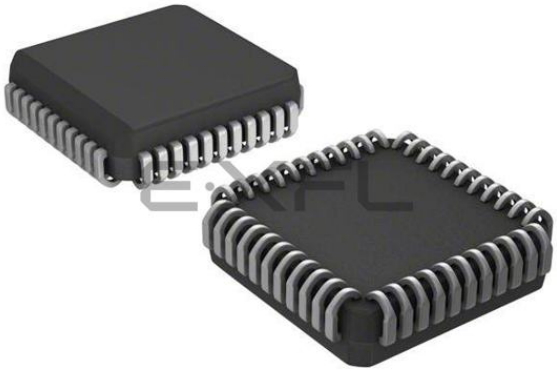


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

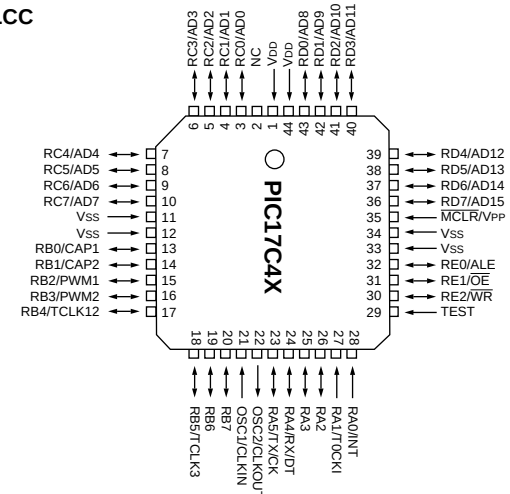
"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" 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"

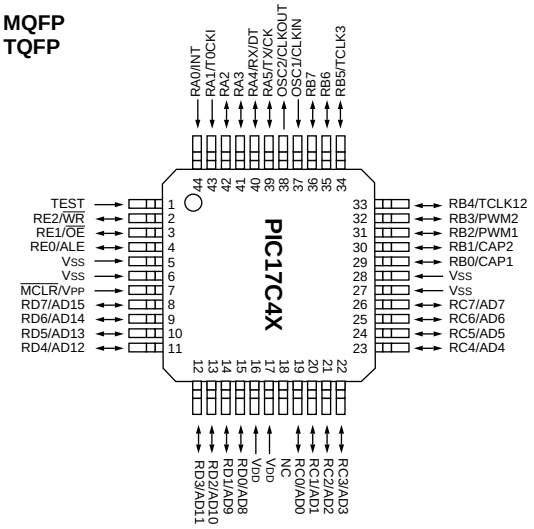
Details	
Product Status	Obsolete
Core Processor	PIC
Core Size	8-Bit
Speed	25MHz
Connectivity	UART/USART
Peripherals	POR, PWM, WDT
Number of I/O	33
Program Memory Size	8KB (4K x 16)
Program Memory Type	OTP
EEPROM Size	-
RAM Size	454 x 8
Voltage - Supply (Vcc/Vdd)	4.5V ~ 6V
Data Converters	-
Oscillator Type	External
Operating Temperature	0°C ~ 70°C (TA)
Mounting Type	Surface Mount
Package / Case	44-LCC (J-Lead)
Supplier Device Package	44-PLCC (16.59x16.59)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic17c43t-25-l

Pin Diagrams Cont.'d

PLCC



MQFP
TQFP



All devices are available in all package types, listed in Section 21.0, with the following exceptions:

- ROM devices are not available in Windowed CERDIP Packages
- TQFP is not available for the PIC17C42.

NOTES:

NOTES:

6.2 Data Memory Organization

Data memory is partitioned into two areas. The first is the General Purpose Registers (GPR) area, while the second is the Special Function Registers (SFR) area. The SFRs control the operation of the device.

Portions of data memory are banked, this is for both areas. The GPR area is banked to allow greater than 232 bytes of general purpose RAM. SFRs are for the registers that control the peripheral functions. Banking requires the use of control bits for bank selection. These control bits are located in the Bank Select Register (BSR). If an access is made to a location outside this banked region, the BSR bits are ignored. Figure 6-5 shows the data memory map organization for the PIC17C42 and Figure 6-6 for all of the other PIC17C4X devices.

Instructions MOVPF and MOVFP provide the means to move values from the peripheral area (“P”) to any location in the register file (“F”), and vice-versa. The definition of the “P” range is from 0h to 1Fh, while the “F” range is 0h to FFh. The “P” range has six more locations than peripheral registers (eight locations for the PIC17C42 device) which can be used as General Purpose Registers. This can be useful in some applications where variables need to be copied to other locations in the general purpose RAM (such as saving status information during an interrupt).

The entire data memory can be accessed either directly or indirectly through file select registers FSR0 and FSR1 (Section 6.4). Indirect addressing uses the appropriate control bits of the BSR for accesses into the banked areas of data memory. The BSR is explained in greater detail in Section 6.8.

6.2.1 GENERAL PURPOSE REGISTER (GPR)

All devices have some amount of GPR area. The GPRs are 8-bits wide. When the GPR area is greater than 232, it must be banked to allow access to the additional memory space.

Only the PIC17C43 and PIC17C44 devices have banked memory in the GPR area. To facilitate switching between these banks, the MOVLR bank instruction has been added to the instruction set. GPRs are not initialized by a Power-on Reset and are unchanged on all other resets.

6.2.2 SPECIAL FUNCTION REGISTERS (SFR)

The SFRs are used by the CPU and peripheral functions to control the operation of the device (Figure 6-5 and Figure 6-6). These registers are static RAM.

The SFRs can be classified into two sets, those associated with the “core” function and those related to the peripheral functions. Those registers related to the “core” are described here, while those related to a peripheral feature are described in the section for each peripheral feature.

The peripheral registers are in the banked portion of memory, while the core registers are in the unbanked region. To facilitate switching between the peripheral banks, the MOVLB bank instruction has been provided.

TABLE 6-3: SPECIAL FUNCTION REGISTERS (Cont.'d)

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)
Bank 2											
10h	TMR1	Timer1								xxxx xxxx	uuuu uuuu
11h	TMR2	Timer2								xxxx xxxx	uuuu uuuu
12h	TMR3L	TMR3 register; low byte								xxxx xxxx	uuuu uuuu
13h	TMR3H	TMR3 register; high byte								xxxx xxxx	uuuu uuuu
14h	PR1	Timer1 period register								xxxx xxxx	uuuu uuuu
15h	PR2	Timer2 period register								xxxx xxxx	uuuu uuuu
16h	PR3L/CA1L	Timer3 period register, low byte/capture1 register; low byte								xxxx xxxx	uuuu uuuu
17h	PR3H/CA1H	Timer3 period register, high byte/capture1 register; high byte								xxxx xxxx	uuuu uuuu
Bank 3											
10h	PW1DCL	DC1	DC0	—	—	—	—	—	—	xx-- ----	uu-- ----
11h	PW2DCL	DC1	DC0	TM2PW2	—	—	—	—	—	xx0- ----	uu0- ----
12h	PW1DCH	DC9	DC8	DC7	DC6	DC5	DC4	DC3	DC2	xxxx xxxx	uuuu uuuu
13h	PW2DCH	DC9	DC8	DC7	DC6	DC5	DC4	DC3	DC2	xxxx xxxx	uuuu uuuu
14h	CA2L	Capture2 low byte								xxxx xxxx	uuuu uuuu
15h	CA2H	Capture2 high byte								xxxx xxxx	uuuu uuuu
16h	TCON1	CA2ED1	CA2ED0	CA1ED1	CA1ED0	T16	TMR3CS	TMR2CS	TMR1CS	0000 0000	0000 0000
17h	TCON2	CA2OVF	CA1OVF	PWM2ON	PWM1ON	CA1/$\overline{PR3}$	TMR3ON	TMR2ON	TMR1ON	0000 0000	0000 0000
Unbanked											
18h (5)	PRODL	Low Byte of 16-bit Product (8 x 8 Hardware Multiply)								xxxx xxxx	uuuu uuuu
19h (5)	PRODH	High Byte of 16-bit Product (8 x 8 Hardware Multiply)								xxxx xxxx	uuuu uuuu

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.

6.2.2.1 ALU STATUS REGISTER (ALUSTA)

The ALUSTA register contains the status bits of the Arithmetic and Logic Unit and the mode control bits for the indirect addressing register.

As with all the other registers, the ALUSTA register can be the destination for any instruction. If the ALUSTA register is the destination for an instruction that affects the Z, DC or C bits, then the write to these three bits is disabled. These bits are set or cleared according to the device logic. Therefore, the result of an instruction with the ALUSTA register as destination may be different than intended.

For example, `CLRF ALUSTA` will clear the upper four bits and set the Z bit. This leaves the ALUSTA register as `0000u1uu` (where u = unchanged).

It is recommended, therefore, that only `BCF`, `BSF`, `SWAPF` and `MOVWF` instructions be used to alter the ALUSTA register because these instructions do not affect any status bit. To see how other instructions affect the status bits, see the "Instruction Set Summary."

Note 1: The C and DC bits operate as a $\overline{\text{borrow}}$ out bit in subtraction. See the `SUBLW` and `SUBWF` instructions for examples.

Note 2: The overflow bit will be set if the 2's complement result exceeds +127 or is less than -128.

Arithmetic and Logic Unit (ALU) is capable of carrying out arithmetic or logical operations on two operands or a single operand. All single operand instructions operate either on the WREG register or a file register. For two operand instructions, one of the operands is the WREG register and the other one is either a file register or an 8-bit immediate constant.

FIGURE 6-7: ALUSTA REGISTER (ADDRESS: 04h, UNBANKED)

R/W - 1	R/W - 1	R/W - 1	R/W - 1	R/W - x	R/W - x	R/W - x	R/W - x
FS3	FS2	FS1	FS0	OV	Z	DC	C
bit7							bit0

R = Readable bit
W = Writable bit
-n = Value at POR reset
(x = unknown)

bit 7-6: **FS3:FS2**: FSR1 Mode Select bits
00 = Post auto-decrement FSR1 value
01 = Post auto-increment FSR1 value
1x = FSR1 value does not change

bit 5-4: **FS1:FS0**: FSR0 Mode Select bits
00 = Post auto-decrement FSR0 value
01 = Post auto-increment FSR0 value
1x = FSR0 value does not change

bit 3: **OV**: Overflow bit
This bit is used for signed arithmetic (2's complement). It indicates an overflow of the 7-bit magnitude, which causes the sign bit (bit7) to change state.
1 = Overflow occurred for signed arithmetic, (in this arithmetic operation)
0 = No overflow occurred

bit 2: **Z**: Zero bit
1 = The result of an arithmetic or logic operation is zero
0 = The results of an arithmetic or logic operation is not zero

bit 1: **DC**: Digit carry/$\overline{\text{borrow}}$ bit
For `ADDWF` and `ADDLW` instructions.
1 = A carry-out from the 4th low order bit of the result occurred
0 = No carry-out from the 4th low order bit of the result
Note: For borrow the polarity is reversed.

bit 0: **C**: carry/$\overline{\text{borrow}}$ bit
For `ADDWF` and `ADDLW` instructions.
1 = A carry-out from the most significant bit of the result occurred
Note that a subtraction is executed by adding the two's complement of the second operand. For rotate (`RRCF`, `RLCF`) instructions, this bit is loaded with either the high or low order bit of the source register.
0 = No carry-out from the most significant bit of the result
Note: For borrow the polarity is reversed.

Example 8-3 shows the sequence to do a 16 x 16 unsigned multiply. Equation 8-1 shows the algorithm that is used. The 32-bit result is stored in 4 registers RES3:RES0.

EQUATION 8-1: 16 x 16 UNSIGNED MULTIPLICATION ALGORITHM

$$
\begin{aligned}
RES3:RES0 &= ARG1H:ARG1L * ARG2H:ARG2L \\
&= (ARG1H * ARG2H * 2^{16}) + \\
&\quad (ARG1H * ARG2L * 2^{8}) + \\
&\quad (ARG1L * ARG2H * 2^{8}) + \\
&\quad (ARG1L * ARG2L)
\end{aligned}
$$

EXAMPLE 8-3: 16 x 16 MULTIPLY ROUTINE

```
   MOVFP    ARG1L, WREG
   MULWF    ARG2L       ; ARG1L * ARG2L ->
                        ;   PRODH:PRODL
   MOVPF    PRODH, RES1 ;
   MOVPF    PRODL, RES0 ;
;
   MOVFP    ARG1H, WREG
   MULWF    ARG2H       ; ARG1H * ARG2H ->
                        ;   PRODH:PRODL
   MOVPF    PRODH, RES3 ;
   MOVPF    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     ;
```

9.3 PORTC and DDRC Registers

PORTC is an 8-bit bi-directional port. The corresponding data direction register is DDRC. A '1' in DDRC configures the corresponding port pin as an input. A '0' in the DDRC register configures the corresponding port pin as an output. Reading PORTC reads the status of the pins, whereas writing to it will write to the port latch. PORTC is multiplexed with the system bus. When operating as the system bus, PORTC is the low order byte of the address/data bus (AD7:AD0). The timing for the system bus is shown in the Electrical Characteristics section.

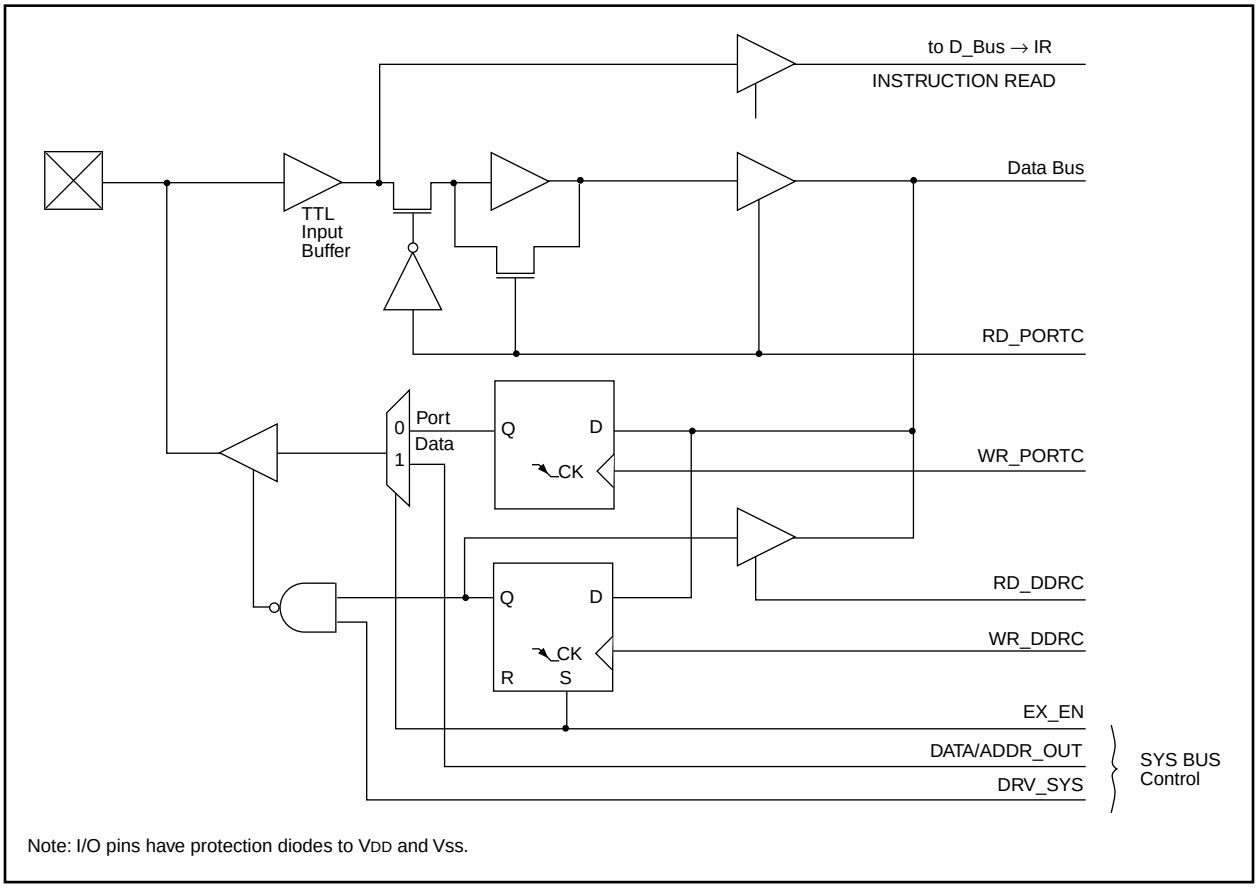
Note:	This port is configured as the system bus when the device's configuration bits are selected to Microprocessor or Extended Microcontroller modes. In the two other microcontroller modes, this port is a general purpose I/O.

Example 9-2 shows the instruction sequence to initialize PORTC. The Bank Select Register (BSR) must be selected to Bank 1 for the port to be initialized.

EXAMPLE 9-2: INITIALIZING PORTC

```
MOVLB  1          ;  Select Bank 1
CLRF   PORTC      ;  Initialize PORTC data
                  ;   latches before setting
                  ;   the data direction
                  ;   register
MOVLW  0xCF       ;  Value used to initialize
                  ;   data direction
MOVWF  DDRC       ;  Set RC<3:0> as inputs
                  ;   RC<5:4> as outputs
                  ;   RC<7:6> as inputs
```

FIGURE 9-6: BLOCK DIAGRAM OF RC<7:0> PORT PINS



Note: I/O pins have protection diodes to VDD and Vss.

12.2.3 EXTERNAL CLOCK INPUT FOR TIMER3

When TMR3CS is set, the 16-bit TMR3 increments on the falling edge of clock input TCLK3. The input on the RB5/TCLK3 pin is sampled and synchronized by the internal phase clocks twice every instruction cycle. This causes a delay from the time a falling edge appears on TCLK3 to the time TMR3 is actually incremented. For the external clock input timing requirements, see the Electrical Specification section. Figure 12-9 shows the timing diagram when operating from an external clock.

12.2.4 READING/WRITING TIMER3

Since Timer3 is a 16-bit timer and only 8-bits at a time can be read or written, care should be taken when reading or writing while the timer is running. The best method to read or write the timer is to stop the timer, perform any read or write operation, and then restart Timer3 (using the TMR3ON bit). However, if it is necessary to keep Timer3 free-running, care must be taken. For writing to the 16-bit TMR3, Example 12-2 may be used. For reading the 16-bit TMR3, Example 12-3 may be used. Interrupts must be disabled during this routine.

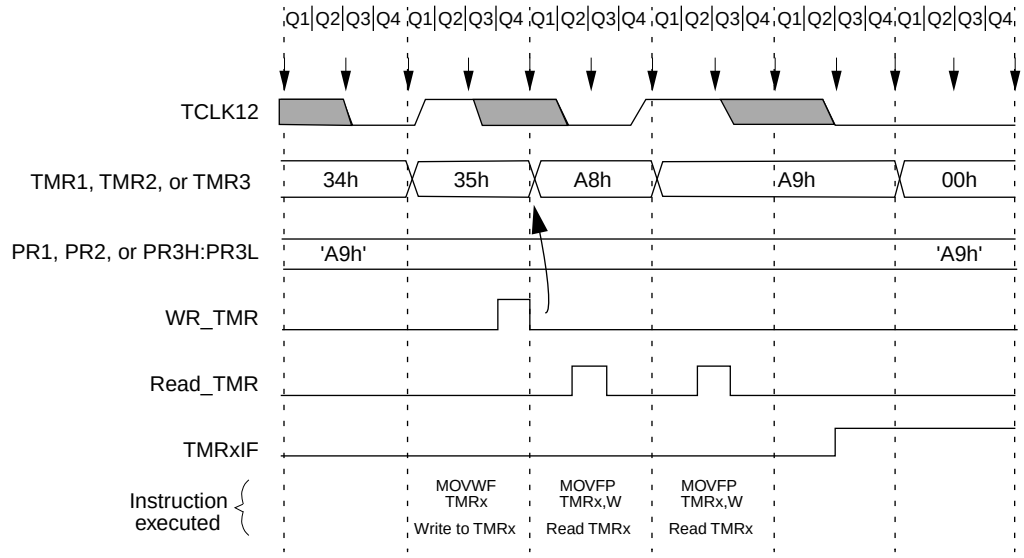
EXAMPLE 12-2: WRITING TO TMR3

```
BSF   CPUSTA, GLINTD ;Disable interrupt
MOVFP RAM_L,  TMR3L  ;
MOVFP RAM_H,  TMR3H  ;
BCF   CPUSTA, GLINTD ;Done,enable interrupt
```

EXAMPLE 12-3: READING FROM TMR3

```
MOVPF   TMR3L, TMPLO    ;read low tmr0
MOVPF   TMR3H, TMPHI    ;read high tmr0
MOVFP   TMPLO, WREG     ;tmplo -> wreg
CPFSLT  TMR3L, WREG     ;tmr0l < wreg?
RETURN                  ;no then return
MOVPF   TMR3L, TMPLO    ;read low tmr0
MOVPF   TMR3H, TMPHI    ;read high tmr0
RETURN                  ;return
```

FIGURE 12-9: TMR1, TMR2, AND TMR3 OPERATION IN EXTERNAL CLOCK MODE



Note 1: TCLK12 is sampled in Q2 and Q4.
2: ↓ indicates a sampling point.
3: The latency from TCLK12 ↓ to timer increment is between 2Tosc and 6Tosc.

13.2 USART Asynchronous Mode

In this mode, the USART uses standard nonreturn-to-zero (NRZ) format (one start bit, eight or nine data bits, and one stop bit). The most common data format is 8-bits. An on-chip dedicated 8-bit baud rate generator can be used to derive standard baud rate frequencies from the oscillator. The USART's transmitter and receiver are functionally independent but use the same data format and baud rate. The baud rate generator produces a clock x64 of the bit shift rate. Parity is not supported by the hardware, but can be implemented in software (and stored as the ninth data bit). Asynchronous mode is stopped during SLEEP.

The asynchronous mode is selected by clearing the SYNC bit (TXSTA<4>).

The USART Asynchronous module consists of the following important elements:

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

13.2.1 USART ASYNCHRONOUS TRANSMITTER

The USART transmitter block diagram is shown in Figure 13-3. The heart of the transmitter is the transmit 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 stop bit has been transmitted from the previous load. As soon as the stop bit is transmitted, the TSR is loaded with new data from the TXREG (if available). Once TXREG transfers the data to the TSR (occurs in one T_{CY} at the end of the current BRG cycle), the TXREG is empty and an interrupt bit, TXIF (PIR<1>) is set. This interrupt can be enabled or disabled by the TXIE bit (PIE<1>). TXIF will be set regardless of TXIE and cannot be reset in software. It will reset only when new data is loaded into TXREG. While TXIF indicates the status of the TXREG, the TRMT (TXSTA<1>) bit 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.

Note:	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 and the baud rate generator (BRG) has produced a shift clock (Figure 13-5). The transmission can also be started by first loading TXREG and then setting TXEN. Normally when transmission is first started, the TSR is empty, so a transfer to TXREG will result in an immediate transfer to TSR resulting in an empty TXREG. A back-to-back transfer is thus possible (Figure 13-6). Clearing TXEN during a transmission will cause the transmission to be aborted. This will reset the transmitter and the RA5/TX/CK pin will revert to hi-impedance.

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 the 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).

Steps to follow when setting up an Asynchronous Transmission:

1. Initialize the SPBRG register for the appropriate baud rate.
2. Enable the asynchronous serial port by clearing the SYNC bit and setting the SPEN bit.
3. If interrupts are desired, then set the TXIE bit.
4. If 9-bit transmission is desired, then set the TX9 bit.
5. Load data to the TXREG register.
6. If 9-bit transmission is selected, the ninth bit should be loaded in TX9D.
7. Enable the transmission by setting TXEN (starts transmission).

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

FIGURE 13-5: ASYNCHRONOUS MASTER TRANSMISSION

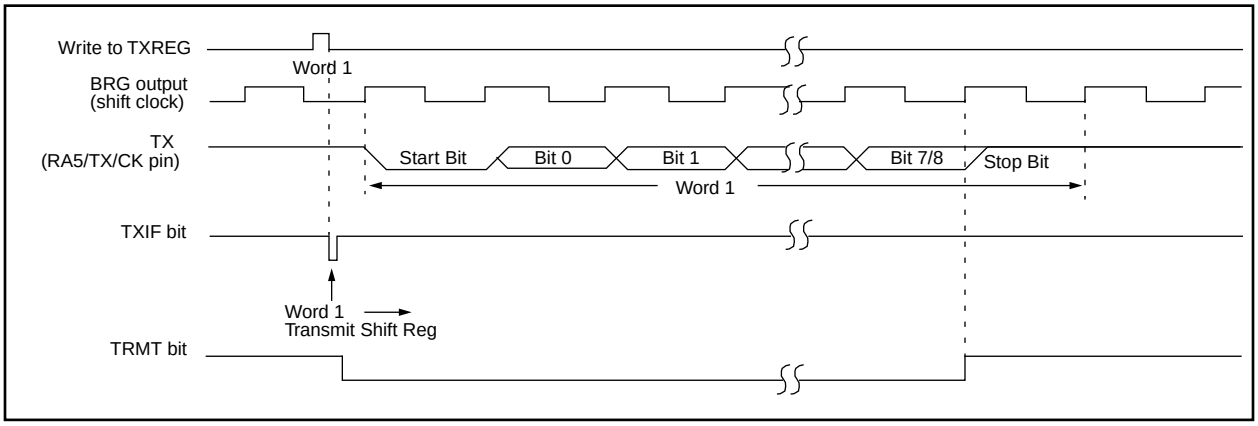
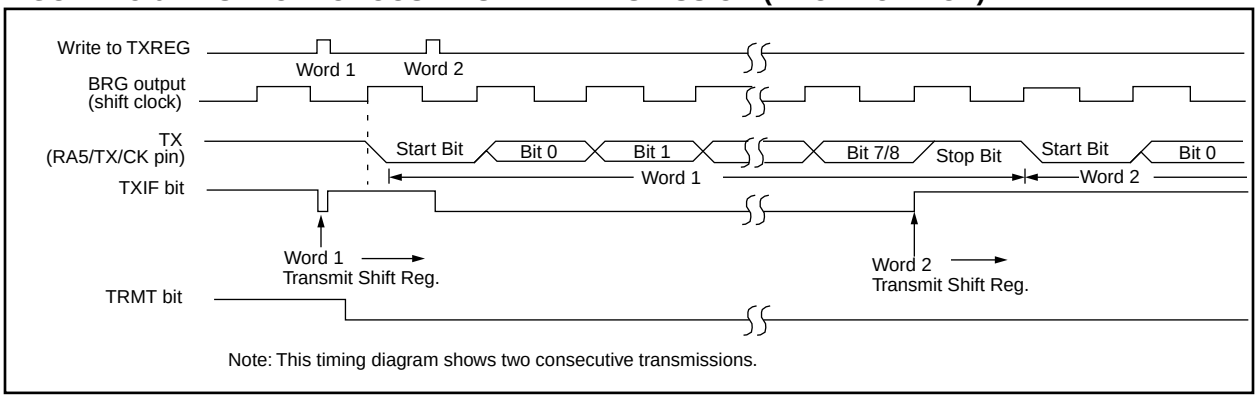


FIGURE 13-6: ASYNCHRONOUS MASTER TRANSMISSION (BACK TO BACK)



Note: This timing diagram shows two consecutive transmissions.

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.

RLNCF **Rotate Left f (no carry)**

Syntax: [*label*] RLNCF f,d

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

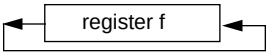
Operation: f<n> → d<n+1>;
f<7> → d<0>

Status Affected: None

Encoding:

0010	001d	ffff	ffff

Description: The contents of register 'f' are rotated one bit to the left. If 'd' is 0 the result is placed 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: RLNCF REG, 1

Before Instruction
C = 0
REG = 1110 1011

After Instruction
C =
REG = 1101 0111

RRCF **Rotate Right f through Carry**

Syntax: [*label*] RRCF f,d

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

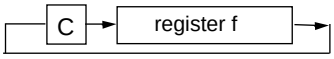
Operation: f<n> → d<n-1>;
f<0> → C;
C → d<7>

Status Affected: C

Encoding:

0001	100d	ffff	ffff

Description: The contents of register 'f' are rotated one bit to the right through the Carry Flag. If 'd' is 0 the result is placed in 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: RRCF REG1,0

Before Instruction
REG1 = 1110 0110
C = 0

After Instruction
REG1 = 1110 0110
WREG = 0111 0011
C = 0

RRNCF **Rotate Right f (no carry)**

Syntax: [*label*] RRNCF f,d

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

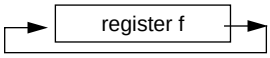
Operation: f<n> → d<n-1>;
f<0> → d<7>

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

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
```

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 (*fuzzy*TECH-MP)

*fuzzy*TECH-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, *fuzzy*TECH-MP, edition for implementing more complex systems.

Both versions include Microchip's *fuzzy*LAB™ 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.

Applicable Devices	42	R42	42A	43	R43	44

FIGURE 17-4: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER TIMING

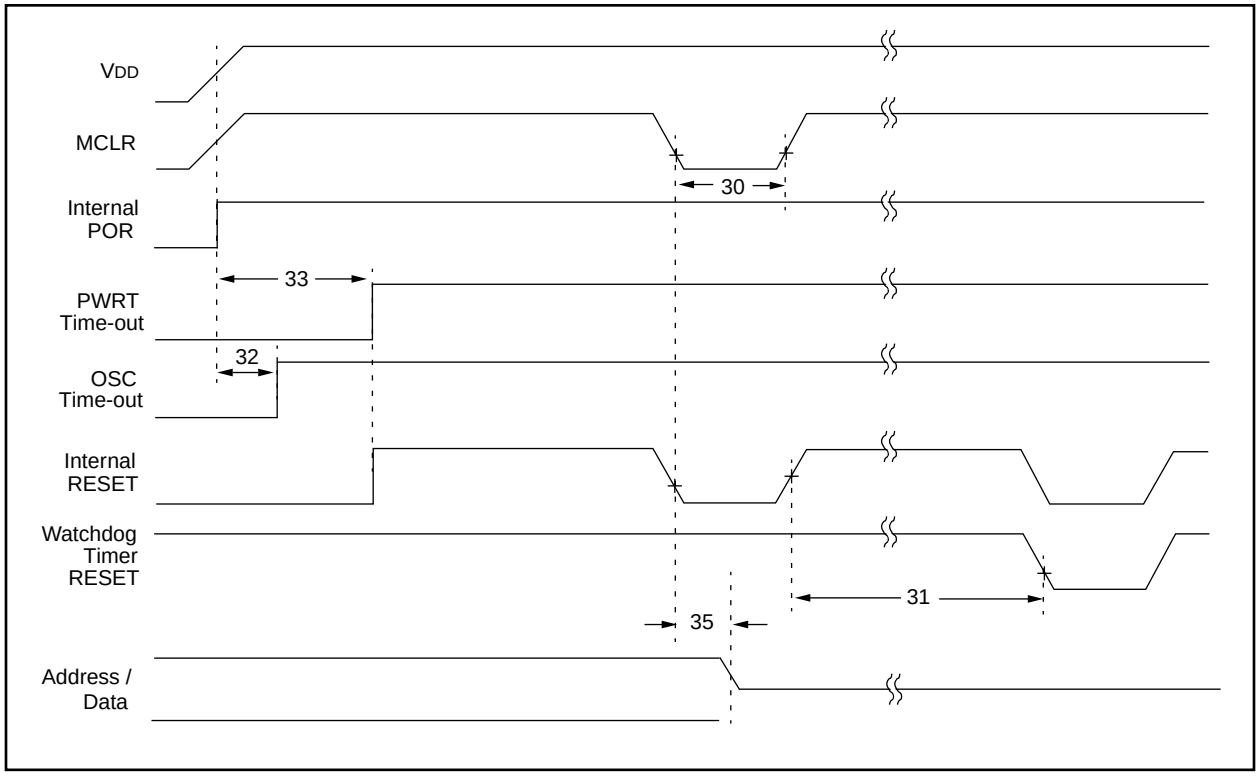


TABLE 17-4: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER REQUIREMENTS

Parameter No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
30	TmcL	$\overline{\text{MCLR}}$ Pulse Width (low)	100 *	—	—	ns	
31	Twdt	Watchdog Timer Time-out Period (Prescale = 1)	5 *	12	25 *	ms	
32	Tost	Oscillation Start-up Timer Period		1024 TOSC §		ms	TOSC = OSC1 period
33	Tpwrt	Power-up Timer Period	40 *	96	200 *	ms	
35	TmcL2adl	$\overline{\text{MCLR}}$ to System Interface bus (AD15:AD0) invalid	—	—	100 *	ns	

* 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.

§ This specification ensured by design.

Applicable Devices	42	R42	42A	43	R43	44

FIGURE 18-11: TYPICAL IPD vs. VDD WATCHDOG ENABLED 25°C

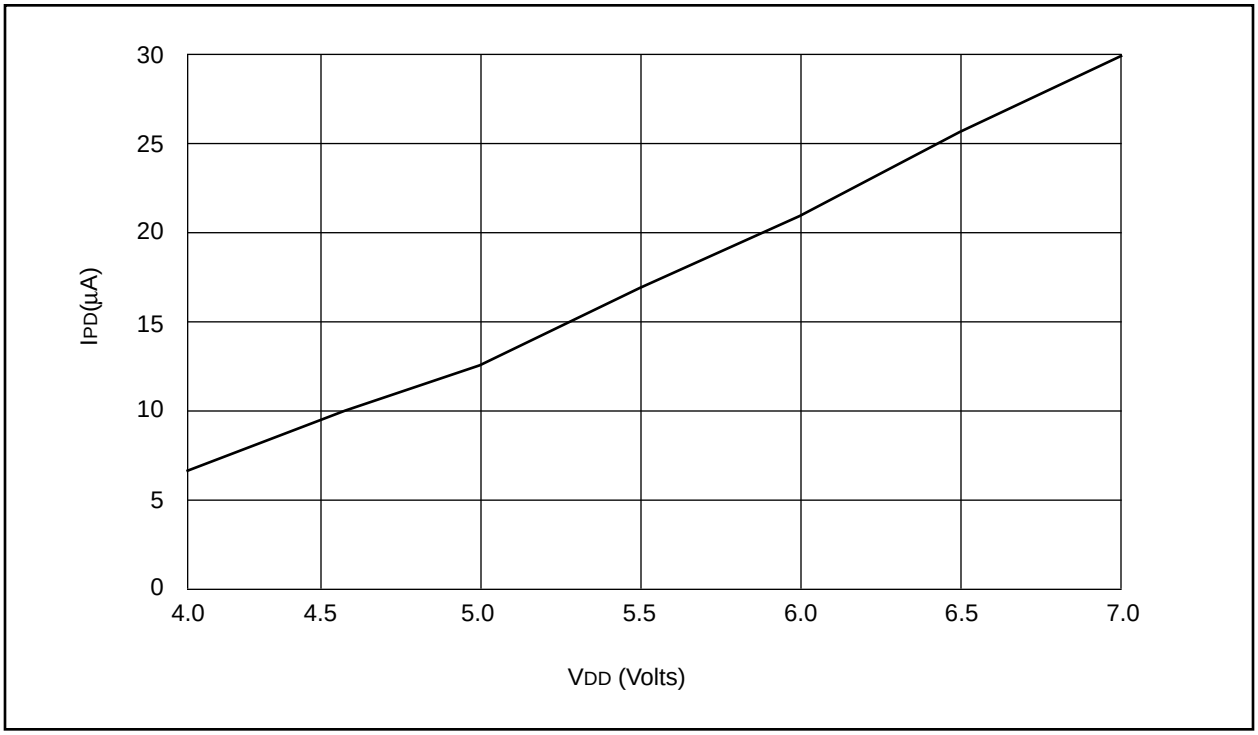
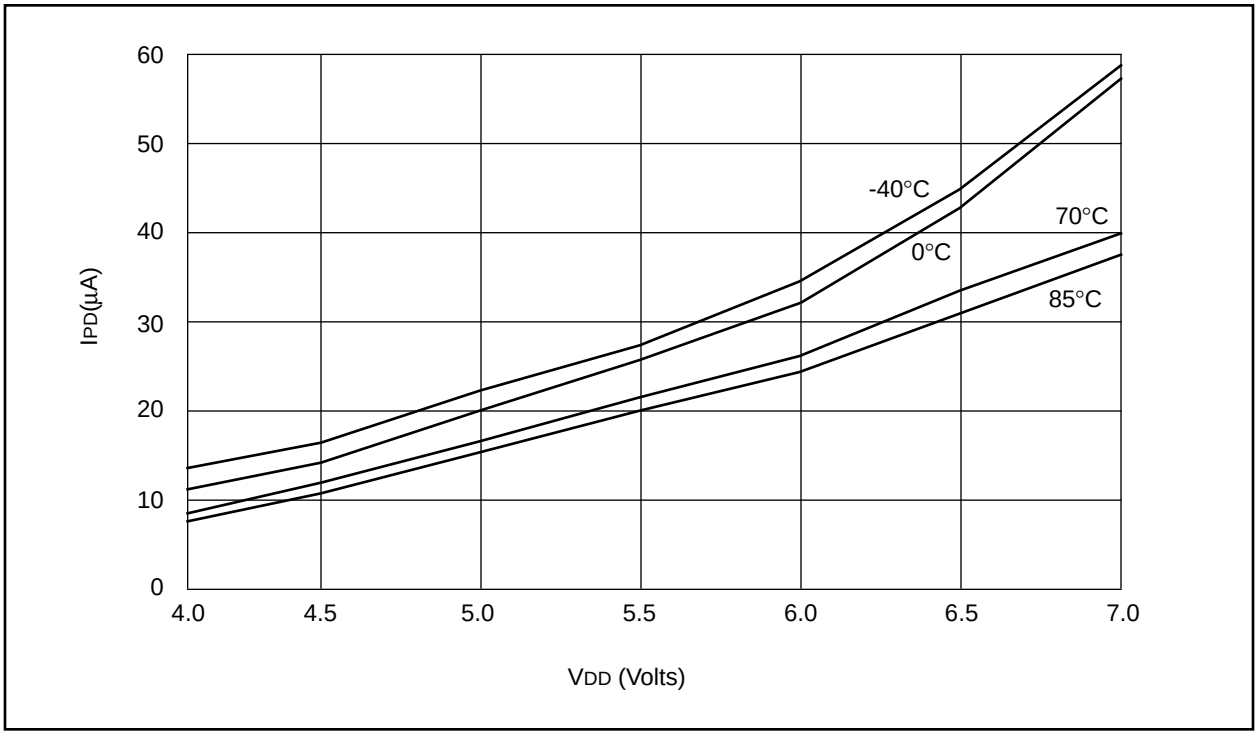


FIGURE 18-12: MAXIMUM IPD vs. VDD WATCHDOG ENABLED



Applicable Devices	42	R42	42A	43	R43	44

FIGURE 18-19: V_{TH}, V_{IL} of I/O PINS (SCHMITT TRIGGER) vs. V_{DD}

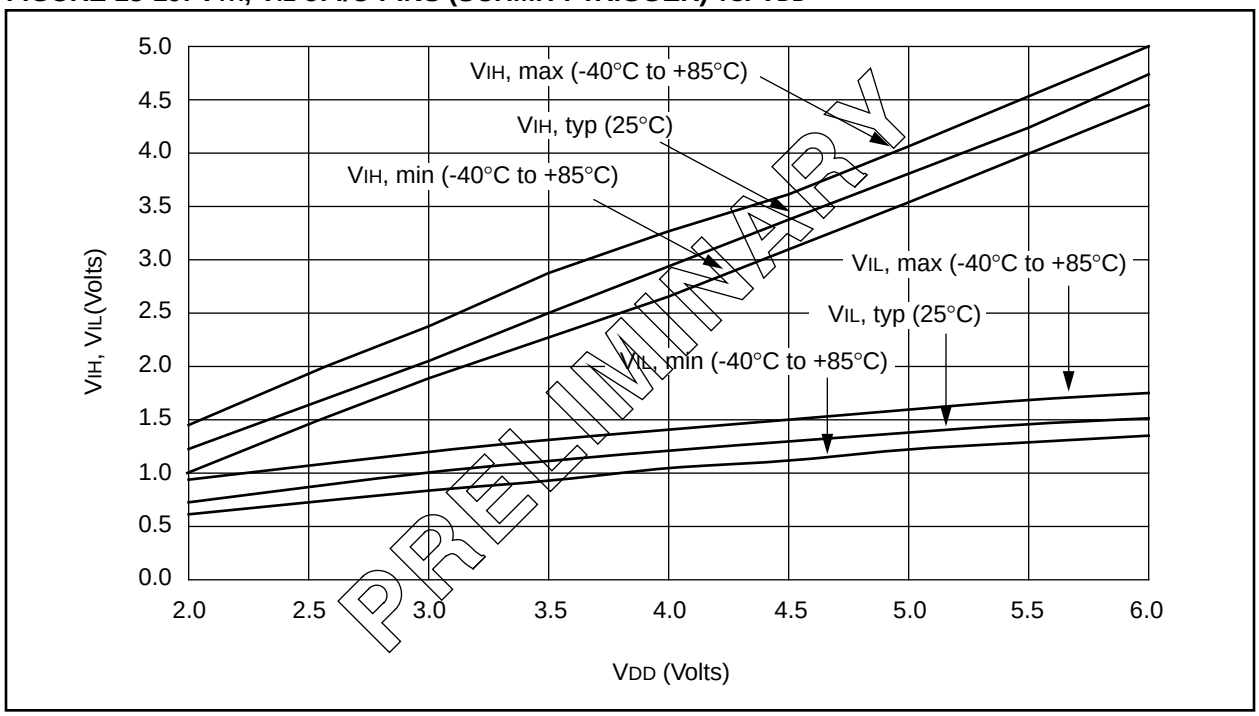
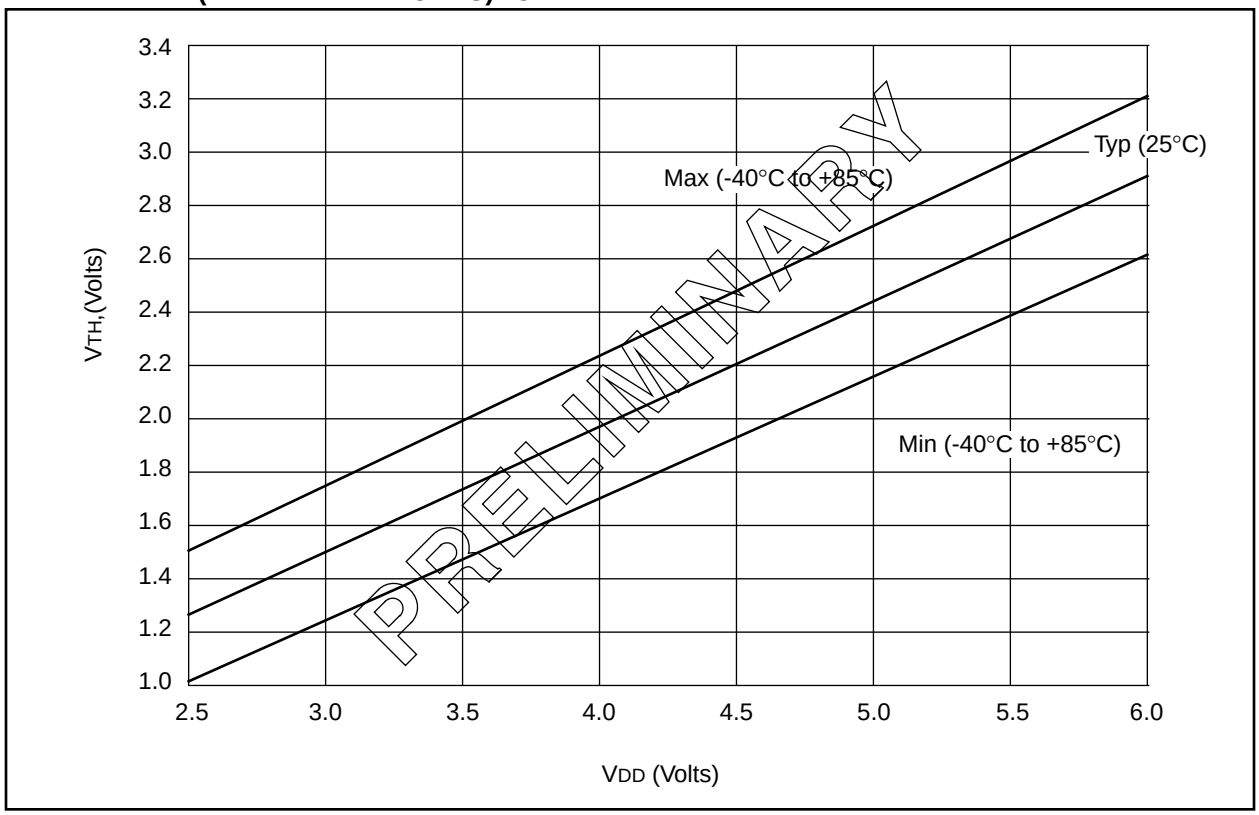


FIGURE 18-20: V_{TH} (INPUT THRESHOLD VOLTAGE) OF OSC1 INPUT (IN XT AND LF MODES) vs. V_{DD}



Applicable Devices	42	R42	42A	43	R43	44

DC CHARACTERISTICS

Standard Operating Conditions (unless otherwise stated)
Operating temperature
-40°C ≤ TA ≤ +85°C for industrial and
0°C ≤ TA ≤ +70°C for commercial
Operating voltage VDD range as described in Section 19.1

Parameter No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
		Output Low Voltage					
D080	VOL	I/O ports (except RA2 and RA3)					IOL = VDD/1.250 mA
			–	–	0.1VDD	V	4.5V ≤ VDD ≤ 6.0V
			–	–	0.1VDD *	V	VDD = 2.5V
D081		with TTL buffer	–	–	0.4	V	IOL = 6 mA, VDD = 4.5V Note 6
D082		RA2 and RA3	–	–	3.0	V	IOL = 60.0 mA, VDD = 6.0V
D083		OSC2/CLKOUT	–	–	0.4	V	IOL = 1 mA, VDD = 4.5V
D084		(RC and EC osc modes)	–	–	0.1VDD *	V	IOL = VDD/5 mA (PIC17LC43/LC44 only)
		Output High Voltage (Note 3)					
D090	VOH	I/O ports (except RA2 and RA3)					IOH = -VDD/2.500 mA
			0.9VDD	–	–	V	4.5V ≤ VDD ≤ 6.0V
			0.9VDD *	–	–	V	VDD = 2.5V
D091		with TTL buffer	2.4	–	–	V	IOH = -6.0 mA, VDD=4.5V Note 6
D092		RA2 and RA3	–	–	12	V	Pulled-up to externally applied voltage
D093		OSC2/CLKOUT	2.4	–	–	V	IOH = -5 mA, VDD = 4.5V
D094		(RC and EC osc modes)	0.9VDD *	–	–	V	IOH = -VDD/5 mA (PIC17LC43/LC44 only)
		Capacitive Loading Specs on Output Pins					
D100	COSC2	OSC2/CLKOUT pin	–	–	25	pF	In EC or RC osc modes when OSC2 pin is outputting CLKOUT. external clock is used to drive OSC1.
D101	CIO	All I/O pins and OSC2 (in RC mode)	–	–	50	pF	
D102	CAD	System Interface Bus (PORTC, PORTD and PORTE)	–	–	50	pF	In Microprocessor or Extended Microcontroller mode

* 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 $\overline{MCLR}$ 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 $\overline{MCLR}$/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.

Applicable Devices	42	R42	42A	43	R43	44

19.5 Timing Diagrams and Specifications

FIGURE 19-2: EXTERNAL CLOCK TIMING

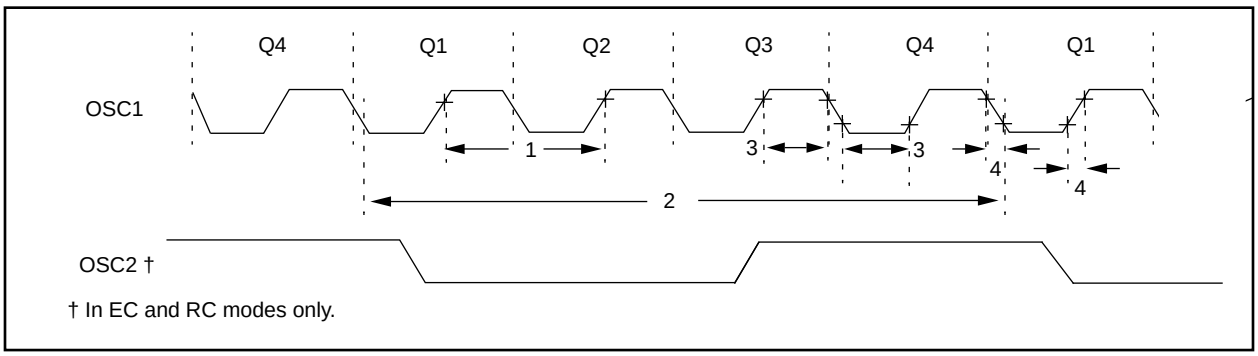


TABLE 19-2: EXTERNAL CLOCK TIMING REQUIREMENTS

Param No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
	Fosc	**External CLKIN Frequency (Note 1)**	DC	—	8	MHz	EC osc mode - 08 devices (8 MHz devices)
			DC	—	16	MHz	- 16 devices (16 MHz devices)
			DC	—	25	MHz	- 25 devices (25 MHz devices)
			DC	—	33	MHz	- 33 devices (33 MHz devices)
		Oscillator Frequency (Note 1)	DC	—	4	MHz	RC osc mode
			1	—	8	MHz	XT osc mode - 08 devices (8 MHz devices)
			1	—	16	MHz	- 16 devices (16 MHz devices)
			1	—	25	MHz	- 25 devices (25 MHz devices)
			1	—	33	MHz	- 33 devices (33 MHz devices)
			DC	—	2	MHz	LF osc mode
1	Tosc	**External CLKIN Period (Note 1)**	125	—	—	ns	EC osc mode - 08 devices (8 MHz devices)
			62.5	—	—	ns	- 16 devices (16 MHz devices)
			40	—	—	ns	- 25 devices (25 MHz devices)
			30.3	—	—	ns	- 33 devices (33 MHz devices)
		Oscillator Period (Note 1)	250	—	—	ns	RC osc mode
			125	—	1,000	ns	XT osc mode - 08 devices (8 MHz devices)
			62.5	—	1,000	ns	- 16 devices (16 MHz devices)
			40	—	1,000	ns	- 25 devices (25 MHz devices)
			30.3	—	1,000	ns	- 33 devices (33 MHz devices)
			500	—	—	ns	LF osc mode
2	TCY	**Instruction Cycle Time (Note 1)**	121.2	4/Fosc	DC	ns	
3	TosL, TosH	**Clock in (OSC1) high or low time**	10 ‡	—	—	ns	EC oscillator
4	TosR, TosF	**Clock in (OSC1) rise or fall time**	—	—	5 ‡	ns	EC oscillator

† 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: Instruction cycle period (TCY) equals four times the input oscillator time base period. All specified values are based on characterization data for that particular oscillator type under standard operating conditions with the device executing code. Exceeding these specified limits may result in an unstable oscillator operation and/or higher than expected current consumption. All devices are tested to operate at "min." values with an external clock applied to the OSC1/CLKIN pin. When an external clock input is used, the "max." cycle time limit is "DC" (no clock) for all devices.

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 microcontroller 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

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

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