution PWM output. Since the CCP1 pin is multiplexed with the PORTC data latch, the TRISC<2> bit must be cleared to make the CCP1 pin an output.

**Note:** Clearing the CCP1CON register will force the CCP1 PWM output latch to the default low level. This is not the PORTC I/O data latch.

Figure 8-3 shows a simplified block diagram of the CCP module in PWM mode.

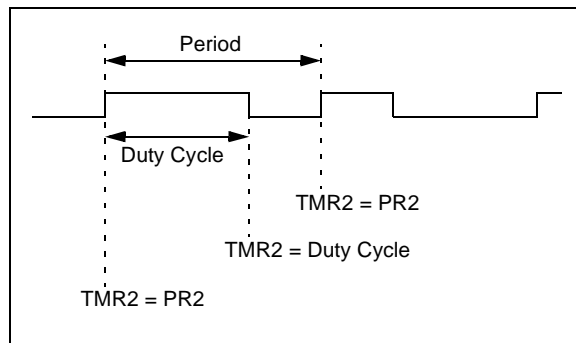
For a step-by-step procedure on how to set up the CCP module for PWM operation, see Section 8.3.3.

**FIGURE 8-3: SIMPLIFIED PWM BLOCK DIAGRAM**



A PWM output (Figure 8-4) has a time-base (period) and a time that the output stays high (duty cycle). The frequency of the PWM is the inverse of the period (1/period).

**FIGURE 8-4: PWM OUTPUT**



### 8.3.1 PWM PERIOD

The PWM period is specified by writing to the PR2 register. The PWM period can be calculated using the following formula:

$$\text{PWM period} = [(PR2) + 1] \cdot 4 \cdot T_{osc} \cdot (\text{TMR2 prescale value})$$

PWM frequency is defined as  $1 / [\text{PWM period}]$ .

When TMR2 is equal to PR2, the following three events occur on the next increment cycle:

- TMR2 is cleared
- The CCP1 pin is set (exception: if PWM duty cycle = 0%, the CCP1 pin will not be set)
- The PWM duty cycle is latched from CCPR1L into CCPR1H

**Note:** The Timer2 postscaler (see Section 7.1) is not used in the determination of the PWM frequency. The postscaler could be used to have a servo update rate at a different frequency than the PWM output.

### 8.3.2 PWM DUTY CYCLE

The PWM duty cycle is specified by writing to the CCPR1L register and to the CCP1CON<5:4> bits. Up to 10-bit resolution is available. The CCPR1L contains the eight MSBs and the CCP1CON<5:4> contains the two LSBs. This 10-bit value is represented by CCPR1L:CCP1CON<5:4>. The following equation is used to calculate the PWM duty cycle in time:

$$\text{PWM duty cycle} = (\text{CCPR1L:CCP1CON<5:4>}) \cdot T_{osc} \cdot (\text{TMR2 prescale value})$$

CCPR1L and CCP1CON<5:4> can be written to at any time, but the duty cycle value is not latched into CCPR1H until after a match between PR2 and TMR2 occurs (i.e., the period is complete). In PWM mode, CCPR1H is a read-only register.

The CCPR1H register and a 2-bit internal latch are used to double buffer the PWM duty cycle. This double buffering is essential for glitch-free PWM operation.

When the CCPR1H and 2-bit latch match TMR2, concatenated with an internal 2-bit Q clock, or 2 bits of the TMR2 prescaler, the CCP1 pin is cleared.

The maximum PWM resolution (bits) for a given PWM frequency is given by the formula:

$$\text{Resolution} = \frac{\log\left(\frac{F_{osc}}{F_{PWM}}\right)}{\log(2)} \text{ bits}$$

**Note:** If the PWM duty cycle value is longer than the PWM period, the CCP1 pin will not be cleared.





# PIC16F87X

**TABLE 9-1: REGISTERS ASSOCIATED WITH SPI OPERATION**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on: MCLR, WDT
0Bh, 8Bh, 10Bh, 18Bh	INTCON	GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	0000 000u
0Ch	PIR1	PSPIF <sup>(1)</sup>	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
8Ch	PIE1	PSPIE <sup>(1)</sup>	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
13h	SSPBUF	Synchronous Serial Port Receive Buffer/Transmit Register								xxxx xxxx	uuuu uuuu
14h	SSPCON	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	0000 0000
94h	SSPSTAT	SMP	CKE	D/Ā	P	S	R/Ā	UA	BF	0000 0000	0000 0000

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are not used by the SSP in SPI mode.

**Note 1:** These bits are reserved on PIC16F873/876 devices; always maintain these bits clear.

**TABLE 9-2: DATA TRANSFER RECEIVED BYTE ACTIONS**

Status Bits as Data Transfer is Received		SSPSR → SSPBUF	Generate $\overline{\text{ACK}}$ Pulse	Set bit SSPIF (SSP Interrupt occurs if enabled)
BF	SSPOV			
0	0	Yes	Yes	Yes
1	0	No	No	Yes
1	1	No	No	Yes
0	1	Yes	No	Yes

**Note:** Shaded cells show the conditions where the user software did not properly clear the overflow condition.

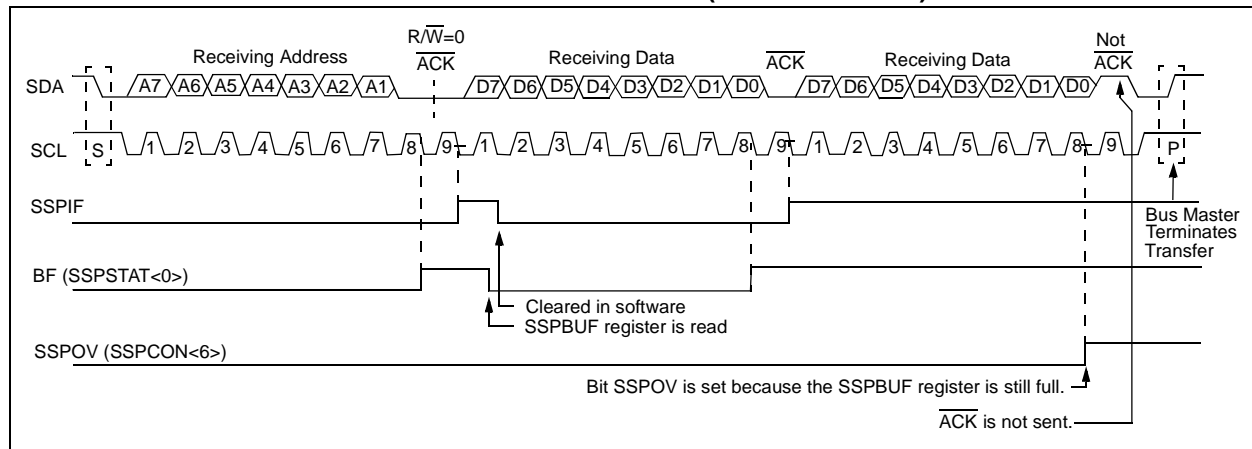
## 9.2.1.3 Slave Transmission

When the R/W bit of the incoming address byte is set and an address match occurs, the R/W bit of the SSPSTAT register is set. The received address is loaded into the SSPBUF register. The  $\overline{\text{ACK}}$  pulse will be sent on the ninth bit, and the SCL pin is held low. The transmit data must be loaded into the SSPBUF register, which also loads the SSPSR register. Then, the SCL pin should be enabled by setting bit CKP (SSPCON<4>). The master must monitor the SCL pin prior to asserting another clock pulse. The slave devices may be holding off the master by stretching the clock. The eight data bits are shifted out on the falling edge of the SCL input. This ensures that the SDA signal is valid during the SCL high time (Figure 9-7).

An SSP interrupt is generated for each data transfer byte. The SSPIF flag bit must be cleared in software and the SSPSTAT register is used to determine the status of the byte transfer. The SSPIF flag bit is set on the falling edge of the ninth clock pulse.

As a slave-transmitter, the  $\overline{\text{ACK}}$  pulse from the master receiver is latched on the rising edge of the ninth SCL input pulse. If the SDA line is high (not  $\overline{\text{ACK}}$ ), then the data transfer is complete. When the not  $\overline{\text{ACK}}$  is latched by the slave, the slave logic is reset and the slave then monitors for another occurrence of the START bit. If the SDA line was low ( $\overline{\text{ACK}}$ ), the transmit data must be loaded into the SSPBUF register, which also loads the SSPSR register. Then the SCL pin should be enabled by setting the CKP bit.

**FIGURE 9-6: I<sup>2</sup>C WAVEFORMS FOR RECEPTION (7-BIT ADDRESS)**



# PIC16F87X

## 9.2.18.1 Bus Collision During a START Condition

During a START condition, a bus collision occurs if:

- SDA or SCL are sampled low at the beginning of the START condition (Figure 9-20).
- SCL is sampled low before SDA is asserted low (Figure 9-21).

During a START condition, both the SDA and the SCL pins are monitored. If either the SDA pin or the SCL pin is already low, then these events all occur:

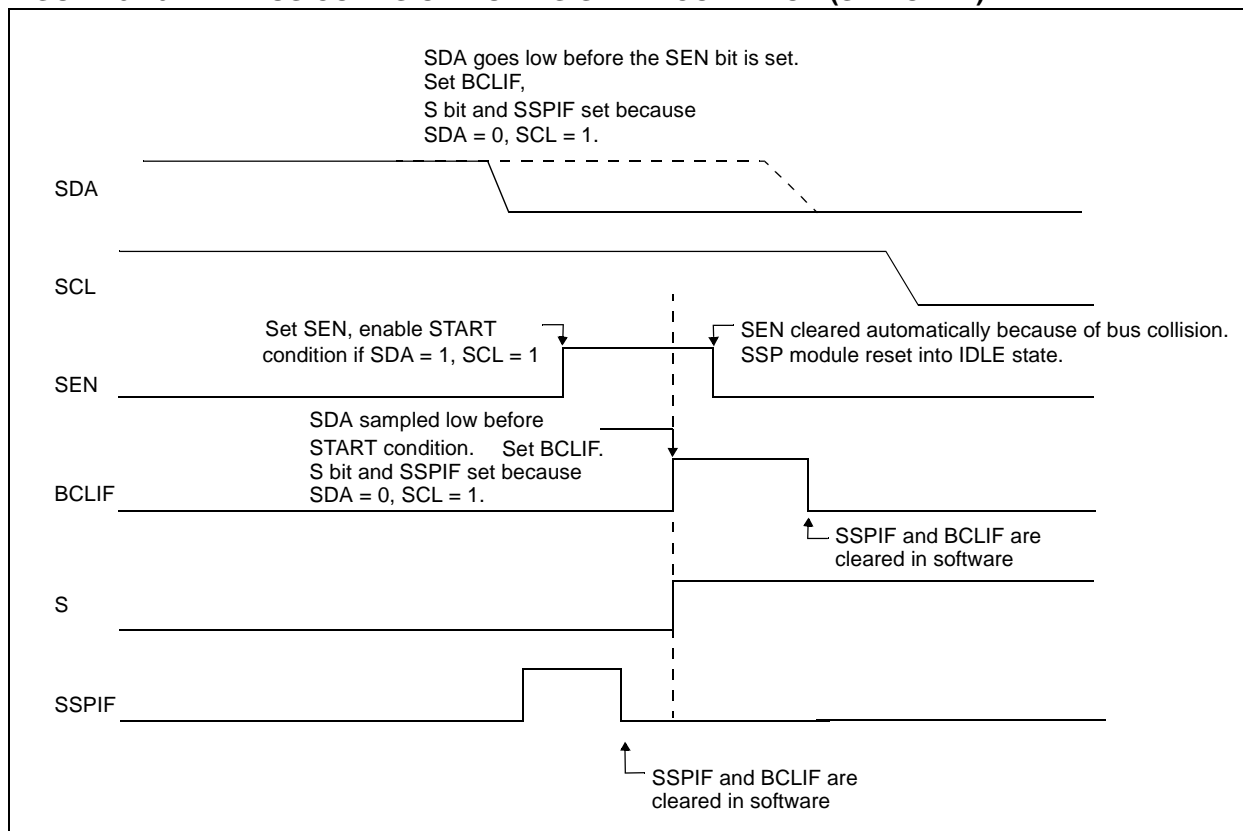
- the START condition is aborted,
- and the BCLIF flag is set,
- and the SSP module is reset to its IDLE state (Figure 9-20).

The START condition begins with the SDA and SCL pins de-asserted. When the SDA pin is sampled high, the baud rate generator is loaded from SSPADD<6:0> and counts down to 0. If the SCL pin is sampled low while SDA is high, a bus collision occurs, because it is assumed that another master is attempting to drive a data '1' during the START condition.

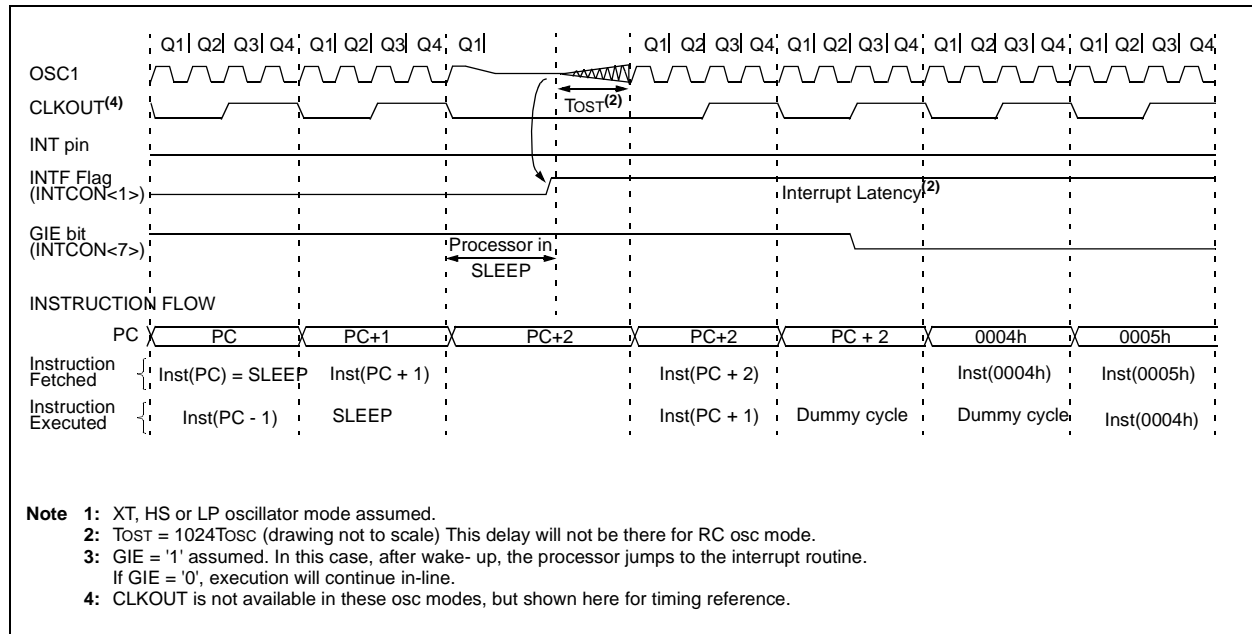
If the SDA pin is sampled low during this count, the BRG is reset and the SDA line is asserted early (Figure 9-22). If, however, a '1' is sampled on the SDA pin, the SDA pin is asserted low at the end of the BRG count. The baud rate generator is then reloaded and counts down to 0. During this time, if the SCL pins are sampled as '0', a bus collision does not occur. At the end of the BRG count, the SCL pin is asserted low.

**Note:** The reason that bus collision is not a factor during a START condition is that no two bus masters can assert a START condition at the exact same time. Therefore, one master will always assert SDA before the other. This condition does not cause a bus collision, because the two masters must be allowed to arbitrate the first address following the START condition. If the address is the same, arbitration must be allowed to continue into the data portion, Repeated START, or STOP conditions.

**FIGURE 9-20: BUS COLLISION DURING START CONDITION (SDA ONLY)**



**FIGURE 12-11: WAKE-UP FROM SLEEP THROUGH INTERRUPT**



## 12.14 In-Circuit Debugger

When the DEBUG bit in the configuration word is programmed to a '0', the In-Circuit Debugger functionality is enabled. This function allows simple debugging functions when used with MPLAB® ICD. When the microcontroller has this feature enabled, some of the resources are not available for general use. Table 12-8 shows which features are consumed by the background debugger.

**TABLE 12-8: DEBUGGER RESOURCES**

I/O pins	RB6, RB7
Stack	1 level
Program Memory	Address 0000h must be NOP Last 100h words
Data Memory	0x070 (0x0F0, 0x170, 0x1F0) 0x1EB - 0x1EF

To use the In-Circuit Debugger function of the microcontroller, the design must implement In-Circuit Serial Programming connections to MCLR/VPP, VDD, GND, RB7 and RB6. This will interface to the In-Circuit Debugger module available from Microchip, or one of the third party development tool companies.

## 12.15 Program Verification/Code Protection

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

## 12.16 ID Locations

Four memory locations (2000h - 2003h) are designated as ID locations, where the user can store checksum or other code identification numbers. These locations are not accessible during normal execution, but are readable and writable during program/verify. It is recommended that only the 4 Least Significant bits of the ID location are used.

## RLF Rotate Left f through Carry

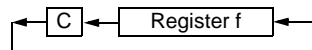
**Syntax:** [ *label* ] RLF f,d

**Operands:**  $0 \leq f \leq 127$   
 $d \in [0,1]$

**Operation:** See description below

**Status Affected:** C

**Description:** The contents of register 'f' are rotated one bit to the left through the Carry Flag. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is stored back in register 'f'.



## SLEEP

**Syntax:** [ *label* ] SLEEP

**Operands:** None

**Operation:** 00h → WDT,  
 0 → WDT prescaler,  
 1 →  $\overline{TO}$ ,  
 0 →  $\overline{PD}$

**Status Affected:**  $\overline{TO}$ ,  $\overline{PD}$

**Description:** The power-down status bit,  $\overline{PD}$  is cleared. Time-out status bit,  $\overline{TO}$  is set. Watchdog Timer and its prescaler are cleared. The processor is put into SLEEP mode with the oscillator stopped.

## RETURN Return from Subroutine

**Syntax:** [ *label* ] RETURN

**Operands:** None

**Operation:** TOS → PC

**Status Affected:** None

**Description:** Return from subroutine. The stack is POPed and the top of the stack (TOS) is loaded into the program counter. This is a two-cycle instruction.

## SUBLW Subtract W from Literal

**Syntax:** [ *label* ] SUBLW k

**Operands:**  $0 \leq k \leq 255$

**Operation:**  $k - (W) \rightarrow (W)$

**Status Affected:** C, DC, Z

**Description:** The W register is subtracted (2's complement method) from the eight-bit literal 'k'. The result is placed in the W register.

## RRF Rotate Right f through Carry

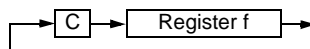
**Syntax:** [ *label* ] RRF f,d

**Operands:**  $0 \leq f \leq 127$   
 $d \in [0,1]$

**Operation:** See description below

**Status Affected:** C

**Description:** The contents of register 'f' are rotated one bit to the right through the Carry Flag. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is placed back in register 'f'.



## SUBWF Subtract W from f

**Syntax:** [ *label* ] SUBWF f,d

**Operands:**  $0 \leq f \leq 127$   
 $d \in [0,1]$

**Operation:**  $(f) - (W) \rightarrow (\text{destination})$

**Status Affected:** C, DC, Z

**Description:** Subtract (2's complement method) W register from register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'.

# PIC16F87X

---

## **SWAPF**      **Swap Nibbles in f**

---

Syntax:      `[label] SWAPF f,d`

Operands:       $0 \leq f \leq 127$   
                     $d \in [0,1]$

Operation:       $(f<3:0>) \rightarrow (\text{destination}<7:4>)$ ,  
                     $(f<7:4>) \rightarrow (\text{destination}<3:0>)$

Status Affected:      None

Description:      The upper and lower nibbles of register 'f' are exchanged. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is placed in register 'f'.

## **XORWF**      **Exclusive OR W with f**

---

Syntax:      `[label] XORWF f,d`

Operands:       $0 \leq f \leq 127$   
                     $d \in [0,1]$

Operation:       $(W) .XOR. (f) \rightarrow (\text{destination})$

Status Affected:      Z

Description:      Exclusive OR the contents of the W register with register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'.

## **XORLW**      **Exclusive OR Literal with W**

---

Syntax:      `[label] XORLW k`

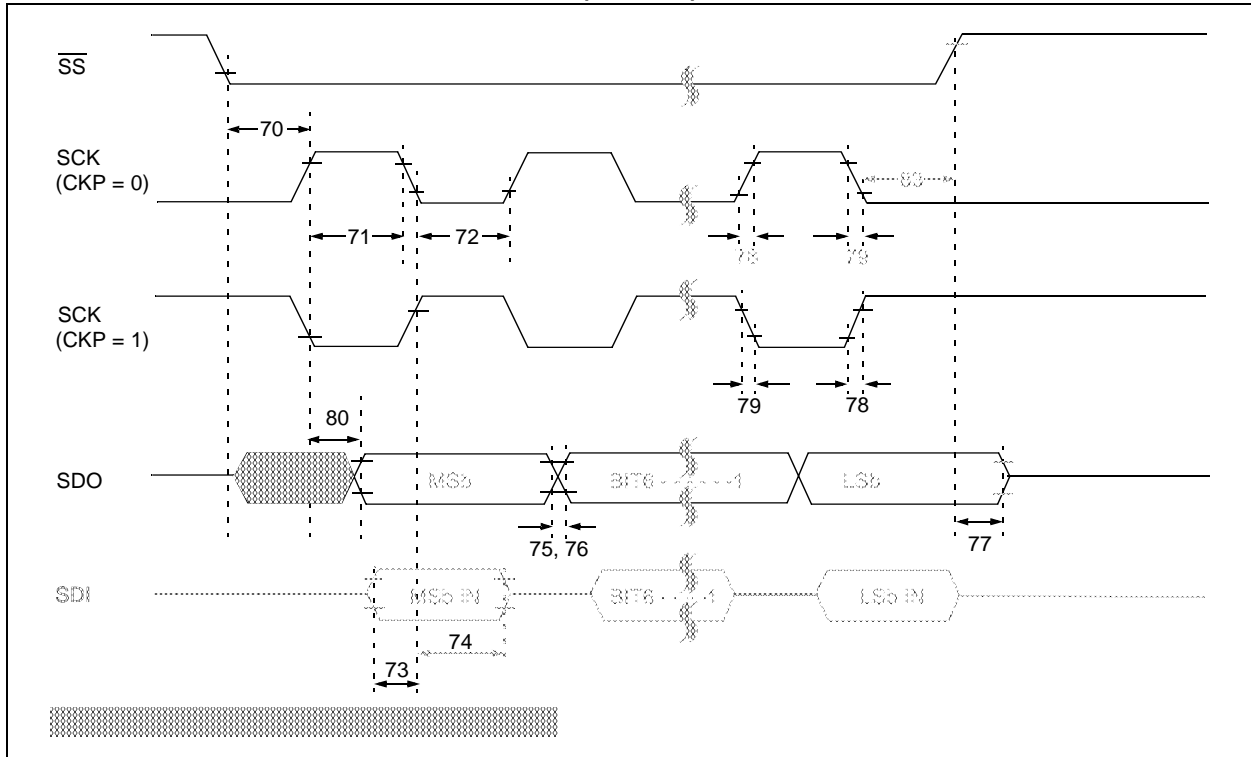
Operands:       $0 \leq k \leq 255$

Operation:       $(W) .XOR. k \rightarrow (W)$

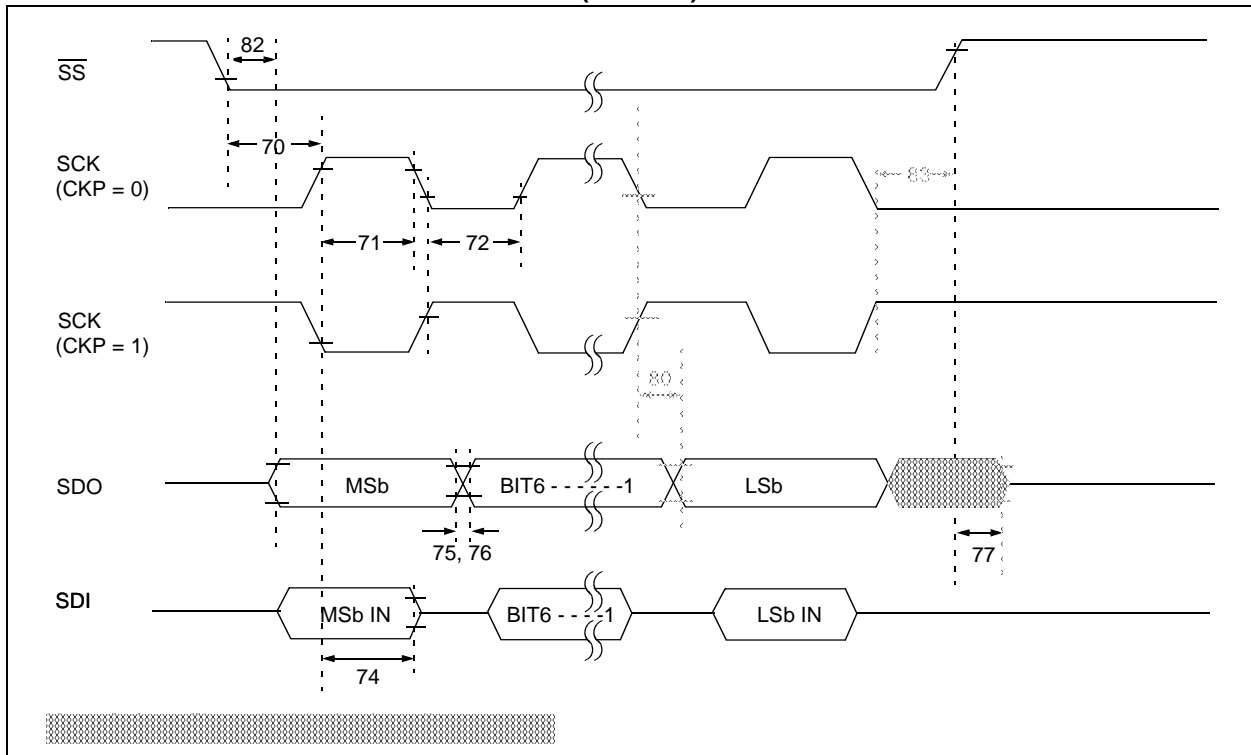
Status Affected:      Z

Description:      The contents of the W register are XOR'ed with the eight-bit literal 'k'. The result is placed in the W register.

**FIGURE 15-15: SPI SLAVE MODE TIMING (CKE = 0)**

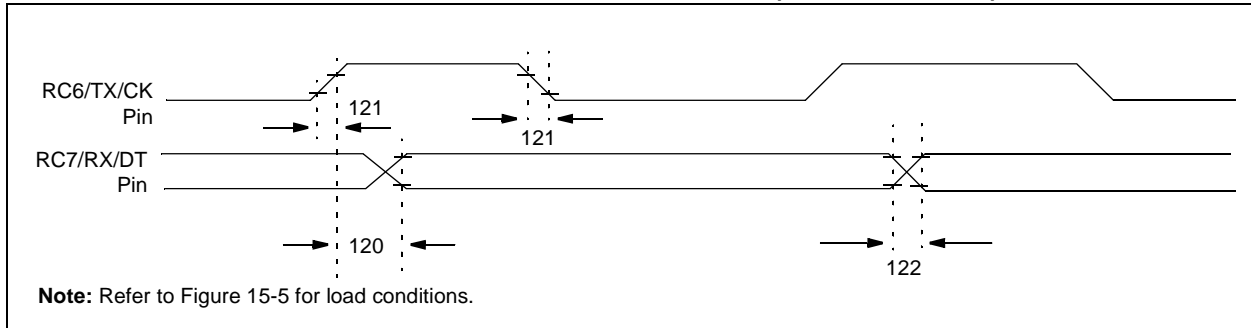


**FIGURE 15-16: SPI SLAVE MODE TIMING (CKE = 1)**





**FIGURE 15-19: USART SYNCHRONOUS TRANSMISSION (MASTER/SLAVE) TIMING**

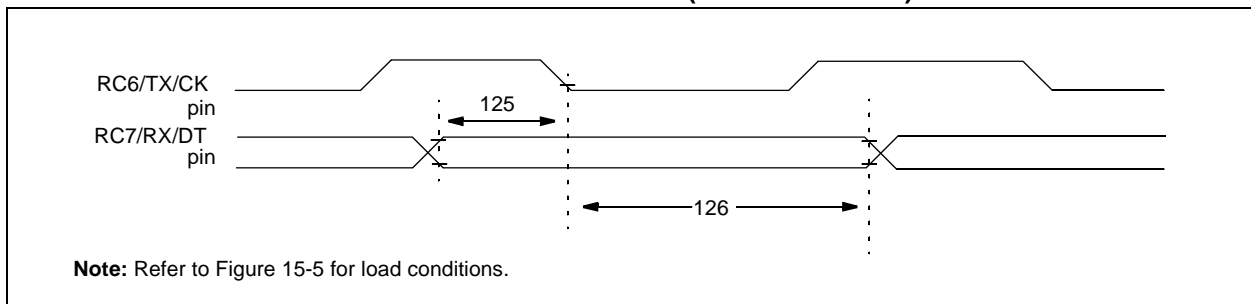


**TABLE 15-10: USART SYNCHRONOUS TRANSMISSION REQUIREMENTS**

Param No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
120	TckH2dtV	<u>SYNC XMIT (MASTER &amp; SLAVE)</u> Clock high to data out valid	—	—	80	ns	
			—	—	100	ns	
121	Tckrf	Clock out rise time and fall time (Master mode)	—	—	45	ns	
			—	—	50	ns	
122	TdtV	Data out rise time and fall time	—	—	45	ns	
			—	—	50	ns	

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

**FIGURE 15-20: USART SYNCHRONOUS RECEIVE (MASTER/SLAVE) TIMING**



**TABLE 15-11: USART SYNCHRONOUS RECEIVE REQUIREMENTS**

Parameter No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
125	TdtV2ckL	<u>SYNC RCV (MASTER &amp; SLAVE)</u> Data setup before CK ↓ (DT setup time)	15	—	—	ns	
126	TckL2dtL	Data hold after CK ↓ (DT hold time)	15	—	—	ns	

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

# PIC16F87X

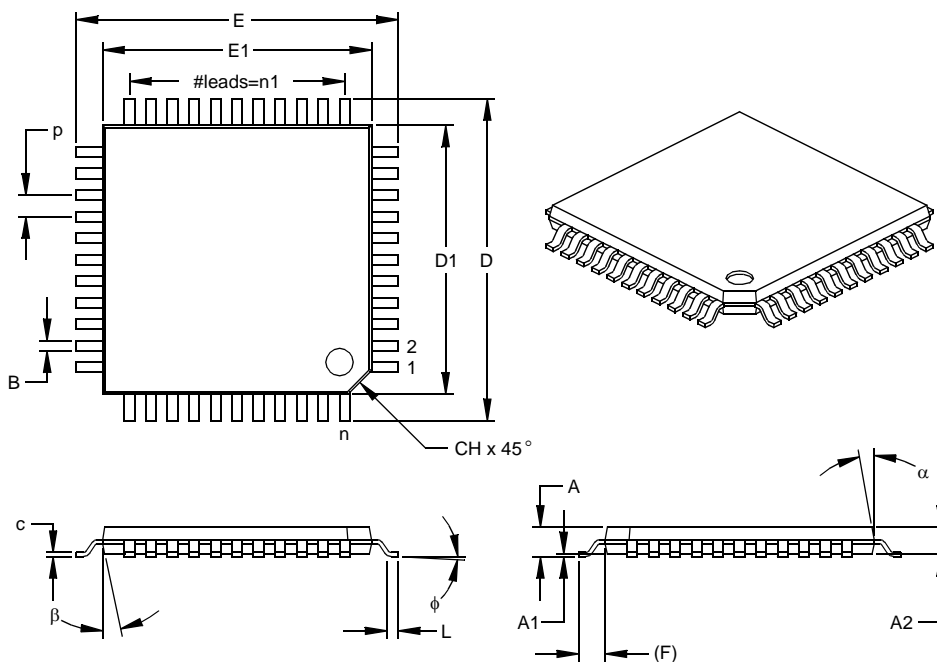
---

NOTES:

# PIC16F87X

## 44-Lead Plastic Thin Quad Flatpack (PT) 10x10x1 mm Body, 1.0/0.10 mm Lead Form (TQFP)

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



Units		INCHES			MILLIMETERS*		
Dimension Limits		MIN	NOM	MAX	MIN	NOM	MAX
Number of Pins	n		44			44	
Pitch	p		.031			0.80	
Pins per Side	n1		11			11	
Overall Height	A	.039	.043	.047	1.00	1.10	1.20
Molded Package Thickness	A2	.037	.039	.041	0.95	1.00	1.05
Standoff §	A1	.002	.004	.006	0.05	0.10	0.15
Foot Length	L	.018	.024	.030	0.45	0.60	0.75
Footprint (Reference)	(F)		.039		1.00		
Foot Angle	φ	0	3.5	7	0	3.5	7
Overall Width	E	.463	.472	.482	11.75	12.00	12.25
Overall Length	D	.463	.472	.482	11.75	12.00	12.25
Molded Package Width	E1	.390	.394	.398	9.90	10.00	10.10
Molded Package Length	D1	.390	.394	.398	9.90	10.00	10.10
Lead Thickness	c	.004	.006	.008	0.09	0.15	0.20
Lead Width	B	.012	.015	.017	0.30	0.38	0.44
Pin 1 Corner Chamfer	CH	.025	.035	.045	0.64	0.89	1.14
Mold Draft Angle Top	α	5	10	15	5	10	15
Mold Draft Angle Bottom	β	5	10	15	5	10	15

\* Controlling Parameter

§ Significant Characteristic

### Notes:

Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.

JEDEC Equivalent: MS-026

Drawing No. C04-076

Bus Collision During a Repeated START Condition (Case 1) .....	92
Bus Collision During a Repeated START Condition (Case2) .....	92
Bus Collision During a START Condition (SCL = 0) .....	91
Bus Collision During a STOP Condition .....	93
Bus Collision for Transmit and Acknowledge .....	89
Capture/Compare/PWM .....	166
CLKOUT and I/O .....	163
I <sup>2</sup> C Bus Data .....	171
I <sup>2</sup> C Bus START/STOP bits .....	170
I <sup>2</sup> C Master Mode First START Bit Timing .....	80
I <sup>2</sup> C Master Mode Reception Timing .....	85
I <sup>2</sup> C Master Mode Transmission Timing .....	83
Master Mode Transmit Clock Arbitration .....	88
Power-up Timer .....	164
Repeat START Condition .....	81
RESET .....	164
SPI Master Mode .....	70
SPI Slave Mode (CKE = 1) .....	71
SPI Slave Mode Timing (CKE = 0) .....	71
Start-up Timer .....	164
STOP Condition Receive or Transmit .....	87
Time-out Sequence on Power-up .....	127, 128
Timer0 .....	165
Timer1 .....	165
USART Asynchronous Master Transmission .....	100
USART Asynchronous Reception .....	102
USART Synchronous Receive .....	173
USART Synchronous Reception .....	108
USART Synchronous Transmission .....	106, 173
USART, Asynchronous Reception .....	104
Wake-up from SLEEP via Interrupt .....	133
Watchdog Timer .....	164
TMR0 .....	17
TMR0 Register .....	15
TMR1CS bit .....	51
TMR1H .....	17
TMR1H Register .....	15
TMR1L .....	17
TMR1L Register .....	15
TMR1ON bit .....	51
TMR2 .....	17
TMR2 Register .....	15
TMR2ON bit .....	55
TOUTPS0 bit .....	55
TOUTPS1 bit .....	55
TOUTPS2 bit .....	55
TOUTPS3 bit .....	55
TRISA Register .....	16
TRISB Register .....	16
TRISC Register .....	16
TRISD Register .....	16
TRISE Register .....	16, 36, 37
IBF Bit .....	37
IBOV Bit .....	37
OBF Bit .....	37
PSPMODE Bit .....	35, 36, 37, 38
TXREG .....	17

TXSTA Register .....	95
BRGH Bit .....	95
CSRC Bit .....	95
SYNC Bit .....	95
TRMT Bit .....	95
TX9 Bit .....	95
TX9D Bit .....	95
TXEN Bit .....	95

## U

UA .....	66
Universal Synchronous Asynchronous Receiver Transmitter. See USART	
Update Address, UA .....	66
USART .....	95
Address Detect Enable (ADDEN Bit) .....	96
Asynchronous Mode .....	99
Asynchronous Receive .....	101
Associated Registers .....	102
Block Diagram .....	101
Asynchronous Receive (9-bit Mode) .....	103
Associated Registers .....	104
Block Diagram .....	103
Timing Diagram .....	104
Asynchronous Receive with Address Detect. See Asynchronous Receive (9-bit Mode).	
Asynchronous Reception .....	102
Asynchronous Transmitter .....	99
Baud Rate Generator (BRG) .....	97
Baud Rate Formula .....	97
Baud Rates, Asynchronous Mode (BRGH=0) ...	98
High Baud Rate Select (BRGH Bit) .....	95
Sampling .....	97
Clock Source Select (CSRC Bit) .....	95
Continuous Receive Enable (CREN Bit) .....	96
Framing Error (FERR Bit) .....	96
Mode Select (SYNC Bit) .....	95
Overrun Error (OERR Bit) .....	96
RC6/TX/CK Pin .....	7, 9
RC7/RX/DT Pin .....	7, 9
RCSTA Register .....	96
Receive Data, 9th bit (RX9D Bit) .....	96
Receive Enable, 9-bit (RX9 Bit) .....	96
Serial Port Enable (SPEN Bit) .....	95, 96
Single Receive Enable (SREN Bit) .....	96
Synchronous Master Mode .....	105
Synchronous Master Reception .....	107
Associated Registers .....	107
Synchronous Master Transmission .....	105
Associated Registers .....	106
Synchronous Slave Mode .....	108
Synchronous Slave Reception .....	109
Associated Registers .....	109
Synchronous Slave Transmit .....	108
Associated Registers .....	108
Transmit Block Diagram .....	99
Transmit Data, 9th Bit (TX9D) .....	95
Transmit Enable (TXEN Bit) .....	95
Transmit Enable, Nine-bit (TX9 Bit) .....	95
Transmit Shift Register Status (TRMT Bit) .....	95
TXSTA Register .....	95

