



Welcome to E-XFL.COM

What is "[Embedded - Microcontrollers](#)"?

"[Embedded - Microcontrollers](#)" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "[Embedded - Microcontrollers](#)"

Details

Product Status	Obsolete
Core Processor	PIC
Core Size	8-Bit
Speed	16MHz
Connectivity	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)	4.5V ~ 6V
Data Converters	-
Oscillator Type	External
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	44-QFP
Supplier Device Package	44-MQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic17c44t-16e-pq

4.1.3 OSCILLATOR START-UP TIMER (OST)

The Oscillator Start-up Timer (OST) provides a 1024 oscillator cycle (1024Tosc) delay after $\overline{\text{MCLR}}$ is detected high or a wake-up from SLEEP event occurs.

The OST time-out is invoked only for XT and LF oscillator modes on a Power-on Reset or a Wake-up from SLEEP.

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

4.1.4 TIME-OUT SEQUENCE

On power-up the time-out sequence is as follows: First the internal POR signal goes high when the POR trip point is reached. If $\overline{\text{MCLR}}$ is high, then both the OST and PWRT timers start. In general the PWRT time-out is longer, except with low frequency crystals/resonators. The total time-out also varies based on oscillator configuration. Table 4-1 shows the times that are associated with the oscillator configuration. Figure 4-2 and Figure 4-3 display these time-out sequences.

If the device voltage is not within electrical specification at the end of a time-out, the $\overline{\text{MCLR}}/\text{VPP}$ pin must be held low until the voltage is within the device specification. The use of an external RC delay is sufficient for many of these applications.

TABLE 4-1: TIME-OUT IN VARIOUS SITUATIONS

Oscillator Configuration	Power-up	Wake up from SLEEP	MCLR Reset
XT, LF	Greater of: 96 ms or 1024Tosc	1024Tosc	—
EC, RC	Greater of: 96 ms or 1024Tosc	—	—

The time-out sequence begins from the first rising edge of $\overline{\text{MCLR}}$.

Table 4-3 shows the reset conditions for some special registers, while Table 4-4 shows the initialization conditions for all the registers. The shaded registers (in Table 4-4) are for all devices except the PIC17C42. In the PIC17C42, the PRODH and PRODL registers are general purpose RAM.

TABLE 4-2: STATUS BITS AND THEIR SIGNIFICANCE

$\overline{\text{TO}}$	$\overline{\text{PD}}$	Event
1	1	Power-on Reset, $\overline{\text{MCLR}}$ Reset during normal operation, or CLRWDT instruction executed
1	0	$\overline{\text{MCLR}}$ Reset during SLEEP or interrupt wake-up from SLEEP
0	1	WDT Reset during normal operation
0	0	WDT Reset during SLEEP

In Figure 4-2, Figure 4-3 and Figure 4-4, $\text{TPWRT} > \text{TOST}$, as would be the case in higher frequency crystals. For lower frequency crystals, (i.e., 32 kHz) TOST would be greater.

TABLE 4-3: RESET CONDITION FOR THE PROGRAM COUNTER AND THE CPUSTA REGISTER

Event		PCH:PCL	CPUSTA	OST Active
Power-on Reset		0000h	--11 11--	Yes
$\overline{\text{MCLR}}$ Reset during normal operation		0000h	--11 11--	No
$\overline{\text{MCLR}}$ Reset during SLEEP		0000h	--11 10--	Yes ⁽²⁾
WDT Reset during normal operation		0000h	--11 01--	No
WDT Reset during SLEEP ⁽³⁾		0000h	--11 00--	Yes ⁽²⁾
Interrupt wake-up from SLEEP	GLINTD is set	PC + 1	--11 10--	Yes ⁽²⁾
	GLINTD is clear	PC + 1 ⁽¹⁾	--10 10--	Yes ⁽²⁾

Legend: u = unchanged, x = unknown, - = unimplemented read as '0'.

Note 1: On wake-up, this instruction is executed. The instruction at the appropriate interrupt vector is fetched and then executed.

2: The OST is only active when the Oscillator is configured for XT or LF modes.

3: The Program Counter = 0, that is the device branches to the reset vector. This is different from the mid-range devices.

PIC17C4X

TABLE 4-4: INITIALIZATION CONDITIONS FOR SPECIAL FUNCTION REGISTERS (Cont.'d)

Register	Address	Power-on Reset	MCLR Reset WDT Reset	Wake-up from SLEEP through interrupt
Bank 2				
TMR1	10h	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR2	11h	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR3L	12h	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR3H	13h	xxxx xxxx	uuuu uuuu	uuuu uuuu
PR1	14h	xxxx xxxx	uuuu uuuu	uuuu uuuu
PR2	15h	xxxx xxxx	uuuu uuuu	uuuu uuuu
PR3/CA1L	16h	xxxx xxxx	uuuu uuuu	uuuu uuuu
PR3/CA1H	17h	xxxx xxxx	uuuu uuuu	uuuu uuuu
Bank 3				
PW1DCL	10h	xx-- ----	uu-- ----	uu-- ----
PW2DCL	11h	xx-- ----	uu-- ----	uu-- ----
PW1DCH	12h	xxxx xxxx	uuuu uuuu	uuuu uuuu
PW2DCH	13h	xxxx xxxx	uuuu uuuu	uuuu uuuu
CA2L	14h	xxxx xxxx	uuuu uuuu	uuuu uuuu
CA2H	15h	xxxx xxxx	uuuu uuuu	uuuu uuuu
TCON1	16h	0000 0000	0000 0000	uuuu uuuu
TCON2	17h	0000 0000	0000 0000	uuuu uuuu
Unbanked				
PRODL ⁽⁵⁾	18h	xxxx xxxx	uuuu uuuu	uuuu uuuu
PRODH ⁽⁵⁾	19h	xxxx xxxx	uuuu uuuu	uuuu uuuu

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

Note 1: One or more bits in INTSTA, PIR will be affected (to cause wake-up).

- 2: When the wake-up is due to an interrupt and the GLINTD bit is cleared, the PC is loaded with the interrupt vector.
- 3: See Table 4-3 for reset value of specific condition.
- 4: Only applies to the PIC17C42.
- 5: Does not apply to the PIC17C42.

5.1 Interrupt Status Register (INTSTA)

The Interrupt Status/Control register (INTSTA) records the individual interrupt requests in flag bits, and contains the individual interrupt enable bits (not for the peripherals).

The PEIF bit is a read only, bit wise OR of all the peripheral flag bits in the PIR register (Figure 5-4).

Note: T0IF, INTF, T0CKIF, or PEIF will be set by the specified condition, even if the corresponding interrupt enable bit is clear (interrupt disabled) or the GLINTD bit is set (all interrupts disabled).

Care should be taken when clearing any of the INTSTA register enable bits when interrupts are enabled (GLINTD is clear). If any of the INTSTA flag bits (T0IF, INTF, T0CKIF, or PEIF) are set in the same instruction cycle as the corresponding interrupt enable bit is cleared, the device will vector to the reset address (0x00).

When disabling any of the INTSTA enable bits, the GLINTD bit should be set (disabled).

FIGURE 5-2: INTSTA REGISTER (ADDRESS: 07h, UNBANKED)

R - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0
PEIF	T0CKIF	T0IF	INTF	PEIE	T0CKIE	T0IE	INTE
bit7							bit0

R = Readable bit
W = Writable bit
- n = Value at POR reset

bit 7: **PEIF:** Peripheral Interrupt Flag bit
This bit is the OR of all peripheral interrupt flag bits AND'ed with their corresponding enable bits.
1 = A peripheral interrupt is pending
0 = No peripheral interrupt is pending

bit 6: **T0CKIF:** External Interrupt on T0CKI Pin Flag bit
This bit is cleared by hardware, when the interrupt logic forces program execution to vector (18h).
1 = The software specified edge occurred on the RA1/T0CKI pin
0 = The software specified edge did not occur on the RA1/T0CKI pin

bit 5: **T0IF:** TMR0 Overflow Interrupt Flag bit
This bit is cleared by hardware, when the interrupt logic forces program execution to vector (10h).
1 = TMR0 overflowed
0 = TMR0 did not overflow

bit 4: **INTF:** External Interrupt on INT Pin Flag bit
This bit is cleared by hardware, when the interrupt logic forces program execution to vector (08h).
1 = The software specified edge occurred on the RA0/INT pin
0 = The software specified edge did not occur on the RA0/INT pin

bit 3: **PEIE:** Peripheral Interrupt Enable bit
This bit enables all peripheral interrupts that have their corresponding enable bits set.
1 = Enable peripheral interrupts
0 = Disable peripheral interrupts

bit 2: **T0CKIE:** External Interrupt on T0CKI Pin Enable bit
1 = Enable software specified edge interrupt on the RA1/T0CKI pin
0 = Disable interrupt on the RA1/T0CKI pin

bit 1: **T0IE:** TMR0 Overflow Interrupt Enable bit
1 = Enable TMR0 overflow interrupt
0 = Disable TMR0 overflow interrupt

bit 0: **INTE:** External Interrupt on RA0/INT Pin Enable bit
1 = Enable software specified edge interrupt on the RA0/INT pin
0 = Disable software specified edge interrupt on the RA0/INT pin

TABLE 6-3: SPECIAL FUNCTION REGISTERS

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 (3)
Unbanked											
00h	INDF0	Uses contents of FSR0 to address data memory (not a physical register)								---- --	---- --
01h	FSR0	Indirect data memory address pointer 0								xxxx xxxx	uuuu uuuu
02h	PCL	Low order 8-bits of PC								0000 0000	0000 0000
03h ⁽¹⁾	PCLATH	Holding register for upper 8-bits of PC								0000 0000	uuuu uuuu
04h	ALUSTA	FS3	FS2	FS1	FS0	OV	Z	DC	C	1111 xxxx	1111 uuuu
05h	T0STA	INTEDG	T0SE	T0CS	PS3	PS2	PS1	PS0	—	0000 000-	0000 000-
06h ⁽²⁾	CPUSTA	—	—	STKAV	GLINTD	T0	PD	—	—	--11 11--	--11 qq--
07h	INTSTA	PEIF	T0CKIF	T0IF	INTF	PEIE	T0CKIE	T0IE	INTE	0000 0000	0000 0000
08h	INDF1	Uses contents of FSR1 to address data memory (not a physical register)								---- --	---- --
09h	FSR1	Indirect data memory address pointer 1								xxxx xxxx	uuuu uuuu
0Ah	WREG	Working register								xxxx xxxx	uuuu uuuu
0Bh	TMR0L	TMR0 register; low byte								xxxx xxxx	uuuu uuuu
0Ch	TMR0H	TMR0 register; high byte								xxxx xxxx	uuuu uuuu
0Dh	TBLPTRL	Low byte of program memory table pointer								(4)	(4)
0Eh	TBLPTRH	High byte of program memory table pointer								(4)	(4)
0Fh	BSR	Bank select register								0000 0000	0000 0000
Bank 0											
10h	PORTA	RBP0	—	RA5	RA4	RA3	RA2	RA1/T0CKI	RA0/INT	0-xx xxxx	0-uu uuuu
11h	DDRB	Data direction register for PORTB								1111 1111	1111 1111
12h	PORTB	PORTB data latch								xxxx xxxx	uuuu uuuu
13h	RCSTA	SPEN	RX9	SREN	CREN	—	FERR	OERR	RX9D	0000 -00x	0000 -00u
14h	RCREG	Serial port receive register								xxxx xxxx	uuuu uuuu
15h	TXSTA	CSRC	TX9	TXEN	SYNC	—	—	TRMT	TX9D	0000 --1x	0000 --1u
16h	TXREG	Serial port transmit register								xxxx xxxx	uuuu uuuu
17h	SPBRG	Baud rate generator register								xxxx xxxx	uuuu uuuu
Bank 1											
10h	DDRC	Data direction register for PORTC								1111 1111	1111 1111
11h	PORTC	RC7/AD7	RC6/AD6	RC5/AD5	RC4/AD4	RC3/AD3	RC2/AD2	RC1/AD1	RC0/AD0	xxxx xxxx	uuuu uuuu
12h	DDRD	Data direction register for PORTD								1111 1111	1111 1111
13h	PORTD	RD7/AD15	RD6/AD14	RD5/AD13	RD4/AD12	RD3/AD11	RD2/AD10	RD1/AD9	RD0/AD8	xxxx xxxx	uuuu uuuu
14h	DDRE	Data direction register for PORTE								---- -111	---- -111
15h	PORTE	—	—	—	—	—	RE2/W \overline{R}	RE1/O \overline{E}	RE0/ALE	---- -xxx	---- -uuu
16h	PIR	RBIF	TMR3IF	TMR2IF	TMR1IF	CA2IF	CA1IF	TXIF	RCIF	0000 0010	0000 0010
17h	PIE	RBIE	TMR3IE	TMR2IE	TMR1IE	CA2IE	CA1IE	TXIE	RCIE	0000 0000	0000 0000

Legend: x = unknown, u = unchanged, - = unimplemented read as '0', q - value depends on condition. Shaded cells are unimplemented, read as '0'.
 Note 1: The upper byte of the program counter is not directly accessible. PCLATH is a holding register for PC<15:8> whose contents are updated from or transferred to the upper byte of the program counter.

- 2: The T0 and PD status bits in CPUSTA are not affected by a MCLR reset.
- 3: Other (non power-up) resets include: external reset through MCLR and the Watchdog Timer Reset.
- 4: The following values are for both TBLPTRL and TBLPTRH:
 All PIC17C4X devices (Power-on Reset 0000 0000) and (All other resets 0000 0000)
 except the PIC17C42 (Power-on Reset xxxx xxxx) and (All other resets uuuu uuuu)
- 5: The PRODL and PRODH registers are not implemented on the PIC17C42.

8.0 HARDWARE MULTIPLIER

All PIC17C4X devices except the PIC17C42, have an 8 x 8 hardware multiplier included in the ALU of the device. By making the multiply a hardware operation, it completes in a single instruction cycle. This is an unsigned multiply that gives a 16-bit result. The result is stored into the 16-bit PRODUct register (PRODH:PRODL). The multiplier does not affect any flags in the ALUSTA register.

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

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

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

Table 8-1 shows a performance comparison between the PIC17C42 and all other PIC17CXX devices, which have the single cycle hardware multiply.

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

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

EXAMPLE 8-1: 8 x 8 MULTIPLY ROUTINE

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

EXAMPLE 8-2: 8 x 8 SIGNED MULTIPLY ROUTINE

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

BTFSC    ARG2, SB      ; Test Sign Bit
SUBWF    PRODH, F       ; PRODH = PRODH
                        ; - ARG1

MOVFP    ARG2, WREG
BTFSC    ARG1, SB      ; Test Sign Bit
SUBWF    PRODH, F       ; PRODH = PRODH
                        ; - ARG2
```

TABLE 8-1: PERFORMANCE COMPARISON

Routine	Device	Program Memory (Words)	Cycles (Max)	Time	
				@ 25 MHz	@ 33 MHz
8 x 8 unsigned	PIC17C42	13	69	11.04 µs	N/A
	All other PIC17CXX devices	1	1	160 ns	121 ns
8 x 8 signed	PIC17C42	—	—	—	N/A
	All other PIC17CXX devices	6	6	960 ns	727 ns
16 x 16 unsigned	PIC17C42	21	242	38.72 µs	N/A
	All other PIC17CXX devices	24	24	3.84 µs	2.91 µs
16 x 16 signed	PIC17C42	52	254	40.64 µs	N/A
	All other PIC17CXX devices	36	36	5.76 µs	4.36 µs

TABLE 9-7: PORTD FUNCTIONS

Name	Bit	Buffer Type	Function
RD0/AD8	bit0	TTL	Input/Output or system bus address/data pin.
RD1/AD9	bit1	TTL	Input/Output or system bus address/data pin.
RD2/AD10	bit2	TTL	Input/Output or system bus address/data pin.
RD3/AD11	bit3	TTL	Input/Output or system bus address/data pin.
RD4/AD12	bit4	TTL	Input/Output or system bus address/data pin.
RD5/AD13	bit5	TTL	Input/Output or system bus address/data pin.
RD6/AD14	bit6	TTL	Input/Output or system bus address/data pin.
RD7/AD15	bit7	TTL	Input/Output or system bus address/data pin.

Legend: TTL = TTL input.

TABLE 9-8: REGISTERS/BITS ASSOCIATED WITH PORTD

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)
13h, Bank 1	PORTD	RD7/ AD15	RD6/ AD14	RD5/ AD13	RD4/ AD12	RD3/ AD11	RD2/ AD10	RD1/ AD9	RD0/ AD8	xxxx xxxx	uuuu uuuu
12h, Bank 1	DDRD	Data direction register for PORTD								1111 1111	1111 1111

Legend: x = unknown, u = unchanged.

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

12.2.1 ONE CAPTURE AND ONE PERIOD REGISTER MODE

In this mode registers PR3H/CA1H and PR3L/CA1L constitute a 16-bit period register. A block diagram is shown in Figure 12-7. The timer increments until it equals the period register and then resets to 0000h. TMR3 Interrupt Flag bit (TMR3IF) is set at this point. This interrupt can be disabled by clearing the TMR3 Interrupt Enable bit (TMR3IE). TMR3IF must be cleared in software.

This mode is selected if control bit CA1/PR3 is clear. In this mode, the Capture1 register, consisting of high byte (PR3H/CA1H) and low byte (PR3L/CA1L), is configured as the period control register for TMR3. Capture1 is disabled in this mode, and the corresponding Interrupt bit CA1IF is never set. TMR3 increments until it equals the value in the period register and then resets to 0000h.

Capture2 is active in this mode. The CA2ED1 and CA2ED0 bits determine the event on which capture will occur. The possible events are:

- Capture on every falling edge
- Capture on every rising edge
- Capture every 4th rising edge
- Capture every 16th rising edge

When a capture takes place, an interrupt flag is latched into the CA2IF bit. This interrupt can be enabled by setting the corresponding mask bit CA2IE. The Peripheral Interrupt Enable bit (PEIE) must be set and the Global Interrupt Disable bit (GLINTD) must be cleared for the interrupt to be acknowledged. The CA2IF interrupt flag bit must be cleared in software.

When the capture prescale select is changed, the prescaler is not reset and an event may be generated. Therefore, the first capture after such a change will be ambiguous. However, it sets the time-base for the next capture. The prescaler is reset upon chip reset.

Capture pin RB1/CAP2 is a multiplexed pin. When used as a port pin, Capture2 is not disabled. However, the user can simply disable the Capture2 interrupt by clearing CA2IE. If RB1/CAP2 is used as an output pin, the user can activate a capture by writing to the port pin. This may be useful during development phase to emulate a capture interrupt.

The input on capture pin RB1/CAP2 is synchronized internally to internal phase clocks. This imposes certain restrictions on the input waveform (see the Electrical Specification section for timing).

The Capture2 overflow status flag bit is double buffered. The master bit is set if one captured word is already residing in the Capture2 register and another "event" has occurred on the RB1/CA2 pin. The new event will not transfer the Timer3 value to the capture register, protecting the previous unread capture value. When the user reads both the high and the low bytes (in any order) of the Capture2 register, the master overflow bit is transferred to the slave overflow bit (CA2OVF) and then the master bit is reset. The user can then read TCON2 to determine the value of CA2OVF.

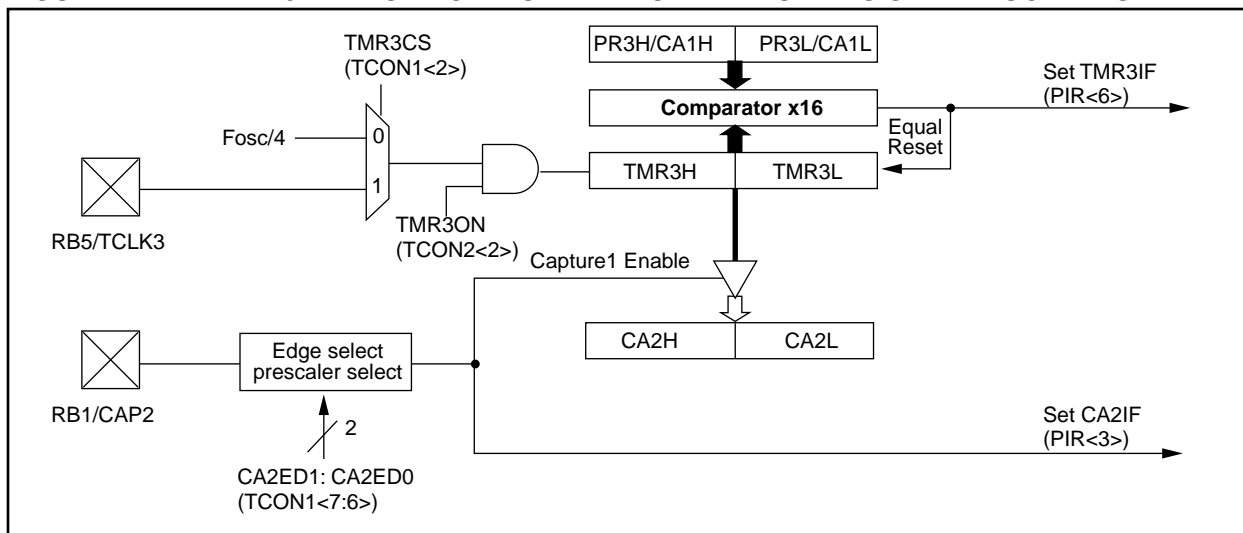
The recommended sequence to read capture registers and capture overflow flag bits is shown in Example 12-1.

EXAMPLE 12-1: SEQUENCE TO READ CAPTURE REGISTERS

```

MOVLB 3           ;Select Bank 3
MOVPF CA2L,LO_BYTE ;Read Capture2 low
                  ;byte, store in LO_BYTE
MOVPF CA2H,HI_BYTE ;Read Capture2 high
                  ;byte, store in HI_BYTE
MOVPF TCON2,STAT_VAL ;Read TCON2 into file
                  ;STAT_VAL
    
```

FIGURE 12-7: TIMER3 WITH ONE CAPTURE AND ONE PERIOD REGISTER BLOCK DIAGRAM



13.3 USART Synchronous Master Mode

In Master Synchronous mode, the data is transmitted in a half-duplex manner; i.e. transmission and reception do not occur at the same time: when transmitting data, the reception is inhibited and vice versa. The synchronous mode is entered by setting the SYNC (TXSTA<4>) bit. In addition, the SPEN (RCSTA<7>) bit is set in order to configure the RA5 and RA4 I/O ports to CK (clock) and DT (data) lines respectively. The Master mode indicates that the processor transmits the master clock on the CK line. The Master mode is entered by setting the CSRC (TXSTA<7>) bit.

13.3.1 USART SYNCHRONOUS MASTER TRANSMISSION

The USART transmitter block diagram is shown in Figure 13-3. The heart of the transmitter is the transmit (serial) shift register (TSR). The shift register obtains its data from the read/write transmit buffer TXREG. TXREG is loaded with data in software. The TSR is not loaded until the last bit has been transmitted from the previous load. As soon as the last bit is transmitted, the TSR is loaded with new data from TXREG (if available). Once TXREG transfers the data to the TSR (occurs in one Tcy at the end of the current BRG cycle), TXREG is empty and the TXIF (PIR<1>) bit is set. This interrupt can be enabled/disabled by setting/clearing the TXIE bit (PIE<1>). TXIF will be set regardless of the state of bit TXIE and cannot be cleared in software. It will reset only when new data is loaded into TXREG. While TXIF indicates the status of TXREG, TRMT (TXSTA<1>) shows the status of the TSR. TRMT is a read only bit which is set when the TSR 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 is empty. The TSR is not mapped in data memory, so it is not available to the user.

Transmission is enabled by setting the TXEN (TXSTA<5>) bit. The actual transmission will not occur until TXREG has been loaded with data. The first data bit will be shifted out on the next available rising edge of the clock on the RA5/TX/CK pin. Data out is stable around the falling edge of the synchronous clock (Figure 13-10). The transmission can also be started by first loading TXREG and then setting TXEN. This is advantageous when slow baud rates are selected, since BRG is kept in RESET when the TXEN, CREN, and SREN bits are clear. Setting the TXEN bit will start the BRG, creating a shift clock immediately. Normally when transmission is first started, the TSR is empty, so a transfer to TXREG will result in an immediate transfer to the TSR, resulting in an empty TXREG. Back-to-back transfers are possible.

Clearing TXEN during a transmission will cause the transmission to be aborted and will reset the transmitter. The RA4/RX/DT and RA5/TX/CK pins will revert to hi-impedance. If either CREN or SREN are set during a transmission, the transmission is aborted and the

RA4/RX/DT pin reverts to a hi-impedance state (for a reception). The RA5/TX/CK pin will remain an output if the CSRC bit is set (internal clock). The transmitter logic is not reset, although it is disconnected from the pins. In order to reset the transmitter, the user has to clear the TXEN bit. If the SREN bit is set (to interrupt an ongoing transmission and receive a single word), then after the single word is received, SREN will be cleared and the serial port will revert back to transmitting, since the TXEN bit is still set. The DT line will immediately switch from hi-impedance receive mode to transmit and start driving. To avoid this, TXEN should be cleared.

In order to select 9-bit transmission, the TX9 (TXSTA<6>) bit 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 TXREG. This is because a data write to TXREG can result in an immediate transfer of the data to the TSR (if the TSR is empty). If the TSR was empty and TXREG was written before writing the "new" TX9D, the "present" value of TX9D is loaded.

Steps to follow when setting up a Synchronous Master Transmission:

1. Initialize the SPBRG register for the appropriate baud rate (see Baud Rate Generator Section for details).
2. Enable the synchronous master serial port by setting the SYNC, SPEN, and CSRC bits.
3. Ensure that the CREN and SREN bits are clear (these bits override transmission when set).
4. If interrupts are desired, then set the TXIE bit (the GLINTD bit must be clear and the PEIE bit must be set).
5. If 9-bit transmission is desired, then set the TX9 bit.
6. Start transmission by loading data to the TXREG register.
7. If 9-bit transmission is selected, the ninth bit should be loaded in TX9D.
8. Enable the transmission by setting TXEN.

Writing the transmit data to the TXREG, then enabling the transmit (setting TXEN) allows transmission to start sooner than doing these two events in the reverse order.

Note: To terminate a transmission, either clear the SPEN bit, or the TXEN bit. This will reset the transmit logic, so that it will be in the proper state when transmit is re-enabled.

ANDWF

AND WREG with f

Syntax:

[/label] ANDWF f,d

Operands:

$0 \leq f \leq 255$

$d \in [0,1]$

Operation:

(WREG) .AND. (f) → (dest)

Status Affected:

Z

Encoding:

0000	101d	ffff	ffff
------	------	------	------

Description:

The contents of WREG are AND'ed with register 'f'. If 'd' is 0 the result is stored in WREG. If 'd' is 1 the result is stored back in register 'f'.

Words:

1

Cycles:

1

Q Cycle Activity:

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

Example: ANDWF REG, 1

Before Instruction
WREG = 0x17
REG = 0xC2
After Instruction
WREG = 0x17
REG = 0x02

BCF

Bit Clear f

Syntax:

[/label] BCF f,b

Operands:

$0 \leq f \leq 255$

$0 \leq b \leq 7$

Operation:

$0 \rightarrow (f)$

Status Affected:

None

Encoding:

1000	1bbb	ffff	ffff
------	------	------	------

Description:

Bit 'b' in register 'f' is cleared.

Words:

1

Cycles:

1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Execute	Write register 'f'

Example: BCF FLAG_REG, 7

Before Instruction
FLAG_REG = 0xC7
After Instruction
FLAG_REG = 0x47

CALL	Subroutine Call				
Syntax:	[<i>label</i>] 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:	<table><tr><td>111k</td><td>kkkk</td><td>kkkk</td><td>kkkk</td></tr></table>	111k	kkkk	kkkk	kkkk
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:	[<i>label</i>] CLRF f,s								
Operands:	$0 \leq f \leq 255$								
Operation:	00h \rightarrow f, s $\in [0,1]$ 00h \rightarrow dest								
Status Affected:	None								
Encoding:	<table border="1"><tr><td>0010</td><td>100s</td><td>ffff</td><td>ffff</td></tr></table>					0010	100s	ffff	ffff
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

SLEEP	Enter SLEEP mode				
Syntax:	[<i>label</i>] SLEEP				
Operands:	None				
Operation:	00h → WDT; 0 → WDT postscaler; 1 → \overline{TO} ; 0 → \overline{PD}				
Status Affected:	\overline{TO} , \overline{PD}				
Encoding:	<table><tr><td>0000</td><td>0000</td><td>0000</td><td>0011</td></tr></table>	0000	0000	0000	0011
0000	0000	0000	0011		
Description:	<p>The power down status bit (\overline{PD}) is cleared. The time-out status bit (\overline{TO}) is set. Watchdog Timer and its prescaler are cleared.</p> <p>The processor is put into SLEEP mode with the oscillator stopped.</p>				
Words:	1				
Cycles:	1				

Q Cycle Activity:	Q1	Q2	Q3	Q4
	Decode	Read register PCLATH	Execute	NOP

Example: SLEEP

Before Instruction

\overline{TO} = ?

\overline{PD} = ?

After Instruction

\overline{TO} = 1 †

\overline{PD} = 0

† If WDT causes wake-up, this bit is cleared

SUBLW	Subtract WREG from Literal				
Syntax:	[<i>label</i>] SUBLW k				
Operands:	$0 \leq k \leq 255$				
Operation:	$k - (WREG) \rightarrow (WREG)$				
Status Affected:	OV, C, DC, Z				
Encoding:	<table><tr><td>1011</td><td>0010</td><td>kkkk</td><td>kkkk</td></tr></table>	1011	0010	kkkk	kkkk
1011	0010	kkkk	kkkk		
Description:	WREG is subtracted from the eight bit literal 'k'. The result is placed in WREG.				
Words:	1				
Cycles:	1				

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Execute	Write to WREG

Example 1: SUBLW 0x02

Before Instruction

WREG = 1

C = ?

After Instruction

WREG = 1

C = 1 ; result is positive

Z = 0

Example 2:

Before Instruction

WREG = 2

C = ?

After Instruction

WREG = 0

C = 1 ; result is zero

Z = 1

Example 3:

Before Instruction

WREG = 3

C = ?

After Instruction

WREG = FF ; (2's complement)

C = 0 ; result is negative

Z = 1

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 18-2: TYPICAL RC OSCILLATOR FREQUENCY vs. VDD

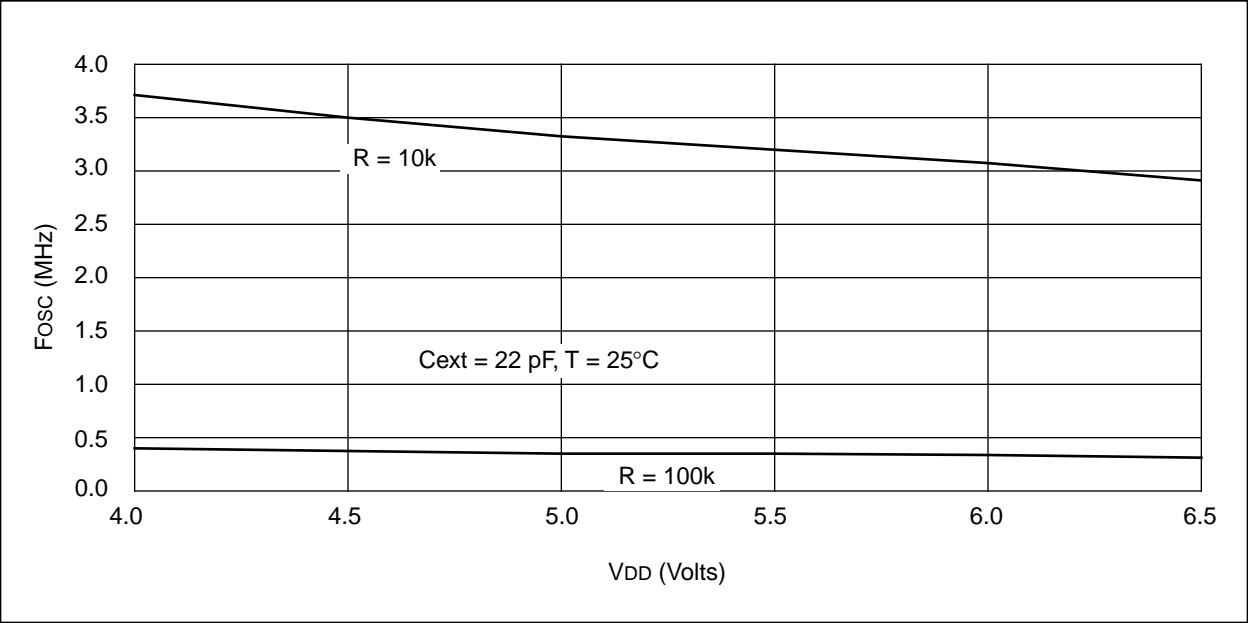


FIGURE 18-3: TYPICAL RC OSCILLATOR FREQUENCY vs. VDD

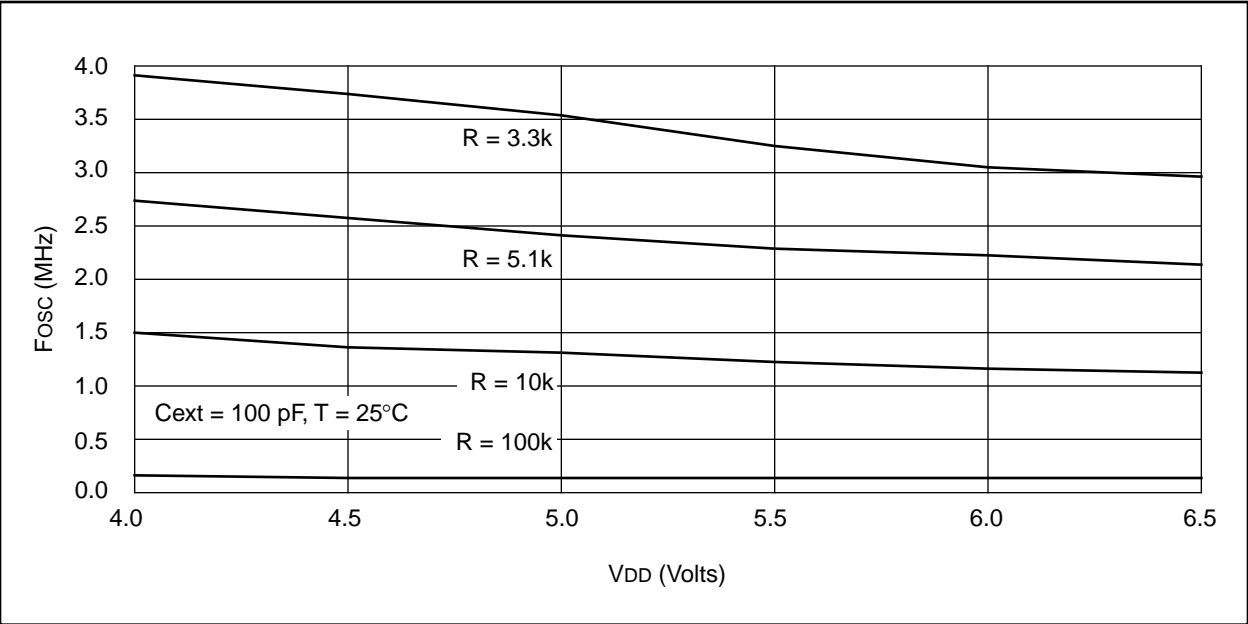


FIGURE 18-4: TYPICAL RC OSCILLATOR FREQUENCY vs. V_{DD}

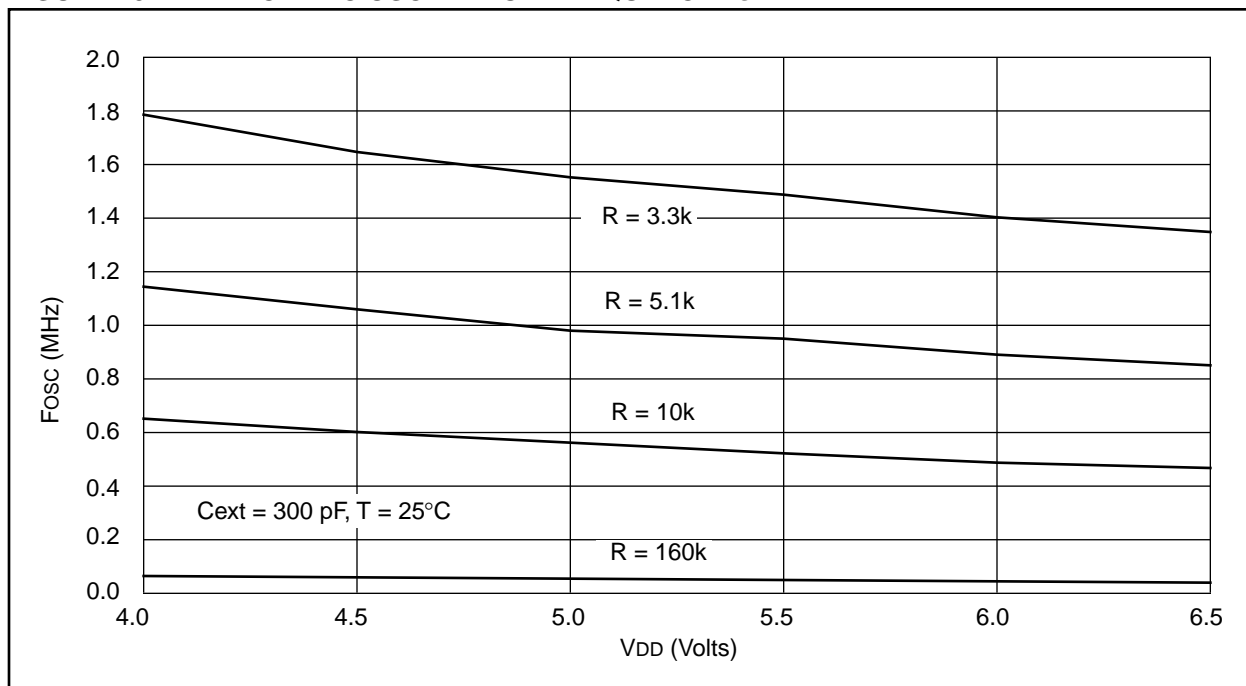


TABLE 18-2: RC OSCILLATOR FREQUENCIES

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%

FIGURE 20-4: TYPICAL RC OSCILLATOR FREQUENCY vs. V_{DD}

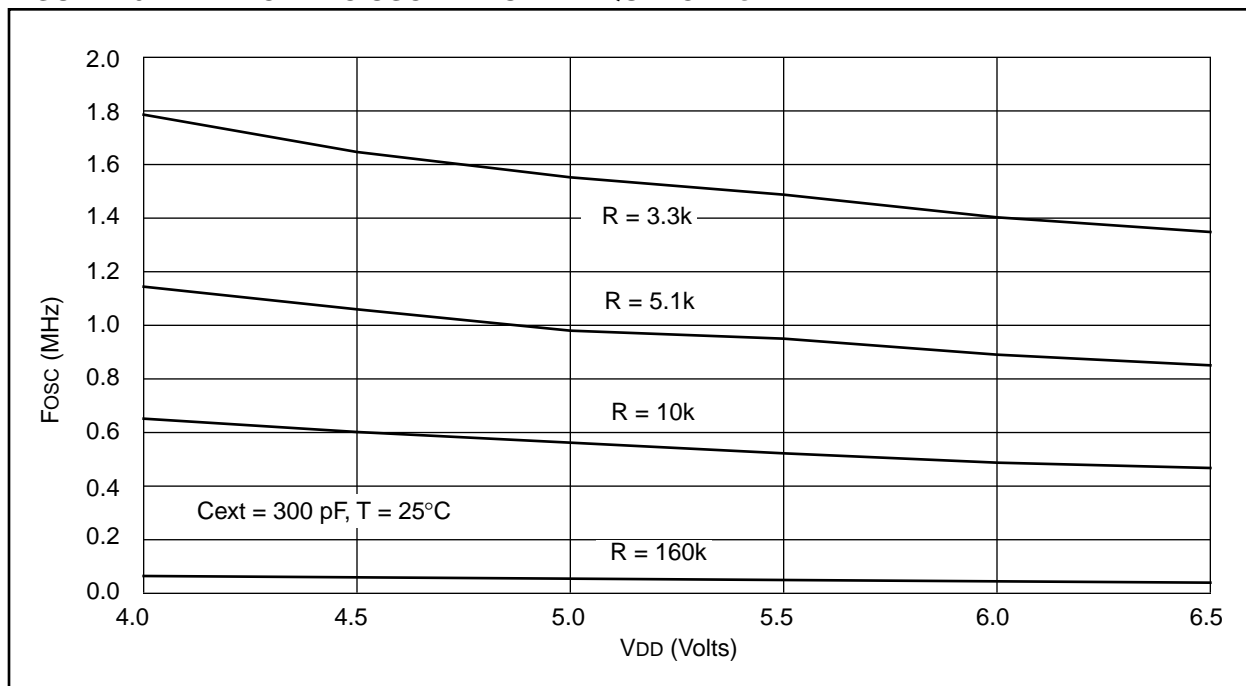


TABLE 20-2: RC OSCILLATOR FREQUENCIES

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%

PIC17C4X

Applicable Devices 42 R42 42A 43 R43 44

FIGURE 20-5: TRANSCONDUCTANCE (gm) OF LF OSCILLATOR vs. VDD

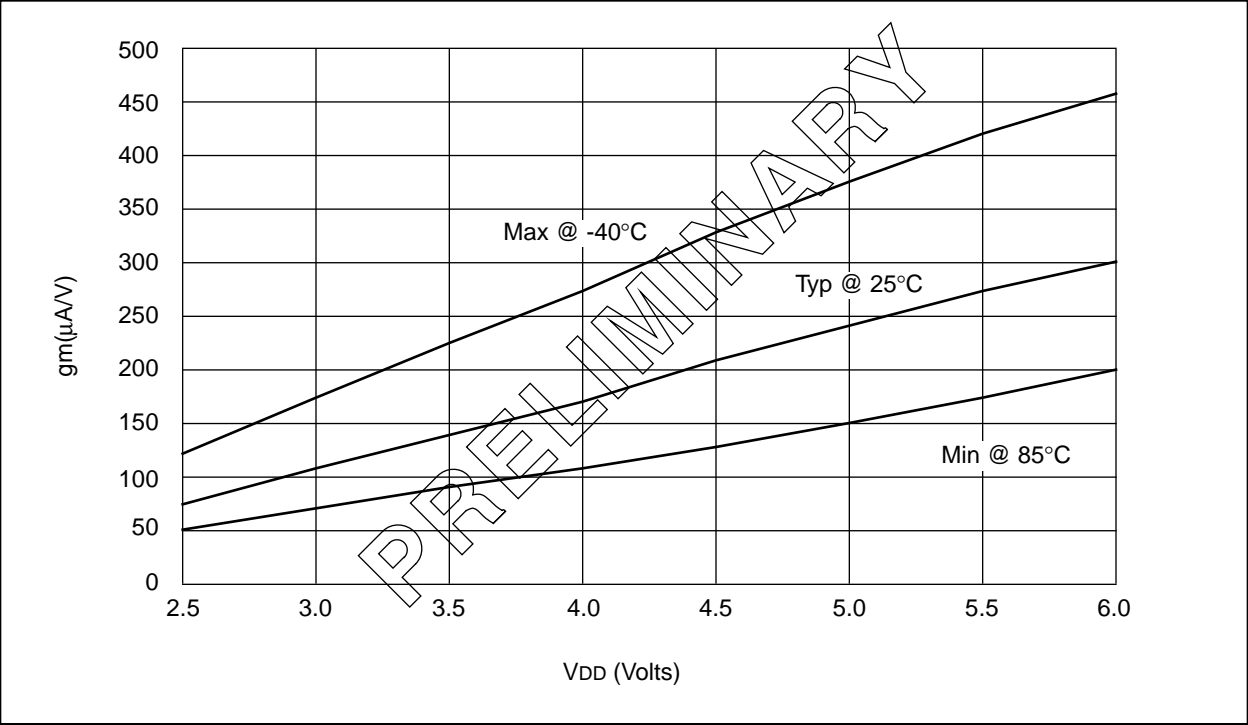


FIGURE 20-6: TRANSCONDUCTANCE (gm) OF XT OSCILLATOR vs. VDD

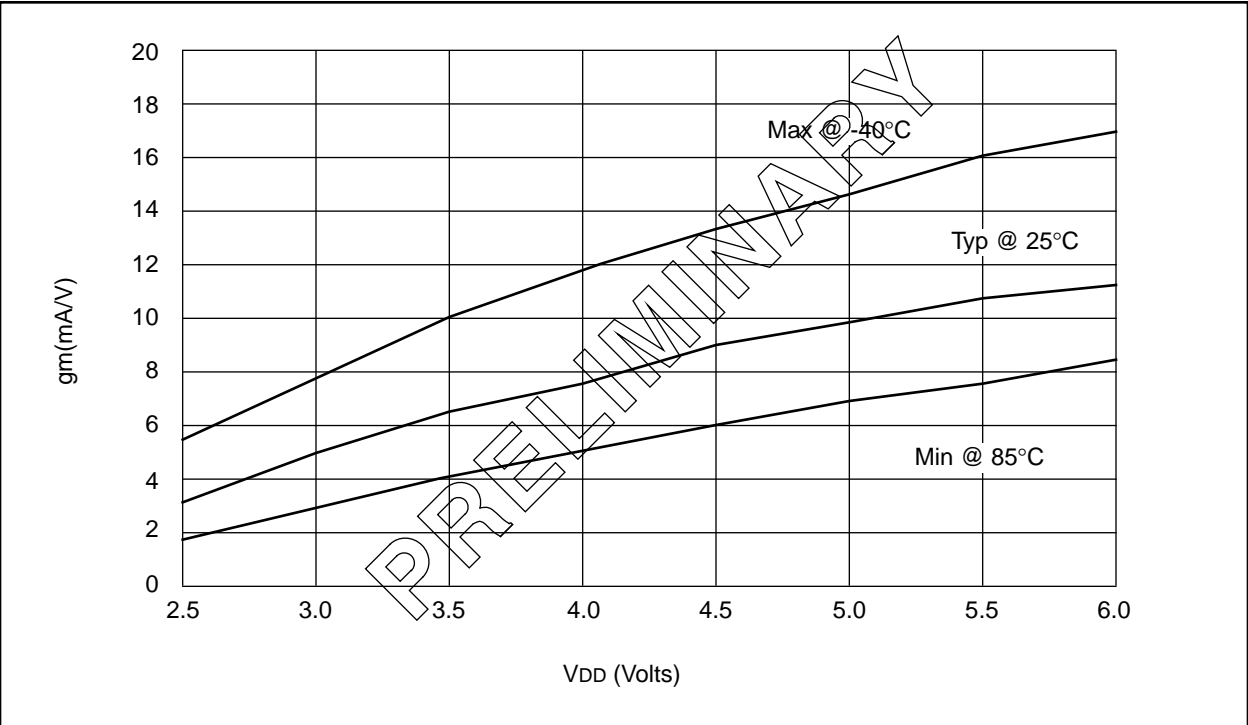


FIGURE 20-19: V_{IH} , V_{IL} of I/O PINS (SCHMITT TRIGGER) vs. V_{DD}

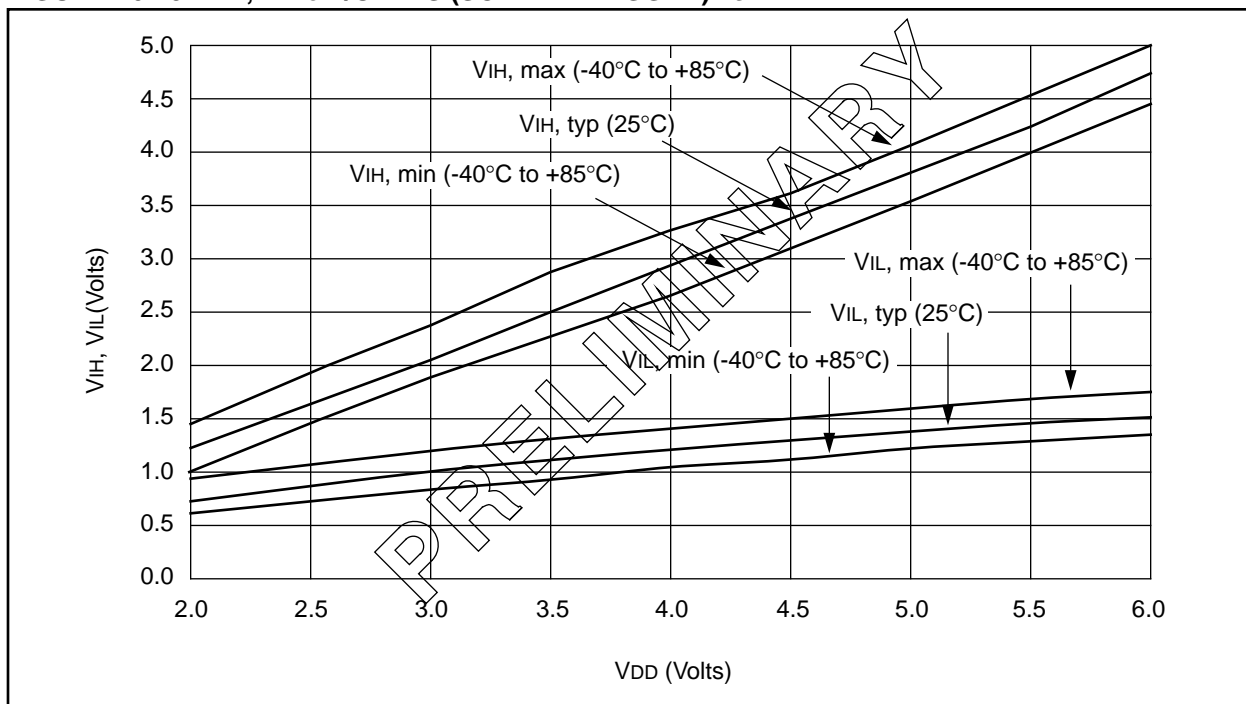
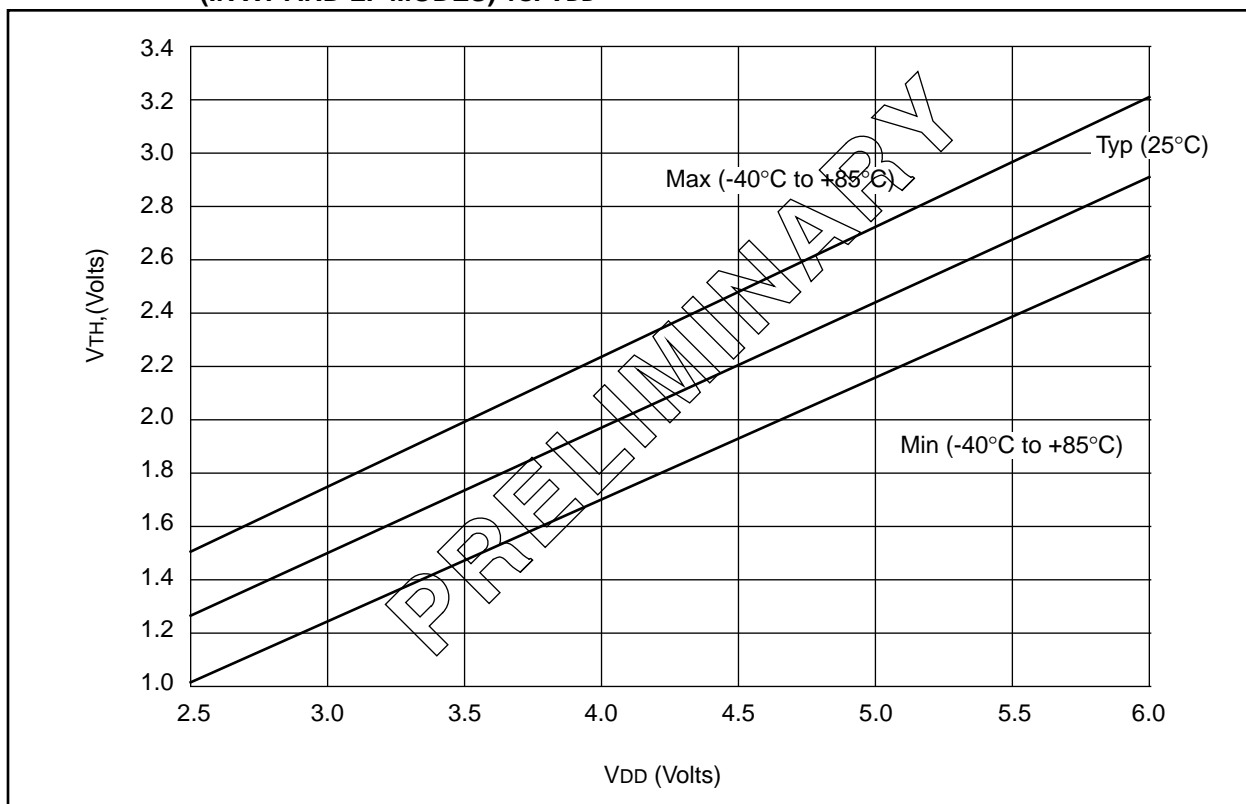


FIGURE 20-20: V_{TH} (INPUT THRESHOLD VOLTAGE) OF OSC1 INPUT (IN XT AND LF MODES) vs. V_{DD}



E.7 PIC16C9XX Family Of Devices

Clock		Memory		Peripherals					Features																				
PIC16C923	8	4K	176	Data Memory (bytes)		Program Memory		Timer Module(s)		Capture/Compare/PWM Module(s)		Serial Ports (SPI/I ² C, USART)		A/D Converter (8-bit) Channels		LCD Module		Interrupt Sources		I/O Pins		Voltage Range (Volts)		In-Circuit Serial Programming		Brown-out Reset		Packages	
				TMR0, TMR1, TMR2	TMR0, TMR1, TMR2	1	SPI/I ² C	—	—	4 Com 32 Seg	8	25	27	3.0-6.0	Yes	—	64-pin SDIP(1), TQFP, 68-pin PLCC, DIE												
PIC16C924	8	4K	176	TMR0, TMR1, TMR2	TMR0, TMR1, TMR2	1	SPI/I ² C	—	5	4 Com 32 Seg	9	25	27	3.0-6.0	Yes	—	64-pin SDIP(1), TQFP, 68-pin PLCC, DIE												

ON-LINE SUPPORT

Microchip provides two methods of on-line support. These are the Microchip BBS and the Microchip World Wide Web (WWW) site.

Use Microchip's Bulletin Board Service (BBS) to get current information and help about Microchip products. Microchip provides the BBS communication channel for you to use in extending your technical staff with micro-controller and memory experts.

To provide you with the most responsive service possible, the Microchip systems team monitors the BBS, posts the latest component data and software tool updates, provides technical help and embedded systems insights, and discusses how Microchip products provide project solutions.

The web site, like the BBS, is used by Microchip as a means to make files and information easily available to customers. To view the site, the user must have access to the Internet and a web browser, such as Netscape or Microsoft Explorer. Files are also available for FTP download from our FTP site.

Connecting to the Microchip Internet Web Site

The Microchip web site is available by using your favorite Internet browser to attach to:

www.microchip.com

The file transfer site is available by using an FTP service to connect to:

[ftp.mchip.com/biz/mchip](ftp://mchip.com/biz/mchip)

The web site and file transfer site provide a variety of services. Users may download files for the latest Development Tools, Data Sheets, Application Notes, User's Guides, Articles and Sample Programs. A variety of Microchip specific business information is also available, including listings of Microchip sales offices, distributors and factory representatives. Other data available for consideration is:

- Latest Microchip Press Releases
- Technical Support Section with Frequently Asked Questions
- Design Tips
- Device Errata
- Job Postings
- Microchip Consultant Program Member Listing
- Links to other useful web sites related to Microchip Products

Connecting to the Microchip BBS

Connect worldwide to the Microchip BBS using either the Internet or the CompuServe® communications network.

Internet:

You can telnet or ftp to the Microchip BBS at the address:

[mchipbbs.microchip.com](telnet://mchipbbs.microchip.com)

CompuServe Communications Network:

When using the BBS via the Compuserve Network, in most cases, a local call is your only expense. The Microchip BBS connection does not use CompuServe membership services, therefore you do not need CompuServe membership to join Microchip's BBS. There is no charge for connecting to the Microchip BBS.

The procedure to connect will vary slightly from country to country. Please check with your local CompuServe agent for details if you have a problem. CompuServe service allow multiple users various baud rates depending on the local point of access.

The following connect procedure applies in most locations.

1. Set your modem to 8-bit, No parity, and One stop (8N1). This is not the normal CompuServe setting which is 7E1.
2. Dial your local CompuServe access number.
3. Depress the <Enter> key and a garbage string will appear because CompuServe is expecting a 7E1 setting.
4. Type +, depress the <Enter> key and "Host Name:" will appear.
5. Type MCHIPBBS, depress the <Enter> key and you will be connected to the Microchip BBS.

In the United States, to find the CompuServe phone number closest to you, set your modem to 7E1 and dial (800) 848-4480 for 300-2400 baud or (800) 331-7166 for 9600-14400 baud connection. After the system responds with "Host Name:", type NETWORK, depress the <Enter> key and follow CompuServe's directions.

For voice information (or calling from overseas), you may call (614) 723-1550 for your local CompuServe number.

Microchip regularly uses the Microchip BBS to distribute technical information, application notes, source code, errata sheets, bug reports, and interim patches for Microchip systems software products. For each SIG, a moderator monitors, scans, and approves or disapproves files submitted to the SIG. No executable files are accepted from the user community in general to limit the spread of computer viruses.

Systems Information and Upgrade Hot Line

The Systems Information and Upgrade Line provides system users a listing of the latest versions of all of Microchip's development systems software products. Plus, this line provides information on how customers can receive any currently available upgrade kits. The Hot Line Numbers are:

1-800-755-2345 for U.S. and most of Canada, and

1-602-786-7302 for the rest of the world.

960513

Trademarks: The Microchip name, logo, PIC, PICSTART, PICMASTER and PRO MATE are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries. FlexROM, MPLAB and fuzzyLAB, are trademarks and SQTP is a service mark of Microchip in the U.S.A.

fuzzyTECH is a registered trademark of Inform Software Corporation. IBM, IBM PC-AT are registered trademarks of International Business Machines Corp. Pentium is a trademark of Intel Corporation. Windows is a trademark and MS-DOS, Microsoft Windows are registered trademarks of Microsoft Corporation. CompuServe is a registered trademark of CompuServe Incorporated.

All other trademarks mentioned herein are the property of their respective companies.

PIC17C4X

NOTES: