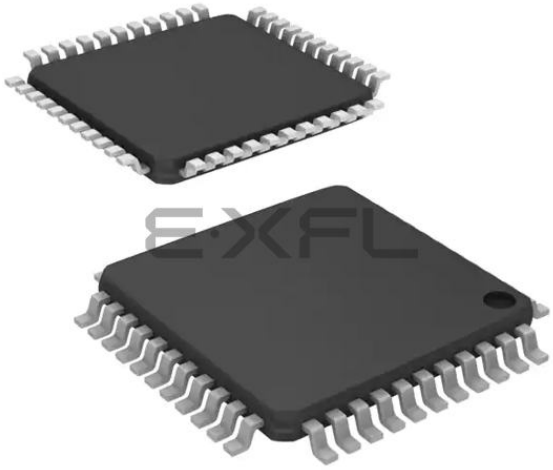


**Welcome to E-XFL.COM**

## 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 | 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 ~ 85°C (TA) |
| Mounting Type | Surface Mount |
| Package / Case | 44-TQFP |
| Supplier Device Package | 44-TQFP (10x10) |
| Purchase URL | https://www.e-xfl.com/product-detail/microchip-technology/pic17c44t-25i-pt |

**FIGURE 6-5: PIC17C42 REGISTER FILE MAP**

| Addr | Unbanked | | | |
|---|---|---|---|---|
| 00h | INDF0 | | | |
| 01h | FSR0 | | | |
| 02h | PCL | | | |
| 03h | PCLATH | | | |
| 04h | ALUSTA | | | |
| 05h | T0STA | | | |
| 06h | CPUSTA | | | |
| 07h | INTSTA | | | |
| 08h | INDF1 | | | |
| 09h | FSR1 | | | |
| 0Ah | WREG | | | |
| 0Bh | TMR0L | | | |
| 0Ch | TMR0H | | | |
| 0Dh | TBLPTRL | | | |
| 0Eh | TBLPTRH | | | |
| 0Fh | BSR | | | |
| | **Bank 0** | **Bank 1 (1)** | **Bank 2 (1)** | **Bank 3 (1)** |
| 10h | PORTA | DDRC | TMR1 | PW1DCL |
| 11h | DDRB | PORTC | TMR2 | PW2DCL |
| 12h | PORTB | DDRD | TMR3L | PW1DCH |
| 13h | RCSTA | PORTD | TMR3H | PW2DCH |
| 14h | RCREG | DDRE | PR1 | CA2L |
| 15h | TXSTA | PORTE | PR2 | CA2H |
| 16h | TXREG | PIR | PR3L/CA1L | TCON1 |
| 17h | SPBRG | PIE | PR3H/CA1H | TCON2 |
| 18h<br>1Fh | General Purpose RAM | | | |
| 20h<br>FFh | | | | |

Note 1: SFR file locations 10h - 17h are banked. All other SFRs ignore the Bank Select Register (BSR) bits.

**FIGURE 6-6: PIC17CR42/42A/43/R43/44 REGISTER FILE MAP**

| Addr | Unbanked | | | |
|---|---|---|---|---|
| 00h | INDF0 | | | |
| 01h | FSR0 | | | |
| 02h | PCL | | | |
| 03h | PCLATH | | | |
| 04h | ALUSTA | | | |
| 05h | T0STA | | | |
| 06h | CPUSTA | | | |
| 07h | INTSTA | | | |
| 08h | INDF1 | | | |
| 09h | FSR1 | | | |
| 0Ah | WREG | | | |
| 0Bh | TMR0L | | | |
| 0Ch | TMR0H | | | |
| 0Dh | TBLPTRL | | | |
| 0Eh | TBLPTRH | | | |
| 0Fh | BSR | | | |
| | **Bank 0** | **Bank 1 (1)** | **Bank 2 (1)** | **Bank 3 (1)** |
| 10h | PORTA | DDRC | TMR1 | PW1DCL |
| 11h | DDRB | PORTC | TMR2 | PW2DCL |
| 12h | PORTB | DDRD | TMR3L | PW1DCH |
| 13h | RCSTA | PORTD | TMR3H | PW2DCH |
| 14h | RCREG | DDRE | PR1 | CA2L |
| 15h | TXSTA | PORTE | PR2 | CA2H |
| 16h | TXREG | PIR | PR3L/CA1L | TCON1 |
| 17h | SPBRG | PIE | PR3H/CA1H | TCON2 |
| 18h | PRODL | | | |
| 19h | PRODH | | | |
| 1Ah<br>1Fh | General Purpose RAM (2) | | | |
| 20h<br>FFh | | General Purpose RAM (2) | | |

Note 1: SFR file locations 10h - 17h are banked. All other SFRs ignore the Bank Select Register (BSR) bits.

2: General Purpose Registers (GPR) locations 20h - FFh and 120h - 1FFh are banked. All other GPRs ignore the Bank Select Register (BSR) bits.

## 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 9-9: PORTE FUNCTIONS**

| Name | Bit | Buffer Type | Function |
|---|---|---|---|
| RE0/ALE | bit0 | TTL | Input/Output or system bus Address Latch Enable (ALE) control pin. |
| RE1/$\overline{OE}$ | bit1 | TTL | Input/Output or system bus Output Enable ($\overline{OE}$) control pin. |
| RE2/$\overline{WR}$ | bit2 | TTL | Input/Output or system bus Write ($\overline{WR}$) control pin. |

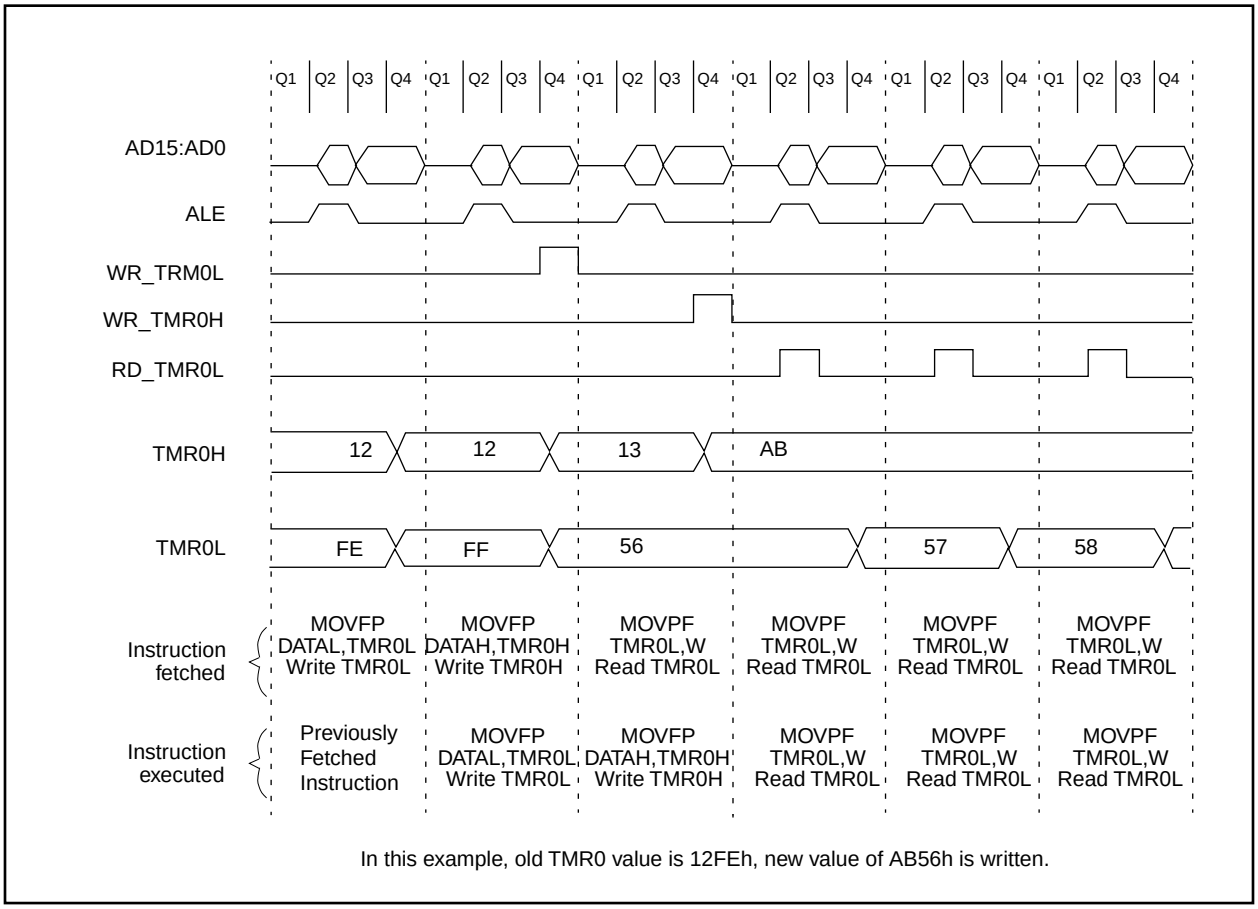
Legend: TTL = TTL input.

**TABLE 9-10: REGISTERS/BITS ASSOCIATED WITH PORTE**

| 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) |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 15h, Bank 1 | PORTE | — | — | — | — | — | RE2/$\overline{WR}$ | RE1/$\overline{OE}$ | RE0/ALE | ---- -xxx | ---- -uuu |
| 14h, Bank 1 | DDRE | Data direction register for PORTE | | | | | | | | ---- -111 | ---- -111 |

Legend: x = unknown, u = unchanged, - = unimplemented read as '0'. Shaded cells are not used by PORTE.
Note 1: Other (non power-up) resets include: external reset through $\overline{MCLR}$ and the Watchdog Timer Reset.

**FIGURE 11-5: TMR0 READ/WRITE IN TIMER MODE**




In this example, old TMR0 value is 12FEh, new value of AB56h is written.

**TABLE 11-1: REGISTERS/BITS ASSOCIATED WITH TIMER0**

| 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) |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 05h, Unbanked | T0STA | INTEDG | T0SE | T0CS | PS3 | PS2 | PS1 | PS0 | — | 0000 000- | 0000 000- |
| 06h, Unbanked | CPUSTA | — | — | STKAV | GLINTD | $\overline{TO}$ | $\overline{PD}$ | — | — | --11 11-- | --11 qq-- |
| 07h, Unbanked | INTSTA | PEIF | T0CKIF | T0IF | INTF | PEIE | T0CKIE | T0IE | INTE | 0000 0000 | 0000 0000 |
| 0Bh, Unbanked | TMR0L | TMR0 register; low byte | | | | | | | | xxxx xxxx | uuuu uuuu |
| 0Ch, Unbanked | TMR0H | TMR0 register; high byte | | | | | | | | xxxx xxxx | uuuu uuuu |

Legend: x = unknown, u = unchanged, - = unimplemented read as a '0', q - value depends on condition, Shaded cells are not used by Timer0.
Note 1: Other (non power-up) resets include: external reset through $\overline{MCLR}$ and the Watchdog Timer Reset.

### 13.1 USART Baud Rate Generator (BRG)

The BRG supports both the Asynchronous and Synchronous modes of the USART. It is a dedicated 8-bit baud rate generator. The SPBRG register controls the period of a free running 8-bit timer. Table 13-1 shows the formula for computation of the baud rate for different USART modes. These only apply when the USART is in synchronous master mode (internal clock) and asynchronous mode.

Given the desired baud rate and Fosc, the nearest integer value between 0 and 255 can be calculated using the formula below. The error in baud rate can then be determined.

**TABLE 13-1: BAUD RATE FORMULA**

| SYNC | Mode | Baud Rate |
|---|---|---|
| 0 | Asynchronous | FOSC/(64(X+1)) |
| 1 | Synchronous | FOSC/(4(X+1)) |

X = value in SPBRG (0 to 255)

Example 13-1 shows the calculation of the baud rate error for the following conditions:

FOSC = 16 MHz
Desired Baud Rate = 9600
SYNC = 0

**EXAMPLE 13-1: CALCULATING BAUD RATE ERROR**

Desired Baud rate=Fosc / (64 (X + 1))

9600 = 16000000 /(64 (X + 1))

X = 25.042 = 25

Calculated Baud Rate=16000000 / (64 (25 + 1))

= 9615

$$\text{Error} = \frac{\text{(Calculated Baud Rate - Desired Baud Rate)}}{\text{Desired Baud Rate}}$$

= (9615 - 9600) / 9600

= 0.16%

Writing a new value to the SPBRG, causes the BRG timer to be reset (or cleared), this ensures that the BRG does not wait for a timer overflow before outputting the new baud rate.

**TABLE 13-2: REGISTERS ASSOCIATED WITH BAUD RATE GENERATOR**

| 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 0 | RCSTA | SPEN | RX9 | SREN | CREN | — | FERR | OERR | RX9D | 0000 -00x | 0000 -00u |
| 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 by the Baud Rate Generator.
Note 1: Other (non power-up) resets include: external reset through $\overline{\text{MCLR}}$ and Watchdog Timer Reset.

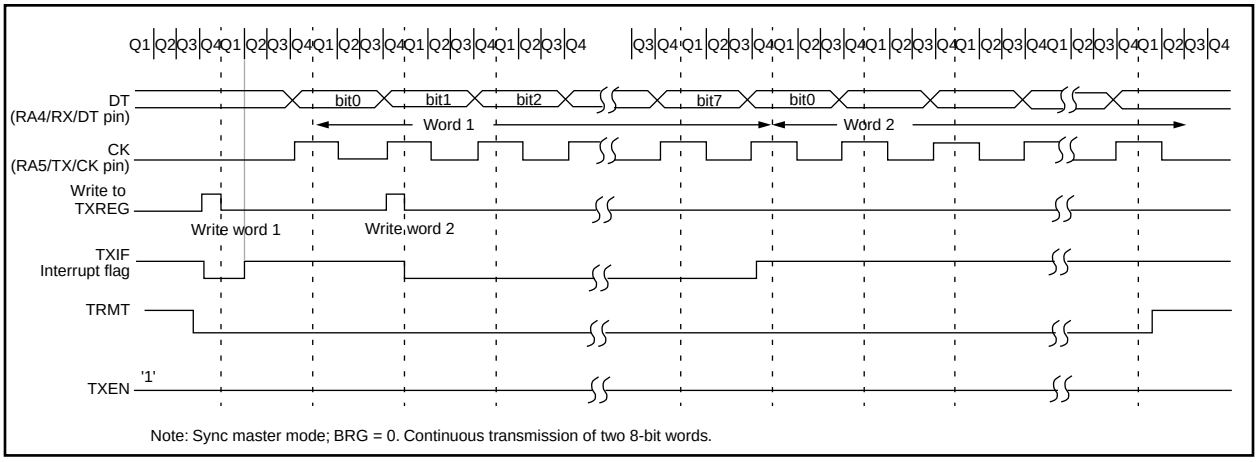
### TABLE 13-7: REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER TRANSMISSION

| 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




Note: Sync master mode; BRG = 0. Continuous transmission of two 8-bit words.

### FIGURE 13-10: SYNCHRONOUS TRANSMISSION (THROUGH TXEN)

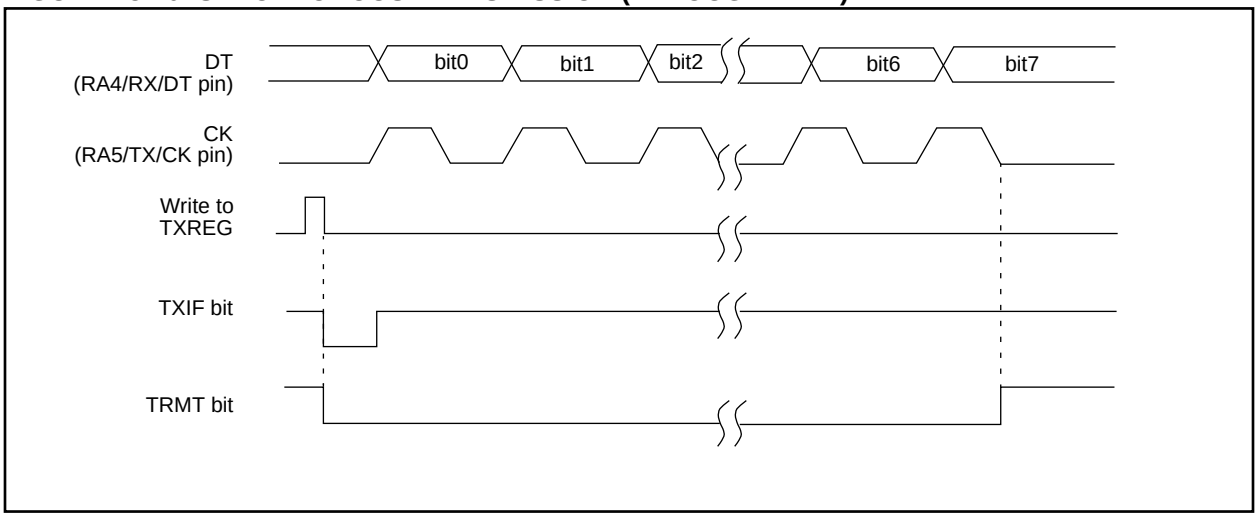


Table 15-2 lists the instructions recognized by the MPASM assembler.

**Note 1:** Any unused opcode is Reserved. Use of any reserved opcode may cause unexpected operation.

**Note 2:** The shaded instructions are not available in the PIC17C42

All instruction examples use the following format to represent a hexadecimal number:

0xhh

where h signifies a hexadecimal digit.

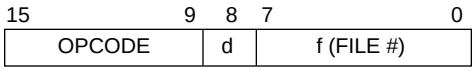
To represent a binary number:

`0000 0100b`

where b signifies a binary string.

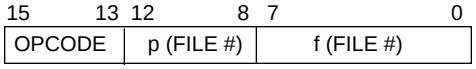
**FIGURE 15-1: GENERAL FORMAT FOR INSTRUCTIONS**

**Byte-oriented** file register operations



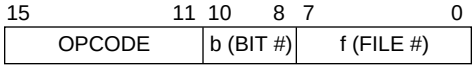

d = 0 for destination WREG
d = 1 for destination f
f = 8-bit file register address

**Byte to Byte** move operations



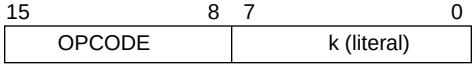

p = peripheral register file address
f = 8-bit file register address

**Bit-oriented** file register operations



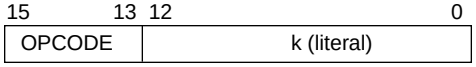

b = 3-bit address
f = 8-bit file register address

**Literal and control** operations




k = 8-bit immediate value

**Call and GOTO** operations




k = 13-bit immediate value

## 15.1 Special Function Registers as Source/Destination

The PIC17C4X's orthogonal instruction set allows read and write of all file registers, including special function registers. There are some special situations the user should be aware of:

### 15.1.1 ALUSTA AS DESTINATION

If an instruction writes to ALUSTA, the Z, C, DC and OV bits may be set or cleared as a result of the instruction and overwrite the original data bits written. For example, executing `CLRF ALUSTA` will clear register ALUSTA, and then set the Z bit leaving `0000 0100b` in the register.

### 15.1.2 PCL AS SOURCE OR DESTINATION

Read, write or read-modify-write on PCL may have the following results:

| | |
|---|---|
| Read PC: | PCH → PCLATH; PCL → dest |
| Write PCL: | PCLATH → PCH;<br>8-bit destination value → PCL |
| Read-Modify-Write: | PCL→ ALU operand<br>PCLATH → PCH;<br>8-bit result → PCL |

Where PCH = program counter high byte (not an addressable register), PCLATH = Program counter high holding latch, dest = destination, WREG or f.

### 15.1.3 BIT MANIPULATION

All bit manipulation instructions are done by first reading the entire register, operating on the selected bit and writing the result back (read-modify-write). The user should keep this in mind when operating on special function registers, such as ports.

**ANDWF** **AND WREG with f**

Syntax: [ *label* ] ANDWF f,d

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

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: $0 \le f \le 255$
$0 \le b \le 7$

Operation: 0 → (f<b>)

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

**INCF** **Increment f**

Syntax: [ *label* ] INCF f,d

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

Operation: (f) + 1 → (dest)

Status Affected: OV, C, DC, Z

Encoding:

| | | | |
|---|---|---|---|
| 0001 | 010d | ffff | ffff |

Description: The contents of register 'f' are incremented. 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:
```
INCF    CNT, 1
```
Before Instruction
CNT = 0xFF
Z = 0
C = ?

After Instruction
CNT = 0x00
Z = 1
C = 1

**INCFSZ** **Increment f, skip if 0**

Syntax: [ *label* ] INCFSZ f,d

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

Operation: (f) + 1 → (dest)
skip if result = 0

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 0001 | 111d | ffff | ffff |

Description: The contents of register 'f' are incremented. 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 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    INCFSZ  CNT, 1
NZERO   :
ZERO    :
```
Before Instruction
PC = Address (HERE)

After Instruction
CNT = CNT + 1
If CNT = 0;
PC = Address(ZERO)
If CNT ≠ 0;
PC = Address(NZERO)

**RETURN** **Return from Subroutine**

Syntax: [ *label* ] RETURN

Operands: None

Operation: TOS → PC;

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 0000 | 0000 | 0000 | 0010 |

Description: Return from subroutine. The stack is popped and the top of the stack (TOS) is loaded into the program counter.

Words: 1

Cycles: 2

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read register PCL* | Execute | NOP |
| Forced NOP | NOP | Execute | NOP |

* Remember reading PCL causes PCLATH to be updated. This will be the high address of where the RETURN instruction is located.

Example: RETURN

After Interrupt
PC = TOS

**RLCF** **Rotate Left f through Carry**

Syntax: [ *label* ] RLCF f,d

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

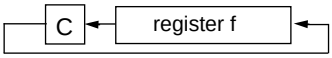
Operation: f<n> → d<n+1>;
f<7> → C;
C → d<0>

Status Affected: C

Encoding:

| | | | |
|---|---|---|---|
| 0001 | 101d | ffff | ffff |

Description: The contents of register 'f' are rotated one bit to the left through the Carry Flag. If 'd' is 0 the result is placed in 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:
```
RLCF    REG,0
```

Before Instruction
```
REG  =  1110 0110
C    =  0
```

After Instruction
```
REG  =  1110 0110
WREG =  1100 1100
C    =  1
```

**SWAPF** **Swap f**

Syntax: [ *label* ] SWAPF f,d

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

Operation: f<3:0> → dest<7:4>;
f<7:4> → dest<3:0>

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 0001 | 110d | ffff | ffff |

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

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: `SWAPF REG, 0`

Before Instruction
REG = 0x53

After Instruction
REG = 0x35

**TABLRD** **Table Read**

Syntax: [ *label* ] TABLRD t,i,f

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

Operation: If t = 1,
TBLATH → f;
If t = 0,
TBLATL → f;
Prog Mem (TBLPTR) → TBLAT;
If i = 1,
TBLPTR + 1 → TBLPTR

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 1010 | 10ti | ffff | ffff |

Description:
1. A byte of the table latch (TBLAT) is moved to register file 'f'.
If t = 0: the high byte is moved;
If t = 1: the low byte is moved
2. Then the contents of the program memory location pointed to by the 16-bit Table Pointer (TBLPTR) is loaded into the 16-bit Table Latch (TBLAT).
3. If i = 1: TBLPTR is incremented;
If i = 0: TBLPTR is not incremented

Words: 1

Cycles: 2 (3 cycle if f = PCL)

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read register TBLATH or TBLATL | Execute | Write register 'f' |

### 16.6 PICDEM-1 Low-Cost PIC16/17 Demonstration Board

The PICDEM-1 is a simple board which demonstrates the capabilities of several of Microchip's microcontrollers. The microcontrollers supported are: PIC16C5X (PIC16C54 to PIC16C58A), PIC16C61, PIC16C62X, PIC16C71, PIC16C8X, PIC17C42, PIC17C43 and PIC17C44. All necessary hardware and software is included to run basic demo programs. The users can program the sample microcontrollers provided with the PICDEM-1 board, on a PRO MATE II or PICSTART-16B programmer, and easily test firmware. The user can also connect the PICDEM-1 board to the PICMASTER emulator and download the firmware to the emulator for testing. Additional prototype area is available for the user to build some additional hardware and connect it to the microcontroller socket(s). Some of the features include an RS-232 interface, a potentiometer for simulated analog input, push-button switches and eight LEDs connected to PORTB.

### 16.7 PICDEM-2 Low-Cost PIC16CXX Demonstration Board

The PICDEM-2 is a simple demonstration board that supports the PIC16C62, PIC16C64, PIC16C65, PIC16C73 and PIC16C74 microcontrollers. All the necessary hardware and software is included to run the basic demonstration programs. The user can program the sample microcontrollers provided with the PICDEM-2 board, on a PRO MATE II programmer or PICSTART-16C, and easily test firmware. The PICMASTER emulator may also be used with the PICDEM-2 board to test firmware. Additional prototype area has been provided to the user for adding additional hardware and connecting it to the microcontroller socket(s). Some of the features include a RS-232 interface, push-button switches, a potentiometer for simulated analog input, a Serial EEPROM to demonstrate usage of the $I^2C$ bus and separate headers for connection to an LCD module and a keypad.

### 16.8 PICDEM-3 Low-Cost PIC16CXXX Demonstration Board

The PICDEM-3 is a simple demonstration board that supports the PIC16C923 and PIC16C924 in the PLCC package. It will also support future 44-pin PLCC microcontrollers with a LCD Module. All the necessary hardware and software is included to run the basic demonstration programs. The user can program the sample microcontrollers provided with the PICDEM-3 board, on a PRO MATE II programmer or PICSTART Plus with an adapter socket, and easily test firmware. The PICMASTER emulator may also be used with the PICDEM-3 board to test firmware. Additional prototype area has been provided to the user for adding hardware and connecting it to the microcontroller socket(s). Some of the features include an RS-232 interface, push-button switches, a potentiometer for simulated analog input, a thermistor and separate headers for connection to an external LCD module and a keypad. Also provided on the PICDEM-3 board is an LCD panel, with 4 commons and 12 segments, that is capable of displaying time, temperature and day of the week. The PICDEM-3 provides an additional RS-232 interface and Windows 3.1 software for showing the demultiplexed LCD signals on a PC. A simple serial interface allows the user to construct a hardware demultiplexer for the LCD signals. PICDEM-3 will be available in the 3rd quarter of 1996.

### 16.9 MPLAB Integrated Development Environment Software

The MPLAB IDE Software brings an ease of software development previously unseen in the 8-bit microcontroller market. MPLAB is a windows based application which contains:

- A full featured editor
- Three operating modes
  - editor
  - emulator
  - simulator
- A project manager
- Customizable tool bar and key mapping
- A status bar with project information
- Extensive on-line help

MPLAB allows you to:

- Edit your source files (either assembly or 'C')
- One touch assemble (or compile) and download to PIC16/17 tools (automatically updates all project information)
- Debug using:
  - source files
  - absolute listing file
- Transfer data dynamically via DDE (soon to be replaced by OLE)
- Run up to four emulators on the same PC

The ability to use MPLAB with Microchip's simulator allows a consistent platform and the ability to easily switch from the low cost simulator to the full featured emulator with minimal retraining due to development tools.

### 16.10 Assembler (MPASM)

The MPASM Universal Macro Assembler is a PC-hosted symbolic assembler. It supports all microcontroller series including the PIC12C5XX, PIC14000, PIC16C5X, PIC16CXXX, and PIC17CXX families.

MPASM offers full featured Macro capabilities, conditional assembly, and several source and listing formats. It generates various object code formats to support Microchip's development tools as well as third party programmers.

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

**TABLE 17-1: CROSS REFERENCE OF DEVICE SPECS FOR OSCILLATOR CONFIGURATIONS AND FREQUENCIES OF OPERATION (COMMERCIAL DEVICES)**

| OSC | PIC17C42-16 | PIC17C42-25 |
|---|---|---|
| RC | VDD: 4.5V to 5.5V<br>IDD: 6 mA max.<br>IPD: 5 μA max. at 5.5V (WDT disabled)<br>Freq: 4 MHz max. | VDD: 4.5V to 5.5V<br>IDD: 6 mA max.<br>IPD: 5 μA max. at 5.5V (WDT disabled)<br>Freq: 4 MHz max. |
| XT | VDD: 4.5V to 5.5V<br>IDD: 24 mA max.<br>IPD: 5 μA max. at 5.5V (WDT disabled)<br>Freq: 16 MHz max. | VDD: 4.5V to 5.5V<br>IDD: 38 mA max.<br>IPD: 5 μA max. at 5.5V (WDT disabled)<br>Freq: 25 MHz max. |
| EC | VDD: 4.5V to 5.5V<br>IDD: 24 mA max.<br>IPD: 5 μA max. at 5.5V (WDT disabled)<br>Freq: 16 MHz max. | VDD: 4.5V to 5.5V<br>IDD: 38 mA max.<br>IPD: 5 μA max. at 5.5V (WDT disabled)<br>Freq: 25 MHz max. |
| LF | VDD: 4.5V to 5.5V<br>IDD: 150 μA max. at 32 kHz (WDT enabled)<br>IPD: 5 μA max. at 5.5V (WDT disabled)<br>Freq: 2 MHz max. | VDD: 4.5V to 5.5V<br>IDD: 150 μA max. at 32 kHz (WDT enabled)<br>IPD: 5 μA max. at 5.5V (WDT disabled)<br>Freq: 2 MHz max. |

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

| DC CHARACTERISTICS | | | Standard Operating Conditions (unless otherwise stated)<br>Operating temperature<br>-40°C ≤ TA ≤ +40°C<br>Operating voltage VDD range as described in Section 17.1 | | | | |
|---|---|---|---|---|---|---|---|
| **Parameter No.** | **Sym** | **Characteristic** | **Min** | **Typ†** | **Max** | **Units** | **Conditions** |
| | | **Internal Program Memory Programming Specs** (Note 4) | | | | | |
| D110 | VPP | Voltage on $\overline{MCLR}$/VPP pin | 12.75 | – | 13.25 | V | Note 5 |
| D111 | VDDP | Supply voltage during programming | 4.75 | 5.0 | 5.25 | V | |
| D112 | IPP | Current into $\overline{MCLR}$/VPP pin | – | 25 ‡ | 50 ‡ | mA | |
| D113 | IDDP | Supply current during programming | – | – | 30 ‡ | mA | |
| D114 | TPROG | Programming pulse width | 10 | 100 | 1000 | µs | Terminated via internal/external interrupt or a reset |

* These parameters are characterized but not tested.

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

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

Note 1: In RC oscillator configuration, the OSC1/CLKIN pin is a Schmitt Trigger input. It is not recommended that the PIC17CXX devices be driven with external clock in RC mode.

2: The leakage current on the $\overline{MCLR}$ 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.

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

| 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 | $\overline{RD}$ |
| dt | Data in | rw | $\overline{RD}$ or $\overline{WR}$ |
| in | INT pin | t0 | T0CKI |
| io | I/O port | t123 | TCLK12 and TCLK3 |
| mc | $\overline{MCLR}$ | wdt | Watchdog Timer |
| oe | $\overline{OE}$ | wr | $\overline{WR}$ |
| 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 |
|---|---|---|---|---|---|---|

## 19.4 Timing Parameter Symbology

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

1. TppS2ppS
2. TppS
3. TCC:ST ($I^2C$ specifications only)
4. Ts ($I^2C$ specifications only)

| 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 | $\overline{RD}$ |
| dt | Data in | rw | $\overline{RD}$ or $\overline{WR}$ |
| in | INT pin | t0 | T0CKI |
| io | I/O port | t123 | TCLK12 and TCLK3 |
| mc | $\overline{MCLR}$ | wdt | Watchdog Timer |
| oe | $\overline{OE}$ | wr | $\overline{WR}$ |
| 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 19-4: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER, AND POWER-UP TIMER TIMING**

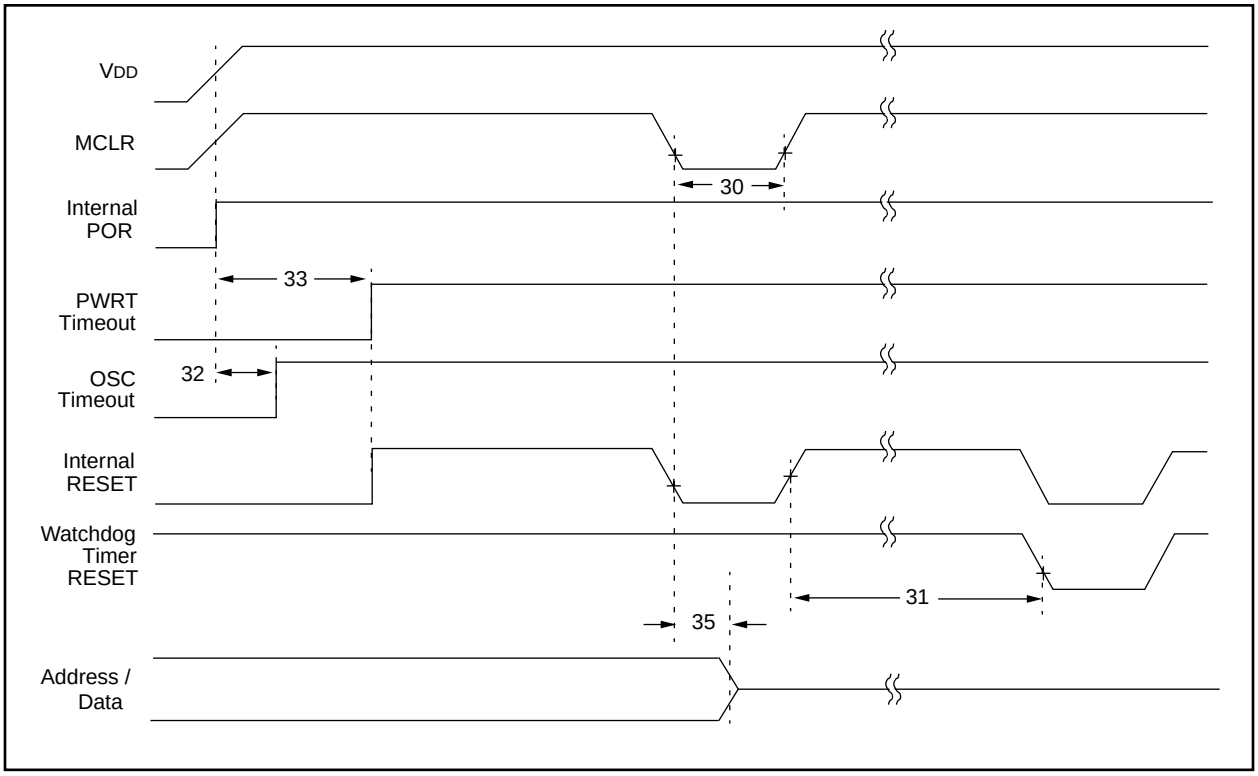



**TABLE 19-4: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER REQUIREMENTS**

| Parameter No. | Sym | Characteristic | | Min | Typ† | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|---|
| 30 | TmcL | $\overline{\text{MCLR}}$ Pulse Width (low) | | 100 * | — | — | ns | VDD = 5V |
| 31 | Twdt | Watchdog Timer Time-out Period (Prescale = 1) | | 5 * | 12 | 25 * | ms | VDD = 5V |
| 32 | Tost | Oscillation Start-up Timer Period | | — | 1024TOSC§ | — | ms | TOSC = OSC1 period |
| 33 | Tpwrt | Power-up Timer Period | | 40 * | 96 | 200 * | ms | VDD = 5V |
| 35 | TmcL2adl | $\overline{\text{MCLR}}$ to System Interface bus (AD15:AD0>) invalid | PIC17CR42/42A/43/R43/44 | — | — | 100 * | ns | |
| | | | PIC17LCR42/42A/43/R43/44 | — | — | 120 * | 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 20-19: $V_{TH}$, $V_{IL}$ of I/O PINS (SCHMITT TRIGGER) vs. $V_{DD}$**

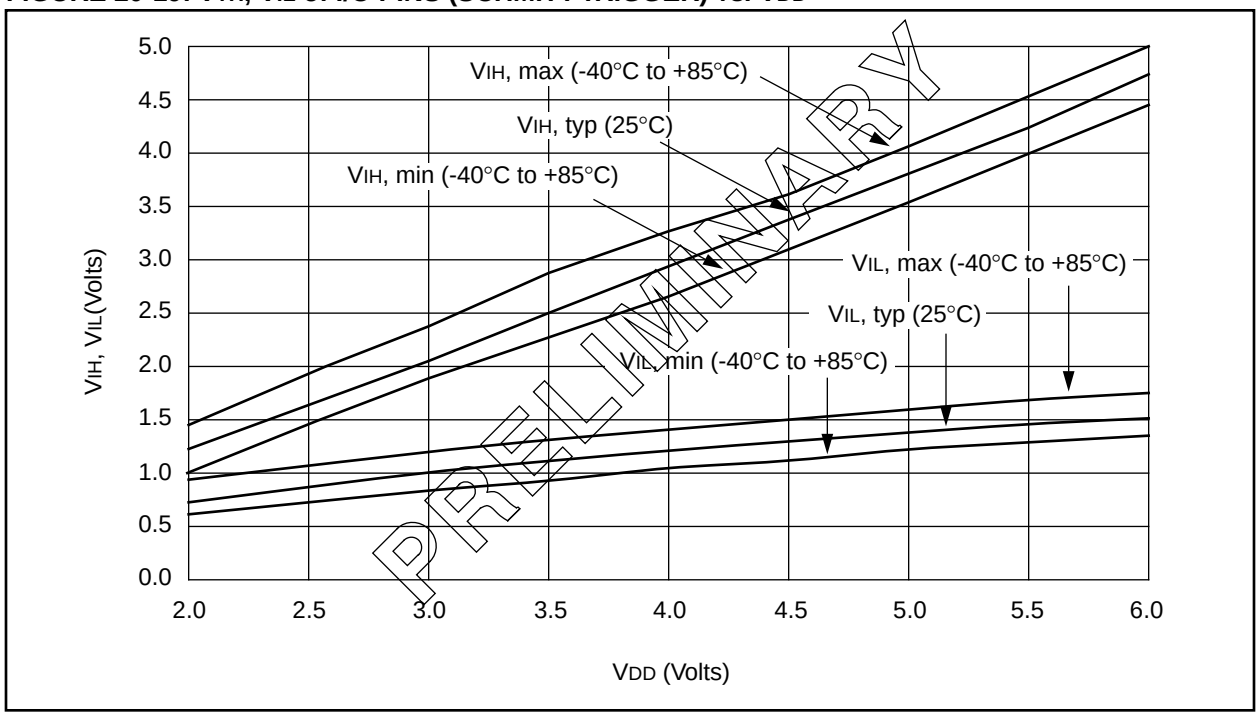
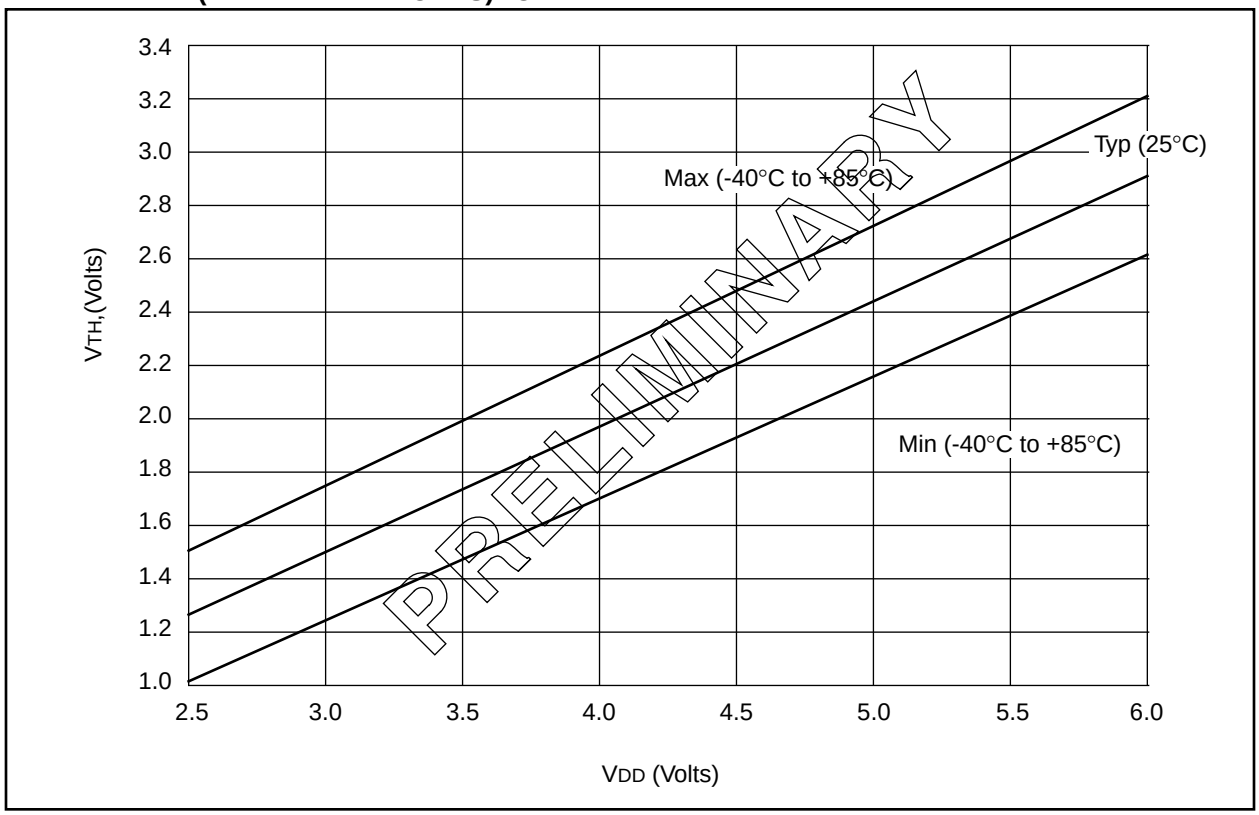



**FIGURE 20-20: $V_{TH}$ (INPUT THRESHOLD VOLTAGE) OF OSC1 INPUT (IN XT AND LF MODES) vs. $V_{DD}$**



## E.7 PIC16C9XX Family Of Devices

| | Clock | Memory | | | Peripherals | | | | | | | Features | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Maximum Frequency of Operation (MHz) | Program Memory | EPROM | Data Memory (bytes) | Timer Module(s) | Capture/Compare/PWM Module(s) | Serial Port(s) (SPI/I²C, USART) | Parallel Slave Port | A/D Converter (8-bit) Channels | LCD Module | Interrupt Sources | I/O Pins | Input Pins | Voltage Range (Volts) | In-Circuit Serial Programming | Brown-out Reset | Packages |
| PIC16C923 | 8 | 4K | 176 | | TMR0, TMR1, TMR2 | 1 | SPI/I²C | — | — | 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 | 1 | SPI/I²C | — | 5 | 4 Com 32 Seg | 9 | 25 | 27 | 3.0-6.0 | Yes | — | 64-pin SDIP(1), TQFP, 68-pin PLCC, DIE |

All PIC16/17 Family devices have Power-on Reset, selectable Watchdog Timer, selectable code protect and high I/O current capability.
All PIC16CXX Family devices use serial programming with clock pin RB6 and data pin RB7.

Note 1: Please contact your local Microchip representative for availability of this package.

Figure 6-12: Program Counter using The CALL and GOTO Instructions ..... 41
Figure 6-13: BSR Operation (PIC17C43/R43/44) ..... 42
Figure 7-1: TLWT Instruction Operation ..... 43
Figure 7-2: TABLWT Instruction Operation ..... 43
Figure 7-3: TLRD Instruction Operation ..... 44
Figure 7-4: TABLRD Instruction Operation ..... 44
Figure 7-5: TABLWT Write Timing (External Memory) ..... 46
Figure 7-6: Consecutive TABLWT Write Timing (External Memory) ..... 47
Figure 7-7: TABLRD Timing ..... 48
Figure 7-8: TABLRD Timing (Consecutive TABLRD Instructions) ..... 48
Figure 9-1: RA0 and RA1 Block Diagram ..... 53
Figure 9-2: RA2 and RA3 Block Diagram ..... 54
Figure 9-3: RA4 and RA5 Block Diagram ..... 54
Figure 9-4: Block Diagram of RB<7:4> and RB<1:0> Port Pins ..... 55
Figure 9-5: Block Diagram of RB3 and RB2 Port Pins .. 56
Figure 9-6: Block Diagram of RC<7:0> Port Pins ..... 58
Figure 9-7: PORTD Block Diagram (in I/O Port Mode) ..... 60
Figure 9-8: PORTE Block Diagram (in I/O Port Mode) ..... 62
Figure 9-9: Successive I/O Operation ..... 64
Figure 11-1: T0STA Register (Address: 05h, Unbanked) ..... 67
Figure 11-2: Timer0 Module Block Diagram ..... 68
Figure 11-3: TMR0 Timing with External Clock (Increment on Falling Edge) ..... 68
Figure 11-4: TMR0 Timing: Write High or Low Byte ..... 69
Figure 11-5: TMR0 Read/Write in Timer Mode ..... 70
Figure 12-1: TCON1 Register (Address: 16h, Bank 3) ... 71
Figure 12-2: TCON2 Register (Address: 17h, Bank 3) ... 72
Figure 12-3: Timer1 and Timer2 in Two 8-bit Timer/Counter Mode ..... 73
Figure 12-4: TMR1 and TMR2 in 16-bit Timer/Counter Mode ..... 74
Figure 12-5: Simplified PWM Block Diagram ..... 75
Figure 12-6: PWM Output ..... 75
Figure 12-7: Timer3 with One Capture and One Period Register Block Diagram ..... 78
Figure 12-8: Timer3 with Two Capture Registers Block Diagram ..... 79
Figure 12-9: TMR1, TMR2, and TMR3 Operation in External Clock Mode ..... 80
Figure 12-10: TMR1, TMR2, and TMR3 Operation in Timer Mode ..... 81
Figure 13-1: TXSTA Register (Address: 15h, Bank 0) .... 83
Figure 13-2: RCSTA Register (Address: 13h, Bank 0) ... 84
Figure 13-3: USART Transmit ..... 85
Figure 13-4: USART Receive ..... 85
Figure 13-5: Asynchronous Master Transmission ..... 90
Figure 13-6: Asynchronous Master Transmission (Back to Back) ..... 90
Figure 13-7: RX Pin Sampling Scheme ..... 91
Figure 13-8: Asynchronous Reception ..... 92
Figure 13-9: Synchronous Transmission ..... 94
Figure 13-10: Synchronous Transmission (Through TXEN) ..... 94
Figure 13-11: Synchronous Reception (Master Mode, SREN) ..... 95
Figure 14-1: Configuration Word ..... 99
Figure 14-2: Crystal or Ceramic Resonator Operation (XT or LF OSC Configuration) ..... 100
Figure 14-3: Crystal Operation, Overtone Crystals (XT OSC Configuration) ..... 101
Figure 14-4: External Clock Input Operation (EC OSC Configuration) ..... 101
Figure 14-5: External Parallel Resonant Crystal Oscillator Circuit ..... 102
Figure 14-6: External Series Resonant Crystal Oscillator Circuit ..... 102
Figure 14-7: RC Oscillator Mode ..... 102
Figure 14-8: Watchdog Timer Block Diagram ..... 104
Figure 14-9: Wake-up From Sleep Through Interrupt ... 105
Figure 15-1: General Format for Instructions ..... 108
Figure 15-2: Q Cycle Activity ..... 109
Figure 17-1: Parameter Measurement Information ..... 154
Figure 17-2: External Clock Timing ..... 155
Figure 17-3: CLKOUT and I/O Timing ..... 156
Figure 17-4: Reset, Watchdog Timer, Oscillator Start-Up Timer and Power-Up Timer Timing ..... 157
Figure 17-5: Timer0 Clock Timings ..... 158
Figure 17-6: Timer1, Timer2, And Timer3 Clock Timings ..... 158
Figure 17-7: Capture Timings ..... 159
Figure 17-8: PWM Timings ..... 159
Figure 17-9: USART Module: Synchronous Transmission (Master/Slave) Timing ..... 160
Figure 17-10: USART Module: Synchronous Receive (Master/Slave) Timing ..... 160
Figure 17-11: Memory Interface Write Timing ..... 161
Figure 17-12: Memory Interface Read Timing ..... 162
Figure 18-1: Typical RC Oscillator Frequency vs. Temperature ..... 163
Figure 18-2: Typical RC Oscillator Frequency vs. VDD ..... 164
Figure 18-3: Typical RC Oscillator Frequency vs. VDD ..... 164
Figure 18-4: Typical RC Oscillator Frequency vs. VDD ..... 165
Figure 18-5: Transconductance (gm) of LF Oscillator vs. VDD ..... 166
Figure 18-6: Transconductance (gm) of XT Oscillator vs. VDD ..... 166
Figure 18-7: Typical IDD vs. Frequency (External Clock 25°C) ..... 167
Figure 18-8: Maximum IDD vs. Frequency (External Clock 125°C to -40°C) ..... 167
Figure 18-9: Typical IPD vs. VDD Watchdog Disabled 25°C ..... 168
Figure 18-10: Maximum IPD vs. VDD Watchdog Disabled ..... 168
Figure 18-11: Typical IPD vs. VDD Watchdog Enabled 25°C ..... 169
Figure 18-12: Maximum IPD vs. VDD Watchdog Enabled ..... 169
Figure 18-13: WDT Timer Time-Out Period vs. VDD ..... 170
Figure 18-14: IOH vs. VOH, VDD = 3V ..... 170
Figure 18-15: IOH vs. VOH, VDD = 5V ..... 171
Figure 18-16: IOL vs. VOL, VDD = 3V ..... 171
Figure 18-17: IOL vs. VOL, VDD = 5V ..... 172
Figure 18-18: VTH (Input Threshold Voltage) of I/O Pins (TTL) vs. VDD ..... 172
Figure 18-19: VTH, VIL of I/O Pins (Schmitt Trigger) vs. VDD ..... 173
Figure 18-20: VTH (Input Threshold Voltage) of OSC1 Input (In XT and LF Modes) vs. VDD ..... 173
Figure 19-1: Parameter Measurement Information ..... 183