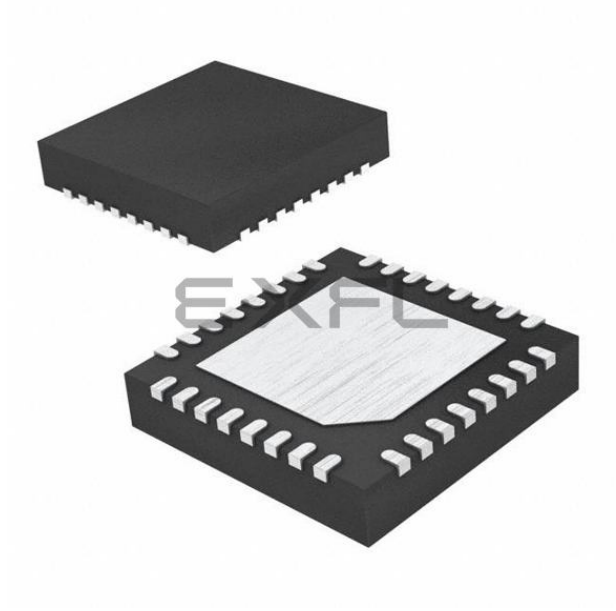
Microchip Technology - PIC18F2331T-I/MM Datasheet



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)	4.2V ~ 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/pic18f2331t-i-mm

TABLE 1-3: PIC18F4331/4431 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number			Pin Type	Buffer Type	Description
	PDIP	TQFP	QFN			
						PORTE is a bidirectional I/O port.
RE0/AN6	8	25	25			
RE0				I/O	ST	Digital I/O.
AN6				I	Analog	Analog Input 6.
RE1/AN7	9	26	26			
RE1				I/O	ST	Digital I/O.
AN7				I	Analog	Analog Input 7.
RE2/AN8	10	27	27			
RE2				I/O	ST	Digital I/O.
AN8				I	Analog	Analog Input 8.
VSS	12, 31	6, 29	6, 30, 31	P	—	Ground reference for logic and I/O pins.
VDD	11, 32	7, 28	7, 8, 28, 29	P	—	Positive supply for logic and I/O pins.
NC	—	12, 13, 33, 34	13	NC	NC	No connect.

Legend: TTL = TTL compatible input; CMOS = CMOS compatible input or output; ST = Schmitt Trigger input with CMOS levels; I = Input; O = Output; P = Power

Note 1: RC3 is the alternate pin for T0CKI/T5CKI; RC4 is the alternate pin for SDI/SDA; RC5 is the alternate pin for SCK/SCL; RC7 is the alternate pin for SDO.

2: RD4 is the alternate pin for $\overline{\text{FLTA}}$.

3: RD5 is the alternate pin for PWM4.

6.1.1 PROGRAM COUNTER

The Program Counter (PC) specifies the address of the instruction to fetch for execution. The PC is 21 bits wide and contained in three 8-bit registers. The low byte, known as the PCL register, is both readable and writable. The high byte (PCH register) contains the PC<15:8> bits and is not directly readable or writable.

Updates to the PCH register are performed through the PCLATH register. The upper byte is the PCU register and contains the bits, PC<20:16>. This register is also not directly readable or writable. Updates to the PCU register are performed through the PCLATU register.

The contents of PCLATH and PCLATU are transferred to the program counter by any operation that writes to the PCL. Similarly, the upper two bytes of the program counter are transferred to PCLATH and PCLATU by an operation that reads PCL. This is useful for computed offsets to the PC (see **Section 6.1.4.1 "Computed GOTO"**).

The PC addresses bytes in the program memory. To prevent the PC from becoming misaligned with word instructions, the Least Significant bit of the PCL is fixed to a value of '0'. The PC increments by two to address sequential instructions in the program memory.

The CALL, RCALL, GOTO and program branch instructions write to the program counter directly. For these instructions, the contents of PCLATH and PCLATU are not transferred to the program counter.

6.1.2 RETURN ADDRESS STACK

The return address stack allows any combination of up to 31 program calls and interrupts to occur. The PC (Program Counter) is pushed onto the stack when a CALL or RCALL instruction is executed, or an interrupt is Acknowledged. The PC value is pulled off the stack on a RETURN, RETLW or a RETFIE instruction. PCLATU and PCLATH are not affected by any of the RETURN or CALL instructions.

The stack operates as a 31-word by 21-bit RAM and a 5-bit Stack Pointer, with the Stack Pointer initialized to 00000b after all Resets. There is no RAM associated with Stack Pointer, 00000b. This is only a Reset value. During a CALL type instruction, causing a push onto the stack, the Stack Pointer is first incremented and the RAM location pointed to by the Stack Pointer is written with the contents of the PC (already pointing to the instruction following the CALL). During a RETURN type instruction, causing a pop from the stack, the contents of the RAM location pointed to by the STKPTR are transferred to the PC and then the Stack Pointer is decremented.

The stack space is not part of either program or data space. The Stack Pointer is readable and writable, and the address on the top of the stack is readable and writable through the Top-of-Stack (TOS) Special Function Registers. Data can also be pushed to, or popped from, the stack using the Top-of-Stack SFRs. Status bits indicate if the stack is full, has overflowed or underflowed.

6.1.2.1 Top-of-Stack Access

The top of the stack is readable and writable. Three register locations, TOSU, TOSH and TOSL, hold the contents of the stack location pointed to by the STKPTR register (Figure 6-3). This allows users to implement a software stack if necessary. After a CALL, RCALL or interrupt, the software can read the pushed value by reading the TOSU, TOSH and TOSL registers. These values can be placed on a user-defined software stack. At return time, the software can replace the TOSU, TOSH and TOSL and do a return.

The user must disable the global interrupt enable bits while accessing the stack to prevent inadvertent stack corruption.

6.1.2.2 Return Stack Pointer (STKPTR)

The STKPTR register (Register 6-1) contains the Stack Pointer value, the STKFUL (Stack Full) status bit and the STKUNF (Stack Underflow) status bits. The value of the Stack Pointer can be 0 through 31. The Stack Pointer increments before values are pushed onto the stack and decrements after values are popped off the stack. At Reset, the Stack Pointer value will be zero. The user may read and write the Stack Pointer value. This feature can be used by a Real-Time Operating System (RTOS) for return stack maintenance.

After the PC is pushed onto the stack 31 times (without popping any values off the stack), the STKFUL bit is set. The STKFUL bit is cleared by software or by a POR.

The action that takes place when the stack becomes full depends on the state of the STVREN (Stack Overflow Reset Enable) Configuration bit. (Refer to **Section 23.1 "Configuration Bits"** for a description of the device Configuration bits.) If STVREN is set (default), the 31st push will push the (PC + 2) value onto the stack, set the STKFUL bit and reset the device. The STKFUL bit will remain set and the Stack Pointer will be set to zero.

If STVREN is cleared, the STKFUL bit will be set on the 31st push and the Stack Pointer will increment to 31. Any additional pushes will not overwrite the 31st push and STKPTR will remain at 31.

6.5 Data Memory Organization

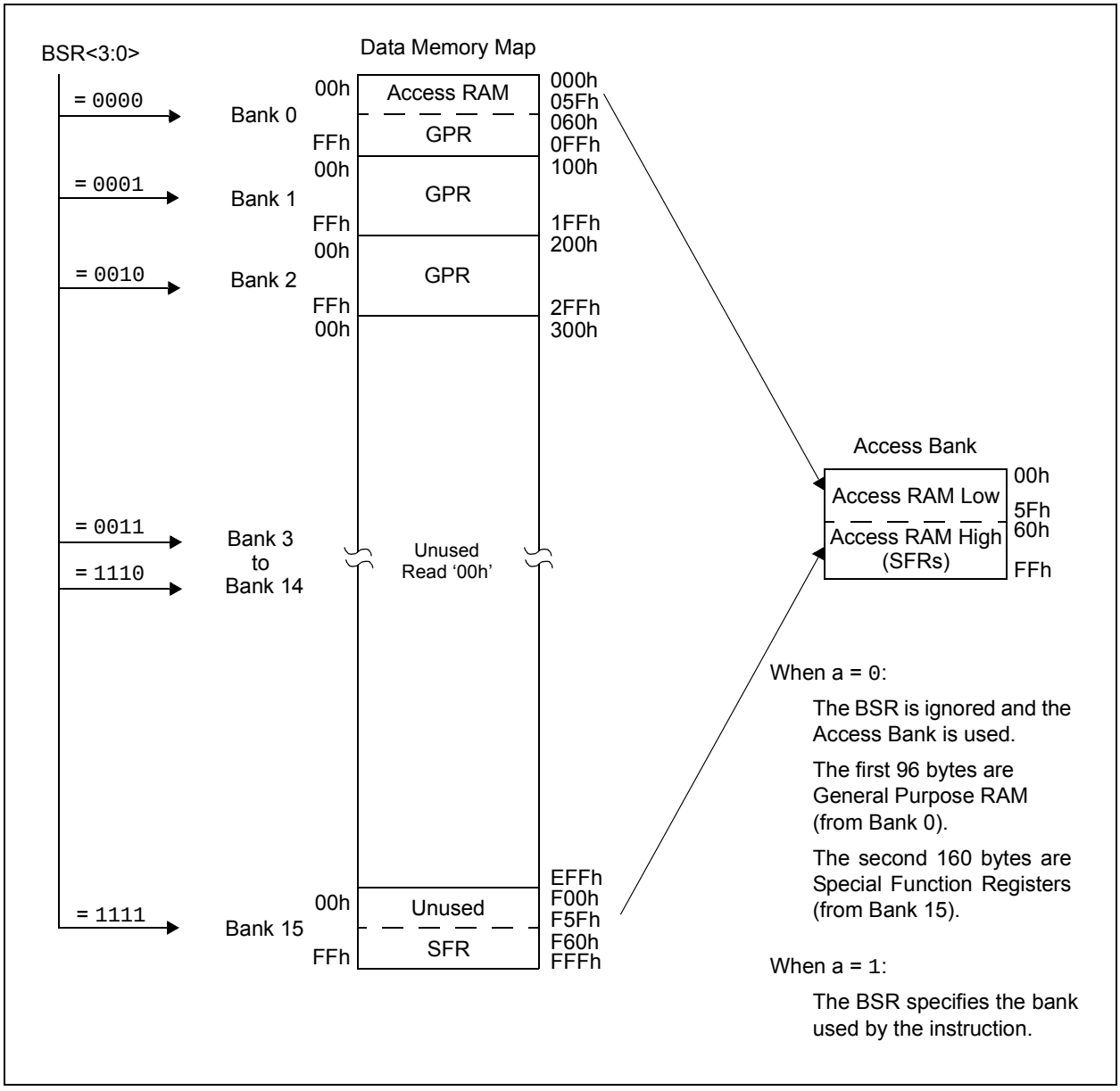
The data memory in PIC18 devices is implemented as static RAM. Each register in the data memory has a 12-bit address, allowing up to 4,096 bytes of data memory. The memory space is divided into as many as 16 banks that contain 256 bytes each. PIC18F2331/2431/4331/4431 devices implement all 16 banks.

Figure 6-6 shows the data memory organization for the PIC18F2331/2431/4331/4431 devices. The data memory contains Special Function Registers (SFRs) and General Purpose Registers (GPRs). The SFRs are used for control and status of the controller and peripheral functions, while GPRs are used for data storage and scratchpad operations in the user's application. Any read of an unimplemented location will read as '0's.

The instruction set and architecture allow operations across all banks. The entire data memory may be accessed by Direct, Indirect or Indexed Addressing modes. Addressing modes are discussed later in this subsection.

To ensure that commonly used registers (SFRs and select GPRs) can be accessed in a single cycle, PIC18 devices implement an Access Bank. This is a 256-byte memory space that provides fast access to SFRs and the lower portion of GPR Bank 0 without using the BSR. **Section 6.5.2 "Access Bank"** provides a detailed description of the Access RAM.

FIGURE 6-6: DATA MEMORY MAP FOR PIC18F2331/2431/4331/4431 DEVICES



8.5 Writing to Flash Program Memory

The programming block size is 4 words or 8 bytes. Word or byte programming is not supported.

Table writes are used internally to load the holding registers needed to program the Flash memory. There are 8 holding registers used by the table writes for programming.

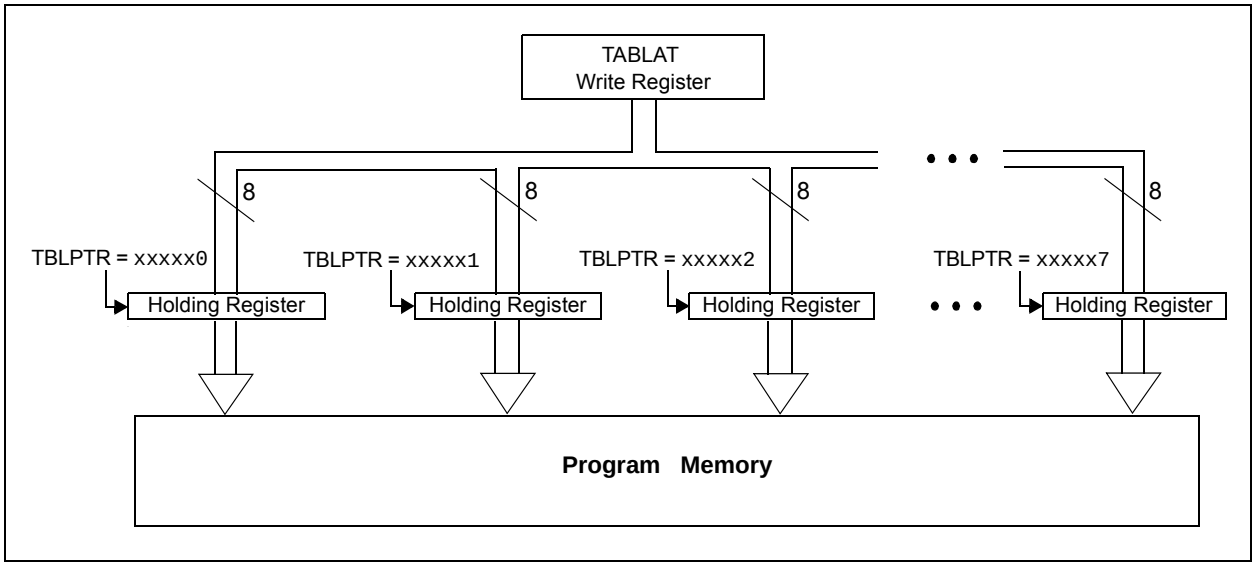
Since the Table Latch (TABLAT) is only a single byte, the TBLWT instruction has to be executed 8 times for each programming operation. All of the table write operations will essentially be short writes, because only the holding registers are written. At the end of updating 8 registers, the EECON1 register must be written to, to start the programming operation with a long write.

The long write is necessary for programming the internal Flash. Instruction execution is halted while in a long write cycle. The long write will be terminated by the internal programming timer.

The EEPROM on-chip timer controls the write time. The write/erase voltages are generated by an on-chip charge pump, rated to operate over the voltage range of the device.

Note:	The default value of the holding registers on device Resets and after write operations is FFh. A write of FFh to a holding register does not modify that byte. This means that individual bytes of program memory may be modified, provided that the modification does not attempt to change any bit from a '0' to a '1'. When modifying individual bytes, it is not necessary to load all 64 holding registers before executing a write operation.

FIGURE 8-5: TABLE WRITES TO FLASH PROGRAM MEMORY



11.4 PORTD, TRISD and LATD Registers

Note: PORTD is only available on PIC18F4331/4431 devices.

PORTD is an 8-bit wide, bidirectional port. The corresponding Data Direction register is TRISD. Setting a TRISD bit (= 1) will make the corresponding PORTD pin an input (i.e., put the corresponding output driver in a high-impedance mode). Clearing a TRISD bit (= 0) will make the corresponding PORTD pin an output (i.e., put the contents of the output latch on the selected pin).

The Data Latch register (LATD) is also memory mapped. Read-modify-write operations on the LATD register read and write the latched output value for PORTD.

All pins on PORTD are implemented with Schmitt Trigger input buffers. Each pin is individually configurable as an input or output.

Note: On a Power-on Reset, these pins are configured as digital inputs.

PORTD includes PWM<7:6> complementary fourth channel PWM outputs. PWM4 is the complementary output of PWM5 (the third channel), which is multiplexed with the RB5 pin. This output can be used as the alternate output using the PWM4MX Configuration bit in CONFIG3H when the Single-Supply Programming pin (PGM) is used on RB5.

RD1, RD2 and RD3 can be used as the alternate output for SDO, SDI/SDA and SCK/SCL using the SSPMX Configuration bit in CONFIG3H.

RD4 an be used as the alternate output for $\overline{\text{FLTA}}$ using the FLTAMX Configuration bit in CONFIG3H.

EXAMPLE 11-4: INITIALIZING PORTD

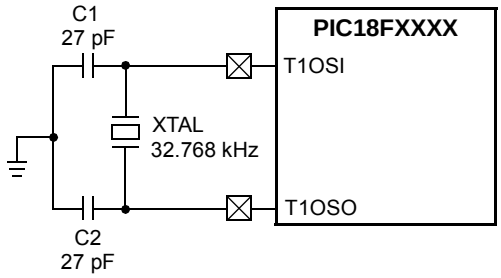
```
CLRF    PORTD   ; Initialize PORTD by
                ; clearing output
                ; data latches
CLRF    LATD    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   0xCF    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISD   ; Set RD<3:0> as inputs
                ; RD<5:4> as outputs
                ; RD<7:6> as inputs
```

13.2 Timer1 Oscillator

A crystal oscillator circuit is built in-between pins, T1OSI (input) and T1OSO (amplifier output). It is enabled by setting control bit, T1OSCEN (T1CON<3>). The oscillator is a low-power oscillator rated for 32 kHz crystals. It will continue to run during all power-managed modes. The circuit for a typical LP oscillator is shown in Figure 13-3. Table 13-1 shows the capacitor selection for the Timer1 oscillator.

The user must provide a software time delay to ensure proper start-up of the Timer1 oscillator.

FIGURE 13-3: EXTERNAL COMPONENTS FOR THE TIMER1 LP OSCILLATOR



Note: See the notes with Table 13-1 for additional information about capacitor selection.

TABLE 13-1: CAPACITOR SELECTION FOR THE TIMER OSCILLATOR

Osc Type	Freq	C1	C2
LP	32 kHz	27 pF(1)	27 pF(1)

Note 1: Microchip suggests this value as a starting point in validating the oscillator circuit.

2: Higher capacitance increases the stability of the oscillator, but also increases the start-up time.

3: Since each resonator/crystal has its own characteristics, the user should consult the resonator/crystal manufacturer for appropriate values of external components.

4: Capacitor values are for design guidance only.

13.3 Timer1 Oscillator Layout Considerations

The Timer1 oscillator for PIC18F2331/2431/4331/4431 devices incorporates an additional low-power feature. When this option is selected, it allows the oscillator to automatically reduce its power consumption when the microcontroller is in Sleep mode. During normal device operation, the oscillator draws full current. As high noise environments may cause excessive oscillator instability in Sleep mode, this option is best suited for low noise applications, where power conservation is an important design consideration.

The low-power option is enabled by clearing the T1OSCMX bit (CONFIG3L<5>). By default, the option is disabled, which results in a more or less constant current draw for the Timer1 oscillator.

Due to the low-power nature of the oscillator, it may also be sensitive to rapidly changing signals in close proximity.

The oscillator circuit, shown in Figure 13-3, should be located as close as possible to the microcontroller. There should be no circuits passing within the oscillator circuit boundaries other than VSS or VDD. Refer to **Section 2.0 "Guidelines for Getting Started with PIC18F Microcontrollers"** for additional information

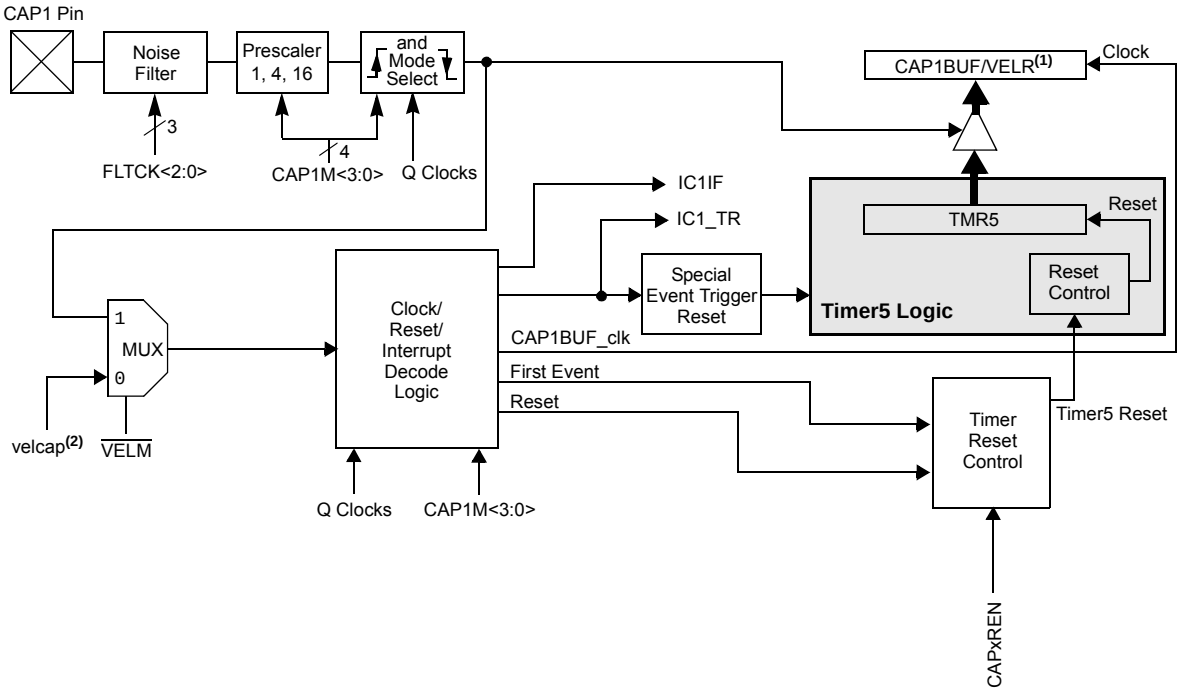
17.1 Input Capture

The Input Capture (IC) submodule implements the following features:

- Three channels of independent input capture (16-bits/channel) on the CAP1, CAP2 and CAP3 pins
- Edge-Trigger, Period or Pulse-Width Measurement Operating modes for each channel
- Programmable prescaler on every input capture channel
- Special Event Trigger output (IC1 only)
- Selectable noise filters on each capture input

Input Channel 1 (IC1) includes a Special Event Trigger that can be configured for use in Velocity Measurement mode. Its block diagram is shown in Figure 17-2. IC2 and IC3 are similar, but lack the Special Event Trigger features or additional velocity measurement logic. A representative block diagram is shown in Figure 17-3. Please note that the time base is Timer5.

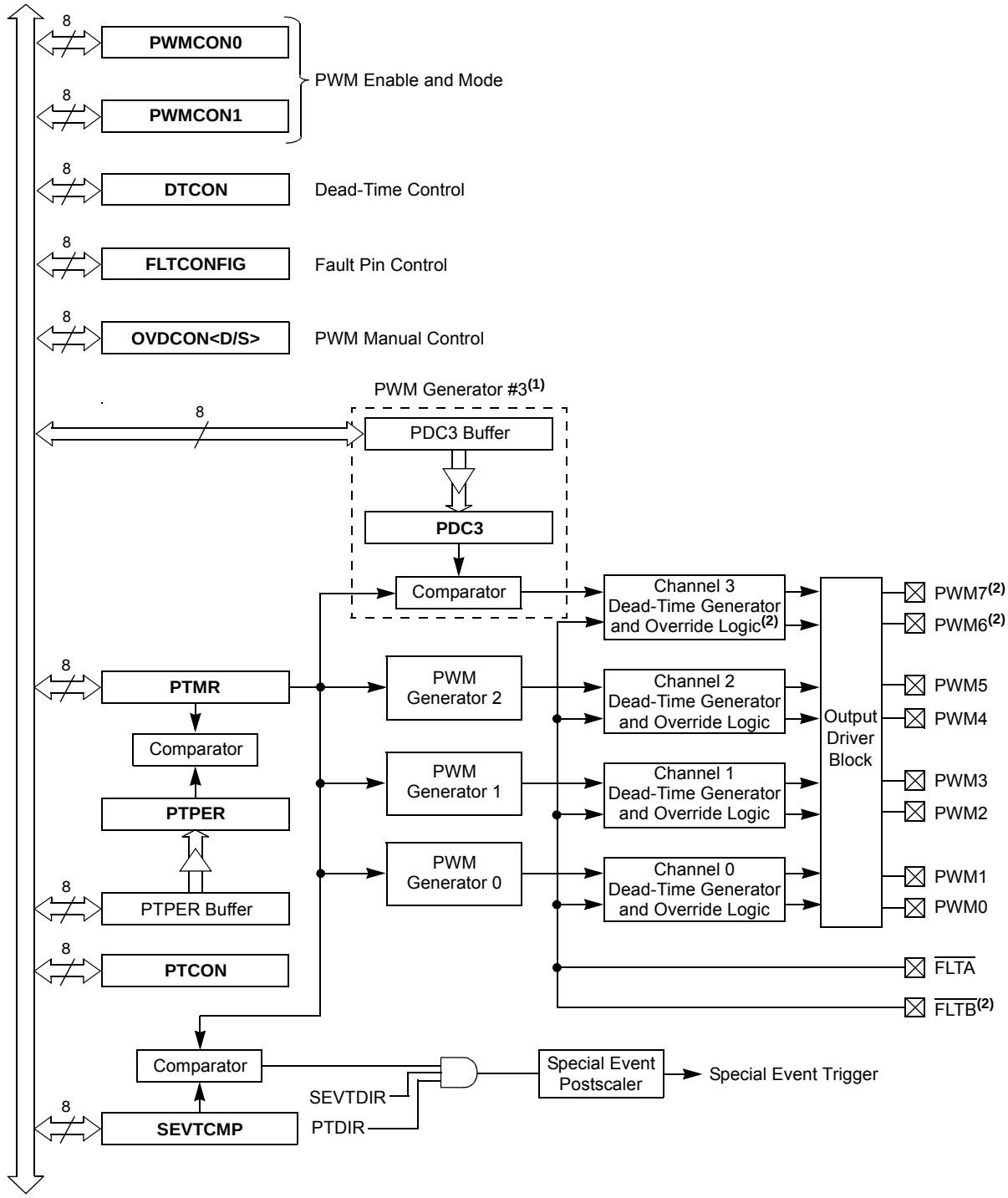
FIGURE 17-2: INPUT CAPTURE BLOCK DIAGRAM FOR IC1



Note 1: CAP1BUF register is reconfigured as VELR register when QEI mode is active.

2: QEI generated velocity pulses, vel_out, are downsampled to produce this velocity capture signal.

FIGURE 18-1: POWER CONTROL PWM MODULE BLOCK DIAGRAM



Note 1: Only PWM Generator 3 is shown in detail. The other generators are identical; their details are omitted for clarity.

2: PWM Generator 3 and its logic, PWM Channels 6 and 7, and $\overline{FLTB}$ and its associated logic are not implemented on PIC18F2331/2431 devices.

NOTES:

21.9 A/D Conversions

Figure 21-3 shows the operation of the A/D Converter after the GO/$\overline{\text{DONE}}$ bit has been set and the ACQT<2:0> bits are cleared. A conversion is started after the following instruction to allow entry into Sleep mode before the conversion begins. The internal A/D RC oscillator must be selected to perform a conversion in Sleep.

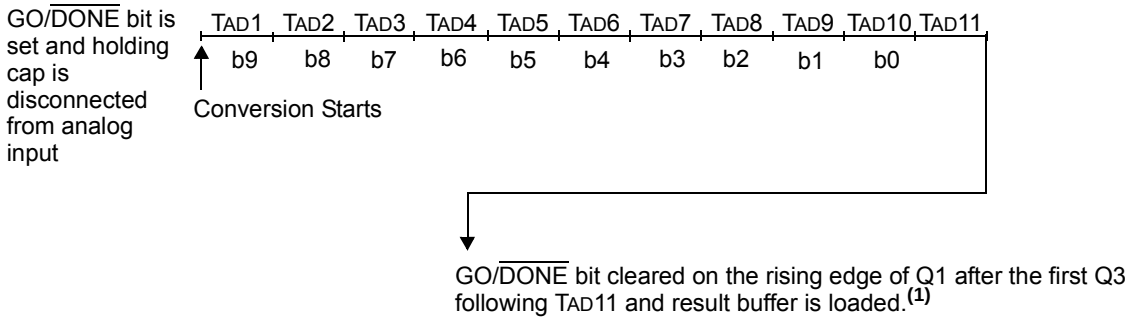
Figure 21-4 shows the operation of the A/D Converter after the GO/$\overline{\text{DONE}}$ bit has been set, the ACQT<3:0> bits are set to '010' and a 4 TAD acquisition time is selected before the conversion starts.

Clearing the GO/$\overline{\text{DONE}}$ bit during a conversion will abort the current conversion. The resulting buffer location will contain the partially completed A/D conversion sample. This will not set the ADIF flag, therefore, the user must read the buffer location before a conversion sequence overwrites it.

After the A/D conversion is completed or aborted, a 2 TAD wait is required before the next acquisition can be started. After this wait, acquisition on the selected channel is automatically started.

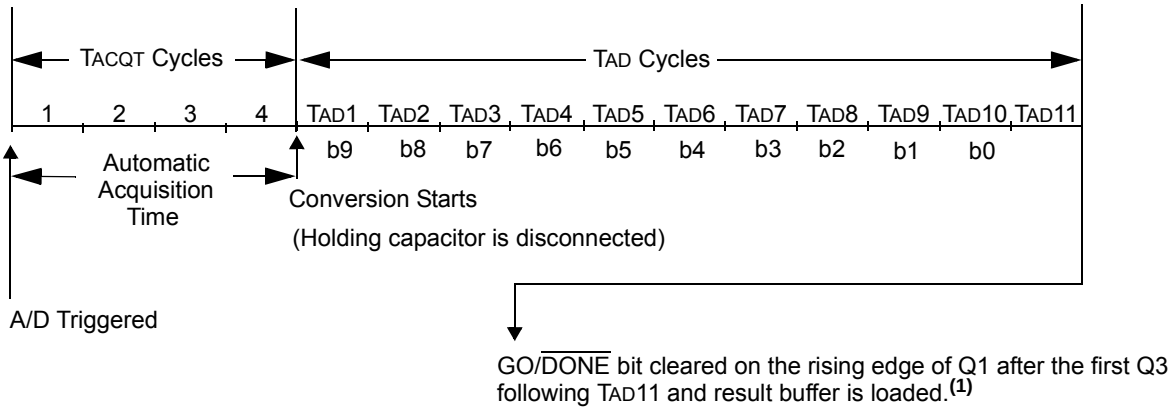
Note:	The GO/$\overline{\text{DONE}}$ bit should **NOT** be set in the same instruction that turns on the A/D.

FIGURE 21-3: A/D CONVERSION TAD CYCLES (ACQT<2:0> = 000, TACQ = 0)



Note 1: Conversion time is a minimum of 11 TAD + 2 TCY and a maximum of 11 TAD + 6 TCY.

FIGURE 21-4: A/D CONVERSION TAD CYCLES (ACQT<3:0> = 0010, TACQ = 4 TAD)



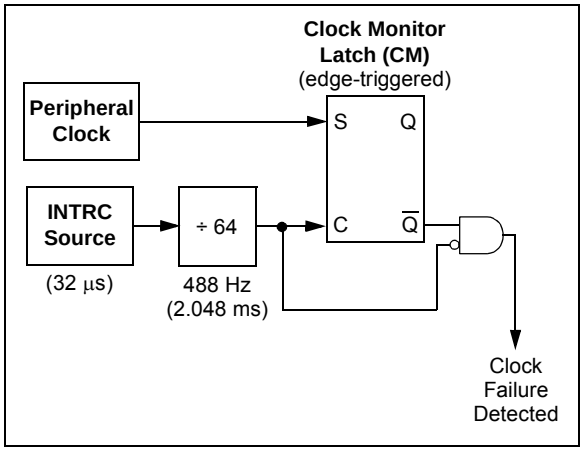
Note 1: In Continuous modes, next conversion starts at the end of TAD12.

23.4 Fail-Safe Clock Monitor

The Fail-Safe Clock Monitor (FSCM) allows the microcontroller to continue operation, in the event of an external oscillator failure, by automatically switching the system clock to the internal oscillator block. The FSCM function is enabled by setting the Fail-Safe Clock Monitor Enable bit, FCMEN (CONFIG1H<6>).

When FSCM is enabled, the INTRC oscillator runs at all times to monitor clocks to peripherals and provide an instant backup clock in the event of a clock failure. Clock monitoring (shown in Figure 23-3) is accomplished by creating a sample clock signal, which is the INTRC output divided by 64. This allows ample time between FSCM sample clocks for a peripheral clock edge to occur. The peripheral system clock and the sample clock are presented as inputs to the Clock Monitor latch (CM). The CM is set on the falling edge of the system clock source, but cleared on the rising edge of the sample clock.

FIGURE 23-3: FSCM BLOCK DIAGRAM



Clock failure is tested for on the falling edge of the sample clock. If a sample clock falling edge occurs while the CM is still set, a clock failure has been detected (Figure 23-4). This causes the following:

- the FSCM generates an oscillator fail interrupt by setting bit, OSCFIF (PIR2<7>);
- the system clock source is switched to the internal oscillator block (OSCCON is not updated to show the current clock source – this is the fail-safe condition); and
- the WDT is reset.

Since the postscaler frequency from the internal oscillator block may not be sufficiently stable, it may be desirable to select another clock configuration and enter an alternate power-managed mode (see **Section 23.3.1 "Special Considerations for Using Two-Speed Start-up"** and **Section 4.1.4 "Multiple Sleep Commands"** for more details). This can be done to attempt a partial recovery or execute a controlled shutdown.

To use a higher clock speed on wake-up, the INTOSC or postscaler clock sources can be selected to provide a higher clock speed by setting bits, IRCF<2:0>, immediately after Reset. For wake-ups from Sleep, the INTOSC or postscaler clock sources can be selected by setting the IRCF<2:0> bits prior to entering Sleep mode.

Adjustments to the internal oscillator block using the OSCTUNE register also affect the period of the FSCM by the same factor. This can usually be neglected, as the clock frequency being monitored is generally much higher than the sample clock frequency.

The FSCM will detect failures of the primary or secondary clock sources only. If the internal oscillator block fails, no failure would be detected, nor would any action be possible.

23.4.1 FSCM AND THE WATCHDOG TIMER

Both the FSCM and the WDT are clocked by the INTRC oscillator. Since the WDT operates with a separate divider and counter, disabling the WDT has no effect on the operation of the INTRC oscillator when the FSCM is enabled.

As already noted, the clock source is switched to the INTOSC clock when a clock failure is detected. Depending on the frequency selected by the IRCF<2:0> bits, this may mean a substantial change in the speed of code execution. If the WDT is enabled with a small prescale value, a decrease in clock speed allows a WDT time-out to occur and a subsequent device Reset. For this reason, Fail-Safe Clock Monitor events also reset the WDT and postscaler, allowing it to start timing from when execution speed was changed and decreasing the likelihood of an erroneous time-out.

23.4.2 EXITING FAIL-SAFE OPERATION

The fail-safe condition is terminated by either a device Reset, or by entering a power-managed mode. On Reset, the controller starts the primary clock source specified in Configuration Register 1H (with any required start-up delays that are required for the oscillator mode, such as the OST or PLL timer). The INTOSC multiplexer provides the system clock until the primary clock source becomes ready (similar to a Two-Speed Start-up). The clock system source is then switched to the primary clock (indicated by the OSTS bit in the OSCCON register becoming set). The Fail-Safe Clock Monitor then resumes monitoring the peripheral clock.

The primary clock source may never become ready during start-up. In this case, operation is clocked by the INTOSC multiplexer. The OSCCON register will remain in its Reset state until a power-managed mode is entered.

Entering a power-managed mode by loading the OSCCON register and executing a SLEEP instruction will clear the fail-safe condition. When the fail-safe condition is cleared, the clock monitor will resume monitoring the peripheral clock.

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.

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 $\rightarrow$ 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)

GOTO **Unconditional Branch**

Syntax: [*label*] GOTO k

Operands: $0 \le k \le 1048575$

Operation: $k \rightarrow PC<20:1>$

Status Affected: None

Encoding:

1st word (k<7:0>)	1110	1111	k_7kkk	$kkkk_0$
2nd word(k<19:8>)	1111	$k_{19}kkk$	kkkk	$kkkk_8$

Description: GOTO allows an unconditional branch anywhere within entire 2-Mbyte memory range. The 20-bit value, 'k', is loaded into PC<20:1>. GOTO is always a two-cycle instruction.

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'<7:0>,	No operation	Read literal 'k'<19:8>, Write to PC
No operation	No operation	No operation	No operation

Example: GOTO THERE

After Instruction
PC = Address (THERE)

INCF **Increment f**

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

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

Operation: (f) + 1 → dest

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

Encoding:

0010	10da	ffff	ffff

Description: The contents of register, 'f', are incremented. 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' = 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: INCF CNT,

Before Instruction
CNT = 0xFF
Z = 0
C = ?
DC = ?

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

MOVFF **Move f to f**

Syntax: [*label*] MOVFF f_s,f_d

Operands: $0 \le f_s \le 4095$
$0 \le f_d \le 4095$

Operation: $(f_s) \rightarrow f_d$

Status Affected: None

Encoding:

1st word (source)	1100	ffff	ffff	$ffff_s$
2nd word (destin.)	1111	ffff	ffff	$ffff_d$

Description: The contents of source register, 'f_s', are moved to destination register, 'f_d'. Location of source, 'f_s', can be anywhere in the 4096-byte data space (000h to FFFh) and location of destination, 'f_d', can also be anywhere from 000h to FFFh.
Either source or destination can be W (a useful special situation).
MOVFF is particularly useful for transferring a data memory location to a peripheral register (such as the transmit buffer or an I/O port).
The MOVFF instruction cannot use the PCL, TOSU, TOSH or TOSL as the destination register.
The MOVFF instruction should not be used to modify interrupt settings while any interrupt is enabled (see the note on page 97).

Words: 2

Cycles: 2 (3)

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f' (src)	Process Data	No operation
Decode	No operation No dummy read	No operation	Write register 'f' (dest)

Example: MOVFF REG1, REG2

Before Instruction
REG1 = 0x33
REG2 = 0x11
After Instruction
REG1 = 0x33
REG2 = 0x33

MOVLB **Move Literal to Low Nibble in BSR**

Syntax: [*label*] MOVLB k

Operands: $0 \le k \le 255$

Operation: $k \rightarrow BSR$

Status Affected: None

Encoding:

0000	0001	0000	kkkk

Description: The 8-bit literal, 'k', is loaded into the Bank Select Register (BSR).

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write literal 'k' to BSR

Example: MOVLB 5

Before Instruction
BSR register = 0x02
After Instruction
BSR register = 0x05

MOVLW **Move Literal to W**

Syntax: [*label*] MOVLW k

Operands: $0 \le k \le 255$

Operation: $k \rightarrow W$

Status Affected: None

Encoding:

0000	1110	kkkk	kkkk

Description: The 8-bit literal, ‘k’, is loaded into W.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal ‘k’	Process Data	Write to W

Example: MOVLW 0x5A

After Instruction
W = 0x5A

MOVWF **Move W to f**

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

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

Operation: $(W) \rightarrow f$

Status Affected: None

Encoding:

0110	111a	ffff	ffff

Description: Move data from W to register, ‘f’. Location, ‘f’, can be anywhere in the 256-byte bank. 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.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: MOVWF REG

Before Instruction
W = 0x4F
REG = 0xFF
After Instruction
W = 0x4F
REG = 0x4F

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

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**		
	Supply Current (IDD)(2,3)						
	PIC18LF2X31/4X31	5.1	9	µA	-10°C	VDD = 2.0V	FOSC = 32 kHz(4) (**SEC_RUN** mode, Timer1 as clock)
		5.8	9	µA	+25°C		
		7.9	11	µA	+70°C		
	PIC18LF2X31/4X31	7.9	12	µA	-10°C	VDD = 3.0V	
		8.9	12	µA	+25°C		
		10.5	14	µA	+70°C		
	All devices	12.5	20	µA	-10°C	VDD = 5.0V	
		16.3	20	µA	+25°C		
		18.9	25	µA	+70°C		
		150	850	µA	+125°C		
	PIC18LF2X31/4X31	9.2	15	µA	-10°C	VDD = 2.0V	FOSC = 32 kHz(4) (**SEC_IDLE** mode, Timer1 as clock)
		9.6	15	µA	+25°C		
		12.7	18	µA	+70°C		
	PIC18LF2X31/4X31	22.0	30	µA	-10°C	VDD = 3.0V	
		21.0	30	µA	+25°C		
		20.0	35	µA	+70°C		
	All devices	30	80	µA	-10°C	VDD = 5.0V	
		45	80	µA	+25°C		
		45	85	µA	+70°C		
		250	850	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.

26.3 DC Characteristics: PIC18F2331/2431/4331/4431 (Industrial, Extended) PIC18LF2331/2431/4331/4431 (Industrial) (Continued)

DC CHARACTERISTICS			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**	**Max**	**Units**	**Conditions**
	VOL	**Output Low Voltage**				
D080		I/O Ports	—	0.6	V	IOL = 8.5 mA, VDD = 4.5V, -40°C to +85°C
D083		OSC2/CLKO (RC, RCIO, EC, ECIO modes)	—	0.6	V	IOL = 1.6 mA, VDD = 4.5V, -40°C to +85°C
	VOH	**Output High Voltage(3)**				
D090		I/O Ports	VDD – 0.7	—	V	IOH = -3.0 mA, VDD = 4.5V, -40°C to +85°C
D092		OSC2/CLKO (RC, RCIO, EC, ECIO modes)	VDD – 0.7	—	V	IOH = -1.3 mA, VDD = 4.5V, -40°C to +85°C
		Capacitive Loading Specs on Output Pins				
D100	COSC2	OSC2 Pin	—	15	pF	In XT, HS and LP modes when external clock is used to drive OSC1
D101	CIO	All I/O Pins and OSC2 (in RC mode)	—	50	pF	To meet the AC Timing Specifications
D102	CB	SCL, SDA	—	400	pF	I^2C™ Specification

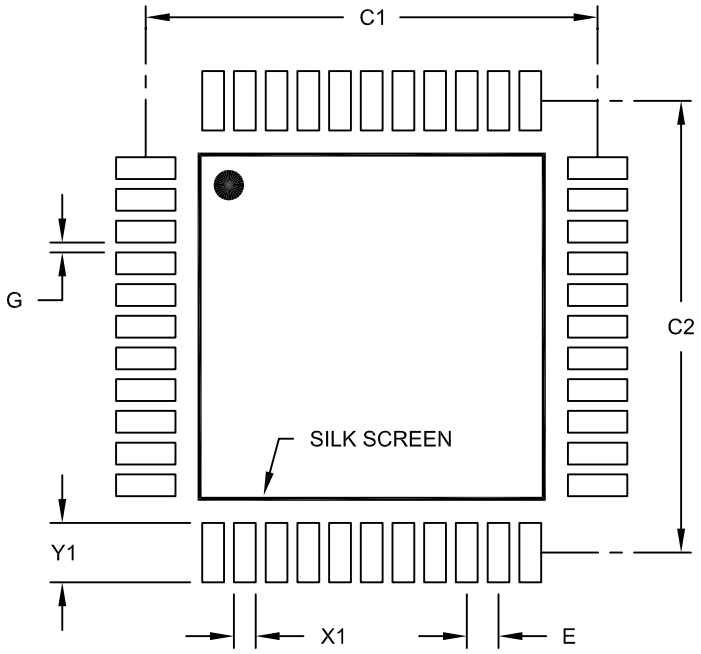
Note 1: In RC oscillator configuration, the OSC1/CLKI pin is a Schmitt Trigger input. It is not recommended that the PIC® device be driven with an external clock while in RC mode.

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.

44-Lead Plastic Thin Quad Flatpack (PT) – 10x10x1 mm Body, 2.00 mm [TQFP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E		0.80 BSC	
Contact Pad Spacing	C1		11.40	
Contact Pad Spacing	C2		11.40	
Contact Pad Width (X44)	X1			0.55
Contact Pad Length (X44)	Y1			1.50
Distance Between Pads	G	0.25		

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2076A

APPENDIX C: CONVERSION CONSIDERATIONS

This appendix discusses the considerations for converting from previous versions of a device to the ones listed in this data sheet. Typically, these changes are due to the differences in the process technology used. An example of this type of conversion is from a PIC16C74A to a PIC16C74B.

Not Applicable

APPENDIX D: MIGRATION FROM BASELINE TO ENHANCED DEVICES

This section discusses how to migrate from a baseline device (i.e., PIC16C5X) to an enhanced MCU device (i.e., PIC18FXXX).

The following are the list of modifications over the PIC16C5X microcontroller family:

Not Currently Available