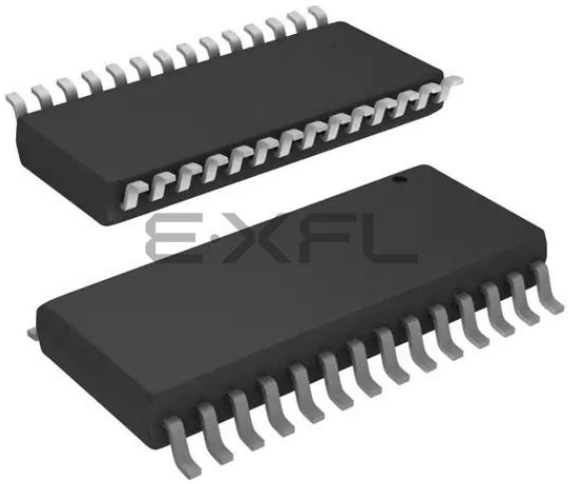


**Welcome to E-XFL.COM**

## 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 | Active |
| Core Processor | PIC |
| Core Size | 8-Bit |
| Speed | 40MHz |
| Connectivity | I²C, SPI, UART/USART |
| Peripherals | Brown-out Detect/Reset, LVD, POR, PWM, WDT |
| Number of I/O | 25 |
| Program Memory Size | 8KB (4K x 16) |
| Program Memory Type | FLASH |
| EEPROM Size | 256 x 8 |
| RAM Size | 512 x 8 |
| Voltage - Supply (Vcc/Vdd) | 2V ~ 5.5V |
| Data Converters | A/D 10x10b |
| Oscillator Type | Internal |
| Operating Temperature | -40°C ~ 85°C (TA) |
| Mounting Type | Surface Mount |
| Package / Case | 28-SOIC (0.295", 7.50mm Width) |
| Supplier Device Package | 28-SOIC |
| Purchase URL | https://www.e-xfl.com/product-detail/microchip-technology/pic18lf2320t-i-so |

## TO OUR VALUED CUSTOMERS

It is our intention to provide our valued customers with the best documentation possible to ensure successful use of your Microchip products. To this end, we will continue to improve our publications to better suit your needs. Our publications will be refined and enhanced as new volumes and updates are introduced.

If you have any questions or comments regarding this publication, please contact the Marketing Communications Department via E-mail at **docerrors@mail.microchip.com** or fax the **Reader Response Form** in the back of this data sheet to (480) 792-4150. We welcome your feedback.

### Most Current Data Sheet

To obtain the most up-to-date version of this data sheet, please register at our Worldwide Web site at:

http://www.microchip.com

You can determine the version of a data sheet by examining its literature number found on the bottom outside corner of any page. The last character of the literature number is the version number, (e.g., DS30000A is version A of document DS30000).

### Errata

An errata sheet, describing minor operational differences from the data sheet and recommended workarounds, may exist for current devices. As device/documentation issues become known to us, we will publish an errata sheet. The errata will specify the revision of silicon and revision of document to which it applies.

To determine if an errata sheet exists for a particular device, please check with one of the following:

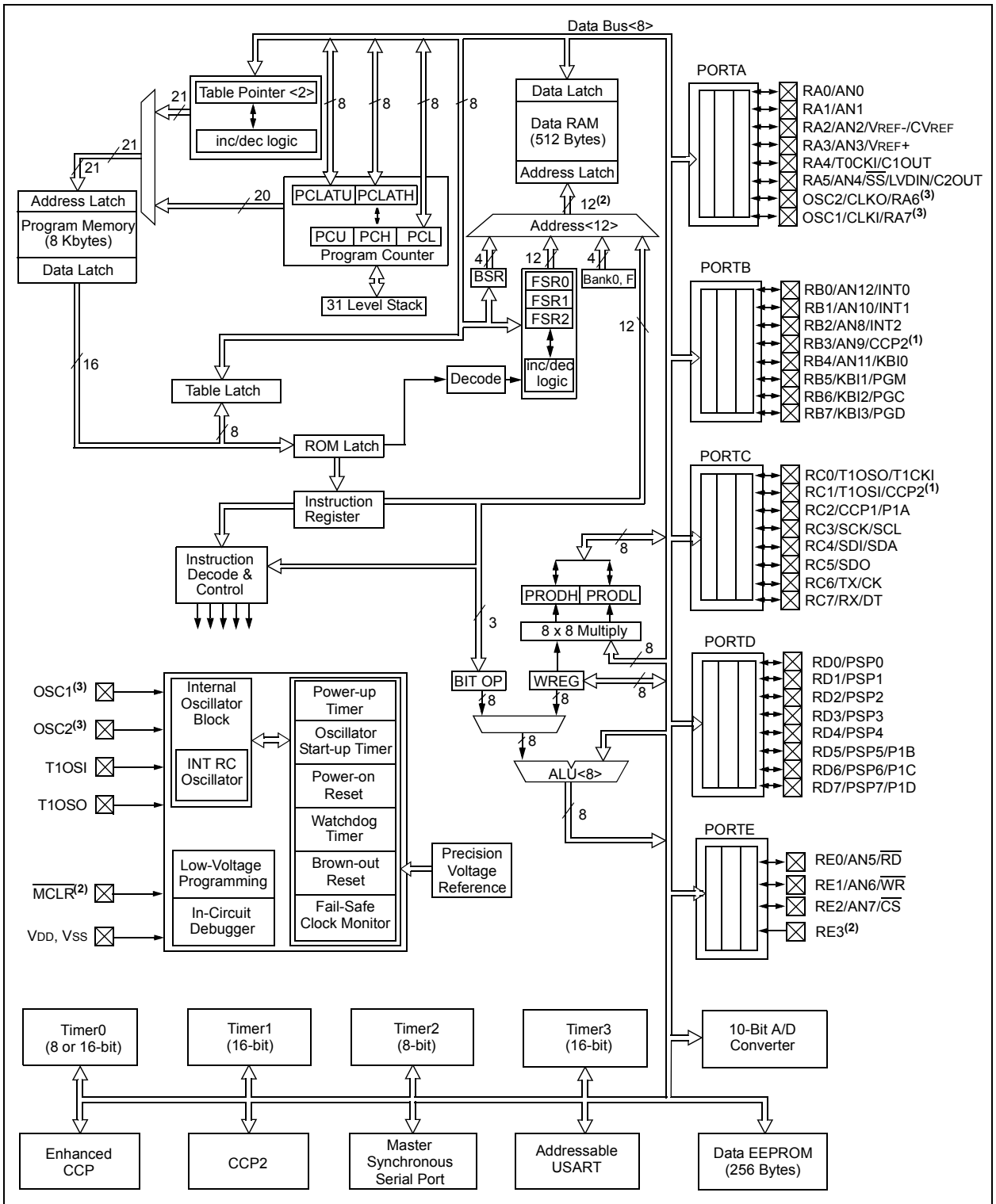
- Microchip's Worldwide Web site; http://www.microchip.com
- Your local Microchip sales office (see last page)
- The Microchip Corporate Literature Center; U.S. FAX: (480) 792-7277

When contacting a sales office or the literature center, please specify which device, revision of silicon and data sheet (include literature number) you are using.

### Customer Notification System

Register on our web site at **www.microchip.com/cn** to receive the most current information on all of our products.

**FIGURE 1-2: PIC18F4220/4320 BLOCK DIAGRAM**




**Note 1:** Optional multiplexing of CCP2 input/output with RB3 is enabled by selection of the CCP2MX Configuration bit.

**2:** RE3 is available only when the MCLR Resets are disabled.

**3:** OSC1, OSC2, CLKI and CLKO are only available in select oscillator modes and when these pins are not being used as digital I/O. Refer to **Section 2.0 "Oscillator Configurations"** for additional information.

### 5.9.2 SPECIAL FUNCTION REGISTERS

The Special Function Registers (SFRs) are registers used by the CPU and peripheral modules for controlling the desired operation of the device. These registers are implemented as static RAM. A list of these registers is given in Table 5-1 and Table 5-2.

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 in this section, while those related to the operation of the peripheral features are described in the section of that peripheral feature.

The SFRs are typically distributed among the peripherals whose functions they control.

The unused SFR locations will be unimplemented and read as '0's.

**TABLE 5-1: SPECIAL FUNCTION REGISTER MAP FOR PIC18F2X20/4X20 DEVICES**

| Address | Name | Address | Name | Address | Name | Address | Name |
|---|---|---|---|---|---|---|---|
| FFFh | TOSU | FDFh | INDF2(2) | FBFh | CCPR1H | F9Fh | IPR1 |
| FFEh | TOSH | FDEh | POSTINC2(2) | FBEh | CCPR1L | F9Eh | PIR1 |
| FFDh | TOSL | FDDh | POSTDEC2(2) | FBDh | CCP1CON | F9Dh | PIE1 |
| FFCh | STKPTR | FDCh | PREINC2(2) | FBCh | CCPR2H | F9Ch | — |
| FFBh | PCLATU | FDBh | PLUSW2(2) | FBBh | CCPR2L | F9Bh | OSCTUNE |
| FFAh | PCLATH | FDAh | FSR2H | FBAh | CCP2CON | F9Ah | OSCTUN2 |
| FF9h | PCL | FD9h | FSR2L | FB9h | — | F99h | — |
| FF8h | TBLPTRU | FD8h | STATUS | FB8h | — | F98h | — |
| FF7h | TBLPTRH | FD7h | TMR0H | FB7h | PWM1CON(1) | F97h | — |
| FF6h | TBLPTRL | FD6h | TMR0L | FB6h | ECCPAS(1) | F96h | TRISE(1) |
| FF5h | TABLAT | FD5h | T0CON | FB5h | CVRCON | F95h | TRISD(1) |
| FF4h | PRODH | FD4h | — | FB4h | CMCON | F94h | TRISC |
| FF3h | PRODL | FD3h | OSCCON | FB3h | TMR3H | F93h | TRISB |
| FF2h | INTCON | FD2h | LVDCON | FB2h | TMR3L | F92h | TRISA |
| FF1h | INTCON2 | FD1h | WDTCON | FB1h | T3CON | F91h | — |
| FF0h | INTCON3 | FD0h | RCON | FB0h | — | F90h | — |
| FEFh | INDF0(2) | FCFh | TMR1H | FAFh | SPBRG | F8Fh | — |
| FEEh | POSTINC0(2) | FCEh | TMR1L | FAEh | RCREG | F8Eh | — |
| FEDh | POSTDEC0(2) | FCDh | T1CON | FADh | TXREG | F8Dh | LATE(1) |
| FECh | PREINC0(2) | FCCh | TMR2 | FACh | TXSTA | F8Ch | LATD(1) |
| FEBh | PLUSW0(2) | FCBh | PR2 | FABh | RCSTA | F8Bh | LATC |
| FEAh | FSR0H | FCAh | T2CON | FAAh | — | F8Ah | LATB |
| FE9h | FSR0L | FC9h | SSPBUF | FA9h | EEADR | F89h | LATA |
| FE8h | WREG | FC8h | SSPADD | FA8h | EEDATA | F88h | — |
| FE7h | INDF1(2) | FC7h | SSPSTAT | FA7h | EECON2 | F87h | — |
| FE6h | POSTINC1(2) | FC6h | SSPCON1 | FA6h | EECON1 | F86h | — |
| FE5h | POSTDEC1(2) | FC5h | SSPCON2 | FA5h | — | F85h | — |
| FE4h | PREINC1(2) | FC4h | ADRESH | FA4h | — | F84h | PORTE(1) |
| FE3h | PLUSW1(2) | FC3h | ADRESL | FA3h | — | F83h | PORTD(1) |
| FE2h | FSR1H | FC2h | ADCON0 | FA2h | IPR2 | F82h | PORTC |
| FE1h | FSR1L | FC1h | ADCON1 | FA1h | PIR2 | F81h | PORTB |
| FE0h | BSR | FC0h | ADCON2 | FA0h | PIE2 | F80h | PORTA |

**Legend:** — = Unimplemented registers, read as '0'.

**Note 1:** This register is not available on PIC18F2X20 devices.

**2:** This is not a physical register.

## 9.5 RCON Register

The RCON register contains bits used to determine the cause of the last Reset or wake-up from power-managed mode. RCON also contains the bit that enables interrupt priorities (IPEN).

**REGISTER 9-10: RCON: RESET CONTROL REGISTER**

| R/W-0 | U-0 | U-0 | R/W-1 | R-1 | R-1 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| IPEN | — | — | $\overline{RI}$ | $\overline{TO}$ | $\overline{PD}$ | $\overline{POR}$ | $\overline{BOR}$ |
| bit 7 | | | | | | | bit 0 |

| **Legend:** | | | |
|---|---|---|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' | |
| -n = Value at POR | '1' = Bit is set | '0' = Bit is cleared | x = Bit is unknown |

bit 7 **IPEN:** Interrupt Priority Enable bit
1 = Enable priority levels on interrupts
0 = Disable priority levels on interrupts (PIC16CXXX Compatibility mode)

bit 6-5 **Unimplemented:** Read as '0'

bit 4 **$\overline{RI}$:** RESET Instruction Flag bit
1 = The RESET instruction was not executed (set by firmware only)
0 = The RESET instruction was executed causing a device Reset (must be set in software after a Brown-out Reset occurs)

bit 3 **$\overline{TO}$:** Watchdog Time-out Flag bit
1 = Set by power-up, CLRWDT instruction or SLEEP instruction
0 = A WDT time-out occurred

bit 2 **$\overline{PD}$:** Power-Down Detection Flag bit
1 = Set by power-up or by the CLRWDT instruction
0 = Cleared by execution of the SLEEP instruction

bit 1 **$\overline{POR}$:** Power-on Reset Status bit
1 = A Power-on Reset has not occurred (set by firmware only)
0 = A Power-on Reset occurred (must be set in software after a Power-on Reset occurs)

bit 0 **$\overline{BOR}$:** Brown-out Reset Status bit
1 = A Brown-out Reset has not occurred (set by firmware only)
0 = A Brown-out Reset occurred (must be set in software after a Brown-out Reset occurs)

## 15.1 CCP1 Module

Capture/Compare/PWM Register 1 (CCPR1) is comprised of two 8-bit registers: CCPR1L (low byte) and CCPR1H (high byte). The CCP1CON register controls the operation of CCP1. All are readable and writable.

**TABLE 15-1: CCP MODE - TIMER RESOURCE**

| CCP Mode | Timer Resource |
|---|---|
| Capture | Timer1 or Timer3 |
| Compare | Timer1 or Timer3 |
| PWM | Timer2 |

## 15.2 CCP2 Module

Capture/Compare/PWM Register 2 (CCPR2) is comprised of two 8-bit registers: CCPR2L (low byte) and CCPR2H (high byte). The CCP2CON register controls the operation of CCP2. All are readable and writable.

CCP2 functions identically to CCP1 except for the enhanced PWM modes offered by CCP2

**TABLE 15-2: INTERACTION OF TWO CCP MODULES**

| CCPx Mode | CCPy Mode | Interaction |
|---|---|---|
| Capture | Capture | TMR1 or TMR3 time base. Time base can be different for each CCP. |
| Capture | Compare | The compare could be configured for the Special Event Trigger which clears either TMR1 or TMR3 depending upon which time base is used. |
| Compare | Compare | The compare(s) could be configured for the Special Event Trigger which clears TMR1 or TMR3 depending upon which time base is used. |
| PWM | PWM | The PWMs will have the same frequency and update rate (TMR2 interrupt). |
| PWM | Capture | None. |
| PWM | Compare | None. |

## 19.3 Selecting and Configuring Automatic Acquisition Time

The ADCON2 register allows the user to select an acquisition time that occurs each time the GO/DONE bit is set.

When the GO/DONE bit is set, sampling is stopped and a conversion begins. The user is responsible for ensuring the required acquisition time has passed between selecting the desired input channel and setting the GO/DONE bit. This occurs when the ACQT2:ACQT0 bits (ADCON2<5:3>) remain in their Reset state ('000') and is compatible with devices that do not offer programmable acquisition times.

If desired, the ACQT bits can be set to select a programmable acquisition time for the A/D module. When the GO/DONE bit is set, the A/D module continues to sample the input for the selected acquisition time, then automatically begins a conversion. Since the acquisition time is programmed, there may be no need to wait for an acquisition time between selecting a channel and setting the GO/DONE bit.

In either case, when the conversion is completed, the GO/DONE bit is cleared, the ADIF flag is set and the A/D begins sampling the currently selected channel again. If an acquisition time is programmed, there is nothing to indicate if the acquisition time has ended or if the conversion has begun.

## 19.4 Selecting the A/D Conversion Clock

The A/D conversion time per bit is defined as TAD. The A/D conversion requires 11 TAD per 10-bit conversion. The source of the A/D conversion clock is software selectable. There are seven possible options for TAD:

- 2 TOSC
- 4 TOSC
- 8 TOSC
- 16 TOSC
- 32 TOSC
- 64 TOSC
- Internal RC Oscillator

For correct A/D conversions, the A/D conversion clock (TAD) must be as short as possible, but greater than the minimum TAD (approximately 2 μs, see parameter #130 for more information).

Table 19-1 shows the resultant TAD times derived from the device operating frequencies and the A/D clock source selected.

**TABLE 19-1: TAD vs. DEVICE OPERATING FREQUENCIES**

| AD Clock Source (TAD) | | Maximum Device Frequency | |
|---|---|---|---|
| **Operation** | **ADCS2:ADCS0** | **PIC18FXX20** | **PIC18LFXX20(4)** |
| 2 TOSC | 000 | 1.25 MHz | 666 kHz |
| 4 TOSC | 100 | 2.50 MHz | 1.33 MHz |
| 8 TOSC | 001 | 5.00 MHz | 2.66 MHz |
| 16 TOSC | 101 | 10.0 MHz | 5.33 MHz |
| 32 TOSC | 010 | 20.0 MHz | 10.65 MHz |
| 64 TOSC | 110 | 40.0 MHz | 21.33 MHz |
| RC(3) | x11 | 1.00 MHz(1) | 1.00 MHz(2) |

**Note 1:** The RC source has a typical TAD time of 4 μs.

**2:** The RC source has a typical TAD time of 6 μs.

**3:** For device frequencies above 1 MHz, the device must be in Sleep for the entire conversion or the A/D accuracy may be out of specification.

**4:** Low-power devices only.

**REGISTER 23-8: CONFIG6L: CONFIGURATION REGISTER 6 LOW (BYTE ADDRESS 30000Ah)**

| U-0 | U-0 | U-0 | U-0 | R/P-1 | R/P-1 | R/P-1 | R/P-1 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | WRT3(1) | WRT2(1) | WRT1 | WRT0 |
| bit 7 | | | | | | | bit 0 |

| Legend: | | |
|---|---|---|
| R = Readable bit | P = Programmable bit | U = Unimplemented bit, read as '0' |
| -n = Value when device is unprogrammed | | u = Unchanged from programmed state |

bit 7-4 **Unimplemented:** Read as '0'

bit 3 **WRT3:** Write Protection bit(1)
1 = Block 3 (001800-001FFFh) not write-protected
0 = Block 3 (001800-001FFFh) write-protected

bit 2 **WRT2:** Write Protection bit(1)
1 = Block 2 (001000-0017FFh) not write-protected
0 = Block 2 (001000-0017FFh) write-protected

bit 1 **WRT1:** Write Protection bit
1 = Block 1 (000800-000FFFh) not write-protected
0 = Block 1 (000800-000FFFh) write-protected

bit 0 **WRT0:** Write Protection bit
1 = Block 0 (000200-0007FFh) not write-protected
0 = Block 0 (000200-0007FFh) write-protected

**Note 1:** Unimplemented in PIC18FX220 devices; maintain this bit set.

**REGISTER 23-9: CONFIG6H: CONFIGURATION REGISTER 6 HIGH (BYTE ADDRESS 30000Bh)**

| R/P-1 | R/P-1 | R-1 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| WRTD | WRTB | WRTC(1) | — | — | — | — | — |
| bit 7 | | | | | | | bit 0 |

| Legend: | | |
|---|---|---|
| R = Readable bit | P = Programmable bit | U = Unimplemented bit, read as '0' |
| -n = Value when device is unprogrammed | | u = Unchanged from programmed state |

bit 7 **WRTD:** Data EEPROM Write Protection bit
1 = Data EEPROM is not write-protected
0 = Data EEPROM is write-protected

bit 6 **WRTB:** Boot Block Write Protection bit
1 = Boot block (000000-0001FFh) is not write-protected
0 = Boot block (000000-0001FFh) is write-protected

bit 5 **WRTC:** Configuration Register Write Protection bit(1)
1 = Configuration registers (300000-3000FFh) are not write-protected
0 = Configuration registers (300000-3000FFh) are write-protected

bit 4-0 **Unimplemented:** Read as '0'

**Note 1:** This bit is read-only in normal execution mode; it can be written only in Program mode.

**TABLE 24-2: PIC18FXXX INSTRUCTION SET**

| Mnemonic, Operands | | Description | Cycles | 16-Bit Instruction Word MSb | LSb | Status Affected | Notes |
|---|---|---|---|---|---|---|---|
| **BYTE-ORIENTED FILE REGISTER OPERATIONS** | | | | | | | |
| ADDWF | f, d, a | Add WREG and f | 1 | 0010 01da | ffff ffff | C, DC, Z, OV, N | 1, 2 |
| ADDWFC | f, d, a | Add WREG and Carry bit to f | 1 | 0010 00da | ffff ffff | C, DC, Z, OV, N | 1, 2 |
| ANDWF | f, d, a | AND WREG with f | 1 | 0001 01da | ffff ffff | Z, N | 1,2 |
| CLRF | f, a | Clear f | 1 | 0110 101a | ffff ffff | Z | 2 |
| COMF | f, d, a | Complement f | 1 | 0001 11da | ffff ffff | Z, N | 1, 2 |
| CPFSEQ | f, a | Compare f with WREG, Skip = | 1 (2 or 3) | 0110 001a | ffff ffff | None | 4 |
| CPFSGT | f, a | Compare f with WREG, Skip > | 1 (2 or 3) | 0110 010a | ffff ffff | None | 4 |
| CPFSLT | f, a | Compare f with WREG, Skip < | 1 (2 or 3) | 0110 000a | ffff ffff | None | 1, 2 |
| DECF | f, d, a | Decrement f | 1 | 0000 01da | ffff ffff | C, DC, Z, OV, N | 1, 2, 3, 4 |
| DECFSZ | f, d, a | Decrement f, Skip if 0 | 1 (2 or 3) | 0010 11da | ffff ffff | None | 1, 2, 3, 4 |
| DCFSNZ | f, d, a | Decrement f, Skip if Not 0 | 1 (2 or 3) | 0100 11da | ffff ffff | None | 1, 2 |
| INCF | f, d, a | Increment f | 1 | 0010 10da | ffff ffff | C, DC, Z, OV, N | 1, 2, 3, 4 |
| INCFSZ | f, d, a | Increment f, Skip if 0 | 1 (2 or 3) | 0011 11da | ffff ffff | None | 4 |
| INFSNZ | f, d, a | Increment f, Skip if Not 0 | 1 (2 or 3) | 0100 10da | ffff ffff | None | 1, 2 |
| IORWF | f, d, a | Inclusive OR WREG with f | 1 | 0001 00da | ffff ffff | Z, N | 1, 2 |
| MOVF | f, d, a | Move f | 1 | 0101 00da | ffff ffff | Z, N | 1 |
| MOVFF | $f_s$, $f_d$ | Move $f_s$ (source) to 1st word<br>$f_d$ (destination) 2nd word | 2 | 1100 ffff<br>1111 ffff | ffff ffff<br>ffff ffff | None | |
| MOVWF | f, a | Move WREG to f | 1 | 0110 111a | ffff ffff | None | |
| MULWF | f, a | Multiply WREG with f | 1 | 0000 001a | ffff ffff | None | |
| NEGF | f, a | Negate f | 1 | 0110 110a | ffff ffff | C, DC, Z, OV, N | 1, 2 |
| RLCF | f, d, a | Rotate Left f through Carry | 1 | 0011 01da | ffff ffff | C, Z, N | |
| RLNCF | f, d, a | Rotate Left f (No Carry) | 1 | 0100 01da | ffff ffff | Z, N | 1, 2 |
| RRCF | f, d, a | Rotate Right f through Carry | 1 | 0011 00da | ffff ffff | C, Z, N | |
| RRNCF | f, d, a | Rotate Right f (No Carry) | 1 | 0100 00da | ffff ffff | Z, N | |
| SETF | f, a | Set f | 1 | 0110 100a | ffff ffff | None | |
| SUBFWB | f, d, a | Subtract f from WREG with Borrow | 1 | 0101 01da | ffff ffff | C, DC, Z, OV, N | 1, 2 |
| SUBWF | f, d, a | Subtract WREG from f | 1 | 0101 11da | ffff ffff | C, DC, Z, OV, N | |
| SUBWFB | f, d, a | Subtract WREG from f with Borrow | 1 | 0101 10da | ffff ffff | C, DC, Z, OV, N | 1, 2 |
| SWAPF | f, d, a | Swap Nibbles in f | 1 | 0011 10da | ffff ffff | None | 4 |
| TSTFSZ | f, a | Test f, Skip if 0 | 1 (2 or 3) | 0110 011a | ffff ffff | None | 1, 2 |
| XORWF | f, d, a | Exclusive OR WREG with f | 1 | 0001 10da | ffff ffff | Z, N | |
| **BIT-ORIENTED FILE REGISTER OPERATIONS** | | | | | | | |
| BCF | f, b, a | Bit Clear f | 1 | 1001 bbba | ffff ffff | None | 1, 2 |
| BSF | f, b, a | Bit Set f | 1 | 1000 bbba | ffff ffff | None | 1, 2 |
| BTFSC | f, b, a | Bit Test f, Skip if Clear | 1 (2 or 3) | 1011 bbba | ffff ffff | None | 3, 4 |
| BTFSS | f, b, a | Bit Test f, Skip if Set | 1 (2 or 3) | 1010 bbba | ffff ffff | None | 3, 4 |
| BTG | f, d, a | Bit Toggle f | 1 | 0111 bbba | ffff ffff | None | 1, 2 |

**Note 1:** When a PORT register is modified as a function of itself (e.g., `MOVF PORTB, 1, 0`), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as input and is driven low by an external device, the data will be written back with a '0'.

**2:** If this instruction is executed on the TMR0 register (and where applicable, d = 1), the prescaler will be cleared if assigned.

**3:** If Program Counter (PC) is modified or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a NOP.

**4:** Some instructions are 2-word instructions. The second word of these instructions will be executed as a NOP unless the first word of the instruction retrieves the information embedded in these 16 bits. This ensures that all program memory locations have a valid instruction.

**5:** If the table write starts the write cycle to internal memory, the write will continue until terminated.

**TABLE 24-2: PIC18FXXX INSTRUCTION SET (CONTINUED)**

| Mnemonic, Operands | | Description | Cycles | 16-Bit Instruction Word MSb LSb | Status Affected | Notes |
|---|---|---|---|---|---|---|
| **LITERAL OPERATIONS** | | | | | | |
| ADDLW | k | Add Literal and WREG | 1 | 0000 1111 kkkk kkkk | C, DC, Z, OV, N | |
| ANDLW | k | AND Literal with WREG | 1 | 0000 1011 kkkk kkkk | Z, N | |
| IORLW | k | Inclusive OR Literal with WREG | 1 | 0000 1001 kkkk kkkk | Z, N | |
| LFSR | f, k | Move Literal (12-bit) 2nd word to FSRx 1st word | 2 | 1110 1110 00ff kkkk<br>1111 0000 kkkk kkkk | None | |
| MOVLB | k | Move Literal to BSR<3:0> | 1 | 0000 0001 0000 kkkk | None | |
| MOVLW | k | Move Literal to WREG | 1 | 0000 1110 kkkk kkkk | None | |
| MULLW | k | Multiply Literal with WREG | 1 | 0000 1101 kkkk kkkk | None | |
| RETLW | k | Return with Literal in WREG | 2 | 0000 1100 kkkk kkkk | None | |
| SUBLW | k | Subtract WREG from Literal | 1 | 0000 1000 kkkk kkkk | C, DC, Z, OV, N | |
| XORLW | k | Exclusive OR Literal with WREG | 1 | 0000 1010 kkkk kkkk | Z, N | |
| **DATA MEMORY ↔ PROGRAM MEMORY OPERATIONS** | | | | | | |
| TBLRD* | | Table Read | 2 | 0000 0000 0000 1000 | None | |
| TBLRD*+ | | Table Read with Post-Increment | | 0000 0000 0000 1001 | None | |
| TBLRD*- | | Table Read with Post-Decrement | | 0000 0000 0000 1010 | None | |
| TBLRD+* | | Table Read with Pre-Increment | | 0000 0000 0000 1011 | None | |
| TBLWT* | | Table Write | 2 (5) | 0000 0000 0000 1100 | None | |
| TBLWT*+ | | Table Write with Post-Increment | | 0000 0000 0000 1101 | None | |
| TBLWT*- | | Table Write with Post-Decrement | | 0000 0000 0000 1110 | None | |
| TBLWT+* | | Table Write with Pre-Increment | | 0000 0000 0000 1111 | None | |

**Note 1:** When a PORT register is modified as a function of itself (e.g., `MOVF PORTB, 1, 0`), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as input and is driven low by an external device, the data will be written back with a '0'.

**2:** If this instruction is executed on the TMR0 register (and where applicable, d = 1), the prescaler will be cleared if assigned.

**3:** If Program Counter (PC) is modified or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a `NOP`.

**4:** Some instructions are 2-word instructions. The second word of these instructions will be executed as a `NOP` unless the first word of the instruction retrieves the information embedded in these 16 bits. This ensures that all program memory locations have a valid instruction.

**5:** If the table write starts the write cycle to internal memory, the write will continue until terminated.

**BCF** **Bit Clear f**

Syntax: [ *label* ] BCF f,b[,a]

Operands: $0 \le f \le 255$
$0 \le b \le 7$
$a \in [0,1]$

Operation: $0 \rightarrow f<b>$

Status Affected: None

Encoding:

| 1001 | bbba | ffff | ffff |
|---|---|---|---|

Description: Bit 'b' in register 'f' is cleared. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read register 'f' | Process Data | Write register 'f' |

Example: `BCF FLAG_REG, 7`

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

**BN** **Branch if Negative**

Syntax: [ *label* ] BN n

Operands: $-128 \le n \le 127$

Operation: if Negative bit is '1',
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

| 1110 | 0110 | nnnn | nnnn |
|---|---|---|---|

Description: If the Negative bit is '1', then the program will branch.
The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read literal 'n' | Process Data | Write to PC |
| No operation | No operation | No operation | No operation |

If No Jump:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read literal 'n' | Process Data | No operation |

Example:
```
HERE    BN   Jump
```

Before Instruction
PC = address (HERE)
After Instruction
If Negative = 1;
PC = address (Jump)
If Negative = 0;
PC = address (HERE + 2)

**TSTFSZ** **Test f, Skip if 0**

Syntax: [ *label* ] TSTFSZ f [,a]

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

Operation: skip if f = 0

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 0110 | 011a | ffff | ffff |

Description: If 'f' = 0, the next instruction, fetched during the current instruction execution is discarded and a NOP is executed, making this a two-cycle instruction. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' is '1', then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1(2)
**Note:** 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read register 'f' | Process Data | No operation |

If skip:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| No operation | No operation | No operation | No operation |

If skip and followed by 2-word instruction:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| No operation | No operation | No operation | No operation |
| No operation | No operation | No operation | No operation |

Example:
```
HERE    TSTFSZ  CNT
NZERO     :
ZERO    :
```

Before Instruction
PC = Address (HERE)

After Instruction
If CNT = 0x00,
PC = Address (ZERO)
If CNT ≠ 0x00,
PC = Address (NZERO)

**XORLW** **Exclusive OR Literal with W**

Syntax: [ *label* ] XORLW k

Operands: $0 \le k \le 255$

Operation: (W) .XOR. k → W

Status Affected: N, Z

Encoding:

| | | | |
|---|---|---|---|
| 0000 | 1010 | kkkk | kkkk |

Description: The contents of W are XOR'ed with the 8-bit literal 'k'. The result is placed in W.

Words: 1

Cycles: 1

Q Cycle Activity:

| Q1 | Q2 | Q3 | Q4 |
|---|---|---|---|
| Decode | Read literal 'k' | Process Data | Write to W |

Example:
```
XORLW 0xAF
```

Before Instruction
W = 0xB5

After Instruction
W = 0x1A

## 25.0 DEVELOPMENT SUPPORT

The PIC® microcontrollers are supported with a full range of hardware and software development tools:

- Integrated Development Environment
  - MPLAB® IDE Software
- Assemblers/Compilers/Linkers
  - MPASM™ Assembler
  - MPLAB C18 and MPLAB C30 C Compilers
  - MPLINK™ Object Linker/ MPLIB™ Object Librarian
  - MPLAB ASM30 Assembler/Linker/Library
- Simulators
  - MPLAB SIM Software Simulator
- Emulators
  - MPLAB ICE 2000 In-Circuit Emulator
  - MPLAB REAL ICE™ In-Circuit Emulator
- In-Circuit Debugger
  - MPLAB ICD 2
- Device Programmers
  - PICSTART® Plus Development Programmer
  - MPLAB PM3 Device Programmer
  - PICkit™ 2 Development Programmer
- Low-Cost Demonstration and Development Boards and Evaluation Kits

### 25.1 MPLAB Integrated Development Environment Software

The MPLAB IDE software brings an ease of software development previously unseen in the 8/16-bit microcontroller market. The MPLAB IDE is a Windows® operating system-based application that contains:

- A single graphical interface to all debugging tools
  - Simulator
  - Programmer (sold separately)
  - Emulator (sold separately)
  - In-Circuit Debugger (sold separately)
- A full-featured editor with color-coded context
- A multiple project manager
- Customizable data windows with direct edit of contents
- High-level source code debugging
- Visual device initializer for easy register initialization
- Mouse over variable inspection
- Drag and drop variables from source to watch windows
- Extensive on-line help
- Integration of select third party tools, such as HI-TECH Software C Compilers and IAR C Compilers

The MPLAB IDE allows you to:

- Edit your source files (either assembly or C)
- One touch assemble (or compile) and download to PIC MCU emulator and simulator tools (automatically updates all project information)
- Debug using:
  - Source files (assembly or C)
  - Mixed assembly and C
  - Machine code

MPLAB IDE supports multiple debugging tools in a single development paradigm, from the cost-effective simulators, through low-cost in-circuit debuggers, to full-featured emulators. This eliminates the learning curve when upgrading to tools with increased flexibility and power.

## 26.3 DC Characteristics: PIC18F2220/2320/4220/4320 (Industrial) PIC18LF2220/2320/4220/4320 (Industrial)

| DC CHARACTERISTICS | | | Standard Operating Conditions (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended | | | |
|---|---|---|---|---|---|---|
| **Param No.** | **Symbol** | **Characteristic** | **Min** | **Max** | **Units** | **Conditions** |
| | VIL | **Input Low Voltage** | | | | |
| | | I/O Ports: | | | | |
| D030 | | with TTL Buffer | VSS | 0.15 VDD | V | VDD < 4.5V |
| D030A | | | — | 0.8 | V | 4.5V ≤ VDD ≤ 5.5V |
| D031 | | with Schmitt Trigger Buffer | VSS | 0.2 VDD | V | |
| | | RC3 and RC4 | VSS | 0.3 VDD | V | |
| D032 | | $\overline{MCLR}$ | VSS | 0.2 VDD | V | |
| D032A | | OSC1 and T1OSI | VSS | 0.2 VDD | V | LP, XT, HS, HSPLL modes(1) |
| D033 | | OSC1 | VSS | 0.2 VDD | V | EC mode(1) |
| | VIH | **Input High Voltage** | | | | |
| | | I/O Ports: | | | | |
| D040 | | with TTL Buffer | 0.25 VDD + 0.8V | VDD | V | VDD < 4.5V |
| D040A | | | 2.0 | VDD | V | 4.5V ≤ VDD ≤ 5.5V |
| D041 | | with Schmitt Trigger Buffer | 0.8 VDD | VDD | V | |
| | | RC3 and RC4 | 0.7 VDD | VDD | V | |
| D042 | | $\overline{MCLR}$ | 0.8 VDD | VDD | V | |
| D042A | | OSC1 and T1OSI | 1.6 | VDD | V | LP, XT, HS, HSPLL modes(1) |
| D043 | | OSC1 | 0.8 VDD | VDD | V | EC mode(1) |
| | IIL | **Input Leakage Current(2,3)** | | | | |
| D060 | | I/O Ports | — | ±0.2 | µA | VSS ≤ VPIN ≤ VDD, Pin at high-impedance |
| D061 | | $\overline{MCLR}$, RA4 | — | ±1.0 | µA | Vss ≤ VPIN ≤ VDD |
| D063 | | OSC1 | — | ±1.0 | µA | Vss ≤ VPIN ≤ VDD |
| | IPU | **Weak Pull-up Current** | | | | |
| D070 | IPURB | PORTB Weak Pull-up Current | 50 | 400 | µA | VDD = 5V, VPIN = VSS |

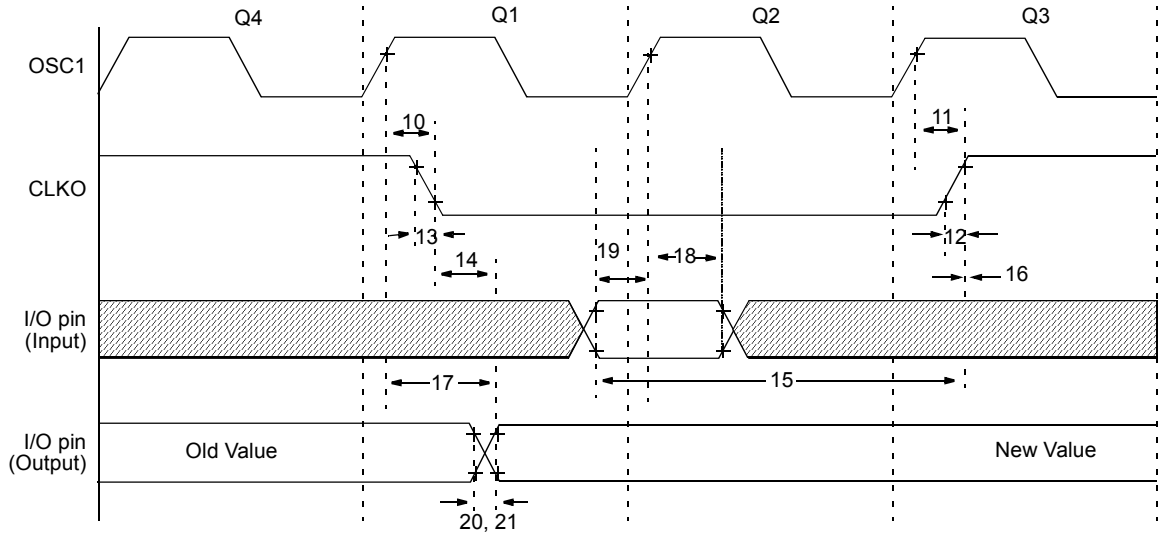
**Note 1:** In RC oscillator configuration, the OSC1/CLKI pin is a Schmitt Trigger input. It is not recommended that the PIC® device be driven with an external clock while in RC mode.

**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 current sourced by the pin.

**4:** Parameter is characterized but not tested.

**FIGURE 26-7: CLKO AND I/O TIMING**




**Note:** Refer to Figure 26-5 for load conditions.

**TABLE 26-9: CLKO AND I/O TIMING REQUIREMENTS**

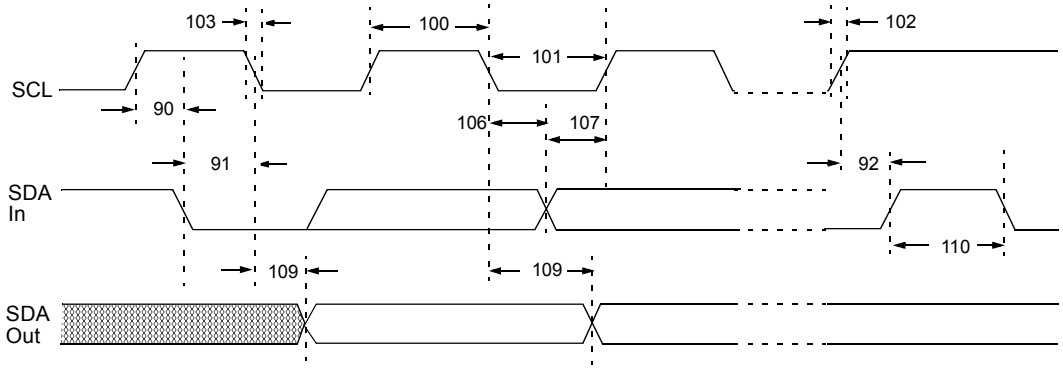
| Param No. | Symbol | Characteristic | | Min | Typ | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|---|
| 10 | TosH2ckL | OSC1 ↑ to CLKO ↓ | | — | 75 | 200 | ns | **(1)** |
| 11 | TosH2ckH | OSC1 ↑ to CLKO ↑ | | — | 75 | 200 | ns | **(1)** |
| 12 | TckR | CLKO Rise Time | | — | 35 | 100 | ns | **(1)** |
| 13 | TckF | CLKO Fall Time | | — | 35 | 100 | ns | **(1)** |
| 14 | TckL2ioV | CLKO ↓ to Port Out Valid | | — | — | 0.5 TCY + 20 | ns | **(1)** |
| 15 | TioV2ckH | Port In Valid before CLKO ↑ | | 0.25 TCY + 25 | — | — | ns | **(1)** |
| 16 | TckH2ioI | Port In Hold after CLKO ↑ | | 0 | — | — | ns | **(1)** |
| 17 | TosH2ioV | OSC1↑ (Q1 cycle) to Port Out Valid | | — | 50 | 150 | ns | |
| 18 | TosH2ioI | OSC1↑ (Q2 cycle) to Port Input Invalid (I/O in hold time) | PIC18**F**XX20 | 100 | — | — | ns | |
| 18A | | | PIC18**LF**XX20 | 200 | — | — | ns | |
| 19 | TioV2osH | Port Input Valid to OSC1↑ (I/O in setup time) | | 0 | — | — | ns | |
| 20 | TioR | Port Output Rise Time | PIC18**F**XX20 | — | 10 | 25 | ns | |
| 20A | | | PIC18**LF**XX20 | — | — | 60 | ns | |
| 21 | TioF | Port Output Fall Time | PIC18**F**XX20 | — | 10 | 25 | ns | |
| 21A | | | PIC18**LF**XX20 | — | — | 60 | ns | |

**Note 1:** Measurements are taken in RC mode, where CLKO output is 4 x TOSC.

**TABLE 26-18: I²C™ BUS START/STOP BITS REQUIREMENTS (SLAVE MODE)**

| Param. No. | Symbol | Characteristic | | Min | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|
| 90 | TSU:STA | Start Condition Setup Time | 100 kHz mode | 4700 | — | ns | Only relevant for Repeated Start condition |
| | | | 400 kHz mode | 600 | — | | |
| 91 | THD:STA | Start Condition Hold Time | 100 kHz mode | 4000 | — | ns | After this period, the first clock pulse is generated |
| | | | 400 kHz mode | 600 | — | | |
| 92 | TSU:STO | Stop Condition Setup Time | 100 kHz mode | 4700 | — | ns | |
| | | | 400 kHz mode | 600 | — | | |
| 93 | THD:STO | Stop Condition Hold Time | 100 kHz mode | 4000 | — | ns | |
| | | | 400 kHz mode | 600 | — | | |

**FIGURE 26-18: I²C™ BUS DATA TIMING**




**Note:** Refer to Figure 26-5 for load conditions.

**FIGURE 27-31: ΔIPD LVD vs. VDD SLEEP MODE, LVD = 2.00V-2.12V**

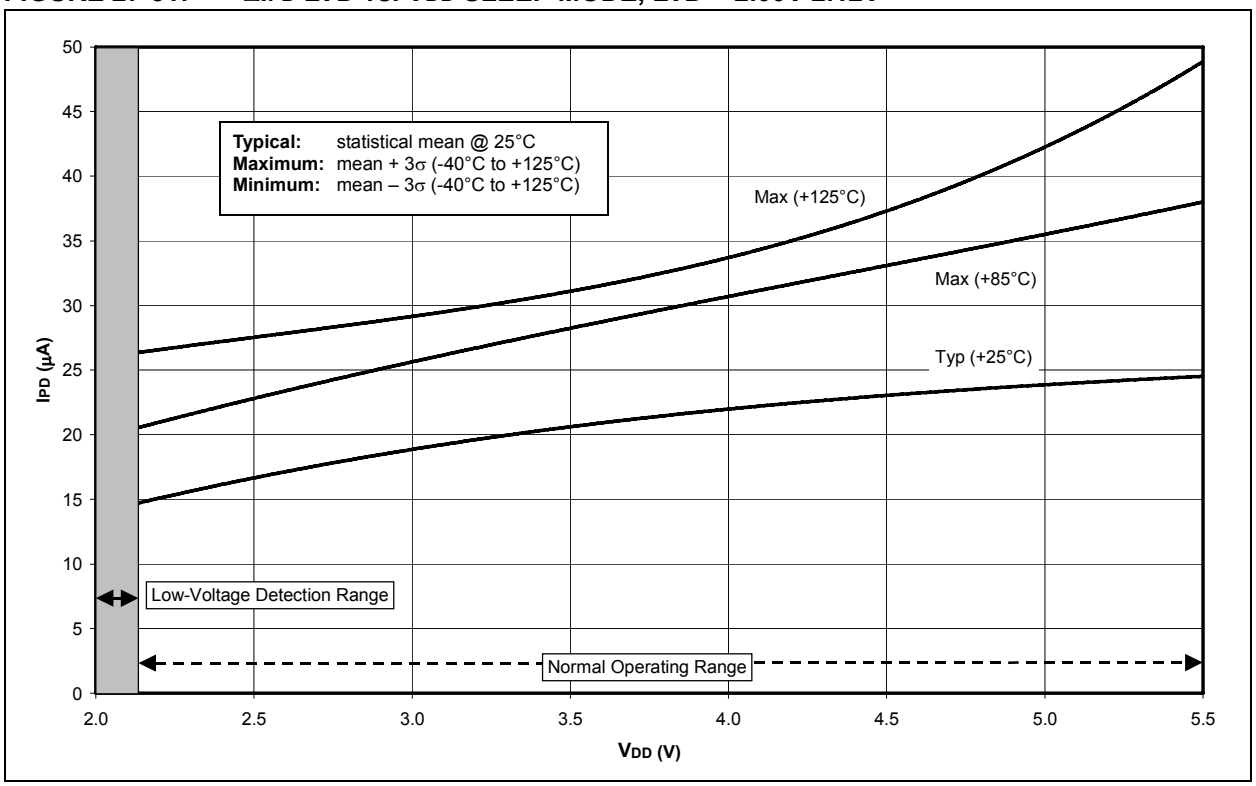
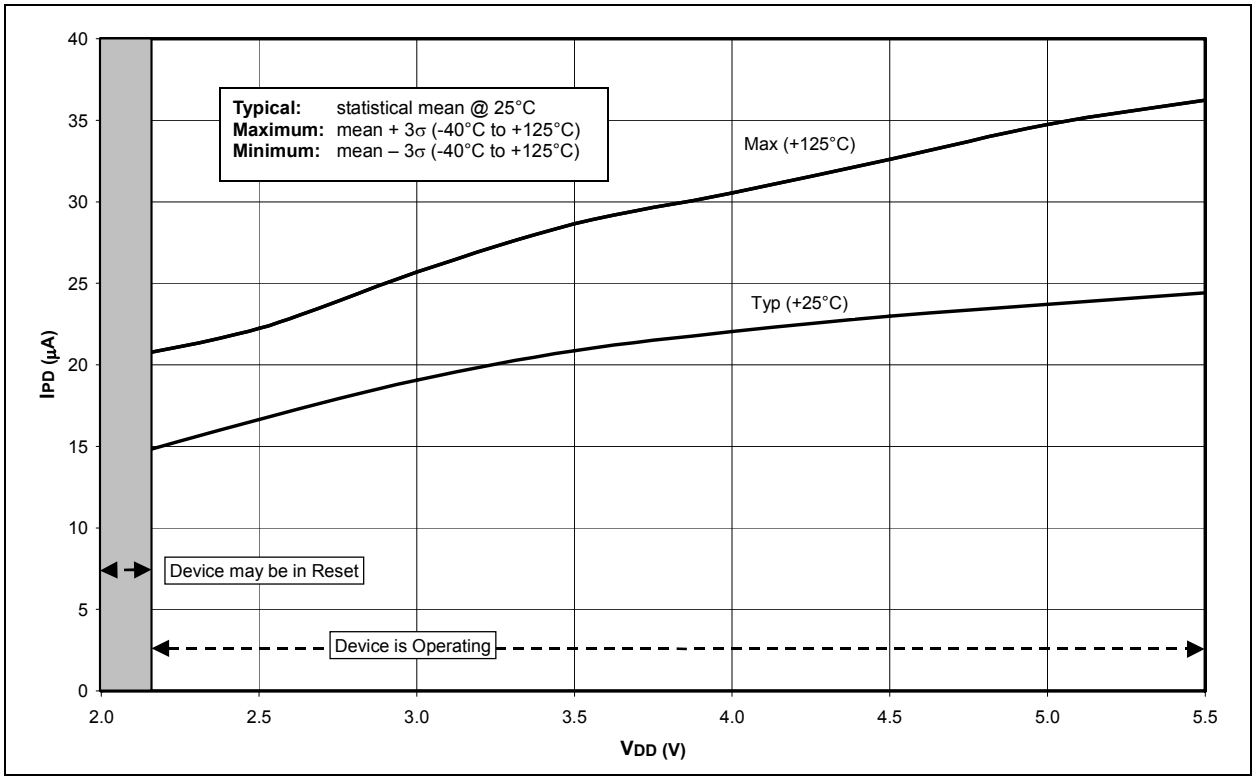
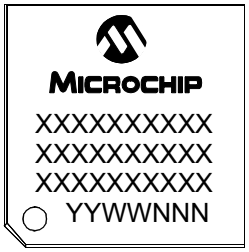



**FIGURE 27-32: ΔIPD BOR vs. VDD, -40°C TO +125°C SLEEP MODE, BOR ENABLED AT 2.00V-2.16V**

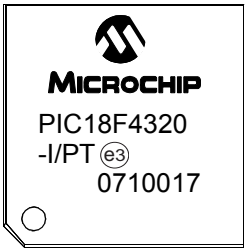


**Package Marking Information (Continued)**

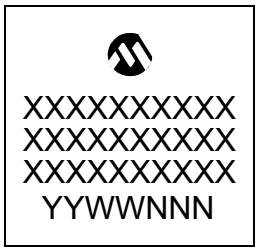
44-Lead TQFP




Example




44-Lead QFN




Example

# INDEX

## A



## B

## S



## T