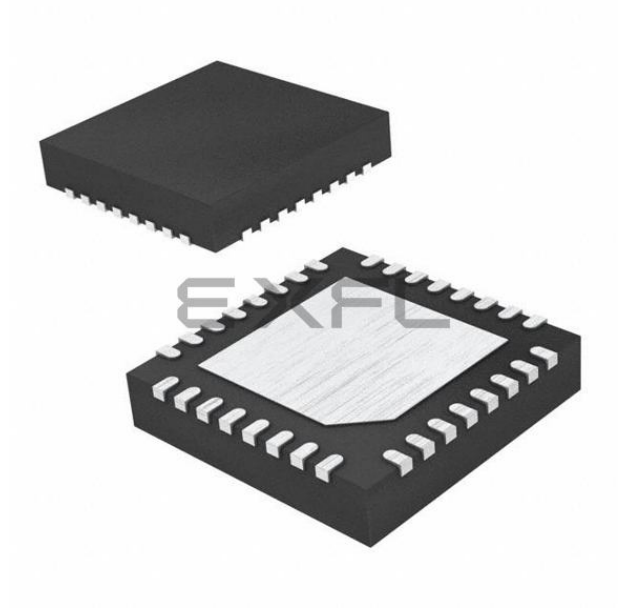


Welcome to E-XFL.COM

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, Power Control PWM, QEI, POR, PWM, WDT
Number of I/O	24
Program Memory Size	8KB (4K x 16)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	768 x 8
Voltage - Supply (Vcc/Vdd)	2V ~ 5.5V
Data Converters	A/D 5x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	28-VQFN Exposed Pad
Supplier Device Package	28-QFN-S (6x6)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18lf2331-i-mm

1.0 DEVICE OVERVIEW

This document contains device-specific information for the following devices:

- PIC18F2331
- PIC18F2431
- PIC18F4331
- PIC18F4431
- PIC18LF2331
- PIC18LF2431
- PIC18LF4331
- PIC18LF4431

This family offers the advantages of all PIC18 microcontrollers – namely, high computational performance at an economical price, with the addition of high-endurance enhanced Flash program memory and a high-speed 10-bit A/D Converter. On top of these features, the PIC18F2331/2431/4331/4431 family introduces design enhancements that make these microcontrollers a logical choice for many high-performance, power control and motor control applications. These special peripherals include:

- 14-Bit Resolution Power Control PWM module (PCPWM) with Programmable Dead-Time Insertion
- Motion Feedback Module (MFM), including a 3-Channel Input Capture (IC) module and Quadrature Encoder Interface (QEI)
- High-Speed 10-Bit A/D Converter (HSADC)

The PCPWM can generate up to eight complementary PWM outputs with dead-band time insertion. Overdrive current is detected by off-chip analog comparators or the digital Fault inputs ($\overline{FLTA}$, $\overline{FLTB}$).

The MFM Quadrature Encoder Interface provides precise rotor position feedback and/or velocity measurement. The MFM 3x input capture or external interrupts can be used to detect the rotor state for electrically commutated motor applications using Hall sensor feedback, such as BLDC motor drives.

PIC18F2331/2431/4331/4431 devices also feature Flash program memory and an internal RC oscillator with built-in LP modes.

1.1 New Core Features

1.1.1 nanoWatt Technology

All of the devices in the PIC18F2331/2431/4331/4431 family incorporate a range of features that can significantly reduce power consumption during operation. Key items include:

- **Alternate Run Modes:** By clocking the controller from the Timer1 source or the internal oscillator block, power consumption during code execution can be reduced by as much as 90%.
- **Multiple Idle Modes:** The controller can also run with its CPU core disabled, but the peripherals are still active. In these states, power consumption can be reduced even further, to as little as 4% of normal operation requirements.
- **On-the-Fly Mode Switching:** The power-managed modes are invoked by user code during operation, allowing the user to incorporate power-saving ideas into their application's software design.
- **Lower Consumption in Key Modules:** The power requirements for both Timer1 and the Watchdog Timer have been reduced by up to 80%, with typical values of 1.1 and 2.1 μA, respectively.

1.1.2 MULTIPLE OSCILLATOR OPTIONS AND FEATURES

All of the devices in the PIC18F2331/2431/4331/4431 family offer nine different oscillator options, allowing users a wide range of choices in developing application hardware. These include:

- Four Crystal modes, using crystals or ceramic resonators.
- Two External Clock modes, offering the option of using two pins (oscillator input and a divide-by-4 clock output) or one pin (oscillator input, with the second pin reassigned as general I/O).
- Two External RC Oscillator modes, with the same pin options as the External Clock modes.
- An internal oscillator block, which provides an 8 MHz clock and an INTRC source (approximately 31 kHz, stable over temperature and V_{DD}), as well as a range of 6 user-selectable clock frequencies (from 125 kHz to 4 MHz) for a total of 8 clock frequencies.
- A Phase Lock Loop (PLL) frequency multiplier, available to both the High-Speed Crystal and Internal Oscillator modes, which allows clock speeds of up to 40 MHz. Used with the internal oscillator, the PLL gives users a complete selection of clock speeds, from 31 kHz to 32 MHz – all without using an external crystal or clock circuit.
- **Fail-Safe Clock Monitor:** This option constantly monitors the main clock source against a reference signal provided by the internal oscillator. If a clock failure occurs, the controller is switched to the internal oscillator block, allowing for continued low-speed operation or a safe application shutdown.
- **Two-Speed Start-up:** This option allows the internal oscillator to serve as the clock source from Power-on Reset, or wake-up from Sleep mode, until the primary clock source is available.

2.5 External Oscillator Pins

Many microcontrollers have options for at least two oscillators: a high-frequency primary oscillator and a low-frequency secondary oscillator (refer to **Section 3.0 "Oscillator Configurations"** for details).

The oscillator circuit should be placed on the same side of the board as the device. Place the oscillator circuit close to the respective oscillator pins with no more than 0.5 inch (12 mm) between the circuit components and the pins. The load capacitors should be placed next to the oscillator itself, on the same side of the board.

Use a grounded copper pour around the oscillator circuit to isolate it from surrounding circuits. The grounded copper pour should be routed directly to the MCU ground. Do not run any signal traces or power traces inside the ground pour. Also, if using a two-sided board, avoid any traces on the other side of the board where the crystal is placed.

Layout suggestions are shown in Figure 2-4. In-line packages may be handled with a single-sided layout that completely encompasses the oscillator pins. With fine-pitch packages, it is not always possible to completely surround the pins and components. A suitable solution is to tie the broken guard sections to a mirrored ground layer. In all cases, the guard trace(s) must be returned to ground.

In planning the application's routing and I/O assignments, ensure that adjacent port pins and other signals in close proximity to the oscillator are benign (i.e., free of high frequencies, short rise and fall times, and other similar noise).

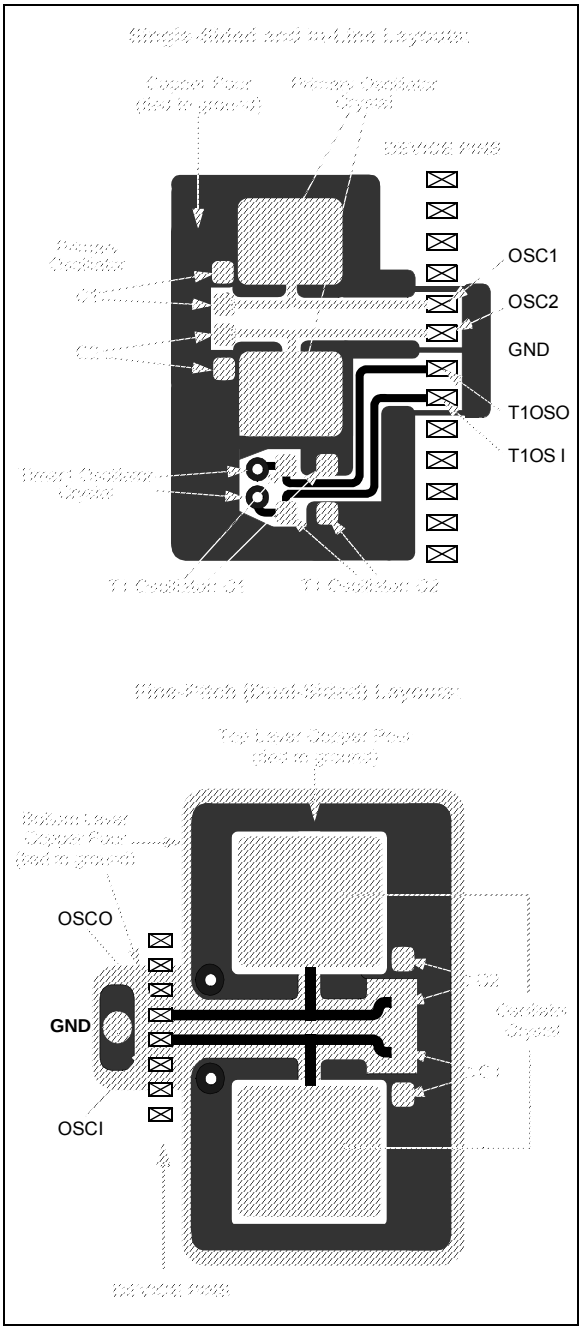
For additional information and design guidance on oscillator circuits, please refer to these Microchip Application Notes, available at the corporate web site (www.microchip.com):

- *AN826, "Crystal Oscillator Basics and Crystal Selection for rfPIC™ and PICmicro® Devices"*
- *AN849, "Basic PICmicro® Oscillator Design"*
- *AN943, "Practical PICmicro® Oscillator Analysis and Design"*
- *AN949, "Making Your Oscillator Work"*

2.6 Unused I/Os

Unused I/O pins should be configured as outputs and driven to a logic low state. Alternatively, connect a 1 kΩ to 10 kΩ resistor to VSS on unused pins and drive the output to logic low.

FIGURE 2-3: SUGGESTED PLACEMENT OF THE OSCILLATOR CIRCUIT



REGISTER 3-2: OSCCON: OSCILLATOR CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R[(1)]	R-0	R/W-0	R/W-0
IDLEN	IRCF2	IRCF1	IRCF0	OSTS	IOFS	SCS1	SCS0
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 **IDLEN:** Idle Enable bit
- 1 = Idle mode enabled; CPU core is not clocked in power-managed modes
- 0 = Run mode enabled; CPU core is clocked in power-managed modes

bit 6-4 **IRCF<2:0>:** Internal Oscillator Frequency Select bits
- 111 = 8 MHz (8 MHz source drives clock directly)
- 110 = 4 MHz (default)
- 101 = 2 MHz
- 100 = 1 MHz
- 011 = 500 kHz
- 010 = 250 kHz
- 001 = 125 kHz
- 000 = 31 kHz (INTRC source drives clock directly)[(2)]

bit 3 **OSTS:** Oscillator Start-up Timer Time-out Status bit[(1)]
- 1 = Oscillator Start-up Timer time-out has expired; primary oscillator is running
- 0 = Oscillator Start-up Timer time-out is running; primary oscillator is not ready

bit 2 **IOFS:** INTOSC Frequency Stable bit
- 1 = INTOSC frequency is stable
- 0 = INTOSC frequency is not stable

bit 1-0 **SCS<1:0>:** System Clock Select bits
- 1x = Internal oscillator block
- 01 = Secondary (Timer1) oscillator
- 00 = Primary oscillator

Note 1: Depends on the state of the IESO bit in Configuration Register 1H.
2: Default output frequency of INTOSC on Reset.

TABLE 5-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices				Power-on Reset, Brown-out Reset	$\overline{\text{MCLR}}$ Resets WDT Reset RESET Instruction Stack Resets	Wake-up via WDT or Interrupt
PTCON0	2331	2431	4331	4431	0000 0000	uuuu uuuu	uuuu uuuu
PTCON1	2331	2431	4331	4431	00-- ----	00-- ----	uu-- ----
PTMRL	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
PTMRH	2331	2431	4331	4431	---- 0000	---- 0000	---- uuuu
PTPERL	2331	2431	4331	4431	1111 1111	1111 1111	uuuu uuuu
PTPERH	2331	2431	4331	4431	---- 1111	---- 1111	---- uuuu
PDC0L	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
PDC0H	2331	2431	4331	4431	--00 0000	--00 0000	--uu uuuu
PDC1L	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
PDC1H	2331	2431	4331	4431	--00 0000	--00 0000	--uu uuuu
PDC2L	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
PDC2H	2331	2431	4331	4431	--00 0000	--00 0000	--uu uuuu
PDC3L	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
PDC3H	2331	2431	4331	4431	--00 0000	--00 0000	--uu uuuu
SEVTCMPL	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
SEVTCMPH	2331	2431	4331	4431	---- 0000	---- 0000	---- uuuu
PWMCON0	2331	2431	4331	4431	-111 0000	-111 0000	-uuu uuuu
PWMCON1	2331	2431	4331	4431	0000 0-00	0000 0-00	uuuu u-uu
DTCON	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
FLTCONFIG	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
OVDCOND	2331	2431	4331	4431	1111 1111	1111 1111	uuuu uuuu
OVDCONS	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
CAP1BUFH/ VELRH	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
CAP1BUFL/ VELRL	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
CAP2BUFH/ POSCNTH	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
CAP2BUFL/ POSCNTL	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
CAP3BUFH/ MAXCNTH	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu

Legend: u = unchanged, x = unknown, - = unimplemented bit, read as '0', q = value depends on condition. Shaded cells indicate conditions do not apply for the designated device.

Note 1: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

2: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

3: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

4: See Table 5-2 for Reset value for specific condition.

5: Bits 6 and 7 of PORTA, LATA and TRISA are enabled depending on the oscillator mode selected. When not enabled as PORTA pins, they are disabled and read '0'.

6: Bit 3 of PORTE and LATE are enabled if $\overline{\text{MCLR}}$ functionality is disabled. When not enabled as the PORTE pin, they are disabled and read as '0'. The 28-pin devices do not have only RE3 implemented.

6.1.4.2 Table Reads and Table Writes

A better method of storing data in program memory allows two bytes of data to be stored in each instruction location. Look-up table data may be stored, two bytes per program word, by using table reads and writes.

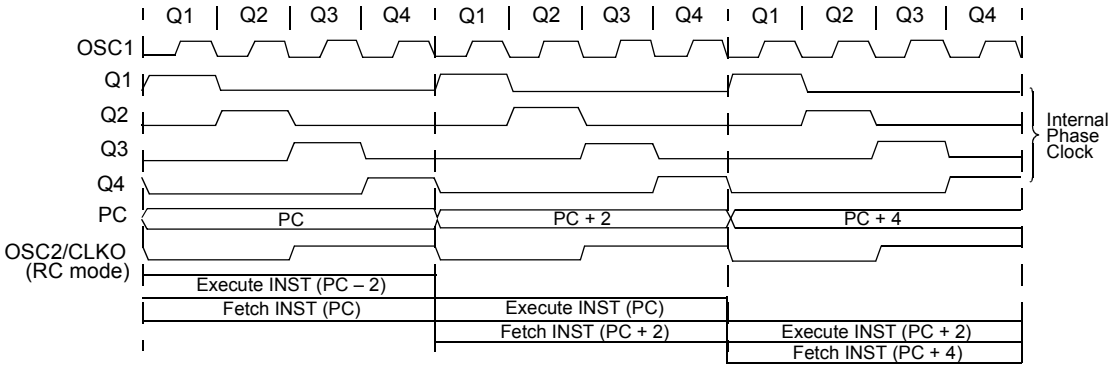
The Table Pointer register (TBLPTR) specifies the byte address and the Table Latch register (TABLAT) contains the data that is read from or written to program memory. Data is transferred to or from program memory, one byte at a time.

Table read and table write operations are discussed further in **Section 8.1 "Table Reads and Table Writes"**.

6.2 Clocking Scheme/Instruction Cycle

The clock input (from OSC1) is internally divided by four to generate four non-overlapping quadrature clocks, namely Q1, Q2, Q3 and Q4. Internally, the Program Counter (PC) is incremented every Q1, the instruction is fetched from the program memory and latched into the Instruction Register (IR) in Q4. The instruction is decoded and executed during the following Q1 through Q4. The clocks and instruction execution flow are shown in Figure 6-4.

FIGURE 6-4: CLOCK/INSTRUCTION CYCLE



6.3 Instruction Flow/Pipelining

An "Instruction Cycle" consists of four Q cycles (Q1, Q2, Q3 and Q4). The instruction fetch and execute are pipelined such that fetch takes one instruction cycle, while decode and execute take another instruction cycle. However, due to the pipelining, each instruction effectively executes in one cycle. If an instruction causes the program counter to change (e.g., GOTO), then two cycles are required to complete the instruction (Example 6-3).

A fetch cycle begins with the Program Counter (PC) incrementing in Q1.

In the execution cycle, the fetched instruction is latched into the "Instruction Register" (IR) in cycle, Q1. This instruction is then decoded and executed during the Q2, Q3 and Q4 cycles. Data memory is read during Q2 (operand read) and written during Q4 (destination write).

EXAMPLE 6-3: INSTRUCTION PIPELINE FLOW

	TCY0	TCY1	TCY2	TCY3	TCY4	TCY5
1. MOVLW 55h	Fetch 1	Execute 1				
2. MOVWF PORTB		Fetch 2	Execute 2			
3. BRA SUB_1			Fetch 3	Execute 3		
4. BSF PORTA, BIT3 (Forced NOP)				Fetch 4	Flush (NOP)	
5. Instruction @ address SUB_1					Fetch SUB_1	Execute SUB_1

All instructions are single cycle, except for any program branches. These take two cycles since the fetch instruction is "flushed" from the pipeline, while the new instruction is being fetched and then executed.

6.4 Instructions in Program Memory

The program memory is addressed in bytes. Instructions are stored as two bytes or four bytes in program memory. The Least Significant Byte of an instruction word is always stored in a program memory location with an even address (LSB = 0). Figure 6-5 shows an example of how instruction words are stored in the program memory. To maintain alignment with instruction boundaries, the PC increments in steps of 2 and the LSB will always read '0'.

The CALL and GOTO instructions have the absolute program memory address embedded into the instruction. Since instructions are always stored on word boundaries, the data contained in the instruction is a word address. The word address is written to PC<20:1>, which accesses the desired byte address in program memory. Instruction 2 in Figure 6-5 shows how the instruction, 'GOTO 000006h', is encoded in the program memory. Program branch instructions, which encode a relative address offset, operate in the same manner. The offset value stored in a branch instruction represents the number of single-word instructions that the PC will be offset by. **Section 24.0 "Instruction Set Summary"** provides further details of the instruction set.

6.4.1 TWO-WORD INSTRUCTIONS

The standard PIC18 instruction set has four two-word instructions: CALL, MOVFF, GOTO and LSFR. In all cases, the second word of the instructions always has '1111' as its four Most Significant bits; the other 12 bits are literal data, usually a data memory address.

The use of '1111' in the four MSbs of an instruction specifies a special form of NOP. If the instruction is executed in proper sequence, immediately after the first word, the data in the second word is accessed and used by the instruction sequence. If the first word is skipped for some reason and the second word is executed by itself, a NOP is executed instead. This is necessary for cases when the two-word instruction is preceded by a conditional instruction that changes the PC. Example 6-4 shows how this works.

Note:	For information on two-word instructions in the extended instruction set, see **Section 24.2 "Instruction Set".**

FIGURE 6-5: INSTRUCTIONS IN PROGRAM MEMORY

			LSB = 1	LSB = 0	Word Address ↓
	Program Memory Byte Locations →				000000h
					000002h
					000004h
					000006h
Instruction 1:	MOVLW	055h	0Fh	55h	000008h
Instruction 2:	GOTO	000006h	EFh	03h	00000Ah
			F0h	00h	00000Ch
Instruction 3:	MOVFF	123h, 456h	C1h	23h	00000Eh
			F4h	56h	000010h
					000012h
					000014h

EXAMPLE 6-4: TWO-WORD INSTRUCTIONS

CASE 1:			
Object Code	**Source Code**		
0110 0110 0000 0000	TSTFSZ	REG1	; is RAM location 0?
1100 0001 0010 0011	MOVFF	REG1, REG2	; No, skip this word
1111 0100 0101 0110			; Execute this word as a NOP
0010 0100 0000 0000	ADDWF	REG3	; continue code
CASE 2:			
Object Code	**Source Code**		
0110 0110 0000 0000	TSTFSZ	REG1	; is RAM location 0?
1100 0001 0010 0011	MOVFF	REG1, REG2	; Yes, execute this word
1111 0100 0101 0110			; 2nd word of instruction
0010 0100 0000 0000	ADDWF	REG3	; continue code

8.2.2 TABLAT – TABLE LATCH REGISTER

The Table Latch (TABLAT) is an 8-bit register mapped into the SFR space. The Table Latch is used to hold 8-bit data during data transfers between program memory and data RAM.

8.2.3 TBLPTR – TABLE POINTER REGISTER

The Table Pointer (TBLPTR) addresses a byte within the program memory. The TBLPTR is comprised of three SFR registers: Table Pointer Upper Byte, Table Pointer High Byte and Table Pointer Low Byte (TBLPTRU:TBLPTRH:TBLPTRL). These three registers join to form a 22-bit wide pointer. The low-order 21 bits allow the device to address up to 2 Mbytes of program memory space. Setting the 22nd bit allows access to the Device ID, the User ID and the Configuration bits.

The TBLPTR is used by the TBLRD and TBLWT instructions. These instructions can update the TBLPTR in one of four ways based on the table operation. These operations are shown in Table 8-1. These operations on the TBLPTR only affect the low-order 21 bits.

8.2.4 TABLE POINTER BOUNDARIES

TBLPTR is used in reads, writes and erases of the Flash program memory.

When a TBLRD is executed, all 22 bits of the Table Pointer determine which byte is read from program or configuration memory into TABLAT.

When a TBLWT is executed, the three LSbs of the Table Pointer (TBLPTR<2:0>) determine which of the eight program memory holding registers is written to. When the timed write to program memory (long write) begins, the 19 MSbs of the Table Pointer, TBLPTR (TBLPTR<21:3>), will determine which program memory block of 8 bytes is written to (TBLPTR<2:0> are ignored). For more detail, see **Section 8.5 "Writing to Flash Program Memory"**.

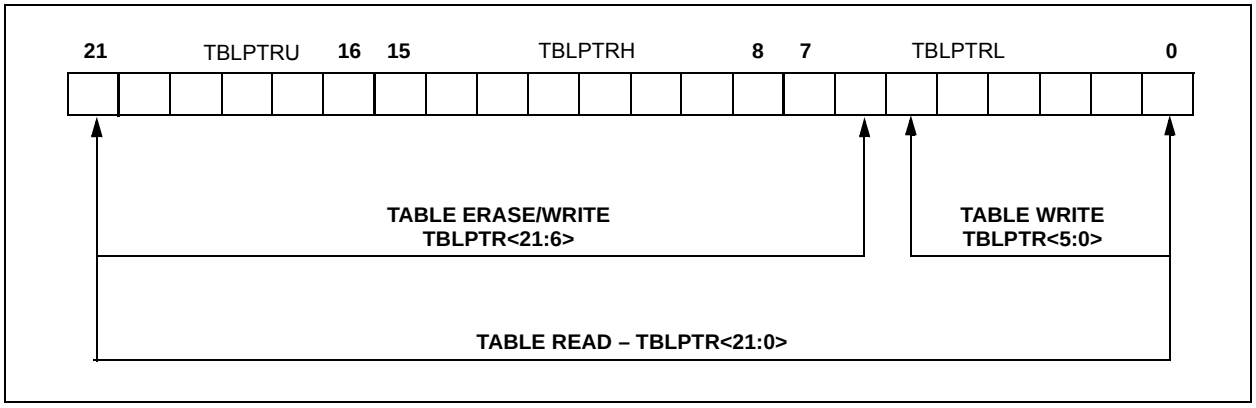
When an erase of program memory is executed, the 16 MSbs of the Table Pointer (TBLPTR<21:6>) point to the 64-byte block that will be erased. The Least Significant bits (TBLPTR<5:0>) are ignored.

Figure 8-3 describes the relevant boundaries of TBLPTR based on Flash program memory operations.

TABLE 8-1: TABLE POINTER OPERATIONS WITH TBLRD AND TBLWT INSTRUCTIONS

Example	Operation on Table Pointer
TBLRD* TBLWT*	TBLPTR is not modified
TBLRD*+ TBLWT*+	TBLPTR is incremented after the read/write
TBLRD*- TBLWT*-	TBLPTR is decremented after the read/write
TBLRD+* TBLWT+*	TBLPTR is incremented before the read/write

FIGURE 8-3: TABLE POINTER BOUNDARIES BASED ON OPERATION



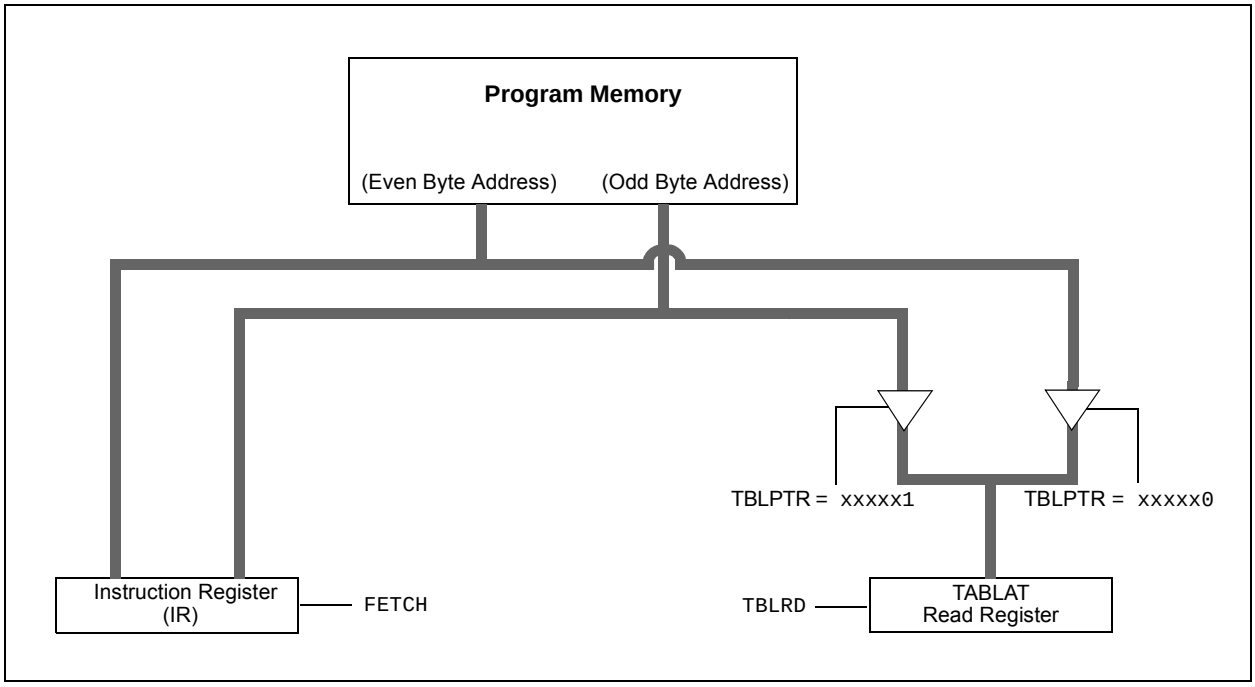
8.3 Reading the Flash Program Memory

The TBLRD instruction is used to retrieve data from program memory and place it into data RAM. Table reads from program memory are performed one byte at a time.

TBLPTR points to a byte address in program space. Executing a TBLRD instruction places the byte pointed to into TABLAT. In addition, TBLPTR can be modified automatically for the next table read operation.

The internal program memory is typically organized by words. The Least Significant bit of the address selects between the high and low bytes of the word. Figure 8-4 shows the interface between the internal program memory and the TABLAT.

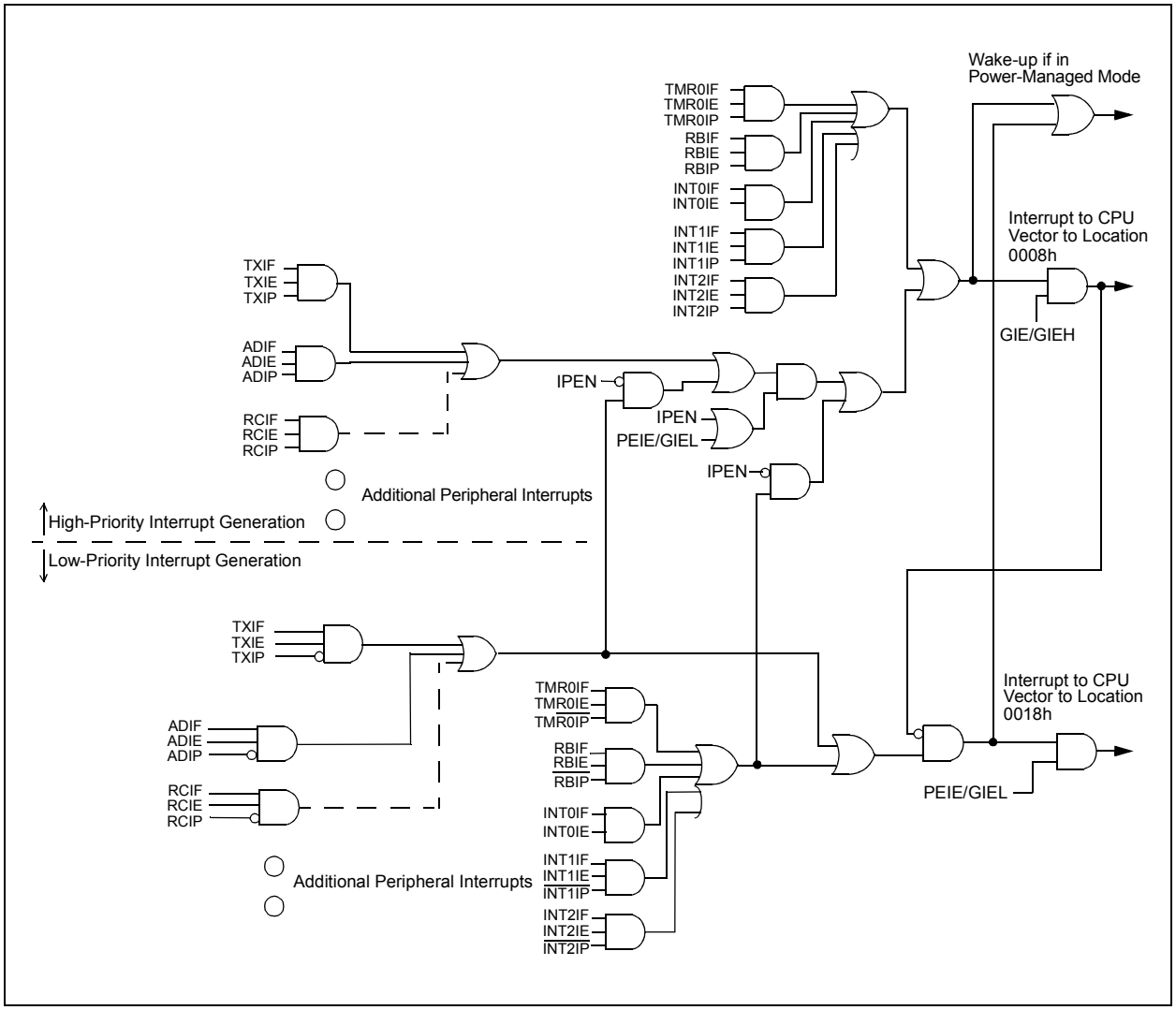
FIGURE 8-4: READS FROM FLASH PROGRAM MEMORY



EXAMPLE 8-1: READING A FLASH PROGRAM MEMORY WORD

```
            MOVLW   CODE_ADDR_UPPER             ; Load TBLPTR with the base
            MOVWF   TBLPTRU                     ; address of the word
            MOVLW   CODE_ADDR_HIGH
            MOVWF   TBLPTRH
            MOVLW   CODE_ADDR_LOW
            MOVWF   TBLPTRL
READ_WORD
            TBLRD*+                             ; read into TABLAT and increment TBLPTR
            MOVF    TABLAT,W                    ; get data
            MOVWF   WORD_EVEN
            TBLRD*+                             ; read into TABLAT and increment TBLPTR
            MOVF    TABLAT,W                    ; get data
            MOVWF   WORD_ODD
```

FIGURE 10-1: INTERRUPT LOGIC



10.1 INTCON Registers

The INTCON registers are readable and writable registers which contain various enable, priority and flag bits.

Note: Interrupt flag bits are set when an interrupt condition occurs, regardless of the state of its corresponding enable bit or the global enable bit. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt. This feature allows for software polling.

REGISTER 10-1: INTCON: INTERRUPT CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-x
GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
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 **GIE/GIEH:** Global Interrupt Enable bit
When IPEN = 0:
1 = Enables all unmasked interrupts
0 = Disables all interrupts
When IPEN = 1:
1 = Enables all high-priority interrupts
0 = Disables all high-priority interrupts

bit 6 **PEIE/GIEL:** Peripheral Interrupt Enable bit
When IPEN = 0:
1 = Enables all unmasked peripheral interrupts
0 = Disables all peripheral interrupts
When IPEN = 1:
1 = Enables all low-priority peripheral interrupts
0 = Disables all low-priority peripheral interrupts

bit 5 **TMR0IE:** TMR0 Overflow Interrupt Enable bit
1 = Enables the TMR0 overflow interrupt
0 = Disables the TMR0 overflow interrupt

bit 4 **INT0IE:** INT0 External Interrupt Enable bit
1 = Enables the INT0 external interrupt
0 = Disables the INT0 external interrupt

bit 3 **RBIE:** RB Port Change Interrupt Enable bit
1 = Enables the RB port change interrupt for RB<7:4> pins
0 = Disables the RB port change interrupt for RB<7:4> pins

bit 2 **TMR0IF:** TMR0 Overflow Interrupt Flag bit
1 = TMR0 register has overflowed (must be cleared in software)
0 = TMR0 register did not overflow

bit 1 **INT0IF:** INT0 External Interrupt Flag bit
1 = The INT0 external interrupt occurred (must be cleared in software)
0 = The INT0 external interrupt did not occur

bit 0 **RBIF:** RB Port Change Interrupt Flag bit
1 = At least one of the RB<7:4> pins changed state (must be cleared in software)
0 = None of the RB<7:4> pins have changed state

NOTES:

18.12.3 PWM OUTPUTS WHILE IN FAULT CONDITION

While in the Fault state (i.e., one or both $\overline{\text{FLTA}}$ and $\overline{\text{FLTB}}$ inputs are active), the PWM output signals are driven into their inactive states. The selection of which PWM outputs are deactivated (while in the Fault state) is determined by the FLTCON bit in the FLTCONFIG register as follows:

- FLTCON = 1: When $\overline{\text{FLTA}}$ or $\overline{\text{FLTB}}$ is asserted, the PWM outputs (i.e., PWM<7:0>) are driven into their inactive state.
- FLTCON = 0: When $\overline{\text{FLTA}}$ or $\overline{\text{FLTB}}$ is asserted, only PWM<5:0> outputs are driven inactive, leaving PWM<7:6> activated.

Note:	Disabling only three PWM channels and leaving one PWM channel enabled when in the Fault state, allows the flexibility to have at least one PWM channel enabled. None of the PWM outputs can be enabled (driven with the PWM Duty Cycle registers) while FLTCON = 1 and the Fault condition is present.

18.12.4 PWM OUTPUTS IN DEBUG MODE

The BRFEN bit in the FLTCONFIG register controls the simulation of a Fault condition, when a breakpoint is hit, while debugging the application using an In-Circuit Emulator (ICE) or an In-Circuit Debugger (ICD). Setting the BRFEN to high, enables the Fault condition on breakpoint, thus driving the PWM outputs to the inactive state. This is done to avoid any continuous keeping of status on the PWM pin, which may result in damage of the power devices connected to the PWM outputs.

If BRFEN = 0, the Fault condition on breakpoint is disabled.

Note:	It is highly recommended to enable the Fault condition on breakpoint if a debugging tool is used while developing the firmware and high-power circuitry. When the device is ready to program after debugging the firmware, the BRFEN bit can be disabled.

TABLE 24-2: PIC18FXXXX INSTRUCTION SET (CONTINUED)

Mnemonic, Operands		Description	Cycles	16-Bit Instruction Word				Status Affected	Notes
				MSb			LSb		
CONTROL OPERATIONS									
BC	n	Branch if Carry	1 (2)	1110	0010	nnnn	nnnn	None	
BN	n	Branch if Negative	1 (2)	1110	0110	nnnn	nnnn	None	
BNC	n	Branch if Not Carry	1 (2)	1110	0011	nnnn	nnnn	None	
BNN	n	Branch if Not Negative	1 (2)	1110	0111	nnnn	nnnn	None	
BNOV	n	Branch if Not Overflow	1 (2)	1110	0101	nnnn	nnnn	None	
BNZ	n	Branch if Not Zero	2	1110	0001	nnnn	nnnn	None	
BOV	n	Branch if Overflow	1 (2)	1110	0100	nnnn	nnnn	None	
BRA	n	Branch Unconditionally	1 (2)	1101	0nnn	nnnn	nnnn	None	
BZ	n	Branch if Zero	1 (2)	1110	0000	nnnn	nnnn	None	
CALL	n, s	Call Subroutine 1st word	2	1110	110s	kkkk	kkkk	None	
		2nd word		1111	kkkk	kkkk	kkkk		
CLRWDT	—	Clear Watchdog Timer	1	0000	0000	0000	0100	$\overline{TO}$, $\overline{PD}$	
DAW	—	Decimal Adjust WREG	1	0000	0000	0000	0111	C, DC	
GOTO	n	Go to Address 1st word	2	1110	1111	kkkk	kkkk	None	
		2nd word		1111	kkkk	kkkk	kkkk		
NOP	—	No Operation	1	0000	0000	0000	0000	None	
NOP	—	No Operation	1	1111	xxxx	xxxx	xxxx	None	4
POP	—	Pop Top of Return Stack (TOS)	1	0000	0000	0000	0110	None	
PUSH	—	Push Top of Return Stack (TOS)	1	0000	0000	0000	0101	None	
RCALL	n	Relative Call	2	1101	1nnn	nnnn	nnnn	None	
RESET		Software Device Reset	1	0000	0000	1111	1111	All	
RETFIE	s	Return from Interrupt Enable	2	0000	0000	0001	000s	GIE/GIEH, PEIE/GIEL	
RETLW	k	Return with Literal in WREG	2	0000	1100	kkkk	kkkk	None	
RETURN	s	Return from Subroutine	2	0000	0000	0001	001s	None	
SLEEP	—	Go into Standby mode	1	0000	0000	0000	0011	$\overline{TO}$, $\overline{PD}$	

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 an 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 the 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.

ADDWFC **ADD W and Carry bit to f**

Syntax: [*label*] ADDWFC f [,d [,a]]

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

Operation: (W) + (f) + (C) → dest

Status Affected: N, OV, C, DC, Z

Encoding:

0010	00da	ffff	ffff

Description: Add W, the Carry flag and data memory location, 'f'. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed in data memory location, 'f'. If 'a' is '0', the Access Bank will be selected. If 'a' is '1', the BSR will not be overridden.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: ADDWFC REG, W

Before Instruction
Carry bit = 1
REG = 0x02
W = 0x4D
After Instruction
Carry bit = 0
REG = 0x02
W = 0x50

ANDLW **AND Literal with W**

Syntax: [*label*] ANDLW k

Operands: $0 \le k \le 255$

Operation: (W) .AND. k → W

Status Affected: N, Z

Encoding:

0000	1011	kkkk	kkkk

Description: The contents of W are ANDed 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: ANDLW 0x5F

Before Instruction
W = 0xA3
After Instruction
W = 0x03

RLNCF **Rotate Left f (No Carry)**

Syntax: [*label*] RLNCF f [,d [,a]]

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

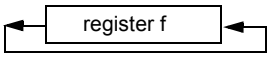
Operation: (f<n>) → dest<n + 1>,
(f<7>) → dest<0>

Status Affected: N, Z

Encoding:

0100	01da	ffff	ffff

Description: The contents of register, ‘f’, are rotated one bit to the left. If ‘d’ is ‘0’, the result is placed in W. If ‘d’ is ‘1’, the result is stored back in register, ‘f’. 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.



Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register ‘f’	Process Data	Write to destination

Example: RLNCF REG

Before Instruction
REG = 1010 1011
After Instruction
REG = 0101 0111

RRCF **Rotate Right f through Carry**

Syntax: [*label*] RRCF f [,d [,a]]

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

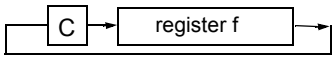
Operation: (f<n>) → dest<n – 1>,
(f<0>) → C,
(C) → dest<7>

Status Affected: C, N, Z

Encoding:

0011	00da	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 W. If ‘d’ is ‘1’, the result is placed back in register, ‘f’. 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.



Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register ‘f’	Process Data	Write to destination

Example: RRCF REG, W

Before Instruction
REG = 1110 0110
C = 0
After Instruction
REG = 1110 0110
W = 0111 0011
C = 0

XORWF **Exclusive OR W with f**

Syntax: [*label*] XORWF f [,d [,a]]

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

Operation: (W) .XOR. (f) → dest

Status Affected: N, Z

Encoding:

0001	10da	ffff	ffff

Description: Exclusive OR the contents of W with register, 'f'. If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in the register, 'f'. 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.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: XORWF REG

Before Instruction
REG = 0xAF
W = 0xB5

After Instruction
REG = 0x1A
W = 0xB5

25.11 PICkit 2 Development Programmer/Debugger and PICkit 2 Debug Express

The PICkit™ 2 Development Programmer/Debugger is a low-cost development tool with an easy to use interface for programming and debugging Microchip's Flash families of microcontrollers. The full featured Windows® programming interface supports baseline (PIC10F, PIC12F5xx, PIC16F5xx), midrange (PIC12F6xx, PIC16F), PIC18F, PIC24, dsPIC30, dsPIC33, and PIC32 families of 8-bit, 16-bit, and 32-bit microcontrollers, and many Microchip Serial EEPROM products. With Microchip's powerful MPLAB Integrated Development Environment (IDE) the PICkit™ 2 enables in-circuit debugging on most PIC® microcontrollers. In-Circuit-Debugging runs, halts and single steps the program while the PIC microcontroller is embedded in the application. When halted at a breakpoint, the file registers can be examined and modified.

The PICkit 2 Debug Express include the PICkit 2, demo board and microcontroller, hookup cables and CDROM with user's guide, lessons, tutorial, compiler and MPLAB IDE software.

25.12 MPLAB PM3 Device Programmer

The MPLAB PM3 Device Programmer is a universal, CE compliant device programmer with programmable voltage verification at V_{DDMIN} and V_{DDMAX} for maximum reliability. It features a large LCD display (128 x 64) for menus and error messages and a modular, detachable socket assembly to support various package types. The ICSP™ cable assembly is included as a standard item. In Stand-Alone mode, the MPLAB PM3 Device Programmer can read, verify and program PIC devices without a PC connection. It can also set code protection in this mode. The MPLAB PM3 connects to the host PC via an RS-232 or USB cable. The MPLAB PM3 has high-speed communications and optimized algorithms for quick programming of large memory devices and incorporates an MMC card for file storage and data applications.

25.13 Demonstration/Development Boards, Evaluation Kits, and Starter Kits

A wide variety of demonstration, development and evaluation boards for various PIC MCUs and dsPIC DSCs allows quick application development on fully functional systems. Most boards include prototyping areas for adding custom circuitry and provide application firmware and source code for examination and modification.

The boards support a variety of features, including LEDs, temperature sensors, switches, speakers, RS-232 interfaces, LCD displays, potentiometers and additional EEPROM memory.

The demonstration and development boards can be used in teaching environments, for prototyping custom circuits and for learning about various microcontroller applications.

In addition to the PICDEM™ and dsPICDEM™ demonstration/development board series of circuits, Microchip has a line of evaluation kits and demonstration software for analog filter design, KEELOQ® security ICs, CAN, IrDA®, PowerSmart battery management, SEEVAL® evaluation system, Sigma-Delta ADC, flow rate sensing, plus many more.

Also available are starter kits that contain everything needed to experience the specified device. This usually includes a single application and debug capability, all on one board.

Check the Microchip web page (www.microchip.com) for the complete list of demonstration, development and evaluation kits.

26.1 DC Characteristics: Supply Voltage PIC18F2331/2431/4331/4431 (Industrial, Extended) PIC18LF2331/2431/4331/4431 (Industrial)

PIC18LF2331/2431/4331/4431 (Industrial)			**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial				
PIC18F2331/2431/4331/4431 (Industrial, Extended)			**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended				
Param No.	**Symbol**	**Characteristic**	**Min**	**Typ**	**Max**	**Units**	**Conditions**
D001	VDD	**Supply Voltage**					
		PIC18LF2X31/4X31	2.0	—	5.5	V	
		PIC18F2X31/4X31	4.2	—	5.5	V	
D001C	AVDD	**Analog Supply Voltage**	VDD – 0.3	—	VDD + 0.3	V	
D001D	AVSS	**Analog Ground Voltage**	VSS – 0.3	—	VSS + 0.3	V	
D002	VDR	**RAM Data Retention Voltage(1)**	1.5	—	—	V	
D003	VPOR	**VDD Start Voltage** to Ensure Internal Power-on Reset Signal	—	—	0.7	V	See section on Power-on Reset for details
D004	SVDD	**VDD Rise Rate** to Ensure Internal Power-on Reset Signal	0.05	—	—	V/ms	See section on Power-on Reset for details
D005A	VBOR	**Brown-out Reset Voltage**					
		PIC18LF2X31/4X31	Industrial Low Voltage (-10°C to +85°C)				
		BORV<1:0> = 11	N/A	N/A	N/A	V	Reserved
		BORV<1:0> = 10	2.50	2.72	2.94	V	
		BORV<1:0> = 01	3.88	4.22	4.56	V	
		BORV<1:0> = 00	4.18	4.54	4.90	V	
D005B		PIC18LF2X31/4X31	Industrial Low Voltage (-40°C to -10°C)				
		BORV<1:0> = 11	N/A	N/A	N/A	V	Reserved
		BORV<1:0>= 10	2.34	2.72	3.10	V	
		BORV<1:0> = 01	3.63	4.22	4.81	V	
		BORV<1:0> = 00	3.90	4.54	5.18	V	
D005C		PIC18F2X31/4X31	Industrial (-10°C to +85°C)				
		BORV<1:0>= 1x	N/A	N/A	N/A	V	Reserved
		BORV<1:0> = 01	3.88	4.22	4.56	V	**(Note 2)**
		BORV<1:0> = 00	4.18	4.54	4.90	V	**(Note 2)**
D005D		PIC18F2X31/4X31	Industrial (-40°C to -10°C)				
		BORV<1:0>= 1x	N/A	N/A	N/A	V	Reserved
		BORV<1:0> = 01	N/A	N/A	N/A	V	Reserved
		BORV<1:0> = 00	3.90	4.54	5.18	V	**(Note 2)**
D005E		PIC18F2X31/4X31	Extended (-10°C to +85°C)				
		BORV<1:0> = 1x	N/A	N/A	N/A	V	Reserved
		BORV<1:0> = 01	3.88	4.22	4.56	V	**(Note 2)**
		BORV<1:0> = 00	4.18	4.54	4.90	V	**(Note 2)**
D005F		PIC18F2X31/4X31	Extended (-40°C to -10°C, +85°C to +125°C)				
		BORV<1:0> = 1x	N/A	N/A	N/A	V	Reserved
		BORV<1:0> = 01	N/A	N/A	N/A	V	Reserved
		BORV<1:0> = 00	3.90	4.54	5.18	V	**(Note 2)**

Legend: Shading of rows is to assist in readability of the table.

Note 1: This is the limit to which VDD can be lowered in Sleep mode, or during a device Reset, without losing RAM data.

2: When BOR is on and BORV<1:0> = 0x, the device will operate correctly at 40 MHz for any VDD at which the BOR allows execution.

26.2 DC Characteristics: Power-Down and Supply Current PIC18F2331/2431/4331/4431 (Industrial, Extended) PIC18LF2331/2431/4331/4431 (Industrial) (Continued)

PIC18LF2331/2431/4331/4431 (Industrial)	**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial
PIC18F2331/2431/4331/4431 (Industrial, Extended)	**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended

Param No.	Device	Typ	Max	Units	Conditions		
	Module Differential Currents (ΔIWDT, ΔIBOR, ΔILVD, ΔIOSCB, ΔIAD)						
D022 (ΔIWDT)	**Watchdog Timer**	1.5	4.0	µA	-40°C	VDD = 2.0V	
		2.2	4.0	µA	+25°C		
		3.1	5.0	µA	+85°C		
		2.5	6.0	µA	-40°C	VDD = 3.0V	
		3.3	6.0	µA	+25°C		
		4.7	7.0	µA	+85°C		
		3.7	10.0	µA	-40°C	VDD = 5.0V	
		4.5	10.0	µA	+25°C		
		6.1	13.0	µA	+85°C		
		22	44	µA	+125°C		
D022A (ΔIBOR)	**Brown-out Reset**	19	35.0	µA	-40°C to +85°C	VDD = 3.0V	
		24	45.0	µA	-40°C to +85°C	VDD = 5.0V	
		40	75	µA	+125°C		
D022B (ΔILVD)	**Low-Voltage Detect**	8.5	25.0	µA	-40°C to +85°C	VDD = 2.0V	
		16	35.0	µA	-40°C to +85°C	VDD = 3.0V	
		20	45.0	µA	-40°C to +85°C	VDD = 5.0V	
		35	66	µA	+125°C		
D025 (ΔIOSCB)	**Timer1 Oscillator**	1.7	3.5	µA	-40°C	VDD = 2.0V	32 kHz on Timer1(4)
		1.8	3.5	µA	+25°C		
		2.1	4.5	µA	+85°C		
		2.2	4.5	µA	-40°C	VDD = 3.0V	32 kHz on Timer1(4)
		2.6	4.5	µA	+25°C		
		2.8	5.5	µA	+85°C		
		3.0	6.0	µA	-40°C	VDD = 5.0V	32 kHz on Timer1(4)
		3.3	6.0	µA	+25°C		
		3.6	7.0	µA	+85°C		
		42	70	µA	+125°C		
D026 (ΔIAD)	**A/D Converter**	1.0	3.0	µA	-40°C to +85°C	VDD = 2.0V	A/D on, not converting
		1.0	4.0	µA	-40°C to +85°C	VDD = 3.0V	
		2.0	10.0	µA	-40°C to +85°C	VDD = 5.0V	
		150	950	µA	+125°C		

Legend: Shading of rows is to assist in readability of the table.

Note 1: The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS, and all features that add delta current disabled (such as WDT, Timer1 Oscillator, BOR, etc.).

2: The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.
The test conditions for all IDD measurements in active operation mode are:
OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;
MCLR = VDD; WDT enabled/disabled as specified.

3: For RC oscillator configurations, current through REXT is not included. The current through the resistor can be estimated by the formula: Ir = VDD/2REXT (mA) with REXT in kΩ.

4: Standard, low-cost 32 kHz crystals have an operating temperature range of -10°C to +70°C. Extended temperature crystals are available at a much higher cost.

THE MICROCHIP WEB SITE

Microchip provides online support via our WWW site at www.microchip.com. This web site is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the web site contains the following information:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQ), technical support requests, online discussion groups, Microchip consultant program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

CUSTOMER CHANGE NOTIFICATION SERVICE

Microchip's customer notification service helps keep customers current on Microchip products. Subscribers will receive e-mail notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, access the Microchip web site at www.microchip.com. Under "Support", click on "Customer Change Notification" and follow the registration instructions.

CUSTOMER SUPPORT

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support
- Development Systems Information Line

Customers should contact their distributor, representative or field application engineer (FAE) for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in the back of this document.

Technical support is available through the web site at: http://support.microchip.com