



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	8MHz
Connectivity	UART/USART
Peripherals	POR, PWM, WDT
Number of I/O	33
Program Memory Size	16KB (8K x 16)
Program Memory Type	OTP
EEPROM Size	-
RAM Size	454 x 8
Voltage - Supply (Vcc/Vdd)	2.5V ~ 6V
Data Converters	-
Oscillator Type	External
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Through Hole
Package / Case	40-DIP (0.600", 15.24mm)
Supplier Device Package	40-PDIP
Purchase URL	<a href="https://www.e-xfl.com/product-detail/microchip-technology/pic17lc44-08i-p">https://www.e-xfl.com/product-detail/microchip-technology/pic17lc44-08i-p</a>

# PIC17C4X

---

NOTES:

## 5.0 INTERRUPTS

The PIC17C4X devices have 11 sources of interrupt:

- External interrupt from the RA0/INT pin
- Change on RB7:RB0 pins
- TMR0 Overflow
- TMR1 Overflow
- TMR2 Overflow
- TMR3 Overflow
- USART Transmit buffer empty
- USART Receive buffer full
- Capture1
- Capture2
- T0CKI edge occurred

There are four registers used in the control and status of interrupts. These are:

- CPUSTA
- INTSTA
- PIE
- PIR

The CPUSTA register contains the GLINTD bit. This is the Global Interrupt Disable bit. When this bit is set, all interrupts are disabled. This bit is part of the controller core functionality and is described in the Memory Organization section.

When an interrupt is responded to, the GLINTD bit is automatically set to disable any further interrupt, the return address is pushed onto the stack and the PC is loaded with the interrupt vector address. There are four interrupt vectors. Each vector address is for a specific interrupt source (except the peripheral interrupts which have the same vector address). These sources are:

- External interrupt from the RA0/INT pin
- TMR0 Overflow
- T0CKI edge occurred
- Any peripheral interrupt

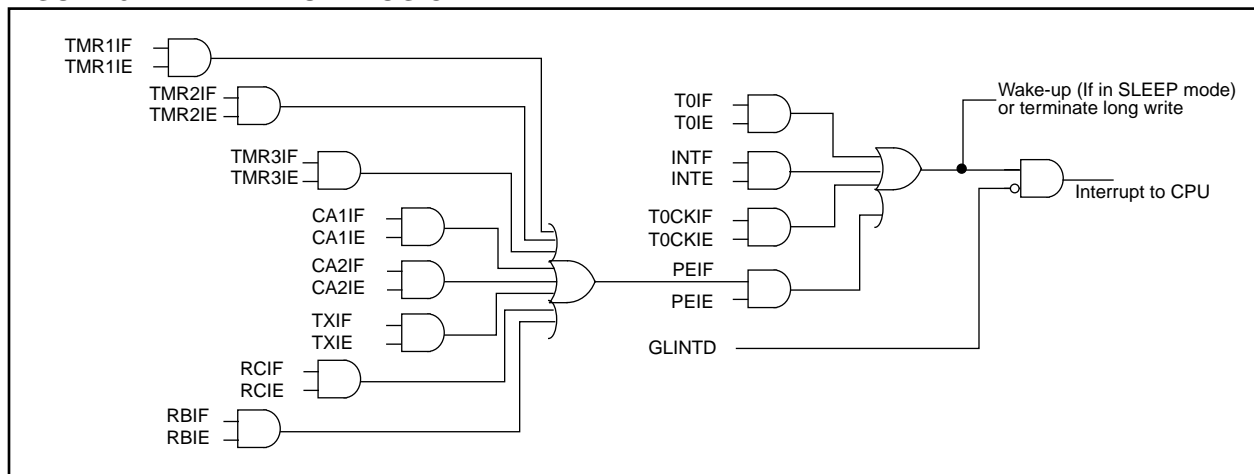
When program execution vectors to one of these interrupt vector addresses (except for the peripheral interrupt address), the interrupt flag bit is automatically cleared. Vectoring to the peripheral interrupt vector address does not automatically clear the source of the interrupt. In the peripheral interrupt service routine, the source(s) of the interrupt can be determined by testing the interrupt flag bits. The interrupt flag bit(s) must be cleared in software before re-enabling interrupts to avoid infinite interrupt requests.

All of the individual interrupt flag bits will be set regardless of the status of their corresponding mask bit or the GLINTD bit.

For external interrupt events, there will be an interrupt latency. For two cycle instructions, the latency could be one instruction cycle longer.

The “return from interrupt” instruction, `RETFIE`, can be used to mark the end of the interrupt service routine. When this instruction is executed, the stack is “POPed”, and the GLINTD bit is cleared (to re-enable interrupts).

**FIGURE 5-1: INTERRUPT LOGIC**



## 5.3 Peripheral Interrupt Request Register (PIR)

This register contains the individual flag bits for the peripheral interrupts.

**Note:** These bits will be set by the specified condition, even if the corresponding interrupt enable bit is cleared (interrupt disabled), or the GLINTD bit is set (all interrupts disabled). Before enabling an interrupt, the user may wish to clear the interrupt flag to ensure that the program does not immediately branch to the peripheral interrupt service routine.

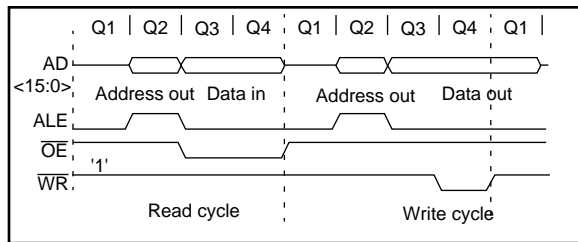
**FIGURE 5-4: PIR REGISTER (ADDRESS: 16h, BANK 1)**

R/W - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0	R - 1	R - 0
RBIF	TMR3IF	TMR2IF	TMR1IF	CA2IF	CA1IF	TXIF	RCIF
bit7							bit0
<p><b>bit 7: RBIF:</b> PORTB Interrupt on Change Flag bit  1 = One of the PORTB inputs changed (Software must end the mismatch condition)  0 = None of the PORTB inputs have changed</p> <p><b>bit 6: TMR3IF:</b> Timer3 Interrupt Flag bit  If Capture1 is enabled (<math>CA1/\overline{PR3} = 1</math>)  1 = Timer3 overflowed  0 = Timer3 did not overflow  If Capture1 is disabled (<math>CA1/\overline{PR3} = 0</math>)  1 = Timer3 value has rolled over to 0000h from equalling the period register (PR3H:PR3L) value  0 = Timer3 value has not rolled over to 0000h from equalling the period register (PR3H:PR3L) value</p> <p><b>bit 5: TMR2IF:</b> Timer2 Interrupt Flag bit  1 = Timer2 value has rolled over to 0000h from equalling the period register (PR2) value  0 = Timer2 value has not rolled over to 0000h from equalling the period register (PR2) value</p> <p><b>bit 4: TMR1IF:</b> Timer1 Interrupt Flag bit  If Timer1 is in 8-bit mode (<math>T16 = 0</math>)  1 = Timer1 value has rolled over to 0000h from equalling the period register (PR) value  0 = Timer1 value has not rolled over to 0000h from equalling the period register (PR2) value  If Timer1 is in 16-bit mode (<math>T16 = 1</math>)  1 = TMR1:TMR2 value has rolled over to 0000h from equalling the period register (PR1:PR2) value  0 = TMR1:TMR2 value has not rolled over to 0000h from equalling the period register (PR1:PR2) value</p> <p><b>bit 3: CA2IF:</b> Capture2 Interrupt Flag bit  1 = Capture event occurred on RB1/CAP2 pin  0 = Capture event did not occur on RB1/CAP2 pin</p> <p><b>bit 2: CA1IF:</b> Capture1 Interrupt Flag bit  1 = Capture event occurred on RB0/CAP1 pin  0 = Capture event did not occur on RB0/CAP1 pin</p> <p><b>bit 1: TXIF:</b> USART Transmit Interrupt Flag bit  1 = Transmit buffer is empty  0 = Transmit buffer is full</p> <p><b>bit 0: RCIF:</b> USART Receive Interrupt Flag bit  1 = Receive buffer is full  0 = Receive buffer is empty</p>							
<p>R = Readable bit  W = Writable bit  -n = Value at POR reset</p>							

## 6.1.2 EXTERNAL MEMORY INTERFACE

When either microprocessor or extended microcontroller mode is selected, PORTC, PORTD and PORTE are configured as the system bus. PORTC and PORTD are the multiplexed address/data bus and PORTE is for the control signals. External components are needed to demultiplex the address and data. This can be done as shown in Figure 6-4. The waveforms of address and data are shown in Figure 6-3. For complete timings, please refer to the electrical specification section.

**FIGURE 6-3: EXTERNAL PROGRAM MEMORY ACCESS WAVEFORMS**



The system bus requires that there is no bus conflict (minimal leakage), so the output value (address) will be capacitively held at the desired value.

As the speed of the processor increases, external EPROM memory with faster access time must be used. Table 6-2 lists external memory speed requirements for a given PIC17C4X device frequency.

In extended microcontroller mode, when the device is executing out of internal memory, the control signals will continue to be active. That is, they indicate the action that is occurring in the internal memory. The external memory access is ignored.

This following selection is for use with Microchip EPROMs. For interfacing to other manufacturers memory, please refer to the electrical specifications of the desired PIC17C4X device, as well as the desired memory device to ensure compatibility.

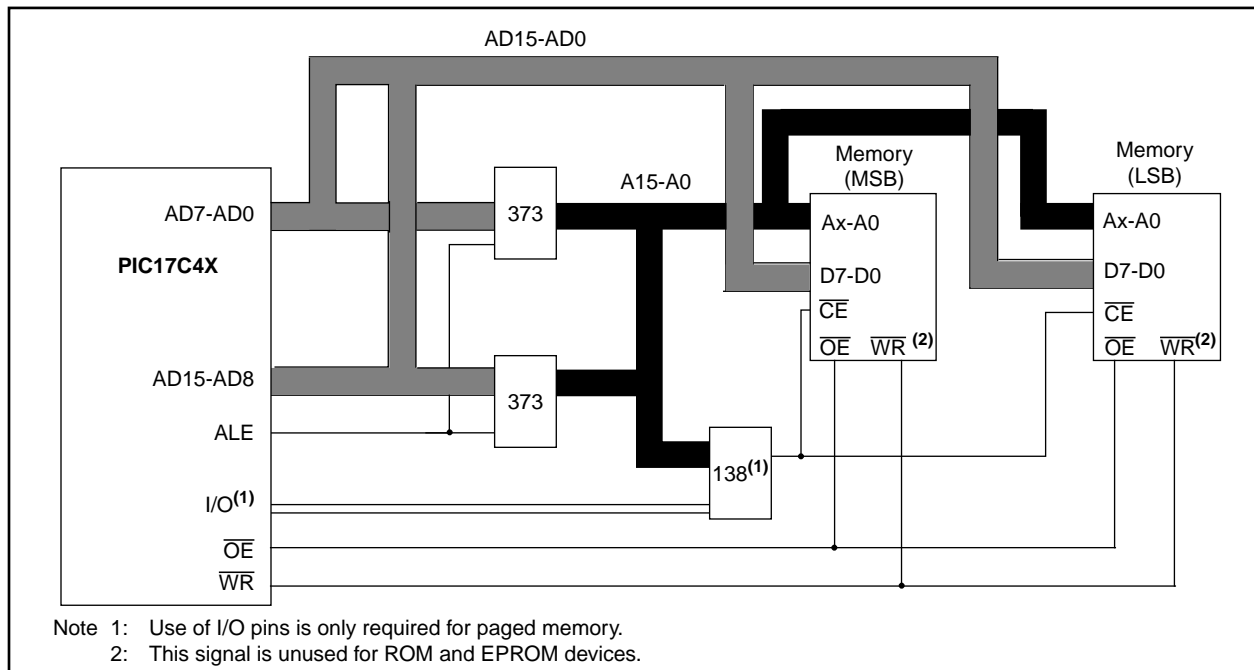
**TABLE 6-2: EPROM MEMORY ACCESS TIME ORDERING SUFFIX**

PIC17C4X Oscillator Frequency	Instruction Cycle Time (Tcy)	EPROM Suffix	
		PIC17C42	PIC17C43 PIC17C44
8 MHz	500 ns	-25	-25
16 MHz	250 ns	-12	-15
20 MHz	200 ns	-90	-10
25 MHz	160 ns	N.A.	-70
33 MHz	121 ns	N.A.	(1)

Note 1: The access times for this requires the use of fast SRAMS.

**Note:** The external memory interface is not supported for the LC devices.

**FIGURE 6-4: TYPICAL EXTERNAL PROGRAM MEMORY CONNECTION DIAGRAM**



# PIC17C4X

## 6.8 Bank Select Register (BSR)

The BSR is used to switch between banks in the data memory area (Figure 6-13). In the PIC17C42, PIC17CR42, and PIC17C42A only the lower nibble is implemented. While in the PIC17C43, PIC17CR43, and PIC17C44 devices, the entire byte is implemented. The lower nibble is used to select the peripheral register bank. The upper nibble is used to select the general purpose memory bank.

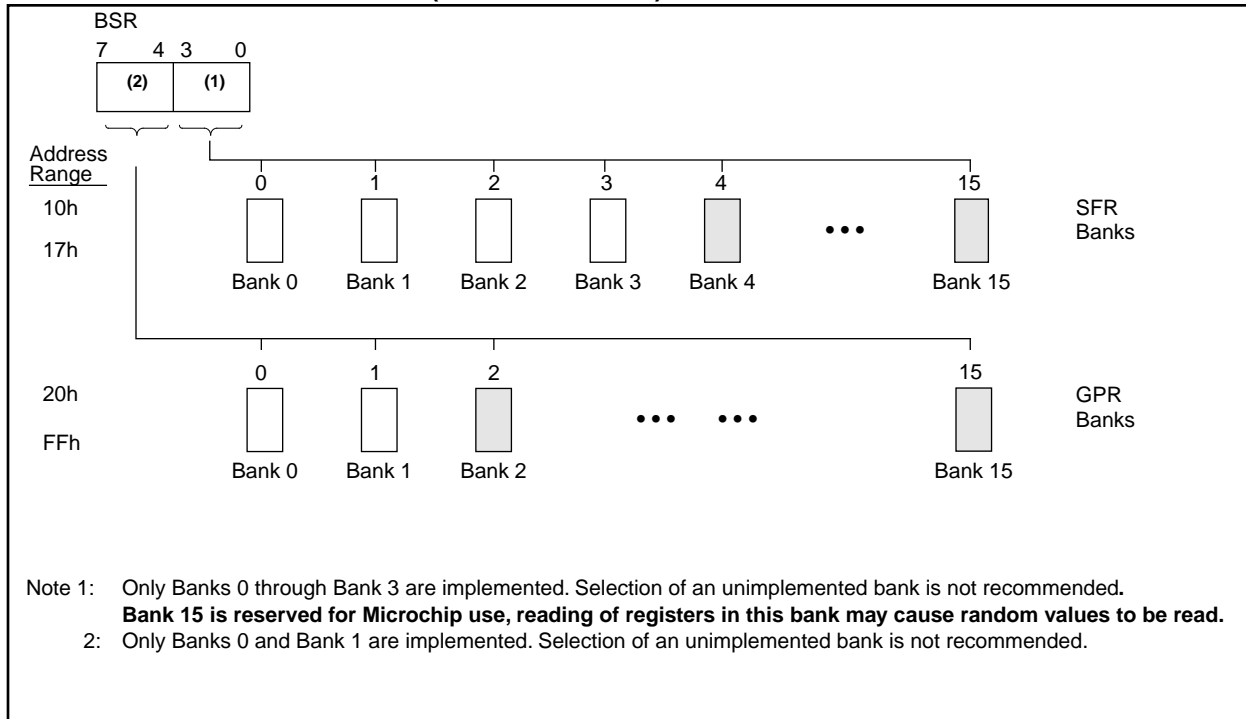
All the Special Function Registers (SFRs) are mapped into the data memory space. In order to accommodate the large number of registers, a banking scheme has been used. A segment of the SFRs, from address 10h to address 17h, is banked. The lower nibble of the bank select register (BSR) selects the currently active "peripheral bank." Effort has been made to group the peripheral registers of related functionality in one bank. However, it will still be necessary to switch from bank to bank in order to address all peripherals related to a single task. To assist this, a `MOVLB` bank instruction is in the instruction set.

For the PIC17C43, PIC17CR43, and PIC17C44 devices, the need for a large general purpose memory space dictated a general purpose RAM banking scheme. The upper nibble of the BSR selects the currently active general purpose RAM bank. To assist this, a `MOVLB` bank instruction has been provided in the instruction set.

If the currently selected bank is not implemented (such as Bank 13), any read will read all '0's. Any write is completed to the bit bucket and the ALU status bits will be set/cleared as appropriate.

**Note:** Registers in Bank 15 in the Special Function Register area, are reserved for Microchip use. Reading of registers in this bank may cause random values to be read.

**FIGURE 6-13: BSR OPERATION (PIC17C43/R43/44)**



Example 8-4 shows the sequence to do an 16 x 16 signed multiply. Equation 8-2 shows the algorithm that used. The 32-bit result is stored in four registers RES3:RES0. To account for the sign bits of the arguments, each argument pairs most significant bit (MSb) is tested and the appropriate subtractions are done.

## EQUATION 8-2: 16 x 16 SIGNED MULTIPLICATION ALGORITHM

RES3:RES0

$$\begin{aligned}
 &= \text{ARG1H:ARG1L} * \text{ARG2H:ARG2L} \\
 &= (\text{ARG1H} * \text{ARG2H} * 2^{16}) &+ \\
 &\quad (\text{ARG1H} * \text{ARG2L} * 2^8) &+ \\
 &\quad (\text{ARG1L} * \text{ARG2H} * 2^8) &+ \\
 &\quad (\text{ARG1L} * \text{ARG2L}) &+ \\
 &\quad (-1 * \text{ARG2H} <7> * \text{ARG1H:ARG1L} * 2^{16}) &+ \\
 &\quad (-1 * \text{ARG1H} <7> * \text{ARG2H:ARG2L} * 2^{16})
 \end{aligned}$$

## EXAMPLE 8-4: 16 x 16 SIGNED MULTIPLY ROUTINE

```

MOVFP ARG1L, WREG
MULWF ARG2L      ; ARG1L * ARG2L ->
                  ; PRODH:PRODL

MOVFP PRODH, RES1 ;
MOVFP PRODL, RES0 ;

;

MOVFP ARG1H, WREG
MULWF ARG2H      ; ARG1H * ARG2H ->
                  ; PRODH:PRODL

MOVFP PRODH, RES3 ;
MOVFP PRODL, RES2 ;

;

MOVFP ARG1L, WREG
MULWF ARG2H      ; ARG1L * ARG2H ->
                  ; PRODH:PRODL

MOVFP PRODL, WREG ;
ADDWF RES1, F    ; Add cross
MOVFP PRODH, WREG ; products
ADDWFC RES2, F   ;
CLRf WREG, F     ;
ADDWFC RES3, F   ;

;

MOVFP ARG1H, WREG ;
MULWF ARG2L      ; ARG1H * ARG2L ->
                  ; PRODH:PRODL

MOVFP PRODL, WREG ;
ADDWF RES1, F    ; Add cross
MOVFP PRODH, WREG ; products
ADDWFC RES2, F   ;
CLRf WREG, F     ;
ADDWFC RES3, F   ;

;

BTfSS ARG2H, 7   ; ARG2H:ARG2L neg?
GOTO SIGN_ARG1  ; no, check ARG1
MOVFP ARG1L, WREG ;
SUBWF RES2      ;
MOVFP ARG1H, WREG ;
SUBWFB RES3     ;

;

SIGN_ARG1
BTfSS ARG1H, 7   ; ARG1H:ARG1L neg?
GOTO CONT_CODE  ; no, done
MOVFP ARG2L, WREG ;
SUBWF RES2      ;
MOVFP ARG2H, WREG ;
SUBWFB RES3     ;

;

CONT_CODE
:

```

## 9.5 I/O Programming Considerations

### 9.5.1 BI-DIRECTIONAL I/O PORTS

Any instruction which writes, operates internally as a read followed by a write operation. For example, the BCF and BSF instructions read the register into the CPU, execute the bit operation, and write the result back to the register. Caution must be used when these instructions are applied to a port with both inputs and outputs defined. For example, a BSF operation on bit5 of PORTB will cause all eight bits of PORTB to be read into the CPU. Then the BSF operation takes place on bit5 and PORTB is written to the output latches. If another bit of PORTB is used as a bi-directional I/O pin (e.g. bit0) and it is defined as an input at this time, the input signal present on the pin itself would be read into the CPU and re-written to the data latch of this particular pin, overwriting the previous content. As long as the pin stays in the input mode, no problem occurs. However, if bit0 is switched into output mode later on, the content of the data latch may now be unknown.

Reading a port reads the values of the port pins. Writing to the port register writes the value to the port latch. When using read-modify-write instructions (BCF, BSF, BTG, etc.) on a port, the value of the port pins is read, the desired operation is performed with this value, and the value is then written to the port latch.

Example 9-5 shows the effect of two sequential read-modify-write instructions on an I/O port

### EXAMPLE 9-5: READ MODIFY WRITE INSTRUCTIONS ON AN I/O PORT

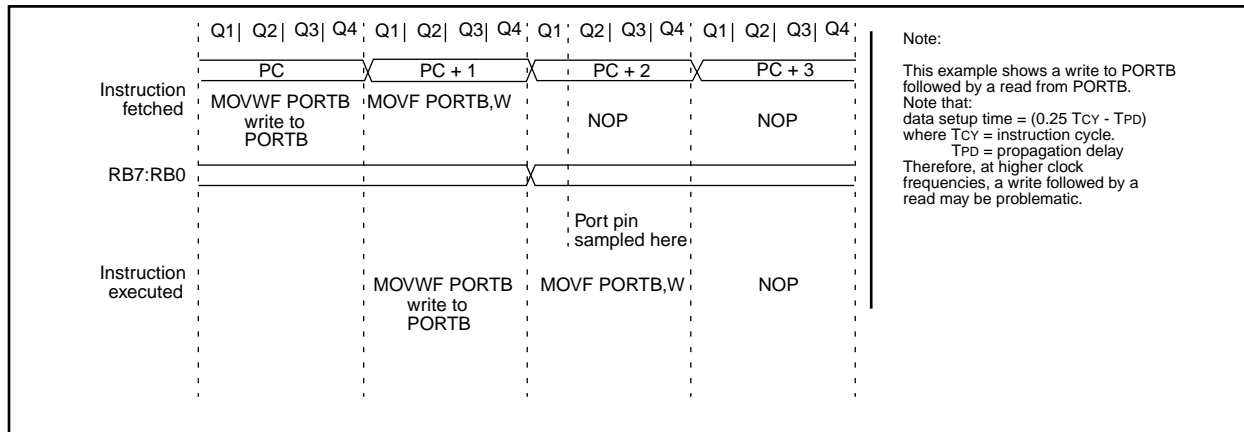
```
; Initial PORT settings: PORTB<7:4> Inputs
;                        PORTB<3:0> Outputs
; PORTB<7:6> have pull-ups and are
; not connected to other circuitry
;
;                        PORT latch  PORT pins
;                        -----
;
;
;   BCF   PORTB, 7      01pp pppp   11pp pppp
;   BCF   PORTB, 6      10pp pppp   11pp pppp
;
;   BCF   DDRB, 7      10pp pppp   11pp pppp
;   BCF   DDRB, 6      10pp pppp   10pp pppp
;
; Note that the user may have expected the
; pin values to be 00pp pppp. The 2nd BCF
; caused RB7 to be latched as the pin value
; (High).
```

**Note:** A pin actively outputting a Low or High should not be driven from external devices in order to change the level on this pin (i.e. "wired-or", "wired-and"). The resulting high output currents may damage the device.

### 9.5.2 SUCCESSIVE OPERATIONS ON I/O PORTS

The actual write to an I/O port happens at the end of an instruction cycle, whereas for reading, the data must be valid at the beginning of the instruction cycle (Figure 9-9). Therefore, care must be exercised if a write followed by a read operation is carried out on the same I/O port. The sequence of instructions should be such to allow the pin voltage to stabilize (load dependent) before executing the instruction that reads the values on that I/O port. Otherwise, the previous state of that pin may be read into the CPU rather than the "new" state. When in doubt, it is better to separate these instructions with a NOP or another instruction not accessing this I/O port.

**FIGURE 9-9: SUCCESSIVE I/O OPERATION**





## 10.0 OVERVIEW OF TIMER RESOURCES

The PIC17C4X has four timer modules. Each module can generate an interrupt to indicate that an event has occurred. These timers are called:

- Timer0 - 16-bit timer with programmable 8-bit prescaler
- Timer1 - 8-bit timer
- Timer2 - 8-bit timer
- Timer3 - 16-bit timer

For enhanced time-base functionality, two input Captures and two Pulse Width Modulation (PWM) outputs are possible. The PWMs use the TMR1 and TMR2 resources and the input Captures use the TMR3 resource.

### 10.1 Timer0 Overview

The Timer0 module is a simple 16-bit overflow counter. The clock source can be either the internal system clock ( $F_{osc}/4$ ) or an external clock.

The Timer0 module also has a programmable prescaler option. The PS3:PS0 bits (T0STA<4:1>) determine the prescaler value. TMR0 can increment at the following rates: 1:1, 1:2, 1:4, 1:8, 1:16, 1:32, 1:64, 1:128, 1:256.

When Timer0's clock source is an external clock, the Timer0 module can be selected to increment on either the rising or falling edge.

Synchronization of the external clock occurs after the prescaler. When the prescaler is used, the external clock frequency may be higher than the device's frequency. The maximum frequency is 50 MHz, given the high and low time requirements of the clock.

### 10.2 Timer1 Overview

The Timer1 module is an 8-bit timer/counter with an 8-bit period register (PR1). When the TMR1 value rolls over from the period match value to 0h, the TMR1IF flag is set, and an interrupt will be generated when enabled. In counter mode, the clock comes from the RB4/TCLK12 pin, which can also be selected to be the clock for the Timer2 module.

TMR1 can be concatenated to TMR2 to form a 16-bit timer. The TMR1 register is the LSB and TMR2 is the MSB. When in the 16-bit timer mode, there is a corresponding 16-bit period register (PR2:PR1). When the TMR2:TMR1 value rolls over from the period match value to 0h, the TMR1IF flag is set, and an interrupt will be generated when enabled.

### 10.3 Timer2 Overview

The TMR2 module is an 8-bit timer/counter with an 8-bit period register (PR2). When the TMR2 value rolls over from the period match value to 0h, the TMR2IF flag is set, and an interrupt will be generated when enabled. In counter mode, the clock comes from the RB4/TCLK12 pin, which can also be selected to be the clock for the TMR1 module.

TMR1 can be concatenated to TMR2 to form a 16-bit timer. The TMR2 register is the MSB and TMR1 is the LSB. When in the 16-bit timer mode, there is a corresponding 16-bit period register (PR2:PR1). When the TMR2:TMR1 value rolls over from the period match value to 0h, the TMR1IF flag is set, and an interrupt will be generated when enabled.

### 10.4 Timer3 Overview

The Timer3 module is a 16-bit timer/counter with a 16-bit period register. When the TMR3H:TMR3L value rolls over to 0h, the TMR3IF bit is set and an interrupt will be generated when enabled. In counter mode, the clock comes from the RB5/TCLK3 pin.

When operating in the dual capture mode, the period registers become the second 16-bit capture register.

### 10.5 Role of the Timer/Counters

The timer modules are general purpose, but have dedicated resources associated with them. Timer1 and Timer2 are the time-bases for the two Pulse Width Modulation (PWM) outputs, while Timer3 is the time-base for the two input captures.

NOTES:

FIGURE 13-5: ASYNCHRONOUS MASTER TRANSMISSION

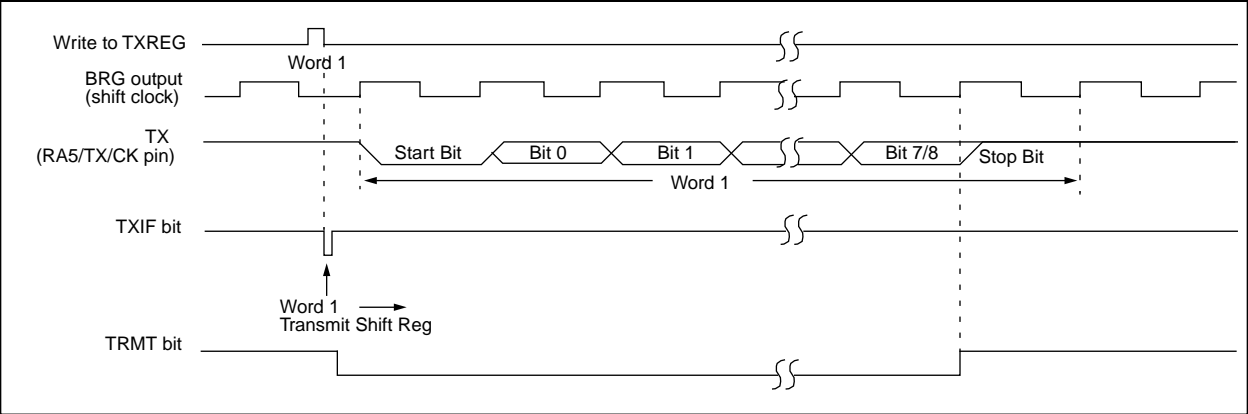


FIGURE 13-6: ASYNCHRONOUS MASTER TRANSMISSION (BACK TO BACK)

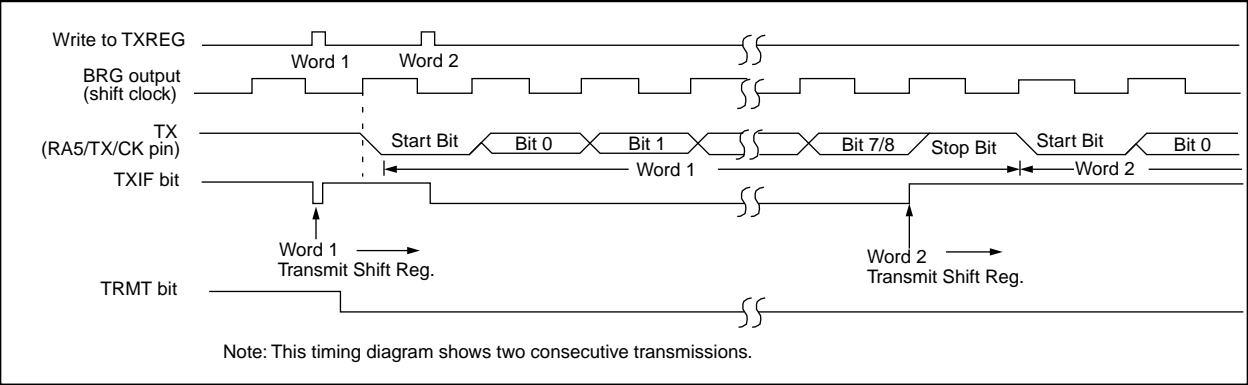


TABLE 13-5: REGISTERS ASSOCIATED WITH ASYNCHRONOUS TRANSMISSION

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on Reset	Value on all other resets (Note1)
16h, Bank 1	PIR	RBIF	TMR3IF	TMR2IF	TMR1IF	CA2IF	CA1IF	TXIF	RCIF	0000 0010	0000 0010
13h, Bank 0	RCSTA	SPEN	RX9	SREN	CREN	—	FERR	OERR	RX9D	0000 -00x	0000 -00u
16h, Bank 0	TXREG	Serial port transmit register								xxxx xxxx	uuuu uuuu
17h, Bank 1	PIE	RBIE	TMR3IE	TMR2IE	TMR1IE	CA2IE	CA1IE	TXIE	RCIE	0000 0000	0000 0000
15h, Bank 0	TXSTA	CSRC	TX9	TXEN	SYNC	—	—	TRMT	TX9D	0000 --1x	0000 --1u
17h, Bank 0	SPBRG	Baud rate generator register								xxxx xxxx	uuuu uuuu

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

Note 1: Other (non power-up) resets include: external reset through  $\overline{\text{MCLR}}$  and Watchdog Timer Reset.

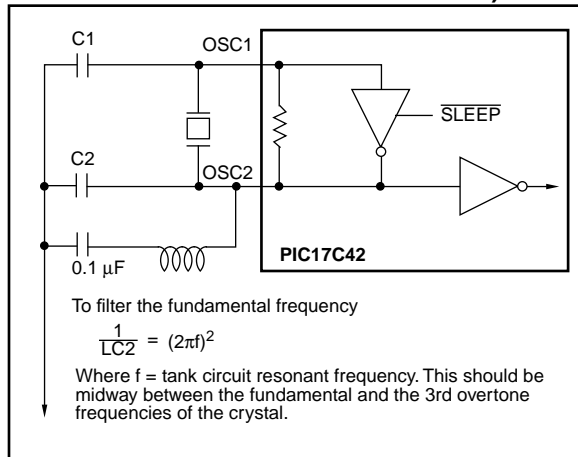
**TABLE 13-8: REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER RECEPTION**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on Reset	Value on all other resets (Note1)
16h, Bank 1	PIR	RBIF	TMR3IF	TMR2IF	TMR1IF	CA2IF	CA1IF	TXIF	RCIF	0000 0010	0000 0010
13h, Bank 0	RCSTA	SPEN	RX9	SREN	CREN	—	FERR	OERR	RX9D	0000 -00x	0000 -00u
14h, Bank 0	RCREG	RX7	RX6	RX5	RX4	RX3	RX2	RX1	RX0	xxxx xxxx	uuuu uuuu
17h, Bank 1	PIE	RBIE	TMR3IE	TMR2IE	TMR1IE	CA2IE	CA1IE	TXIE	RCIE	0000 0000	0000 0000
15h, Bank 0	TXSTA	CSRC	TX9	TXEN	SYNC	—	—	TRMT	TX9D	0000 --1x	0000 --1u
17h, Bank 0	SPBRG	Baud rate generator register								xxxx xxxx	uuuu uuuu

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

Note 1: Other (non power-up) resets include: external reset through  $\overline{\text{MCLR}}$  and Watchdog Timer Reset.

**FIGURE 14-3: CRYSTAL OPERATION, OVERTONE CRYSTALS (XT OSC CONFIGURATION)**



**TABLE 14-2: CAPACITOR SELECTION FOR CERAMIC RESONATORS**

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

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

**Resonators Used:**

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

Resonators used did not have built-in capacitors.

**TABLE 14-3: CAPACITOR SELECTION FOR CRYSTAL OSCILLATOR**

Osc Type	Freq	C1	C2
LF	32 kHz <sup>(1)</sup>	100-150 pF	100-150 pF
	1 MHz	10-33 pF	10-33 pF
	2 MHz	10-33 pF	10-33 pF
XT	2 MHz	47-100 pF	47-100 pF
	4 MHz	15-68 pF	15-68 pF
	8 MHz <sup>(2)</sup>	15-47 pF	15-47 pF
	16 MHz	TBD	TBD
	25 MHz	15-47 pF	15-47 pF
	32 MHz <sup>(3)</sup>	0 <sup>(3)</sup>	0 <sup>(3)</sup>

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

Note 1: For VDD > 4.5V, C1 = C2 ≈ 30 pF is recommended.

2: Rs of 330Ω is required for a capacitor combination of 15/15 pF.

3: Only the capacitance of the board was present.

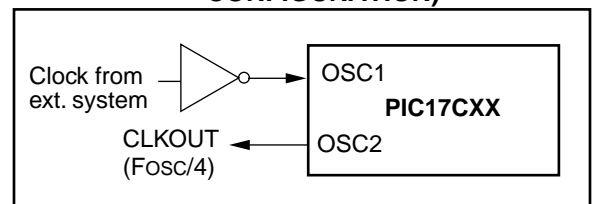
#### Crystals Used:

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

#### 14.2.3 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 14-4: EXTERNAL CLOCK INPUT OPERATION (EC OSC CONFIGURATION)**



**CALL**

**Subroutine Call**

Syntax: [label] CALL k

Operands: 0 ≤ k ≤ 4095

Operation: PC+ 1→ TOS, k → PC<12:0>, k<12:8> → PCLATH<4:0>; PC<15:13> → PCLATH<7:5>

Status Affected: None

Encoding: 

111k	kkkk	kkkk	kkkk
------	------	------	------

Description: Subroutine call within 8K page. First, return address (PC+1) is pushed onto the stack. The 13-bit value is loaded into PC bits<12:0>. Then the upper-eight bits of the PC are copied into PCLATH. Call is a two-cycle instruction. See LCALL for calls outside 8K memory space.

Words: 1  
Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'<7:0>	Execute	NOP
Forced NOP	NOP	Execute	NOP

**Example:**                HERE        CALL    THERE

Before Instruction  
PC = Address ( HERE )

After Instruction  
PC = Address ( THERE )  
TOS = Address ( HERE + 1 )

**CLRF**

**Clear f**

Syntax: [label] CLRF f,s

Operands: 0 ≤ f ≤ 255

Operation: 00h → f, s ∈ [0,1]  
00h → dest

Status Affected: None

Encoding: 

0010	100s	ffff	ffff
------	------	------	------

Description: Clears the contents of the specified register(s).  
s = 0: Data memory location 'f' and WREG are cleared.  
s = 1: Data memory location 'f' is cleared.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Execute	Write register 'f' and other specified register

**Example:**                CLRF        FLAG\_REG

Before Instruction  
FLAG\_REG = 0x5A

After Instruction  
FLAG\_REG = 0x00

MPASM allow full symbolic debugging from the Microchip Universal Emulator System (PICMASTER).

MPASM has the following features to assist in developing software for specific use applications.

- Provides translation of Assembler source code to object code for all Microchip microcontrollers.
- Macro assembly capability.
- Produces all the files (Object, Listing, Symbol, and special) required for symbolic debug with Microchip's emulator systems.
- Supports Hex (default), Decimal and Octal source and listing formats.

MPASM provides a rich directive language to support programming of the PIC16/17. Directives are helpful in making the development of your assemble source code shorter and more maintainable.

## **16.11 Software Simulator (MPLAB-SIM)**

The MPLAB-SIM Software Simulator allows code development in a PC host environment. It allows the user to simulate the PIC16/17 series microcontrollers on an instruction level. On any given instruction, the user may examine or modify any of the data areas or provide external stimulus to any of the pins. The input/output radix can be set by the user and the execution can be performed in; single step, execute until break, or in a trace mode.

MPLAB-SIM fully supports symbolic debugging using MPLAB-C and MPASM. The Software Simulator offers the low cost flexibility to develop and debug code outside of the laboratory environment making it an excellent multi-project software development tool.

## **16.12 C Compiler (MPLAB-C)**

The MPLAB-C Code Development System is a complete 'C' compiler and integrated development environment for Microchip's PIC16/17 family of microcontrollers. The compiler provides powerful integration capabilities and ease of use not found with other compilers.

For easier source level debugging, the compiler provides symbol information that is compatible with the MPLAB IDE memory display (PICMASTER emulator software versions 1.13 and later).

## **16.13 Fuzzy Logic Development System (fuzzyTECH-MP)**

*fuzzyTECH-MP* fuzzy logic development tool is available in two versions - a low cost introductory version, MP Explorer, for designers to gain a comprehensive working knowledge of fuzzy logic system design; and a full-featured version, *fuzzyTECH-MP*, edition for implementing more complex systems.

Both versions include Microchip's *fuzzyLAB™* demonstration board for hands-on experience with fuzzy logic systems implementation.

## **16.14 MP-DriveWay™ – Application Code Generator**

MP-DriveWay is an easy-to-use Windows-based Application Code Generator. With MP-DriveWay you can visually configure all the peripherals in a PIC16/17 device and, with a click of the mouse, generate all the initialization and many functional code modules in C language. The output is fully compatible with Microchip's MPLAB-C C compiler. The code produced is highly modular and allows easy integration of your own code. MP-DriveWay is intelligent enough to maintain your code through subsequent code generation.

## **16.15 SEEVAL® Evaluation and Programming System**

The SEEVAL SEEPROM Designer's Kit supports all Microchip 2-wire and 3-wire Serial EEPROMs. The kit includes everything necessary to read, write, erase or program special features of any Microchip SEEPROM product including Smart Serials™ and secure serials. The Total Endurance™ Disk is included to aid in trade-off analysis and reliability calculations. The total kit can significantly reduce time-to-market and result in an optimized system.

## **16.16 TrueGauge® Intelligent Battery Management**

The TrueGauge development tool supports system development with the MTA11200B TrueGauge Intelligent Battery Management IC. System design verification can be accomplished before hardware prototypes are built. User interface is graphically-oriented and measured data can be saved in a file for exporting to Microsoft Excel.

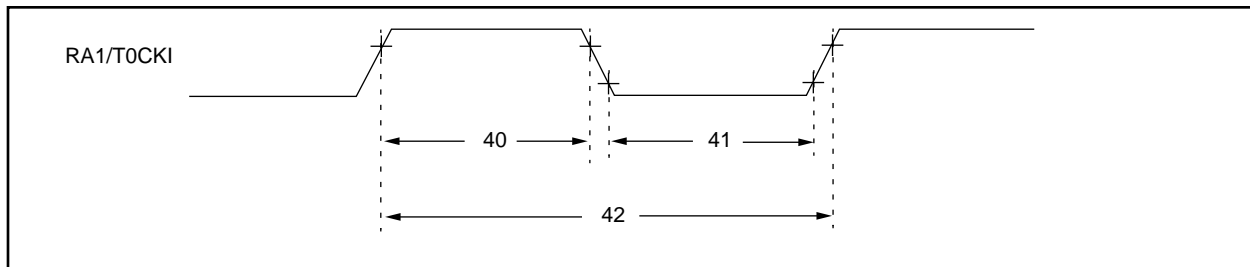
## **16.17 KEELOQ® Evaluation and Programming Tools**

KEELOQ evaluation and programming tools support Microchips HCS Secure Data Products. The HCS evaluation kit includes an LCD display to show changing codes, a decoder to decode transmissions, and a programming interface to program test transmitters.

# PIC17C4X

Applicable Devices 42 R42 42A 43 R43 44

**FIGURE 17-5: TIMER0 CLOCK TIMINGS**



**TABLE 17-5: TIMER0 CLOCK REQUIREMENTS**

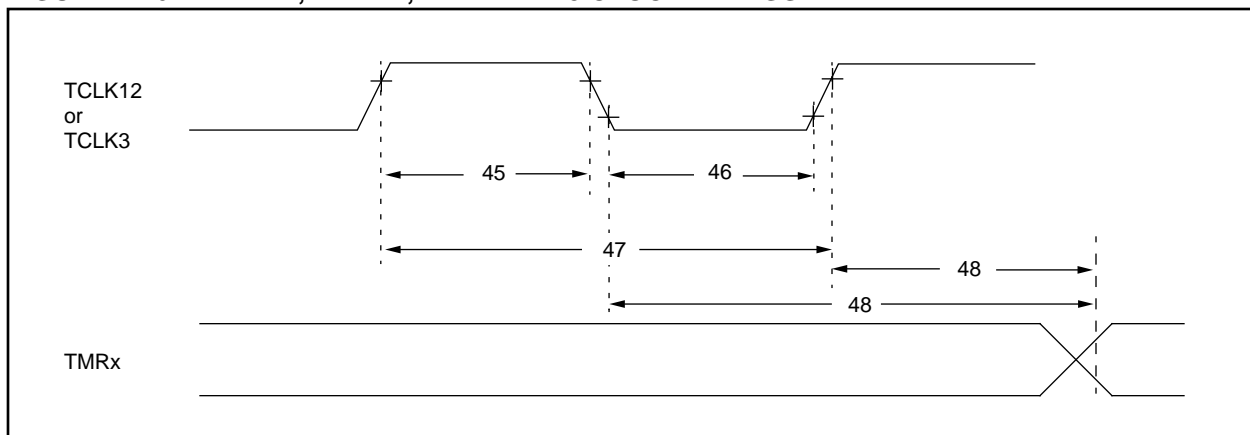
Parameter No.	Sym	Characteristic		Min	Typ†	Max	Units	Conditions
40	Tt0H	T0CKI High Pulse Width	No Prescaler	0.5TCY + 20 §	—	—	ns	
			With Prescaler	10*	—	—	ns	
41	Tt0L	T0CKI Low Pulse Width	No Prescaler	0.5TCY + 20 §	—	—	ns	
			With Prescaler	10*	—	—	ns	
42	Tt0P	T0CKI Period		$\frac{TCY + 40}{N}$ §	—	—	ns	N = prescale value (1, 2, 4, ..., 256)

\* 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.

**FIGURE 17-6: TIMER1, TIMER2, AND TIMER3 CLOCK TIMINGS**



**TABLE 17-6: TIMER1, TIMER2, AND TIMER3 CLOCK REQUIREMENTS**

Parameter No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
45	Tt123H	TCLK12 and TCLK3 high time	0.5 Tcy + 20 §	—	—	ns	N = prescale value (1, 2, 4, 8)
46	Tt123L	TCLK12 and TCLK3 low time	0.5 Tcy + 20 §	—	—	ns	
47	Tt123P	TCLK12 and TCLK3 input period	$\frac{Tcy + 40}{N}$ §	—	—	ns	
48	TckE2tmrl	Delay from selected External Clock Edge to Timer increment	2Tosc §	—	6 Tosc §	—	

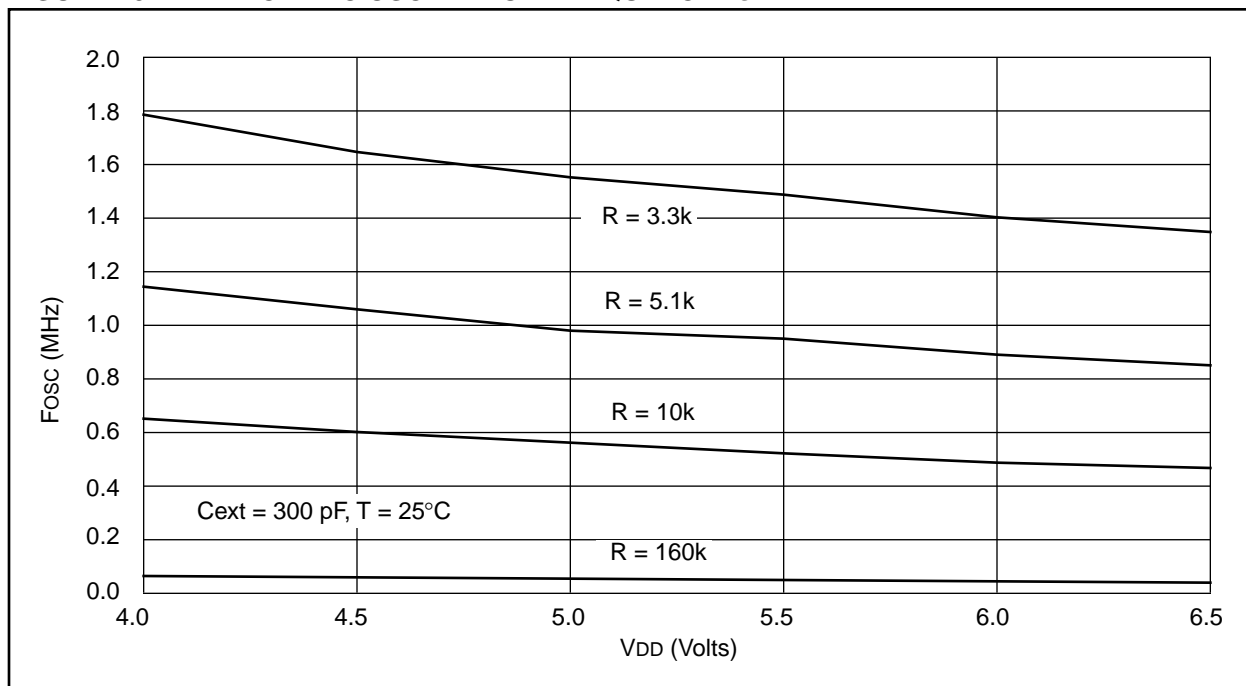
\* These parameters are characterized but not tested.

† 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.



**FIGURE 18-4: TYPICAL RC OSCILLATOR FREQUENCY vs. V<sub>DD</sub>**



**TABLE 18-2: RC OSCILLATOR FREQUENCIES**

Cext	Rext	Average	
		Fosc @ 5V, 25°C	
22 pF	10k	3.33 MHz	± 12%
	100k	353 kHz	± 13%
100 pF	3.3k	3.54 MHz	± 10%
	5.1k	2.43 MHz	± 14%
	10k	1.30 MHz	± 17%
	100k	129 kHz	± 10%
300 pF	3.3k	1.54 MHz	± 14%
	5.1k	980 kHz	± 12%
	10k	564 kHz	± 16%
	160k	35 kHz	± 18%

Applicable Devices	42	R42	42A	43	R43	44
--------------------	----	-----	-----	----	-----	----

DC CHARACTERISTICS		Standard Operating Conditions (unless otherwise stated)					
		Operating temperature					
		-40°C ≤ TA ≤ +40°C					
		Operating voltage VDD range as described in Section 19.1					
Parameter No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
		<b>Internal Program Memory Programming Specs (Note 4)</b>					
D110	VPP	Voltage on $\overline{\text{MCLR}}$ /VPP pin	12.75	–	13.25	V	Note 5
D111	VDDP	Supply voltage during programming	4.75	5.0	5.25	V	
D112	IPP	Current into $\overline{\text{MCLR}}$ /VPP pin	–	25 ‡	50 ‡	mA	
D113	IDDP	Supply current during programming	–	–	30 ‡	mA	
D114	TPROG	Programming pulse width	10	100	1000	μs	Terminated via internal/external interrupt or a reset

\* 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.

‡ These parameters are for design guidance only and are not tested, nor characterized.

Note 1: In RC oscillator configuration, the OSC1/CLKIN pin is a Schmitt Trigger input. It is not recommended that the PIC17CXX 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. The specified levels represent normal operating conditions. Higher leakage current may be measured at different input voltages.

3: Negative current is defined as coming out of 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: PIC17CXX Programming Specifications (Literature number DS30139).

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.

**Note:** When using the Table Write for internal programming, the device temperature must be less than 40°C.

# PIC17C4X

## Timing Diagrams

Asynchronous Master Transmission .....	90
Asynchronous Reception .....	92
Back to Back Asynchronous Master Transmission ....	90
Interrupt (INT, TMR0 Pins) .....	26
PIC17C42 Capture .....	159
PIC17C42 CLKOUT and I/O .....	156
PIC17C42 Memory Interface Read .....	162
PIC17C42 Memory Interface Write .....	161
PIC17C42 PWM Timing .....	159
PIC17C42 RESET, Watchdog Timer, Oscillator	
Start-up Timer and Power-up Timer .....	157
PIC17C42 Timer0 Clock .....	158
PIC17C42 Timer1, Timer2 and Timer3 Clock .....	158
PIC17C42 USART Module, Synchronous	
Receive .....	160
PIC17C42 USART Module, Synchronous	
Transmission .....	160
PIC17C43/44 Capture Timing .....	188
PIC17C43/44 CLKOUT and I/O .....	185
PIC17C43/44 External Clock .....	184
PIC17C43/44 Memory Interface Read .....	191
PIC17C43/44 Memory Interface Write .....	190
PIC17C43/44 PWM Timing .....	188
PIC17C43/44 RESET, Watchdog Timer, Oscillator	
Start-up Timer and Power-up Timer .....	186
PIC17C43/44 Timer0 Clock .....	187
PIC17C43/44 Timer1, Timer2 and Timer3 Clock ....	187
PIC17C43/44 USART Module Synchronous	
Receive .....	189
PIC17C43/44 USART Module Synchronous	
Transmission .....	189
Synchronous Reception .....	95
Synchronous Transmission .....	94
Table Read .....	48
Table Write .....	46
TMR0 .....	68, 69
TMR0 Read/Write in Timer Mode .....	70
TMR1, TMR2, and TMR3 in External Clock Mode ....	80
TMR1, TMR2, and TMR3 in Timer Mode .....	81
Wake-Up from SLEEP .....	105
Timing Diagrams and Specifications .....	155
Timing Parameter Symbolology .....	153
TLRD .....	44, 139
TLWT .....	43, 140
TMR0	
16-bit Read .....	69
16-bit Write .....	69
Clock Timing .....	158
Module .....	68
Operation .....	68
Overview .....	65
Prescaler Assignments .....	69
Read/Write Considerations .....	69
Read/Write in Timer Mode .....	70
Timing .....	68, 69
TMR0 STATUS/Control Register (T0STA) .....	38
TMR0H .....	34
TMR0L .....	34
TMR1 .....	20, 35
8-bit Mode .....	73
External Clock Input .....	73
Overview .....	65
Timer Mode .....	81
Timing in External Clock Mode .....	80
Two 8-bit Timer/Counter Mode .....	73

Using with PWM .....	75
TMR1CS .....	71
TMR1IE .....	23
TMR1IF .....	24
TMR1ON .....	72
TMR2 .....	20, 35
8-bit Mode .....	73
External Clock Input .....	73
In Timer Mode .....	81
Timing in External Clock Mode .....	80
Two 8-bit Timer/Counter Mode .....	73
Using with PWM .....	75
TMR2CS .....	71
TMR2IE .....	23
TMR2IF .....	24
TMR2ON .....	72
TMR3	
Dual Capture1 Register Mode .....	79
Example, Reading From .....	80
Example, Writing To .....	80
External Clock Input .....	80
In Timer Mode .....	81
One Capture and One Period Register Mode .....	78
Overview .....	65
Reading/Writing .....	80
Timing in External Clock Mode .....	80
TMR3CS .....	71, 77
TMR3H .....	20, 35
TMR3IE .....	23
TMR3IF .....	24, 77
TMR3L .....	20, 35
TMR3ON .....	72, 77
$\overline{TO}$ .....	37, 103, 105
Transmit Status and Control Register .....	83
TRMT .....	83
TSTFSZ .....	140
Turning on 16-bit Timer .....	74
TX9 .....	83
TX9d .....	83
TXEN .....	83
TXIE .....	23
TXIF .....	24
TXREG .....	19, 34, 89, 93, 97, 98
TXSTA .....	19, 34, 92, 96, 98

## U

Upward Compatibility .....	5
USART	
Asynchronous Master Transmission .....	90
Asynchronous Mode .....	89
Asynchronous Receive .....	91
Asynchronous Transmitter .....	89
Baud Rate Generator .....	86
Synchronous Master Mode .....	93
Synchronous Master Reception .....	95
Synchronous Master Transmission .....	93
Synchronous Slave Mode .....	97
Synchronous Slave Transmit .....	97

## W

Wake-up from SLEEP .....	105
Wake-up from SLEEP Through Interrupt .....	105
Watchdog Timer .....	99, 103

# PIC17C4X

---

Table 17-9:	Serial Port Synchronous Transmission Requirements .....	160
Table 17-10:	Serial Port Synchronous Receive Requirements .....	160
Table 17-11:	Memory Interface Write Requirements .....	161
Table 17-12:	Memory Interface Read Requirements .....	162
Table 18-1:	Pin Capacitance per Package Type .....	163
Table 18-2:	RC Oscillator Frequencies .....	165
Table 19-1:	Cross Reference of Device Specs for Oscillator Configurations and Frequencies of Operation (Commercial Devices) .....	176
Table 19-2:	External Clock Timing Requirements .....	184
Table 19-3:	CLKOUT and I/O Timing Requirements ...	185
Table 19-4:	Reset, Watchdog Timer, Oscillator Start-Up Timer and Power-Up Timer Requirements .....	186
Table 19-5:	Timer0 Clock Requirements .....	187
Table 19-6:	Timer1, Timer2, and Timer3 Clock Requirements .....	187
Table 19-7:	Capture Requirements .....	188
Table 19-8:	PWM Requirements .....	188
Table 19-9:	Synchronous Transmission Requirements .....	189
Table 19-10:	Synchronous Receive Requirements .....	189
Table 19-11:	Memory Interface Write Requirements (Not Supported in PIC17LC4X Devices) ...	190
Table 19-12:	Memory Interface read Requirements (Not Supported in PIC17LC4X Devices) ...	191
Table 20-1:	Pin Capacitance per Package Type .....	193
Table 20-2:	RC Oscillator Frequencies .....	195
Table E-1:	Pin Compatible Devices .....	221

## LIST OF EQUATIONS

Equation 8-1:	16 x 16 Unsigned Multiplication Algorithm .....	50
Equation 8-2:	16 x 16 Signed Multiplication Algorithm .....	51

# PIC17C4X

---

## READER RESPONSE

It is our intention to provide you with the best documentation possible to ensure successful use of your Microchip product. If you wish to provide your comments on organization, clarity, subject matter, and ways in which our documentation can better serve you, please FAX your comments to the Technical Publications Manager at (602) 786-7578.

Please list the following information, and use this outline to provide us with your comments about this Data Sheet.

To: Technical Publications Manager  
RE: Reader Response  
Total Pages Sent \_\_\_\_\_  
From: Name \_\_\_\_\_  
Company \_\_\_\_\_  
Address \_\_\_\_\_  
City / State / ZIP / Country \_\_\_\_\_  
Telephone: (\_\_\_\_\_) \_\_\_\_\_ - \_\_\_\_\_ FAX: (\_\_\_\_\_) \_\_\_\_\_ - \_\_\_\_\_

Application (optional):

Would you like a reply? \_\_\_Y \_\_\_N

Device: **PIC17C4X**

Literature Number: **DS30412C**

Questions:

1. What are the best features of this document?

---

---

2. How does this document meet your hardware and software development needs?

---

---

3. Do you find the organization of this data sheet easy to follow? If not, why?

---

---

4. What additions to the data sheet do you think would enhance the structure and subject?

---

---

5. What deletions from the data sheet could be made without affecting the overall usefulness?

---

---

6. Is there any incorrect or misleading information (what and where)?

---

---

7. How would you improve this document?

---

---

8. How would you improve our software, systems, and silicon products?

---

---