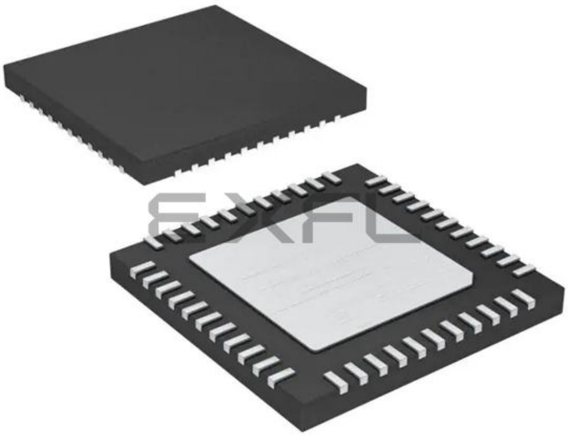


What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Obsolete
Core Processor	PIC
Core Size	8-Bit
Speed	40MHz
Connectivity	I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, LVD, Power Control PWM, QEI, POR, PWM, WDT
Number of I/O	36
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 9x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	44-VQFN Exposed Pad
Supplier Device Package	44-QFN (8x8)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f4331t-i-ml

28/40/44-Pin Enhanced Flash Microcontrollers with nanoWatt Technology, High-Performance PWM and A/D

14-Bit Power Control PWM Module:

- Up to 4 Channels with Complementary Outputs
- Edge or Center-Aligned Operation
- Flexible Dead-Band Generator
- Hardware Fault Protection Inputs
- Simultaneous Update of Duty Cycle and Period:
  - Flexible Special Event Trigger output

Motion Feedback Module:

- Three Independent Input Capture Channels:
  - Flexible operating modes for period and pulse-width measurement
  - Special Hall sensor interface module
  - Special Event Trigger output to other modules
- Quadrature Encoder Interface:
  - 2-phase inputs and one index input from encoder
  - High and low position tracking with direction status and change of direction interrupt
  - Velocity measurement

High-Speed, 200 ksps 10-Bit A/D Converter:

- Up to 9 Channels
- Simultaneous, Two-Channel Sampling
- Sequential Sampling: 1, 2 or 4 Selected Channels
- Auto-Conversion Capability
- 4-Word FIFO with Selectable Interrupt Frequency
- Selectable External Conversion Triggers
- Programmable Acquisition Time

Flexible Oscillator Structure:

- Four Crystal modes up to 40 MHz
- Two External Clock modes up to 40 MHz
- Internal Oscillator Block:
  - 8 user-selectable frequencies: 31 kHz to 8 MHz
  - OSCTUNE can compensate for frequency drift
- Secondary Oscillator using Timer1 @ 32 kHz
- Fail-Safe Clock Monitor:
  - Allows for safe shutdown of device if clock fails

Power-Managed Modes:

- Run: CPU on, Peripherals on
- Idle: CPU off, Peripherals on
- Sleep: CPU off, Peripherals off
- Ultra Low, 50 nA Input Leakage
- Idle mode Currents Down to 5.8 μA, Typical
- Sleep Current Down to 0.1 μA, Typical
- Timer1 Oscillator, 1.8 μA, Typical, 32 kHz, 2V
- Watchdog Timer (WDT), 2.1 μA, typical
- Oscillator Two-Speed Start-up
  - Fast wake from Sleep and Idle, 1 μs, typical

Peripheral Highlights:

- High-Current Sink/Source 25 mA/25 mA
- Three External Interrupts
- Two Capture/Compare/PWM (CCP) modules
- Enhanced USART module:
  - Supports RS-485, RS-232 and LIN/J2602
  - Auto-wake-up on Start bit
  - Auto-Baud Detect

Special Microcontroller Features:

- 100,000 Erase/Write Cycle Enhanced Flash Program Memory, Typical
- 1,000,000 Erase/Write Cycle Data EEPROM Memory, Typical
- Flash/Data EEPROM Retention: 100 Years
- Self-Programmable under Software Control
- Priority Levels for Interrupts
- 8 x 8 Single-Cycle Hardware Multiplier
- Extended Watchdog Timer (WDT):
  - Programmable period from 41 ms to 131s
- Single-Supply In-Circuit Serial Programming™ (ICSP™) via Two Pins
- In-Circuit Debug (ICD) via Two Pins:
  - Drives PWM outputs safely when debugging

Device	Program Memory		Data Memory		I/O	10-Bit A/D (ch)	CCP	SSP		EUSART	Quadrature Encoder	14-Bit PWM (ch)	Timers 8/16-Bit
	Flash (bytes)	# Single-Word Instructions	SRAM (bytes)	EEPROM (bytes)				SPI	Slave I^2C™				
PIC18F2331	8192	4096	768	256	24	5	2	Y	Y	Y	Y	6	1/3
PIC18F2431	16384	8192	768	256	24	5	2	Y	Y	Y	Y	6	1/3
PIC18F4331	8192	4096	768	256	36	9	2	Y	Y	Y	Y	8	1/3
PIC18F4431	16384	8192	768	256	36	9	2	Y	Y	Y	Y	8	1/3

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
IPR2	2331	2431	4331	4431	`1--1 -1-1`	`1--1 -1-1`	`u--u -u-u`
PIR2	2331	2431	4331	4431	`0--0 -0-0`	`0--0 -0-0`	`u--u -u-u`
PIE2	2331	2431	4331	4431	`0--0 -0-0`	`0--0 -0-0`	`u--u -u-u`
IPR1	2331	2431	4331	4431	`-111 1111`	`-111 1111`	`-uuu uuuu`
PIR1	2331	2431	4331	4431	`-000 0000`	`-000 0000`	`-uuu uuuu`(1)
	2331	2431	4331	4431	`-000 0000`	`-000 0000`	`-uuu uuuu`(1)
PIE1	2331	2431	4331	4431	`0000 0000`	`0000 0000`	`uuuu uuuu`
	2331	2431	4331	4431	`-000 0000`	`-000 0000`	`-uuu uuuu`
OSCTUNE	2331	2431	4331	4431	`--00 0000`	`--00 0000`	`--uu uuuu`
TRISE(6)	2331	2431	4331	4431	`---- -111`	`---- -111`	`---- -uuu`
TRISD	2331	2431	4331	4431	`1111 1111`	`1111 1111`	`uuuu uuuu`
TRISC	2331	2431	4331	4431	`1111 1111`	`1111 1111`	`uuuu uuuu`
TRISB	2331	2431	4331	4431	`1111 1111`	`1111 1111`	`uuuu uuuu`
TRISA(5)	2331	2431	4331	4431	`1111 1111`(5)	`1111 1111`(5)	`uuuu uuuu`(5)
PR5H	2331	2431	4331	4431	`1111 1111`	`1111 1111`	`uuuu uuuu`
PR5L	2331	2431	4331	4431	`1111 1111`	`1111 1111`	`uuuu uuuu`
LATE(6)	2331	2431	4331	4431	`---- -xxx`	`---- -uuu`	`---- -uuu`
LATD	2331	2431	4331	4431	`xxxx xxxx`	`uuuu uuuu`	`uuuu uuuu`
LATC	2331	2431	4331	4431	`xxxx xxxx`	`uuuu uuuu`	`uuuu uuuu`
LATB	2331	2431	4331	4431	`xxxx xxxx`	`uuuu uuuu`	`uuuu uuuu`
LATA(5)	2331	2431	4331	4431	`xxxx xxxx`(5)	`uuuu uuuu`(5)	`uuuu uuuu`(5)
TMR5H	2331	2431	4331	4431	`xxxx xxxx`	`uuuu uuuu`	`uuuu uuuu`
TMR5L	2331	2431	4331	4431	`xxxx xxxx`	`uuuu uuuu`	`uuuu uuuu`
PORTE(6)	2331	2431	4331	4431	`---- xxxx`	`---- xxxx`	`---- uuuu`
PORTD	2331	2431	4331	4431	`xxxx xxxx`	`uuuu uuuu`	`uuuu uuuu`
PORTC	2331	2431	4331	4431	`xxxx xxxx`	`uuuu uuuu`	`uuuu uuuu`
PORTB	2331	2431	4331	4431	`xxxx xxxx`	`uuuu uuuu`	`uuuu uuuu`
PORTA(5)	2331	2431	4331	4431	`xx0x 0000`(5)	`uu0u 0000`(5)	`uuuu uuuu`(5)

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.

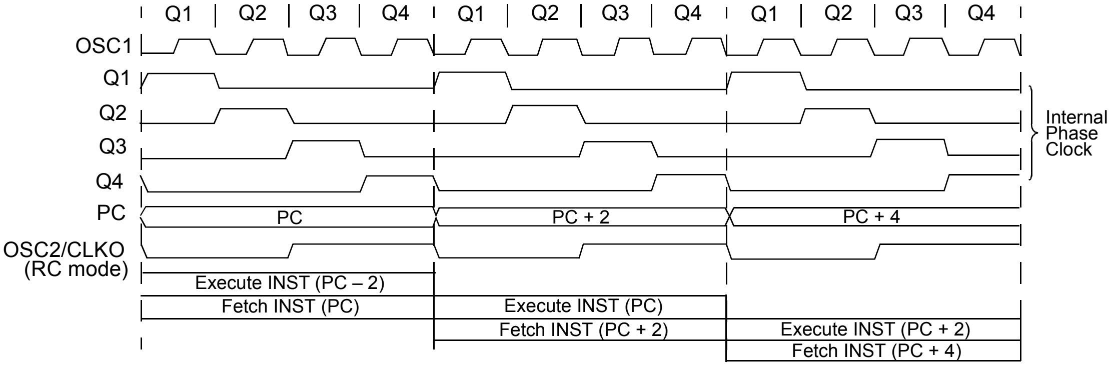
The Table Pointer register (TBLPTR) specifies the byte address and the Table Latch register (TABLAT) contains the data that is read from or written to program memory. Data is transferred to or from program memory, one byte at a time.

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.

TABLE 6-2: REGISTER FILE SUMMARY (PIC18F2331/2431/4331/4431)

File Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR
TOSU	—	—	—	Top-of-Stack Upper Byte (TOS<20:16>)					---0 0000
TOSH	Top-of-Stack High Byte (TOS<15:8>)								0000 0000
TOSL	Top-of-Stack Low Byte (TOS<7:0>)								0000 0000
STKPTR	STKFUL	STKUNF	—	SP4	SP3	SP2	SP1	SP0	00-0 0000
PCLATU	—	—	bit 21(3)	Holding Register for PC<20:16>					---0 0000
PCLATH	Holding Register for PC<15:8>								0000 0000
PCL	PC Low Byte (PC<7:0>)								0000 0000
TBLPTRU	—	—	bit 21(3)	Program Memory Table Pointer Upper Byte (TBLPTR<20:16>)					--00 0000
TBLPTRH	Program Memory Table Pointer High Byte (TBLPTR<15:8>)								0000 0000
TBLPTRL	Program Memory Table Pointer Low Byte (TBLPTR<7:0>)								0000 0000
TABLAT	Program Memory Table Latch								0000 0000
PRODH	Product Register High Byte								xxxx xxxx
PRODL	Product Register Low Byte								xxxx xxxx
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x
INTCON2	$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	—	TMR0IP	—	RBIP	1111 -1-1
INTCON3	INT2IP	INT1IP	—	INT2IE	INT1IE	—	INT2IF	INT1IF	11-0 0-00
INDF0	Uses contents of FSR0 to address data memory – value of FSR0 not changed (not a physical register)								N/A
POSTINC0	Uses contents of FSR0 to address data memory – value of FSR0 post-incremented (not a physical register)								N/A
POSTDEC0	Uses contents of FSR0 to address data memory – value of FSR0 post-decremented (not a physical register)								N/A
PREINC0	Uses contents of FSR0 to address data memory – value of FSR0 pre-incremented (not a physical register)								N/A
PLUSW0	Uses contents of FSR0 to address data memory – value of FSR0 offset by W (not a physical register)								N/A
FSR0H	—	—	—	—	Indirect Data Memory Address Pointer 0 High				---- xxxx
FSR0L	Indirect Data Memory Address Pointer 0 Low Byte								xxxx xxxx
WREG	Working Register								xxxx xxxx
INDF1	Uses contents of FSR1 to address data memory – value of FSR1 not changed (not a physical register)								N/A
POSTINC1	Uses contents of FSR1 to address data memory – value of FSR1 post-incremented (not a physical register)								N/A
POSTDEC1	Uses contents of FSR1 to address data memory – value of FSR1 post-decremented (not a physical register)								N/A
PREINC1	Uses contents of FSR1 to address data memory – value of FSR1 pre-incremented (not a physical register)								N/A
PLUSW1	Uses contents of FSR1 to address data memory – value of FSR1 offset by W (not a physical register)								N/A
FSR1H	—	—	—	—	Indirect Data Memory Address Pointer 1 High Byte				---- 0000
FSR1L	Indirect Data Memory Address Pointer 1 Low Byte								xxxx xxxx
BSR	—	—	—	—	Bank Select Register				---- 0000
INDF2	Uses contents of FSR2 to address data memory – value of FSR2 not changed (not a physical register)								N/A
POSTINC2	Uses contents of FSR2 to address data memory – value of FSR2 post-incremented (not a physical register)								N/A
POSTDEC2	Uses contents of FSR2 to address data memory – value of FSR2 post-decremented (not a physical register)								N/A
PREINC2	Uses contents of FSR2 to address data memory – value of FSR2 pre-incremented (not a physical register)								N/A
PLUSW2	Uses contents of FSR2 to address data memory – value of FSR2 offset by W (not a physical register)								N/A
FSR2H	—	—	—	—	Indirect Data Memory Address Pointer 2 High Byte				---- 0000
FSR2L	Indirect Data Memory Address Pointer 2 Low Byte								xxxx xxxx
STATUS	—	—	—	N	OV	Z	DC	C	---x xxxx
TMR0H	Timer0 Register High Byte								0000 0000
TMR0L	Timer0 Register Low Byte								xxxx xxxx
T0CON	TMR0ON	T016BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0	1111 1111

Legend: x = unknown, u = unchanged, - = unimplemented, q = value depends on condition. Shaded cells are unimplemented.

Note 1: RA6 and associated bits are configured as port pins in RCIO, ECIO and INTIO2 (with port function on RA6) Oscillator modes only and read ‘0’ in all other oscillator modes.

2: RA7 and associated bits are configured as port pins in INTIO2 Oscillator mode only and read ‘0’ in all other modes.

3: Bit 21 of the PC is only available in Test mode and Serial Programming modes.

4: These registers and/or bits are not implemented on the PIC18F2331/2431 devices and read as ‘0’.

5: The RE3 port bit is only available for PIC18F4331/4431 devices when the MCLRE fuse (CONFIG3H<7>) is programmed to ‘0’; otherwise, RE3 reads ‘0’. This bit is read-only.

8.5.1 FLASH PROGRAM MEMORY WRITE SEQUENCE

The sequence of events for programming an internal program memory location should be:

1. Read 64 bytes into RAM.
2. Update data values in RAM as necessary.
3. Load Table Pointer with address being erased.
4. Do the row erase procedure (see **Section 8.4.1 "Flash Program Memory Erase Sequence"**).
5. Load Table Pointer with the address of the first byte being written.
6. Write the first 8 bytes into the holding registers with auto-increment.
7. Set the EECON1 register for the write operation by doing the following:
   - Set the EEPGD bit to point to program memory
   - Clear the CFGS bit to access program memory
   - Set the WREN bit to enable byte writes
8. Disable interrupts.
9. Write 55h to EECON2.
10. Write 0AAh to EECON2.
11. Set the WR bit. This will begin the write cycle.
12. The CPU will stall for the duration of the write (about 2 ms using internal timer).
13. Execute a `NOP`.
14. Re-enable interrupts.
15. Repeat Steps 6-14 seven times to write 64 bytes.
16. Verify the memory (table read).

This procedure will require about 18 ms to update one row of 64 bytes of memory. An example of the required code is given in Example 8-3.

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

REGISTER 10-12: IPR3: PERIPHERAL INTERRUPT PRIORITY REGISTER 3

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	PTIP	IC3DRIP	IC2QEIP	IC1IP	TMR5IP
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 7-5 **Unimplemented:** Read as ‘0’

bit 4 **PTIP:** PWM Time Base Interrupt Priority bit
1 = High priority
0 = Low priority

bit 3 **IC3DRIP:** IC3 Interrupt Priority/Direction Change Interrupt Priority bit
IC3 Enabled (CAP3CON<3:0>):
1 = IC3 interrupt high priority
0 = IC3 interrupt low priority
QEI Enabled (QEIM<2:0>):
1 = Change of direction interrupt high priority
0 = Change of direction interrupt low priority

bit 2 **IC2QEIP:** IC2 Interrupt Priority/QEI Interrupt Priority bit
IC2 Enabled (CAP2CON<3:0>):
1 = IC2 interrupt high priority
0 = IC2 interrupt low priority
QEI Enabled (QEIM<2:0>):
1 = High priority
0 = Low priority

bit 1 **IC1IP:** IC1 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 0 **TMR5IP:** Timer5 Interrupt Priority bit
1 = High priority
0 = Low priority

16.3 Capture Mode

In Capture mode, CCPR1H:CCPR1L captures the 16-bit value of the TMR1 register when an event occurs on pin RC2/CCP1. An event is defined as one of the following:

- every falling edge
- every rising edge
- every 4th rising edge
- every 16th rising edge

The event is selected by control bits, CCP1M<3:0> (CCP1CON<3:0>). When a capture is made, the interrupt request flag bit, CCP1IF (PIR1<2>), is set; it must be cleared in software. If another capture occurs before the value in register CCPR1 is read, the old captured value is overwritten by the new captured value.

16.3.1 CCP PIN CONFIGURATION

In Capture mode, the RC2/CCP1 pin should be configured as an input by setting the TRISC<2> bit.

Note:	If the RC2/CCP1 pin is configured as an output, a write to the port can cause a capture condition.

16.3.2 TIMER1 MODE SELECTION

Timer1 must be running in Timer mode or Synchronized Counter mode to be used with the capture feature. In Asynchronous Counter mode, the capture operation may not work.

16.3.3 SOFTWARE INTERRUPT

When the Capture mode is changed, a false capture interrupt may be generated. The user should keep bit, CCP1IE (PIE1<2>), clear to avoid false interrupts and should clear the flag bit, CCP1IF, following any such change in operating mode.

16.3.4 CCP PRESCALER

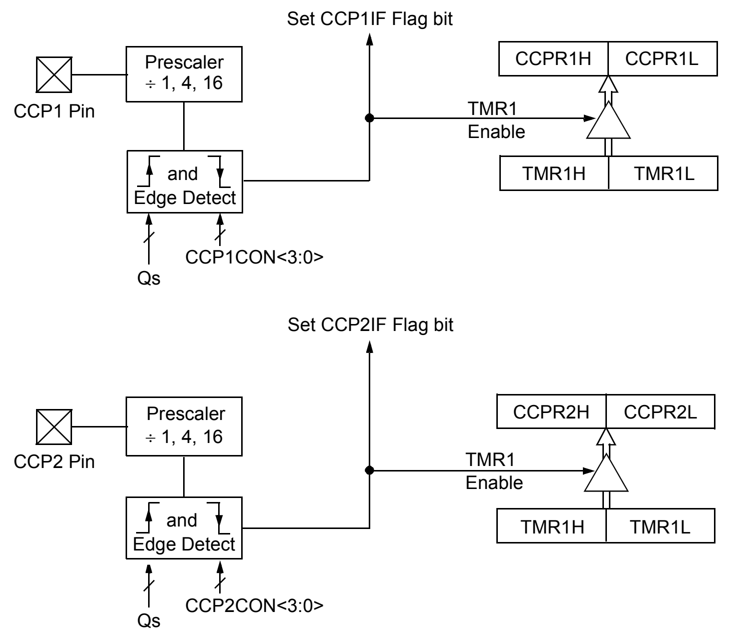
There are four prescaler settings specified by bits CCP1M<3:0>. Whenever the CCP module is turned off, or the CCP module is not in Capture mode, the prescaler counter is cleared. This means that any Reset will clear the prescaler counter.

Switching from one capture prescaler to another may generate an interrupt. Also, the prescaler counter will not be cleared, therefore, the first capture may be from a non-zero prescaler. Example 16-1 shows the recommended method for switching between capture prescalers. This example also clears the prescaler counter and will not generate the “false” interrupt.

EXAMPLE 16-1: CHANGING BETWEEN CAPTURE PRESCALERS

```
CLRF    CCP1CON        ; Turn CCP module off
MOVLW   NEW_CAPT_PS    ; Load WREG with the
                       ; new prescaler mode
                       ; value and CCP ON
MOVWF   CCP1CON        ; Load CCP1CON with
                       ; this value
```

FIGURE 16-1: CAPTURE MODE OPERATION BLOCK DIAGRAM



17.2.2 QEI MODES

Position measurement resolution depends on how often the Position Counter register, POSCNT, is incremented. There are two QEI Update modes to measure the rotor's position: QEI x2 and QEI x4.

TABLE 17-4: QEI MODES

QEIM<2:0>	Mode/ Reset	Description
000	—	QEI disabled.[1]
001	x2 update/ index pulse	Two clocks per QEA pulse. INDX resets POSCNT.
010	x2 update/ period match	Two clocks per QEA pulse. POSCNT is reset by the period match (MAXCNT).
011	—	Unused.
100	—	Unused.
101	x4 update/ index pulse	Four clocks per QEA and QEB pulse pair. INDX resets POSCNT.
110	x4 update/ period match	Four clocks per QEA and QEB pulse pair. POSCNT is reset by the period match (MAXCNT).
111	—	Unused.

Note 1: QEI module is disabled. The position counter and the velocity measurement functions are fully disabled in this mode.

17.2.2.1 QEI x2 Update Mode

QEI x2 Update mode is selected by setting the QEI Mode Select bits (QEIM<2:0>) to '001' or '010'. In this mode, the QEI logic detects every edge on the QEA input only. Every rising and falling edge on the QEA signal clocks the position counter.

The position counter can be reset by either an input on the INDX pin (QEIM<2:0> = 001), or by a period match, even when the POSCNT register pair equals MAXCNT (QEIM<2:0> = 010).

17.2.2.2 QEI x4 Update Mode

QEI x4 Update mode provides for a finer resolution of the rotor position, since the counter increments or decrements more frequently for each QEA/QEB input pulse pair than in QEI x2 mode. This mode is selected by setting the QEI mode select bits to '101' or '110'. In QEI x4, the phase measurement is made on the rising and the falling edges of both QEA and QEB inputs. The position counter is clocked on every QEA and QEB edge.

Like QEI x2 mode, the position counter can be reset by an input on the pin (QEIM<2:0> = 101), or by the period match event (QEIM<2:0> = 010).

17.2.3 QEI OPERATION

The Position Counter register pair (POSCNTH: POSCNTL) acts as an integrator, whose value is proportional to the position of the sensor rotor that corresponds to the number of active edges detected. POSCNT can either increment or decrement, depending on a number of selectable factors which are decoded by the QEI logic block. These include the Count mode selected, the phase relationship of QEA to QEB ("lead/lag"), the direction of rotation and if a Reset event occurs. The logic is detailed in the sections that follow.

17.2.3.1 Edge and Phase Detect

In the first step, the active edges of QEA and QEB are detected, and the phase relationship between them is determined. The position counter is changed based on the selected QEI mode.

In QEI x2 Update mode, the position counter increments or decrements on every QEA edge based on the phase relationship of the QEA and QEB signals.

In QEI x4 Update mode, the position counter increments or decrements on every QEA and QEB edge based on the phase relationship of the QEA and QEB signals. For example, if QEA leads QEB, the position counter is incremented by '1'. If QEB lags QEA, the position counter is decremented by '1'.

17.2.3.2 Direction of Count

The QEI control logic generates a signal that sets the UP/$\overline{\text{DOWN}}$ bit (QEICON<5>); this, in turn, determines the direction of the count. When QEA leads QEB, UP/$\overline{\text{DOWN}}$ is set (= 1) and the position counter increments on every active edge. When QEA lags QEB, UP/$\overline{\text{DOWN}}$ is cleared and the position counter decrements on every active edge.

TABLE 17-5: DIRECTION OF ROTATION

Current Signal Detected	Previous Signal Detected				Pos. Cntrl.[1]
	Rising		Falling		
	QEA	QEB	QEA	QEB	
QEA Rising				x	INC
		x			DEC
QEA Falling				x	DEC
		x			INC
QEB Rising	x				INC
			x		DEC
QEB Falling			x		INC
	x				DEC

Note 1: When UP/$\overline{\text{DOWN}}$ = 1, the position counter is incremented. When UP/$\overline{\text{DOWN}}$ = 0, the position counter is decremented.

18.0 POWER CONTROL PWM MODULE

The Power Control PWM module simplifies the task of generating multiple, synchronized Pulse-Width Modulated (PWM) outputs for use in the control of motor controllers and power conversion applications. In particular, the following power and motion control applications are supported by the PWM module:

- Three-Phase and Single-Phase AC Induction Motors
- Switched Reluctance Motors
- Brushless DC (BLDC) Motors
- Uninterruptible Power Supplies (UPS)
- Multiple DC Brush Motors

The PWM module has the following features:

- Up to eight PWM I/O pins with four duty cycle generators. Pins can be paired to get a complete half-bridge control.
- Up to 14-bit resolution, depending upon the PWM period.
- “On-the-fly” PWM frequency changes.
- Edge and Center-Aligned Output modes.
- Single-Pulse Generation mode.
- Programmable dead-time control between paired PWMs.
- Interrupt support for asymmetrical updates in Center-Aligned mode.
- Output override for Electrically Commutated Motor (ECM) operation; for example, BLDC.
- Special Event Trigger comparator for scheduling other peripheral events.
- PWM outputs disable feature sets PWM outputs to their inactive state when in Debug mode.

The Power Control PWM module supports three PWM generators and six output channels on PIC18F2331/2431 devices, and four generators and eight channels on PIC18F4331/4431 devices. A simplified block diagram of the module is shown in Figure 18-1. Figure 18-2 and Figure 18-3 show how the module hardware is configured for each PWM output pair for the Complementary and Independent Output modes.

Each functional unit of the PWM module will be discussed in subsequent sections.

FIGURE 18-9: PWM PERIOD BUFFER UPDATES IN FREE-RUNNING MODE

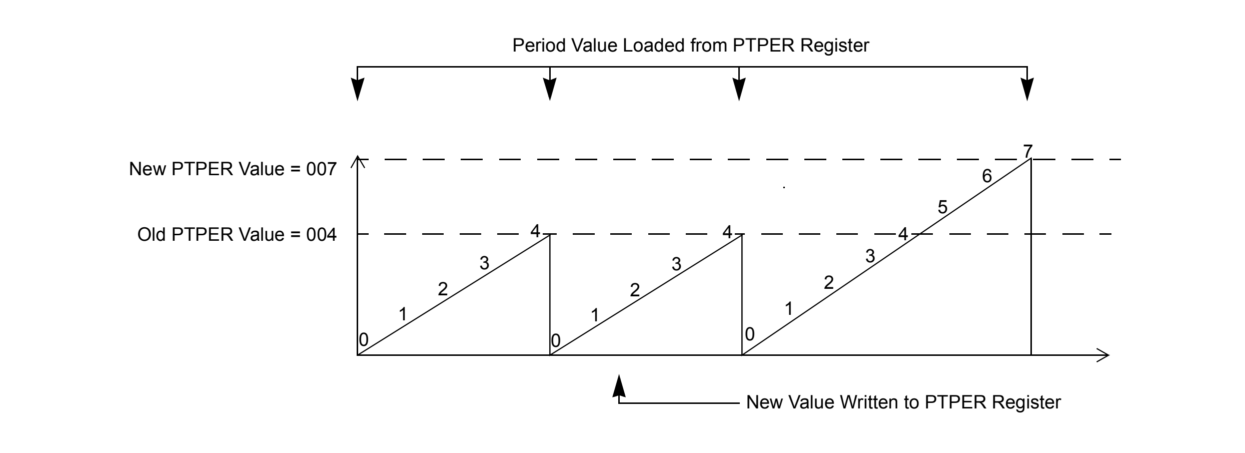

Period Value Loaded from PTPER Register
New PTPER Value = 007
Old PTPER Value = 004
0
1
2
3
4
0
1
2
3
4
0
1
2
3
4
5
6
7
New Value Written to PTPER Register

FIGURE 18-10: PWM PERIOD BUFFER UPDATES IN CONTINUOUS UP/DOWN COUNT MODE

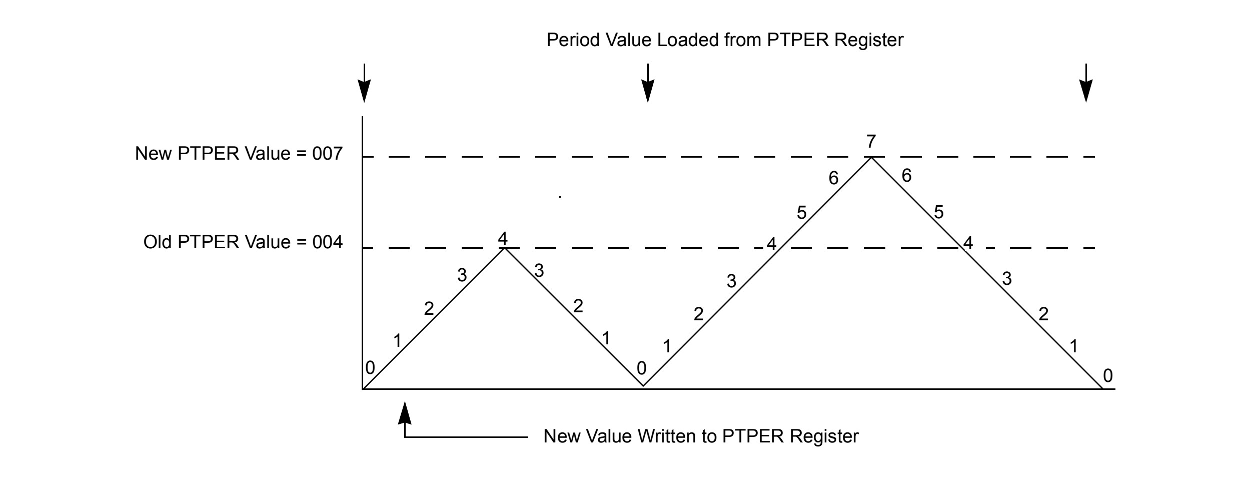

Period Value Loaded from PTPER Register
New PTPER Value = 007
Old PTPER Value = 004
0
1
2
3
4
3
2
1
0
1
2
3
4
5
6
7
6
5
4
3
2
1
0
New Value Written to PTPER Register

FIGURE 18-21: PWM OUTPUT OVERRIDE EXAMPLE #1

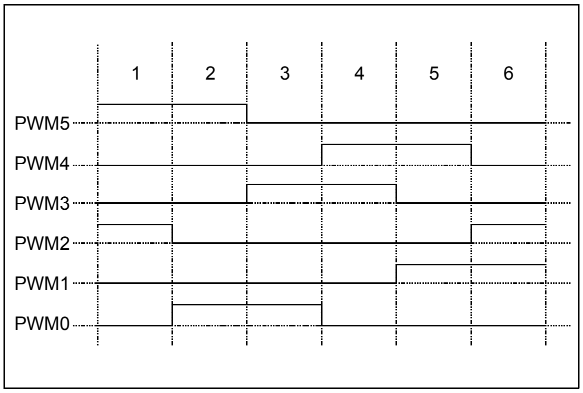


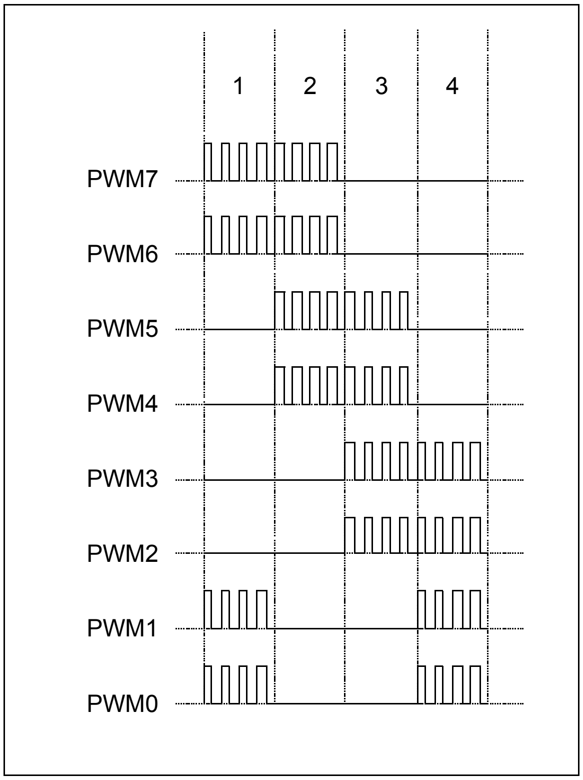
TABLE 18-4: PWM OUTPUT OVERRIDE EXAMPLE #1

State	OVDCOND (POVD)	OVDCONS (POUT)
1	00000000b	00100100b
2	00000000b	00100001b
3	00000000b	00001001b
4	00000000b	00011000b
5	00000000b	00010010b
6	00000000b	00000110b

TABLE 18-5: PWM OUTPUT OVERRIDE EXAMPLE #2

State	OVDCOND (POVD)	OVDCONS (POUT)
1	11000011b	00000000b
2	11110000b	00000000b
3	00111100b	00000000b
4	00001111b	00000000b

FIGURE 18-22: PWM OUTPUT OVERRIDE EXAMPLE #2



19.3.2 MASTER MODE

Master mode of operation is supported in firmware using interrupt generation on the detection of the Start and Stop conditions. The Stop (P) and Start (S) bits are cleared from a Reset or when the SSP module is disabled. The Stop (P) and Start (S) bits will toggle based on the Start and Stop conditions. Control of the I^2C bus may be taken when the P bit is set, or the bus is Idle and both the S and P bits are clear.

In Master mode, the SCL and SDA lines are manipulated by clearing the corresponding TRISC<5:4> or TRISD<3:2> bits. The output level is always low, regardless of the value(s) in PORTC<5:4> or PORTD<3:2>. So when transmitting data, a ‘1’ data bit must have the TRISC<4> bit set (input) and a ‘0’ data bit must have the TRISC<4> bit cleared (output). The same scenario is true for the SCL line with the TRISC<4> or TRISD<2> bit. Pull-up resistors must be provided externally to the SCL and SDA pins for proper operation of the I^2C module.

The following events will cause the SSP Interrupt Flag bit, SSPIF, to be set (SSP interrupt will occur if enabled):

- Start condition
- Stop condition
- Data transfer byte transmitted/received

Master mode of operation can be done with either the Slave mode Idle (SSPM<3:0> = 1011) or with the Slave active. When both Master and Slave modes are enabled, the software needs to differentiate the source(s) of the interrupt.

19.3.3 MULTI-MASTER MODE

In Multi-Master mode, the interrupt generation on the detection of the Start and Stop conditions allows the determination of when the bus is free. The Stop (P) and Start (S) bits are cleared from a Reset or when the SSP module is disabled. The Stop (P) and Start (S) bits will toggle based on the Start and Stop conditions. Control of the I^2C bus may be taken when bit P (SSPSTAT<4>) is set, or the bus is Idle and both the S and P bits clear. When the bus is busy, enabling the SSP interrupt will generate the interrupt when the Stop condition occurs.

In Multi-Master mode, the SDA line must be monitored to see if the signal level is the expected output level. This check only needs to be done when a high level is output. If a high level is expected and a low level is present, the device needs to release the SDA and SCL lines (set TRISC<5:4> or TRISD<3:2>). There are two stages where this arbitration can be lost, these are:

- Address Transfer
- Data Transfer

When the slave logic is enabled, the slave continues to receive. If arbitration was lost during the address transfer stage, communication to the device may be in progress. If addressed, an $\overline{ACK}$ pulse will be generated. If arbitration was lost during the data transfer stage, the device will need to retransfer the data at a later time.

TABLE 19-3: REGISTERS ASSOCIATED WITH I^2C™ OPERATION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	54
PIR1	—	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	57
PIE1	—	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	57
SSPBUF	SSP Receive Buffer/Transmit Register								55
SSPADD	SSP Address Register (I^2C mode)								55
SSPCON	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	55
SSPSTAT	SMP(1)	CKE(1)	$D/\overline{A}$	P	S	$R/\overline{W}$	UA	BF	55
TRISC(2)	PORTC Data Direction Register								57
TRISD(2)	PORTD Data Direction Register								57

Legend: — = unimplemented, read as ‘0’. Shaded cells are not used by the SSP module in I^2C mode.

Note 1: Maintain these bits clear in I^2C mode.

2: Depending upon the setting of SSPMX in CONFIG3H, these pins are multiplexed to PORTC or PORTD.

REGISTER 20-2: RCSTA: RECEIVE STATUS AND CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R-0	R-x
SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
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 **SPEN:** Serial Port Enable bit

1 = Serial port enabled
0 = Serial port disabled

bit 6 **RX9:** 9-Bit Receive Enable bit

1 = Selects 9-bit reception
0 = Selects 8-bit reception

bit 5 **SREN:** Single Receive Enable bit

Asynchronous mode:
Don’t care.

Synchronous mode – Master:
1 = Enables single receive
0 = Disables single receive
This bit is cleared after reception is complete.

Synchronous mode – Slave:
Don’t care.

bit 4 **CREN:** Continuous Receive Enable bit

Asynchronous mode:
1 = Enables receiver
0 = Disables receiver

Synchronous mode:
1 = Enables continuous receive until enable bit, CREN, is cleared (CREN overrides SREN)
0 = Disables continuous receive

bit 3 **ADDEN:** Address Detect Enable bit

Asynchronous mode 9-Bit (RX9 = 1):
1 = Enables address detection, enables interrupt and loads the receive buffer when RSR<8> is set
0 = Disables address detection, all bytes are received and ninth bit can be used as parity bit

Asynchronous mode 8-Bit (RX9 = 0):
Don’t care.

bit 2 **FERR:** Framing Error bit

1 = Framing error (can be cleared by reading RCREGx register and receiving next valid byte)
0 = No framing error

bit 1 **OERR:** Overrun Error bit

1 = Overrun error (can be cleared by clearing bit, CREN)
0 = No overrun error

bit 0 **RX9D:** 9th Bit of Received Data

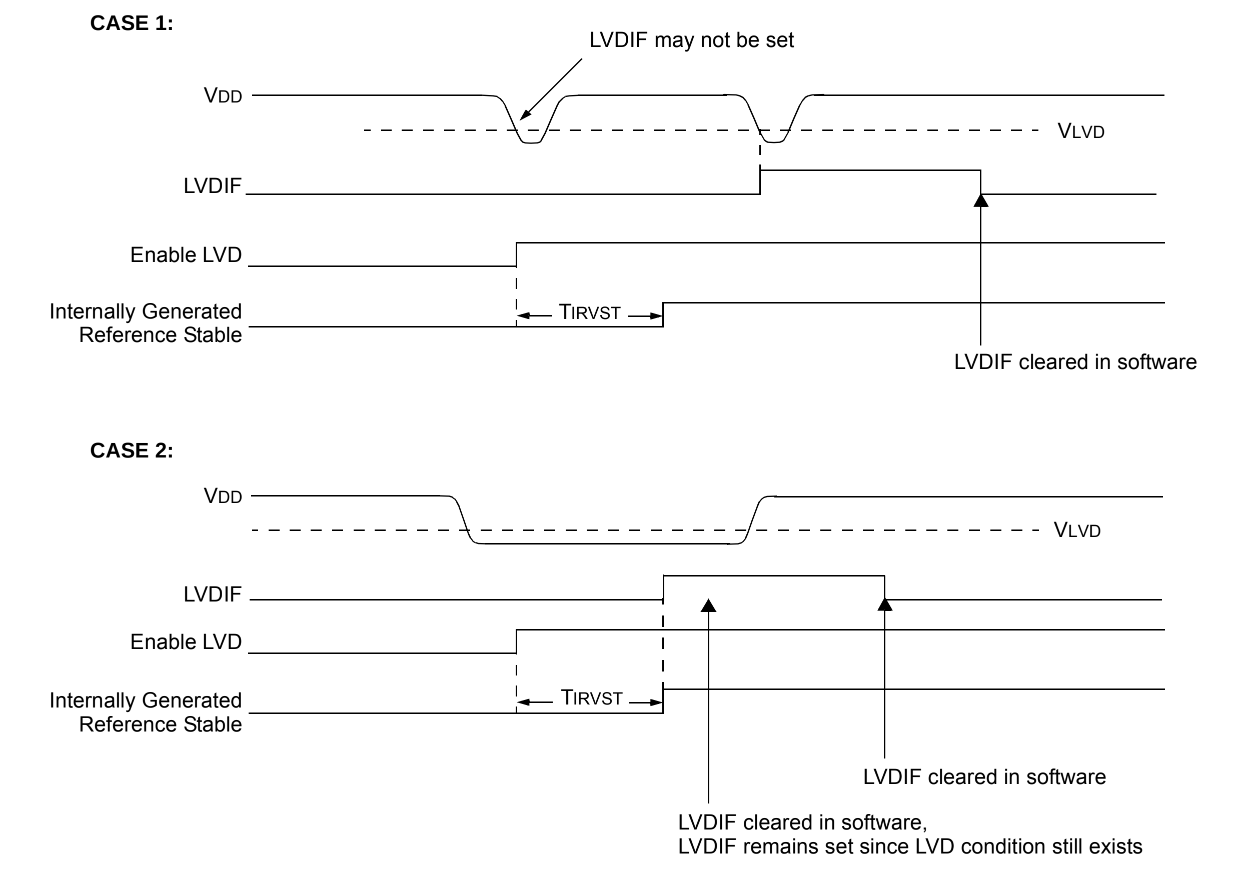
This can be address/data bit or a parity bit and must be calculated by user firmware.

22.4 LVD Start-up Time

The internal reference voltage of the LVD module, specified in electrical specification Parameter D420, may be used by other internal circuitry, such as the Programmable Brown-out Reset. If the LVD, or other circuits using the voltage reference, are disabled to lower the device's current consumption, the reference voltage circuit will require time to become stable before a low-voltage condition can be reliably detected. This start-up time, T_{IRVST}, is an interval that is independent of device clock speed. It is specified in electrical specification Parameter 36.

The LVD interrupt flag is not enabled until T_{IRVST} has expired and a stable reference voltage is reached. For this reason, brief excursions beyond the set point may not be detected during this interval (refer to Figure 22-2).

FIGURE 22-2: LOW-VOLTAGE DETECT WAVEFORMS


CASE 1:
LVDIF may not be set
VDD
VLVD
LVDIF
Enable LVD
Internally Generated Reference Stable
TIRVST
LVDIF cleared in software
CASE 2:
VDD
VLVD
LVDIF
Enable LVD
Internally Generated Reference Stable
TIRVST
LVDIF cleared in software
LVDIF cleared in software,
LVDIF remains set since LVD condition still exists

REGISTER 23-6: CONFIG4L: CONFIGURATION REGISTER 4 LOW (BYTE ADDRESS 300006h)

R/P-1	U-0	U-0	U-0	U-0	R/P-1	U-0	R/P-1
$\overline{\text{DEBUG}}$	—	—	—	—	LVP	—	STVREN
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 **$\overline{\textbf{DEBUG}}$:** Background Debugger Enable bit

`1` = Background debugger is disabled; RB6 and RB7 are configured as general purpose I/O pins
`0` = Background debugger is enabled; RB6 and RB7 are dedicated to In-Circuit Debug

bit 6-3 **Unimplemented:** Read as ‘`0`’

bit 2 **LVP:** Single-Supply ICSP™ Enable bit

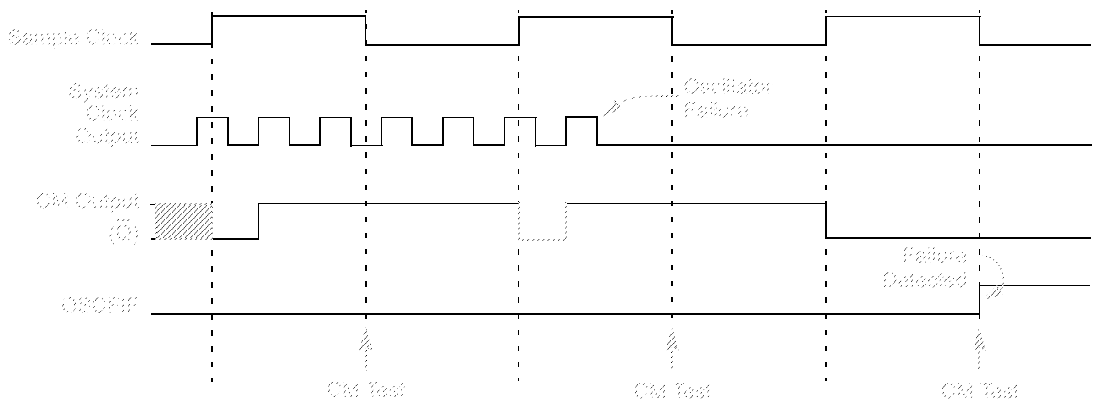
`1` = Single-Supply ICSP is enabled
`0` = Single-Supply ICSP is disabled

bit 1 **Unimplemented:** Read as ‘`0`’

bit 0 **STVREN:** Stack Full/Underflow Reset Enable bit

`1` = Stack full/underflow will cause Reset
`0` = Stack full/underflow will not cause Reset

FIGURE 23-4: FSCM TIMING DIAGRAM



Note: The system clock is normally at a much higher frequency than the sample clock. The relative frequencies in this example have been chosen for clarity.

23.4.3 FSCM INTERRUPTS IN POWER-MANAGED MODES

As previously mentioned, entering a power-managed mode clears the fail-safe condition. By entering a power-managed mode, the clock multiplexer selects the clock source selected by the OSCCON register. Fail-safe monitoring of the power-managed clock source resumes in the power-managed mode.

If an oscillator failure occurs during power-managed operation, the subsequent events depend on whether or not the oscillator failure interrupt is enabled. If enabled (OSCFIF = 1), code execution will be clocked by the INTOSC multiplexer. An automatic transition back to the failed clock source will not occur.

If the interrupt is disabled, the device will not exit the power-managed mode on oscillator failure. Instead, the device will continue to operate as before, but clocked by the INTOSC multiplexer. While in Idle mode, subsequent interrupts will cause the CPU to begin executing instructions while being clocked by the INTOSC multiplexer. The device will not transition to a different clock source until the fail-safe condition is cleared.

23.4.4 POR OR WAKE FROM SLEEP

The FSCM is designed to detect oscillator failure at any point after the device has exited Power-on Reset (POR) or low-power Sleep mode. When the primary system clock is EC, RC or INTRC modes, monitoring can begin immediately following these events.

For oscillator modes involving a crystal or resonator (HS, HSPLL, LP or XT), the situation is somewhat different. Since the oscillator may require a start-up time considerably longer than the FCSM sample clock time, a false clock failure may be detected. To prevent this, the internal oscillator block is automatically configured as the system clock and functions until the primary clock is stable (the OST and PLL timers have timed out). This is identical to Two-Speed Start-up mode. Once the primary clock is stable, the INTRC returns to its role as the FSCM source.

Note:	The same logic that prevents false oscillator failure interrupts on POR or wake from Sleep will also prevent the detection of the oscillator's failure to start at all following these events. This can be avoided by monitoring the OSTS bit and using a timing routine to determine if the oscillator is taking too long to start. Even so, no oscillator failure interrupt will be flagged.

As noted in **Section 23.3.1 "Special Considerations for Using Two-Speed Start-up"**, it is also possible to select another clock configuration, and enter an alternate power-managed mode, while waiting for the primary system clock to become stable. When the new powered-managed mode is selected, the primary clock is disabled.

TABLE 24-2: PIC18FXXXX INSTRUCTION SET (CONTINUED)

Mnemonic, Operands		Description	Cycles	16-Bit Instruction Word				Status Affected	Notes
				MSb			LSb		
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	Load Literal (12-bit) 2nd word	2	1110	1110	00ff	kkkk	None	
		to FSRx 1st word		1111	0000	kkkk	kkkk		
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 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.

RCALL Relative Call

Syntax:	[*label*] RCALL n
Operands:	-1024 ≤ n ≤ 1023
Operation:	(PC) + 2 → TOS, (PC) + 2 + 2n → PC
Status Affected:	None
Encoding:	`1101` `1nnn` `nnnn` `nnnn`
Description:	Subroutine call with a jump up to 1K from the current location. First, return address (PC + 2) is pushed onto the stack. Then, add the 2's complement number '2n' 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 a two-cycle instruction.
Words:	1
Cycles:	2

Q Cycle Activity:

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

Example: `HERE     RCALL Jump`

Before Instruction
PC = Address (`HERE`)

After Instruction
PC = Address (`Jump`)
TOS = Address (`HERE + 2`)

RESET Reset

Syntax:	[*label*] RESET
Operands:	None
Operation:	Reset all registers and flags that are affected by a $\overline{\text{MCLR}}$ Reset.
Status Affected:	All
Encoding:	`0000` `0000` `1111` `1111`
Description:	This instruction provides a way to execute a $\overline{\text{MCLR}}$ Reset in software.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Start Reset	No operation	No operation

Example: `RESET`

After Instruction
Registers = Reset Value
Flags* = Reset Value

TABLE 26-2: LOW-VOLTAGE DETECT CHARACTERISTICS (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.	**Symbol**	**Characteristic**		**Min**	**Typ†**	**Max**	**Units**	**Conditions**
D420B	VLVD	LVD Voltage on VDD Transition High-to-Low		Industrial Low Voltage (-40°C to -10°C)				
		PIC18LF2X31/4X31	LVDL<3:0> = `0000`	N/A	N/A	N/A	V	Reserved
			LVDL<3:0> = `0001`	N/A	N/A	N/A	V	Reserved
			LVDL<3:0> = `0010`	1.99	2.26	2.53	V	
			LVDL<3:0> = `0011`	2.16	2.45	2.75	V	
			LVDL<3:0> = `0100`	2.25	2.55	2.86	V	
			LVDL<3:0> = `0101`	2.43	2.77	3.10	V	
			LVDL<3:0> = `0110`	2.53	2.87	3.21	V	
			LVDL<3:0> = `0111`	2.70	3.07	3.43	V	
			LVDL<3:0> = `1000`	2.96	3.36	3.77	V	
			LVDL<3:0> = `1001`	3.14	3.57	4.00	V	
			LVDL<3:0> = `1010`	3.23	3.67	4.11	V	
			LVDL<3:0> = `1011`	3.41	3.87	4.34	V	
			LVDL<3:0> = `1100`	3.58	4.07	4.56	V	
			LVDL<3:0> = `1101`	3.76	4.28	4.79	V	
			LVDL<3:0> = `1110`	4.04	4.60	5.15	V	
D420C	VLVD	LVD Voltage on VDD Transition High-to-Low		Industrial (-10°C to +85°C)				
		PIC18F2X31/4X31	LVDL<3:0> = `1101`	3.93	4.28	4.62	V	
			LVDL<3:0> = `1110`	4.23	4.60	4.96	V	
D420D	VLVD	LVD Voltage on VDD Transition High-to-Low		Industrial (-40°C to -10°C)				
		PIC18F2X31/4X31	LVDL<3:0> = `1101`	3.76	4.28	4.79	V	Reserved
			LVDL<3:0> = `1110`	4.04	4.60	5.15	V	
D420E	VLVD	LVD Voltage on VDD Transition High-to-Low		Extended (-10°C to +85°C)				
		PIC18F2X31/4X31	LVDL<3:0> = `1101`	3.94	4.28	4.62	V	
			LVDL<3:0> = `1110`	4.23	4.60	4.96	V	
D420F	VLVD	LVD Voltage on VDD Transition High-to-Low		Extended (-40°C to -10°C, +85°C to +125°C)				
		PIC18F2X31/4X31	LVDL<3:0> = `1101`	3.77	4.28	4.79	V	Reserved
			LVDL<3:0> = `1110`	4.05	4.60	5.15	V	

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

† Production tested at TAMB = 25°C. Specifications over temperature limits ensured by characterization.