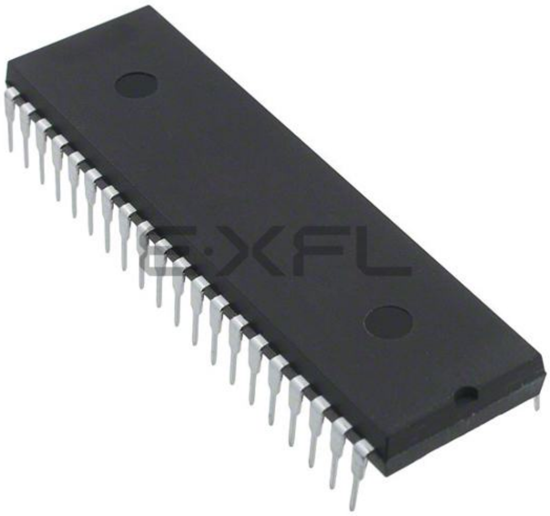


Welcome to E-XFL.COM

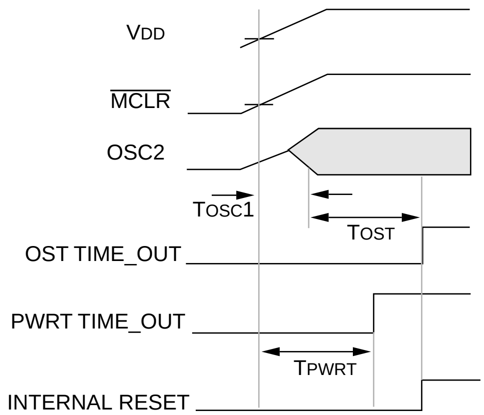
What is "Embedded - Microcontrollers"?

"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	25MHz
Connectivity	UART/USART
Peripherals	POR, PWM, WDT
Number of I/O	33
Program Memory Size	4KB (2K x 16)
Program Memory Type	OTP
EEPROM Size	-
RAM Size	232 x 8
Voltage - Supply (Vcc/Vdd)	4.5V ~ 6V
Data Converters	-
Oscillator Type	External
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Through Hole
Package / Case	40-DIP (0.600", 15.24mm)
Supplier Device Package	40-PDIP
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic17c42a-25e-p

FIGURE 4-5: OSCILLATOR START-UP TIME



This figure shows in greater detail the timings involved with the oscillator start-up timer. In this example the low frequency crystal start-up time is larger than power-up time (T_{PWRT}).
Tosc1 = time for the crystal oscillator to react to an oscillation level detectable by the Oscillator Start-up Timer (ost).
$T_{OST} = 1024T_{OSC}$.

FIGURE 4-6: USING ON-CHIP POR

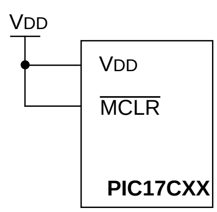
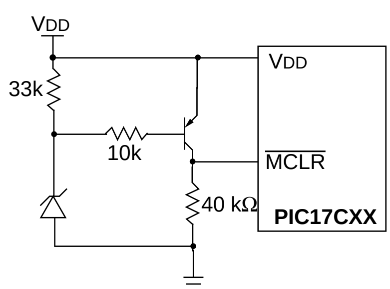
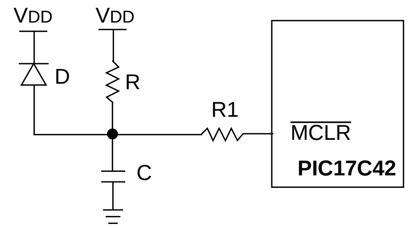


FIGURE 4-7: BROWN-OUT PROTECTION CIRCUIT 1



This circuit will activate reset when V_{DD} goes below (V_Z + 0.7V) where V_Z = Zener voltage.

FIGURE 4-8: PIC17C42 EXTERNAL POWER-ON RESET CIRCUIT (FOR SLOW V_{DD} POWER-UP)

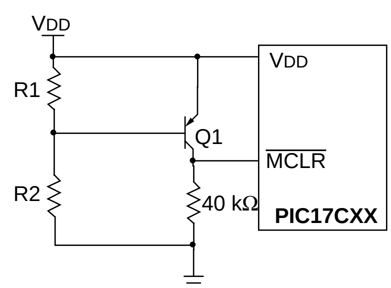


Note 1: An external Power-on Reset circuit is required only if V_{DD} power-up time is too slow. The diode D helps discharge the capacitor quickly when V_{DD} powers down.

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

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

FIGURE 4-9: BROWN-OUT PROTECTION CIRCUIT 2



This brown-out circuit is less expensive, albeit less accurate. Transistor Q1 turns off when V_{DD} is below a certain level such that:

$$V_{DD} \bullet \frac{R1}{R1 + R2} = 0.7V$$

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	$\overline{TO}$	$\overline{PD}$	—	—	--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	$\overline{RBPU}$	—	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/$\overline{WR}$	RE1/$\overline{OE}$	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 $\overline{TO}$ and $\overline{PD}$ 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.

6.2.2.3 TMR0 STATUS/CONTROL REGISTER (T0STA)

This register contains various control bits. Bit7 (INTEDG) is used to control the edge upon which a signal on the RA0/INT pin will set the RB0/INT interrupt flag. The other bits configure the Timer0 prescaler and clock source. (Figure 11-1).

FIGURE 6-9: T0STA REGISTER (ADDRESS: 05h, UNBANKED)

R/W - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0	U - 0
INTEDG	T0SE	T0CS	PS3	PS2	PS1	PS0	—
bit7							bit0

R = Readable bit
W = Writable bit
U = Unimplemented, reads as ‘0’
-n = Value at POR reset

bit 7: **INTEDG**: RA0/INT Pin Interrupt Edge Select bit
This bit selects the edge upon which the interrupt is detected.
1 = Rising edge of RA0/INT pin generates interrupt
0 = Falling edge of RA0/INT pin generates interrupt

bit 6: **T0SE**: Timer0 Clock Input Edge Select bit
This bit selects the edge upon which TMR0 will increment.
When T0CS = 0
1 = Rising edge of RA1/T0CKI pin increments TMR0 and/or generates a T0CKIF interrupt
0 = Falling edge of RA1/T0CKI pin increments TMR0 and/or generates a T0CKIF interrupt
When T0CS = 1
Don’t care

bit 5: **T0CS**: Timer0 Clock Source Select bit
This bit selects the clock source for Timer0.
1 = Internal instruction clock cycle (T_{CY})
0 = T0CKI pin

bit 4-1: **PS3:PS0**: Timer0 Prescale Selection bits
These bits select the prescale value for Timer0.

PS3:PS0	Prescale Value
0000	1:1
0001	1:2
0010	1:4
0011	1:8
0100	1:16
0101	1:32
0110	1:64
0111	1:128
1xxx	1:256

bit 0: **Unimplemented**: Read as '0'

12.1.3.1 PWM PERIODS

The period of the PWM1 output is determined by Timer1 and its period register (PR1). The period of the PWM2 output can be software configured to use either Timer1 or Timer2 as the time-base. When TM2PW2 bit (PW2DCL<5>) is clear, the time-base is determined by TMR1 and PR1. When TM2PW2 is set, the time-base is determined by Timer2 and PR2.

Running two different PWM outputs on two different timers allows different PWM periods. Running both PWMs from Timer1 allows the best use of resources by freeing Timer2 to operate as an 8-bit timer. Timer1 and Timer2 can not be used as a 16-bit timer if either PWM is being used.

The PWM periods can be calculated as follows:

period of PWM1 =[(PR1) + 1] x 4TOSC

period of PWM2 =[(PR1) + 1] x 4TOSC or
[(PR2) + 1] x 4TOSC

The duty cycle of PWMx is determined by the 10-bit value DCx<9:0>. The upper 8-bits are from register PWxDCH and the lower 2-bits are from PWxDCL<7:6> (PWxDCH:PWxDCL<7:6>). Table 12-3 shows the maximum PWM frequency (FPWM) given the value in the period register.

The number of bits of resolution that the PWM can achieve depends on the operation frequency of the device as well as the PWM frequency (FPWM).

Maximum PWM resolution (bits) for a given PWM frequency:

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

The PWMx duty cycle is as follows:

PWMx Duty Cycle = (DCx) x TOSC

where DCx represents the 10-bit value from PWxDCH:PWxDCL.

If DCx = 0, then the duty cycle is zero. If PRx = PWxDCH, then the PWM output will be low for one to four Q-clock (depending on the state of the PWxDCL<7:6> bits). For a Duty Cycle to be 100%, the PWxDCH value must be greater then the PRx value.

The duty cycle registers for both PWM outputs are double buffered. When the user writes to these registers, they are stored in master latches. When TMR1 (or TMR2) overflows and a new PWM period begins, the master latch values are transferred to the slave latches and the PWMx pin is forced high.

Note:	For PW1DCH, PW1DCL, PW2DCH and PW2DCL registers, a write operation writes to the "master latches" while a read operation reads the "slave latches". As a result, the user may not read back what was just written to the duty cycle registers.

The user should also avoid any "read-modify-write" operations on the duty cycle registers, such as: `ADDWF PW1DCH`. This may cause duty cycle outputs that are unpredictable.

TABLE 12-3: PWM FREQUENCY vs. RESOLUTION AT 25 MHz

PWM Frequency	Frequency (kHz)				
	24.4	48.8	65.104	97.66	390.6
PRx Value	0xFF	0x7F	0x5F	0x3F	0x0F
High Resolution	10-bit	9-bit	8.5-bit	8-bit	6-bit
Standard Resolution	8-bit	7-bit	6.5-bit	6-bit	4-bit

12.1.3.2 PWM INTERRUPTS

The PWM module makes use of TMR1 or TMR2 interrupts. A timer interrupt is generated when TMR1 or TMR2 equals its period register and is cleared to zero. This interrupt also marks the beginning of a PWM cycle. The user can write new duty cycle values before the timer roll-over. The TMR1 interrupt is latched into the TMR1IF bit and the TMR2 interrupt is latched into the TMR2IF bit. These flags must be cleared in software.

12.1.3.3 EXTERNAL CLOCK SOURCE

The PWMs will operate regardless of the clock source of the timer. The use of an external clock has ramifications that must be understood. Because the external TCLK12 input is synchronized internally (sampled once per instruction cycle), the time TCLK12 changes to the time the timer increments will vary by as much as TCY (one instruction cycle). This will cause jitter in the duty cycle as well as the period of the PWM output.

This jitter will be ±TCY, unless the external clock is synchronized with the processor clock. Use of one of the PWM outputs as the clock source to the TCLKx input, will supply a synchronized clock.

In general, when using an external clock source for PWM, its frequency should be much less than the device frequency (Fosc).

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 $CA1/\overline{PR3}$ 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

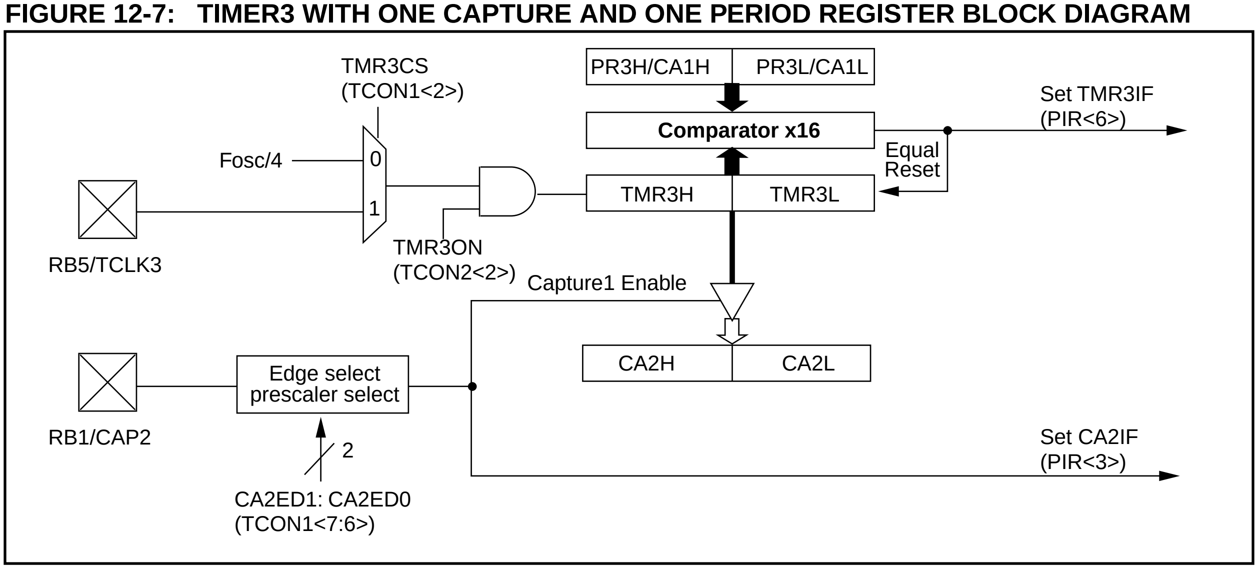


TABLE 13-7: REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER TRANSMISSION

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	TX7	TX6	TX5	TX4	TX3	TX2	TX1	TX0	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 transmission.

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

FIGURE 13-9: SYNCHRONOUS TRANSMISSION

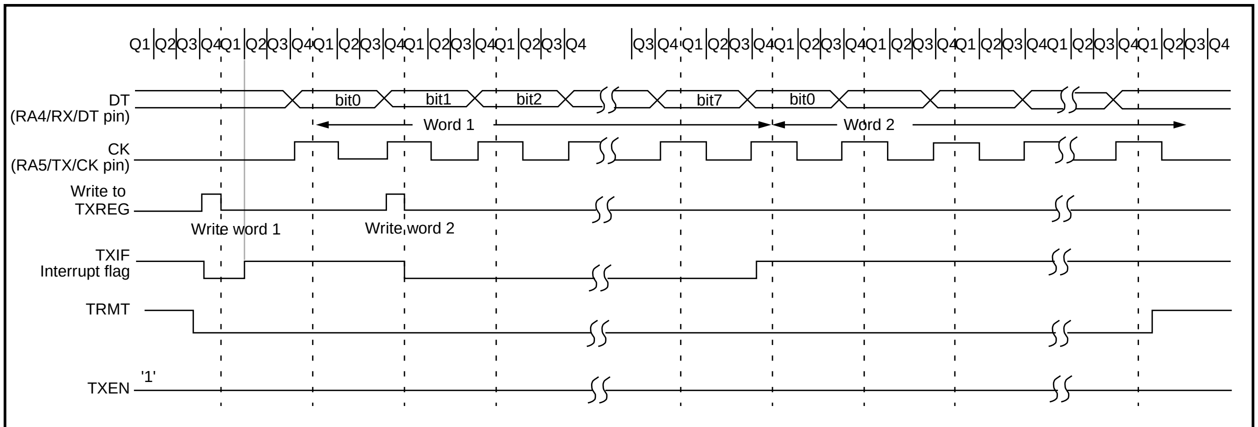
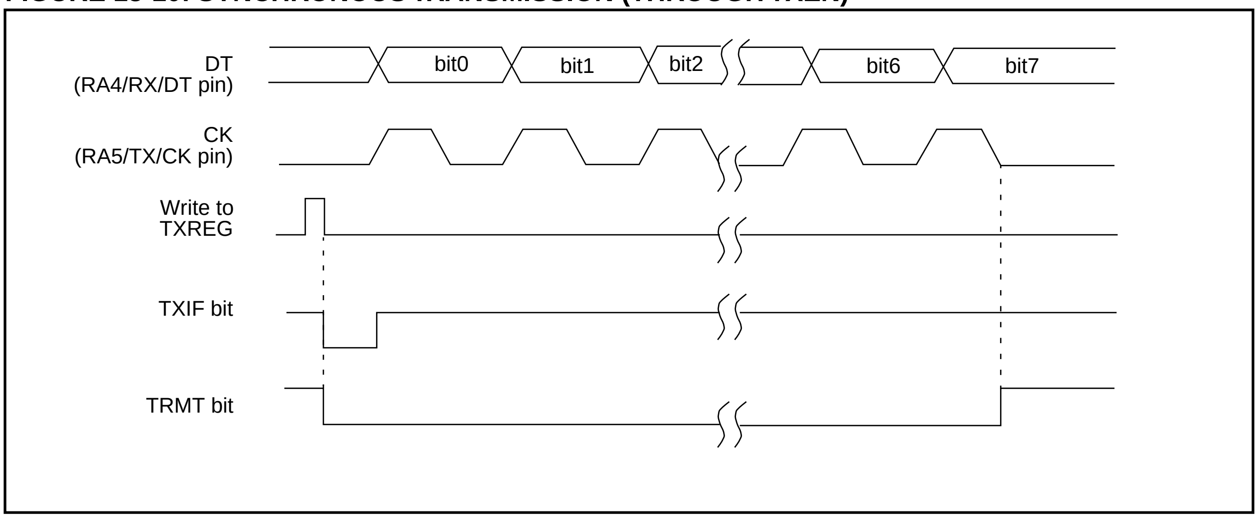


FIGURE 13-10: SYNCHRONOUS TRANSMISSION (THROUGH TXEN)



13.3.2 USART SYNCHRONOUS MASTER RECEPTION

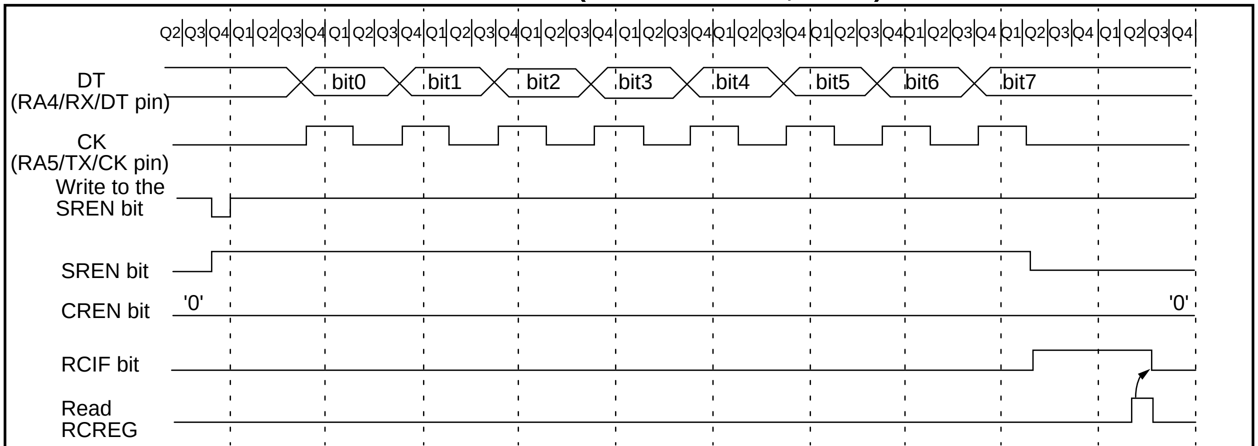
Once synchronous mode is selected, reception is enabled by setting either the SREN (RCSTA<5>) bit or the CREN (RCSTA<4>) bit. Data is sampled on the RA4/RX/DT pin on the falling edge of the clock. If SREN is set, then only a single word is received. If CREN is set, the reception is continuous until CREN is reset. If both bits are set, then CREN takes precedence. After clocking the last bit, the received data in the Receive Shift Register (RSR) is transferred to RCREG (if it is empty). If the transfer is complete, the interrupt bit RCIF (PIR<0>) is set. The actual interrupt can be enabled/disabled by setting/clearing the RCIE (PIE<0>) bit. RCIF is a read only bit which is RESET by the hardware. In this case it is reset when RCREG has been read and is empty. RCREG is a double buffered register; i.e., it is a two deep FIFO. It is possible for two bytes of data to be received and transferred to the RCREG FIFO and a third byte to begin shifting into the RSR. On the clocking of the last bit of the third byte, if RCREG is still full, then the overrun error bit OERR (RCSTA<1>) is set. The word in the RSR will be lost. RCREG can be read twice to retrieve the two bytes in the FIFO. The OERR bit has to be cleared in software. This is done by clearing the CREN bit. If OERR bit is set, transfers from RSR to RCREG are inhibited, so it is essential to clear OERR bit if it is set. The 9th receive bit is buffered the same way as the receive data. Reading the RCREG register will allow the RX9D and FERR bits to be loaded with values for the next received data; therefore, it is essential for the user to read the RCSTA register before reading RCREG in order not to lose the old FERR and RX9D information.

Steps to follow when setting up a Synchronous Master Reception:

1. Initialize the SPBRG register for the appropriate baud rate. See Section 13.1 for details.
2. Enable the synchronous master serial port by setting bits SYNC, SPEN, and CSRC.
3. If interrupts are desired, then set the RCIE bit.
4. If 9-bit reception is desired, then set the RX9 bit.
5. If a single reception is required, set bit SREN. For continuous reception set bit CREN.
6. The RCIF bit will be set when reception is complete and an interrupt will be generated if the RCIE bit was set.
7. Read RCSTA to get the ninth bit (if enabled) and determine if any error occurred during reception.
8. Read the 8-bit received data by reading RCREG.
9. If any error occurred, clear the error by clearing CREN.

Note:	To terminate a reception, either clear the SREN and CREN bits, or the SPEN bit. This will reset the receive logic, so that it will be in the proper state when receive is re-enabled.

FIGURE 13-11: SYNCHRONOUS RECEPTION (MASTER MODE, SREN)



Note: Timing diagram demonstrates SYNC master mode with SREN = 1.

14.3 Watchdog Timer (WDT)

The Watchdog Timer's function is to recover from software malfunction. The WDT uses an internal free running on-chip RC oscillator for its clock source. This does not require any external components. This RC oscillator is separate from the RC oscillator of the OSC1/CLKIN pin. That means that the WDT will run, even if the clock on the OSC1/CLKIN and OSC2/CLKOUT pins of the device has been stopped, for example, by execution of a `SLEEP` instruction. During normal operation and SLEEP mode, a WDT time-out generates a device RESET. The WDT can be permanently disabled by programming the configuration bits WDTPS1:WDTPS0 as '`00`' (Section 14.1).

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

14.3.1 WDT PERIOD

The WDT has a nominal time-out period of 12 ms, (with postscaler = 1). The time-out periods vary with temperature, V_{DD} and process variations from part to part (see DC specs). If longer time-out periods are desired, a postscaler with a division ratio of up to 1:256 can be assigned to the WDT. Thus, typical time-out periods up to 3.0 seconds can be realized.

The `CLRWDT` and `SLEEP` instructions clear the WDT and the postscaler (if assigned to the WDT) and prevent it from timing out thus generating a device RESET condition.

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

14.3.2 CLEARING THE WDT AND POSTSCALER

The WDT and postscaler are cleared when:

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

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

14.3.3 WDT PROGRAMMING CONSIDERATIONS

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

The WDT and postscaler is the Power-up Timer during the Power-on Reset sequence.

14.3.4 WDT AS NORMAL TIMER

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

DCFSNZ — Decrement f, skip if not 0

Syntax: [*label*] DCFSNZ f,d

Operands: $0 \le f \le 255$
$d \in [0,1]$

Operation: (f) – 1 → (dest);
skip if not 0

Status Affected: None

Encoding:

0010	011d	ffff	ffff

Description: The contents of register 'f' are decremented. If 'd' is 0 the result is placed in WREG. If 'd' is 1 the result is placed back in register 'f'.

If the result is not 0, the next instruction, which is already fetched, is discarded, and an NOP is executed instead making it a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:

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

If skip:

Q1	Q2	Q3	Q4
Forced NOP	NOP	Execute	NOP

Example:

```
HERE     DCFSNZ  TEMP, 1
ZERO     :
NZERO    :
```

Before Instruction
TEMP_VALUE = ?

After Instruction
TEMP_VALUE = TEMP_VALUE - 1,
If TEMP_VALUE = 0;
PC = Address (ZERO)
If TEMP_VALUE ≠ 0;
PC = Address (NZERO)

GOTO — Unconditional Branch

Syntax: [*label*] GOTO k

Operands: $0 \le k \le 8191$

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

Status Affected: None

Encoding:

110k	kkkk	kkkk	kkkk

Description: GOTO allows an unconditional branch anywhere within an 8K page boundary. The thirteen bit immediate value is loaded into PC bits <12:0>. Then the upper eight bits of PC are loaded into PCLATH. GOTO is always a two-cycle instruction.

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: GOTO THERE

After Instruction
PC = Address (THERE)

MOVFP — Move f to p

Syntax:	[*label*] MOVFP f,p
Operands:	0 ≤ f ≤ 255 0 ≤ p ≤ 31
Operation:	(f) → (p)
Status Affected:	None
Encoding:	`011p` `pppp` `ffff` `ffff`
Description:	Move data from data memory location 'f' to data memory location 'p'. Location 'f' can be anywhere in the 256 word data space (00h to FFh) while 'p' can be 00h to 1Fh. Either 'p' or 'f' can be WREG (a useful special situation). `MOVFP` is particularly useful for transferring a data memory location to a peripheral register (such as the transmit buffer or an I/O port). Both 'f' and 'p' can be indirectly addressed.
Words:	1
Cycles:	1

Q Cycle Activity:

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

Example: `MOVFP REG1, REG2`

Before Instruction
REG1 = 0x33,
REG2 = 0x11

After Instruction
REG1 = 0x33,
REG2 = 0x33

MOVLB — Move Literal to low nibble in BSR

Syntax:	[*label*] MOVLB k
Operands:	0 ≤ k ≤ 15
Operation:	k → (BSR<3:0>)
Status Affected:	None
Encoding:	`1011` `1000` `uuuu` `kkkk`
Description:	The four bit literal 'k' is loaded in the Bank Select Register (BSR). Only the low 4-bits of the Bank Select Register are affected. The upper half of the BSR is unchanged. The assembler will encode the “u” fields as '0'.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'u:k'	Execute	Write literal 'k' to BSR<3:0>

Example: `MOVLB 0x5`

Before Instruction
BSR register = 0x22

After Instruction
BSR register = 0x25

Note: For the PIC17C42, only the low four bits of the BSR register are physically implemented. The upper nibble is read as '0'.

MOVPF — Move p to f

Syntax: [*label*] MOVPF p,f

Operands: $0 \le f \le 255$
$0 \le p \le 31$

Operation: $(p) \rightarrow (f)$

Status Affected: Z

Encoding:

`010p`	`pppp`	`ffff`	`ffff`

Description: Move data from data memory location 'p' to data memory location 'f'. Location 'f' can be anywhere in the 256 byte data space (00h to FFh) while 'p' can be 00h to 1Fh.

Either 'p' or 'f' can be WREG (a useful special situation).

`MOVPF` is particularly useful for transferring a peripheral register (e.g. the timer or an I/O port) to a data memory location. Both 'f' and 'p' can be indirectly addressed.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: `MOVPF REG1, REG2`

Before Instruction
REG1 = 0x11
REG2 = 0x33

After Instruction
REG1 = 0x11
REG2 = 0x11

MOVWF — Move WREG to f

Syntax: [*label*] MOVWF f

Operands: $0 \le f \le 255$

Operation: $(WREG) \rightarrow (f)$

Status Affected: None

Encoding:

`0000`	`0001`	`ffff`	`ffff`

Description: Move data from WREG to register 'f'. Location 'f' can be anywhere in the 256 word data space.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: `MOVWF REG`

Before Instruction
WREG = 0x4F
REG = 0xFF

After Instruction
WREG = 0x4F
REG = 0x4F

RETFIE — Return from Interrupt

Syntax:	[*label*] RETFIE
Operands:	None
Operation:	TOS → (PC); 0 → GLINTD; PCLATH is unchanged.
Status Affected:	GLINTD
Encoding:	`0000` `0000` `0000` `0101`
Description:	Return from Interrupt. Stack is POP'ed and Top of Stack (TOS) is loaded in the PC. Interrupts are enabled by clearing the GLINTD bit. GLINTD is the global interrupt disable bit (CPUSTA<4>).
Words:	1
Cycles:	2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register T0STA	Execute	NOP
Forced NOP	NOP	Execute	NOP

Example: `RETFIE`

After Interrupt
PC = TOS
GLINTD = 0

RETLW — Return Literal to WREG

Syntax:	[*label*] RETLW k
Operands:	$0 \le k \le 255$
Operation:	k → (WREG); TOS → (PC); PCLATH is unchanged
Status Affected:	None
Encoding:	`1011` `0110` `kkkk` `kkkk`
Description:	WREG is loaded with the eight bit literal 'k'. The program counter is loaded from the top of the stack (the return address). The high address latch (PCLATH) remains unchanged.
Words:	1
Cycles:	2

Q Cycle Activity:

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

Example:

```
   CALL TABLE ; WREG contains table
              ;  offset value
              ;  WREG now has
              ;  table value
   :
TABLE
   ADDWF PC   ; WREG = offset
   RETLW k0   ; Begin table
   RETLW k1   ;
   :
   :
   RETLW kn   ; End of table
```

Before Instruction
WREG = 0x07

After Instruction
WREG = value of k7

RRNCF — Rotate Right f (no carry)

Syntax:	[*label*] RRNCF f,d
Operands:	0 ≤ f ≤ 255 d ∈ [0,1]
Operation:	f<n> → d<n-1>; f<0> → d<7>
Status Affected:	None

Encoding:

0010	000d	ffff	ffff

Description: The contents of register 'f' are rotated one bit to the right. If 'd' is 0 the result is placed in WREG. If 'd' is 1 the result is placed back in register 'f'.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example 1: `RRNCF REG, 1`

Before Instruction
WREG = ?
REG = 1101 0111

After Instruction
WREG = 0
REG = 1110 1011

Example 2: `RRNCF REG, 0`

Before Instruction
WREG = ?
REG = 1101 0111

After Instruction
WREG = 1110 1011
REG = 1101 0111

SETF — Set f

Syntax:	[*label*] SETF f,s
Operands:	0 ≤ f ≤ 255 s ∈ [0,1]
Operation:	FFh → f; FFh → d
Status Affected:	None

Encoding:

0010	101s	ffff	ffff

Description: If 's' is 0, both the data memory location 'f' and WREG are set to FFh. If 's' is 1 only the data memory location 'f' is set to FFh.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example1: `SETF REG, 0`

Before Instruction
REG = 0xDA
WREG = 0x05

After Instruction
REG = 0xFF
WREG = 0xFF

Example2: `SETF REG, 1`

Before Instruction
REG = 0xDA
WREG = 0x05

After Instruction
REG = 0xFF
WREG = 0x05

NOTES:

Applicable Devices	42	R42	42A	43	R43	44

17.3 Timing Parameter Symbology

The timing parameter symbols have been created using one of the following formats:

1. TppS2ppS
2. TppS

T			
F	Frequency	T	Time

Lowercase symbols (pp) and their meanings:

pp			
ad	Address/Data	ost	Oscillator Start-up Timer
al	ALE	pwrt	Power-up Timer
cc	Capture1 and Capture2	rb	PORTB
ck	CLKOUT or clock	rd	$\overline{\text{RD}}$
dt	Data in	rw	$\overline{\text{RD}}$ or $\overline{\text{WR}}$
in	INT pin	t0	T0CKI
io	I/O port	t123	TCLK12 and TCLK3
mc	$\overline{\text{MCLR}}$	wdt	Watchdog Timer
oe	$\overline{\text{OE}}$	wr	$\overline{\text{WR}}$
os	OSC1		

Uppercase symbols and their meanings:

S			
D	Driven	L	Low
E	Edge	P	Period
F	Fall	R	Rise
H	High	V	Valid
I	Invalid (Hi-impedance)	Z	Hi-impedance

Applicable Devices	42	R42	42A	43	R43	44

FIGURE 18-15: IOH vs. VOH, VDD = 5V

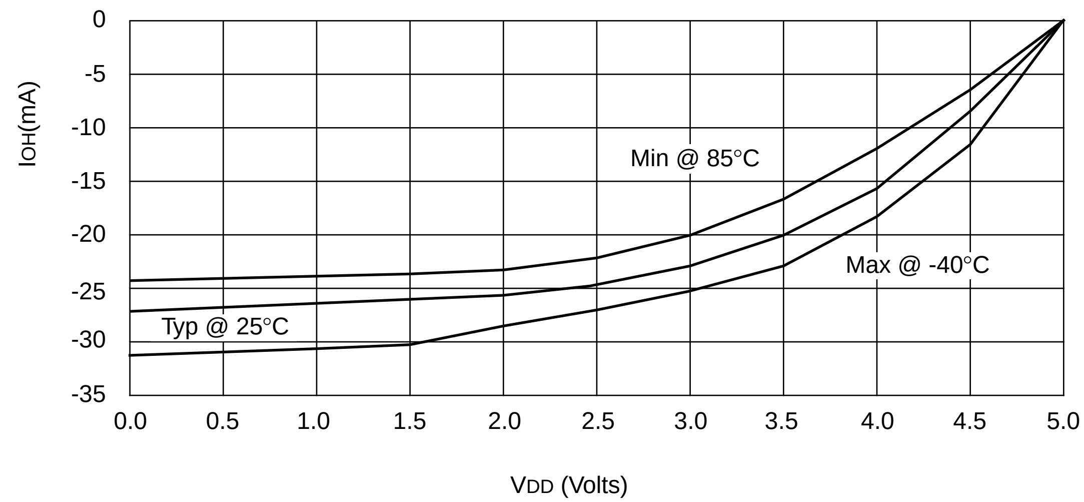
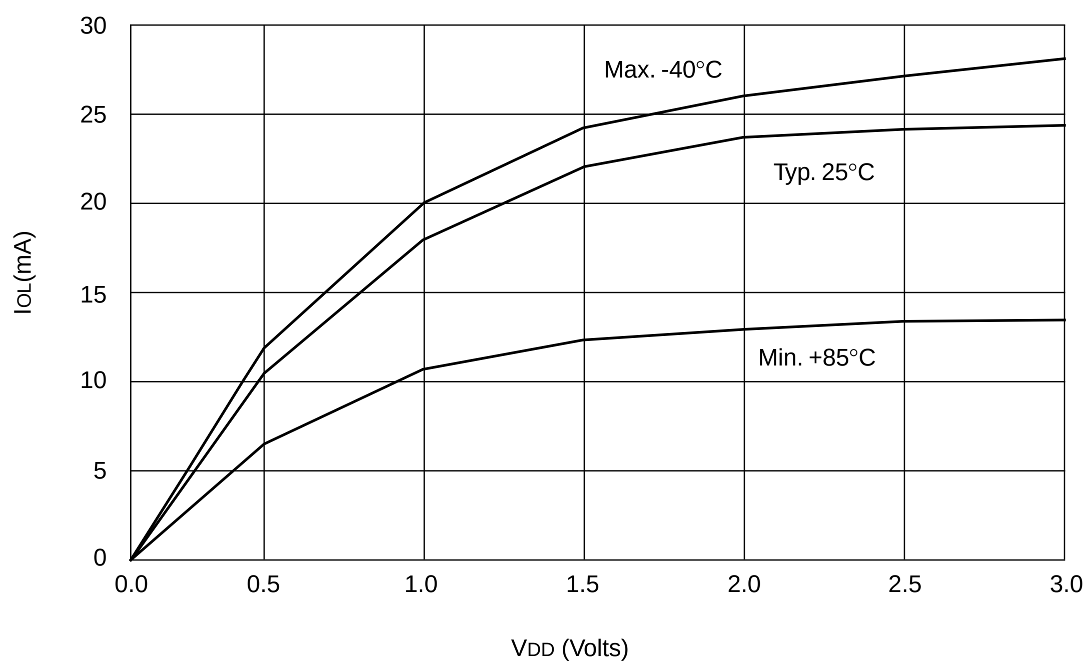


FIGURE 18-16: IOL vs. VOL, VDD = 3V



Applicable Devices	42	R42	42A	43	R43	44

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

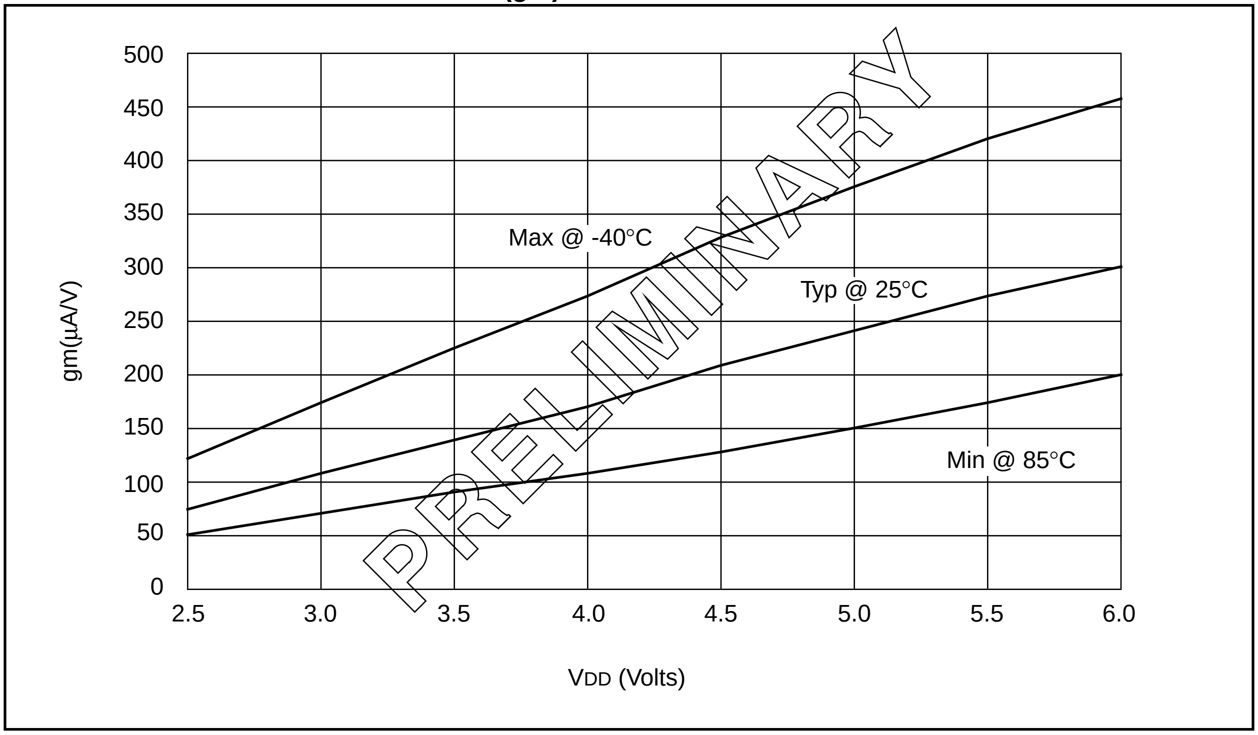
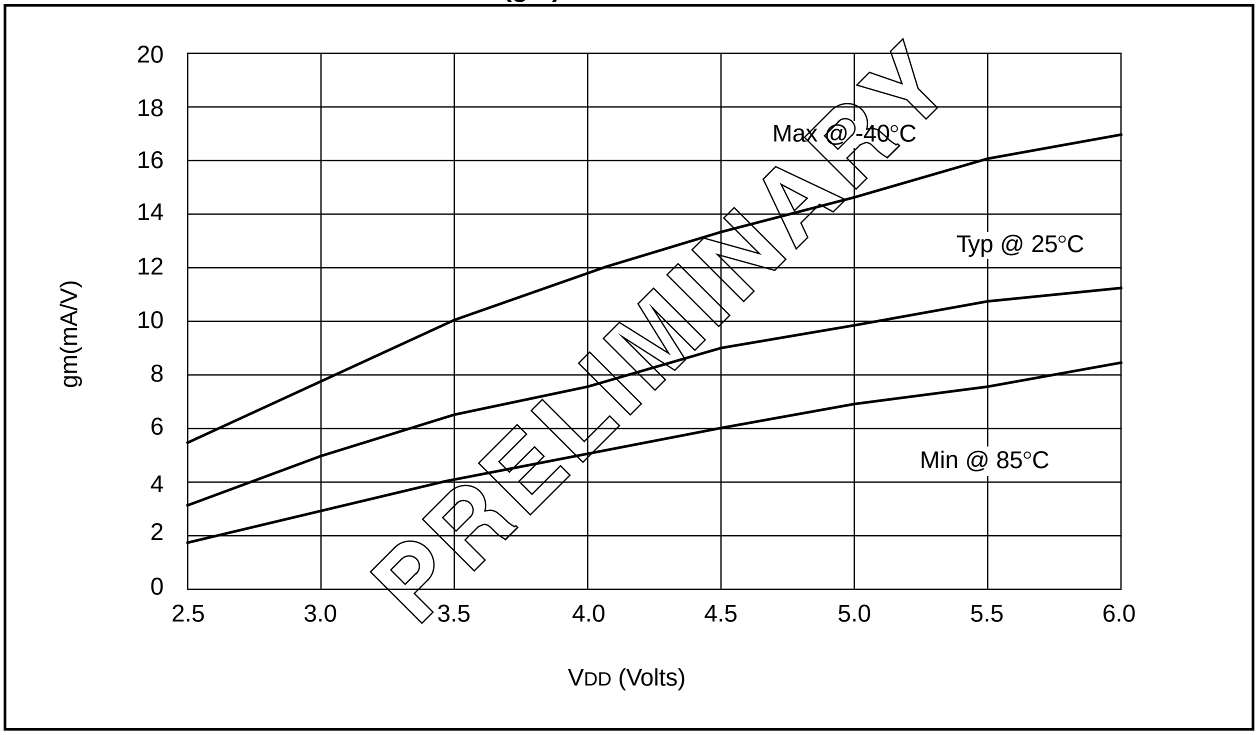


FIGURE 20-6: TRANSCONDUCTANCE (gm) OF XT OSCILLATOR vs. VDD



Applicable Devices	42	R42	42A	43	R43	44

FIGURE 20-11: TYPICAL IPD vs. VDD WATCHDOG ENABLED 25°C

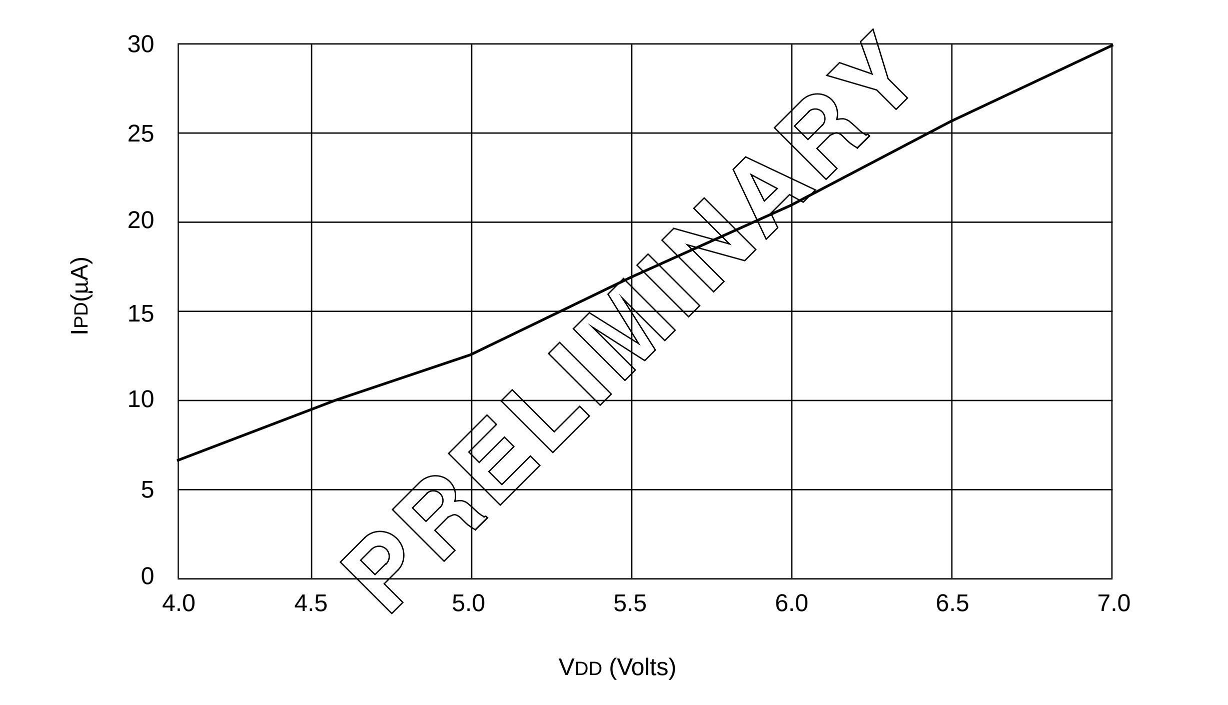
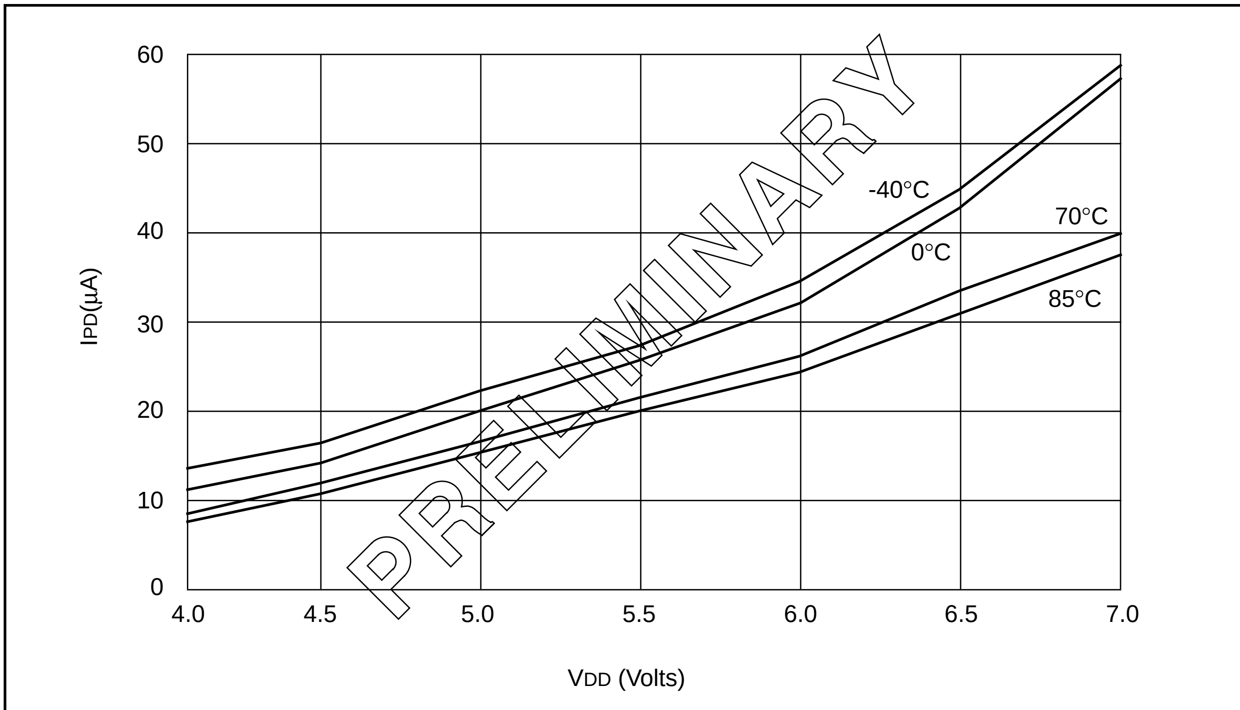


FIGURE 20-12: MAXIMUM IPD vs. VDD WATCHDOG ENABLED



E.3 PIC16CXXX Family of Devices

	Clock	Memory		Peripherals					Features		
	Maximum Frequency of Operation (MHz)	EPROM Program Memory (x14 words)	Data Memory (bytes)	Timer Module(s)	Comparator(s)	Internal Reference Voltage	Interrupt Sources	I/O Pins	Voltage Range (Volts)	Brown-out Reset	Packages
PIC16C554	20	512	80	TMR0	—	—	3	13	2.5-6.0	—	18-pin DIP, SOIC; 20-pin SSOP
PIC16C556	20	1K	80	TMR0	—	—	3	13	2.5-6.0	—	18-pin DIP, SOIC; 20-pin SSOP
PIC16C558	20	2K	128	TMR0	—	—	3	13	2.5-6.0	—	18-pin DIP, SOIC; 20-pin SSOP
PIC16C620	20	512	80	TMR0	2	Yes	4	13	2.5-6.0	Yes	18-pin DIP, SOIC; 20-pin SSOP
PIC16C621	20	1K	80	TMR0	2	Yes	4	13	2.5-6.0	Yes	18-pin DIP, SOIC; 20-pin SSOP
PIC16C622	20	2K	128	TMR0	2	Yes	4	13	2.5-6.0	Yes	18-pin DIP, SOIC; 20-pin SSOP

All PIC16/17 Family devices have Power-on Reset, selectable Watchdog Timer, selectable code protect and high I/O current capability.
All PIC16C6XXX Family devices use serial programming with clock pin RB6 and data pin RB7.

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 pro-vide 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 ser-vice to connect to:

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 vari-ety 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 net-work.

Internet:

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

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 loca-tions.

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 disap-proves 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