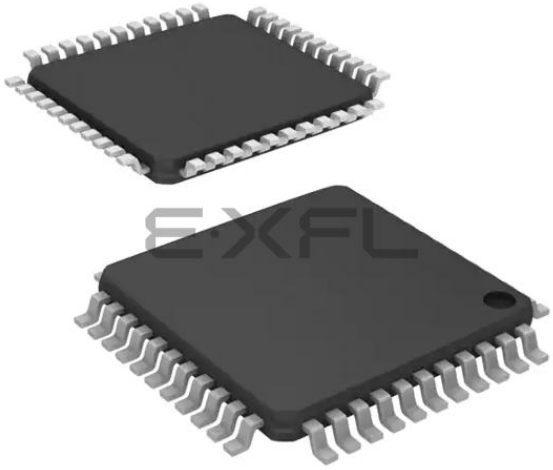


Welcome to E-XFL.COM

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"

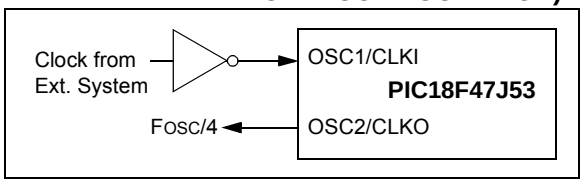
Details	
Product Status	Active
Core Processor	PIC
Core Size	8-Bit
Speed	48MHz
Connectivity	I²C, LINbus, SPI, UART/USART, USB
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	34
Program Memory Size	128KB (64K x 16)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	3.8K x 8
Voltage - Supply (Vcc/Vdd)	2V ~ 2.75V
Data Converters	A/D 13x10b/12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	44-TQFP
Supplier Device Package	44-TQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18lf47j53t-i-pt

3.2.3 EXTERNAL CLOCK INPUT

The EC and ECPLL Oscillator modes require an external clock source to be connected to the OSC1 pin. There is no oscillator start-up time required after a Power-on Reset (POR) or after an exit from Sleep mode.

In the EC Oscillator mode, the oscillator frequency divided by 4, is available on the OSC2 pin. In the ECPLL Oscillator mode, the PLL output, divided by 4, is available on the OSC2 pin. This signal may be used for test purposes or to synchronize other logic. Figure 3-3 displays the pin connections for the EC Oscillator mode.

FIGURE 3-3: EXTERNAL CLOCK INPUT OPERATION (EC AND ECPLL CONFIGURATION)



3.2.4 PLL FREQUENCY MULTIPLIER

PIC18F47J53 family devices include a PLL circuit. This is provided specifically for USB applications with lower speed oscillators and can also be used as a microcontroller clock source.

The PLL can be enabled in HSPLL, ECPLL, INTOSCPLL and INTOSCPLLO Oscillator modes by setting the PLLEN bit (OSCTUNE<6>). It is designed to produce a fixed 96 MHz reference clock from a fixed 4 MHz input. The output can then be divided and used for both the USB and the microcontroller core clock. Because the PLL has a fixed frequency input and output, there are eight prescaling options to match the oscillator input frequency to the PLL. This prescaler allows the PLL to be used with crystals, resonators and external clocks, which are integer multiple frequencies of 4 MHz. For example, a 12 MHz crystal could be used in a prescaler Divide-by-Three mode to drive the PLL.

There is also a CPU divider, which can be used to derive the microcontroller clock from the PLL. This allows the USB peripheral and microcontroller to use the same oscillator input and still operate at different clock speeds. The CPU divider can reduce the incoming frequency by a factor of 1, 2, 3 or 6.

3.2.5 INTERNAL OSCILLATOR BLOCK

The PIC18F47J53 family devices include an internal oscillator block which generates two different clock signals; either can be used as the microcontroller's clock source. The internal oscillator may eliminate the need for external oscillator circuits on the OSC1 and/or OSC2 pins.

The main output (INTOSC) is an 8 MHz clock source which can be used to directly drive the device clock. It also drives the INTOSC postscaler which can provide a range of clock frequencies from 31 kHz to 8 MHz. Additionally, the INTOSC may be used in conjunction with the PLL to generate clock frequencies up to 48 MHz.

The other clock source is the internal RC oscillator (INTRC) which provides a nominal 31 kHz output. INTRC is enabled if it is selected as the device clock source. It is also enabled automatically when any of the following are enabled:

- Power-up Timer
- Fail-Safe Clock Monitor
- Watchdog Timer
- Two-Speed Start-up

These features are discussed in larger detail in **Section 28.0 "Special Features of the CPU"**.

The clock source frequency (INTOSC direct, INTRC direct or INTOSC postscaler) is selected by configuring the IRCF bits of the OSCCON register (page 39).

3.6 Reference Clock Output

In addition to the peripheral clock/4 output in certain oscillator modes, the device clock in the PIC18F47J53 family can also be configured to provide a reference clock output signal to a port pin. This feature is available in all oscillator configurations and allows the user to select a greater range of clock submultiples to drive external devices in the application.

This reference clock output is controlled by the REFOCON register (Register 3-4). Setting the ROON bit (REFOCON<7>) makes the clock signal available on the REFO (RB2) pin. The RODIV<3:0> bits enable the selection of 16 different clock divider options.

The ROSSLP and ROSEL bits (REFOCON<5:4>) control the availability of the reference output during Sleep mode. The ROSEL bit determines if the oscillator is on OSC1 and OSC2, or the current system clock source is used for the reference clock output. The ROSSLP bit determines if the reference source is available on RB2 when the device is in Sleep mode.

To use the reference clock output in Sleep mode, both the ROSSLP and ROSEL bits must be set. The device clock must also be configured for an EC or HS mode; otherwise, the oscillator on OSC1 and OSC2 will be powered down when the device enters Sleep mode. Clearing the ROSEL bit allows the reference output frequency to change as the system clock changes during any clock switches.

REGISTER 3-4: REFOCON: REFERENCE OSCILLATOR CONTROL REGISTER (BANKED F3Dh)

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ROON	—	ROSSLP	ROSEL	RODIV3	RODIV2	RODIV1	RODIV0
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 **ROON:** Reference Oscillator Output Enable bit
1 = Reference oscillator is enabled on REFO pin
0 = Reference oscillator is disabled

bit 6 **Unimplemented:** Read as ‘0’

bit 5 **ROSSLP:** Reference Oscillator Output Stop in Sleep bit
1 = Reference oscillator continues to run in Sleep
0 = Reference oscillator is disabled in Sleep

bit 4 **ROSEL:** Reference Oscillator Source Select bit
1 = Primary oscillator crystal/resonator is used as the base clock(1)
0 = System clock (FOSC) is used as the base clock; the base clock reflects any clock switching of the device

bit 3-0 **RODIV<3:0>:** Reference Oscillator Divisor Select bits
1111 = Base clock value divided by 32,768
1110 = Base clock value divided by 16,384
1101 = Base clock value divided by 8,192
1100 = Base clock value divided by 4,096
1011 = Base clock value divided by 2,048
1010 = Base clock value divided by 1,024
1001 = Base clock value divided by 512
1000 = Base clock value divided by 256
0111 = Base clock value divided by 128
0110 = Base clock value divided by 64
0101 = Base clock value divided by 32
0100 = Base clock value divided by 16
0011 = Base clock value divided by 8
0010 = Base clock value divided by 4
0001 = Base clock value divided by 2
0000 = Base clock value

Note 1: The crystal oscillator must be enabled using the FOSC<2:0> bits; the crystal maintains the operation in Sleep mode.

5.5 Configuration Mismatch (CM)

The Configuration Mismatch (CM) Reset is designed to detect, and attempt to recover from, random memory corrupting events. These include Electrostatic Discharge (ESD) events, which can cause widespread single bit changes throughout the device and result in catastrophic failure.

In PIC18FXXJ Flash devices, the device Configuration registers (located in the configuration memory space) are continuously monitored during operation by comparing their values to complimentary shadow registers. If a mismatch is detected between the two sets of registers, a CM Reset automatically occurs. These events are captured by the $\overline{CM}$ bit (RCON<5>). The state of the bit is set to '0' whenever a CM event occurs; it does not change for any other Reset event.

A CM Reset behaves similarly to a $\overline{MCLR}$, RESET instruction, WDT time-out or Stack Event Resets. As with all hard and power Reset events, the device Configuration Words are reloaded from the Flash Configuration Words in program memory as the device restarts.

5.6 Power-up Timer (PWRT)

PIC18F47J53 family devices incorporate an on-chip PWRT to help regulate the POR process. The PWRT is always enabled. The main function is to ensure that the device voltage is stable before code is executed.

The Power-up Timer (PWRT) of the PIC18F47J53 family devices is a 5-bit counter which uses the INTRC source as the clock input. This yields an approximate time interval of 32 x 32 μs = 1 ms. While the PWRT is counting, the device is held in Reset.

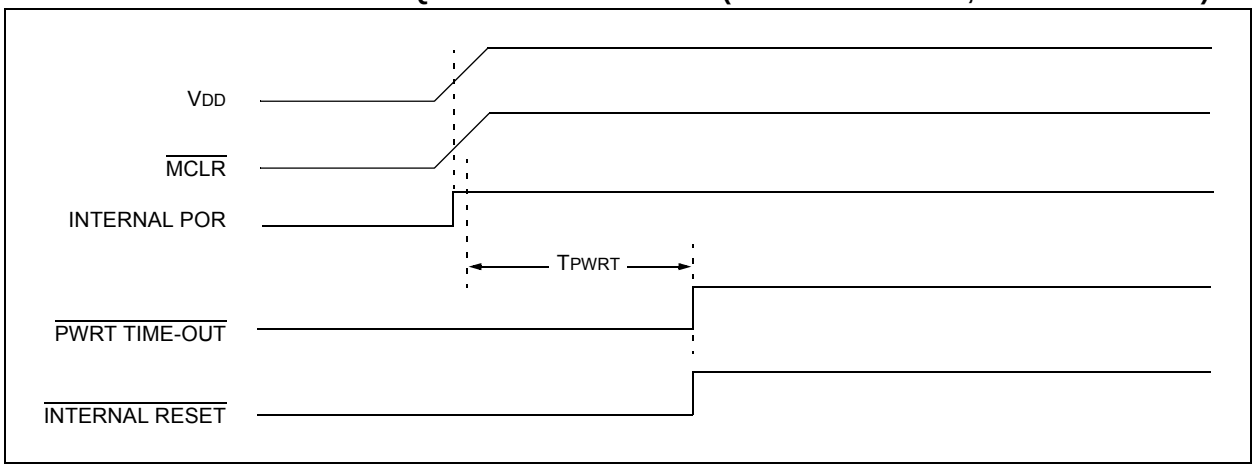
The power-up time delay depends on the INTRC clock and will vary from chip-to-chip due to temperature and process variation. See DC parameter 33 (TPWRT) for details.

5.6.1 TIME-OUT SEQUENCE

The PWRT time-out is invoked after the POR pulse has cleared. The total time-out will vary based on the status of the PWRT. Figure 5-2, Figure 5-3, Figure 5-4 and Figure 5-5 all depict time-out sequences on power-up with the PWRT.

Since the time-outs occur from the POR pulse, if $\overline{MCLR}$ is kept low long enough, the PWRT will expire. Bringing $\overline{MCLR}$ high will begin execution immediately if a clock source is available (Figure 5-4). This is useful for testing purposes or to synchronize more than one PIC18F device operating in parallel.

FIGURE 5-2: TIME-OUT SEQUENCE ON POWER-UP ($\overline{MCLR}$ TIED TO VDD, VDD RISE < TPWRT)



6.4.3.2 FSR Registers and POSTINC, POSTDEC, PREINC and PLUSW

In addition to the INDF operand, each FSR register pair also has four additional indirect operands. Like INDF, these are “virtual” registers that cannot be indirectly read or written to. Accessing these registers actually accesses the associated FSR register pair, but also performs a specific action on its stored value. They are:

- POSTDEC: accesses the FSR value, then automatically decrements it by ‘1’ thereafter
- POSTINC: accesses the FSR value, then automatically increments it by ‘1’ thereafter
- PREINC: increments the FSR value by ‘1’, then uses it in the operation
- PLUSW: adds the signed value of the W register (range of 127 to 128) to that of the FSR and uses the new value in the operation

In this context, accessing an INDF register uses the value in the FSR registers without changing them. Similarly, accessing a PLUSW register gives the FSR value offset by the value in the W register; neither value is actually changed in the operation. Accessing the other virtual registers changes the value of the FSR registers.

Operations on the FSRs with POSTDEC, POSTINC and PREINC affect the entire register pair; that is, roll-overs of the FSRnL register from FFh to 00h carry over to the FSRnH register. On the other hand, results of these operations do not change the value of any flags in the STATUS register (e.g., Z, N, OV, etc.).

The PLUSW register can be used to implement a form of Indexed Addressing in the data memory space. By manipulating the value in the W register, users can reach addresses that are fixed offsets from pointer addresses. In some applications, this can be used to implement some powerful program control structure, such as software stacks, inside of data memory.

6.4.3.3 Operations by FSRs on FSRs

Indirect Addressing operations that target other FSRs or virtual registers represent special cases. For example, using an FSR to point to one of the virtual registers will not result in successful operations. As a specific case, assume that FSR0H:FSR0L contains FE7h, the address of INDF1. Attempts to read the value of the INDF1, using INDF0 as an operand, will return 00h. Attempts to write to INDF1, using INDF0 as the operand, will result in a NOP.

On the other hand, using the virtual registers to write to an FSR pair may not occur as planned. In these cases, the value will be written to the FSR pair but without any incrementing or decrementing. Thus, writing to INDF2 or POSTDEC2 will write the same value to the FSR2H:FSR2L.

Since the FSRs are physical registers mapped in the SFR space, they can be manipulated through all direct operations. Users should proceed cautiously when working on these registers, particularly if their code uses Indirect Addressing.

Similarly, operations by Indirect Addressing are generally permitted on all other SFRs. Users should exercise appropriate caution that they do not inadvertently change settings that might affect the operation of the device.

6.5 Program Memory and the Extended Instruction Set

The operation of program memory is unaffected by the use of the extended instruction set.

Enabling the extended instruction set adds five additional two-word commands to the existing PIC18 instruction set: ADDFSR, CALLW, MOVSF, MOVSS and SUBFSR. These instructions are executed as described in **Section 6.2.4 “Two-Word Instructions”**.

10.4 PORTC, TRISC and LATC Registers

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

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

PORTC is multiplexed with several peripheral functions (see Table). The pins have Schmitt Trigger input buffers.

When enabling peripheral functions, care should be taken in defining TRIS bits for each PORTC pin. Some peripherals override the TRIS bit to make a pin an output, while other peripherals override the TRIS bit to make a pin an input. The user should refer to the corresponding peripheral section for additional information.

Pins RC4 and RC5 are multiplexed with the USB module. Depending on the configuration of the module, they can serve as the differential data lines for the on-chip USB transceiver, or the data inputs from an external USB transceiver. When used as general purpose inputs, both RC4 and RC5 input buffers depend on the level of the voltage applied to the V_{USB} pin, instead of V_{DD} like all other general purpose I/O pins. Therefore, if the RC4 or RC5 general purpose input capability will be used, the V_{USB} pin should not be left floating.

Unlike other PORTC pins, RC4 and RC5 do not have TRISC bits associated with them. As digital ports, they can only function as digital inputs. When configured for USB operation, the data direction is determined by the configuration and status of the USB module at a given time. If an external transceiver is used, RC4 and RC5 always function as inputs from the transceiver. If the on-chip transceiver is used, the data direction is determined by the operation being performed by the module at that time.

Note: On a Power-on Reset, PORTC pins (except RC2, RC4 and RC5) are configured as digital inputs. RC2 will default as an analog input (controlled by the ANCON1 register). To use pins RC4 and RC5 as digital inputs, the USB module must be disabled (UCON<3> = 0) and the on-chip USB transceiver must be disabled (UCFG<3> = 1). The internal USB transceiver has a POR value of enabled.

The contents of the TRISC register are affected by peripheral overrides. Reading TRISC always returns the current contents, even though a peripheral device may be overriding one or more of the pins.

EXAMPLE 10-4: INITIALIZING PORTC

```
CLRF   PORTC    ;  Initialize PORTC by
                ; clearing output
                ; data latches
CLRF   LATC     ; Alternate method
                ; to clear output
                ; data latches
MOVLW  0x3F     ; Value used to
                ; initialize data
                ; direction
MOVWF  TRISC    ; Set RC<5:0> as inputs
                ;  RC<7:6> as outputs
MOVLB  0x0F     ; ANCON register is not in
                  Access Bank
BSF    ANCON1,PCFG11
               ;Configure RC2/AN11 as
                digital input
```

13.0 TIMER1 MODULE

The Timer1 timer/counter module incorporates these features:

- Software selectable operation as a 16-bit timer or counter
- Readable and writable 8-bit registers (TMR1H and TMR1L)
- Selectable clock source (internal or external) with device clock or Timer1 oscillator internal options
- Interrupt-on-overflow
- Reset on ECCP Special Event Trigger
- Device clock status flag (SOSCRUN)
- Timer with gated control

Figure 13-1 displays a simplified block diagram of the Timer1 module.

The module incorporates its own low-power oscillator to provide an additional clocking option. The Timer1 oscillator can also be used as a low-power clock source for the microcontroller in power-managed operation.

Timer1 is controlled through the T1CON Control register (Register 13-1). It also contains the Timer1 Oscillator Enable bit (T1OSCEN). Timer1 can be enabled or disabled by setting or clearing control bit, TMR1ON (T1CON<0>).

The FOSC clock source (TMR1CS<1:0> = 01) should not be used with the ECCP capture/compare features. If the timer will be used with the capture or compare features, always select one of the other timer clocking options.

REGISTER 13-1: T1CON: TIMER1 CONTROL REGISTER (ACCESS FCDh)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
TMR1CS1	TMR1CS0	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	RD16	TMR1ON
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 7-6 **TMR1CS<1:0>:** Timer1 Clock Source Select bits
10 = Timer1 clock source is the T1OSC or T1CKI pin
01 = Timer1 clock source is the system clock (FOSC)(1)
00 = Timer1 clock source is the instruction clock (FOSC/4)

bit 5-4 **T1CKPS<1:0>:** Timer1 Input Clock Prescale Select bits
11 = 1:8 Prescale value
10 = 1:4 Prescale value
01 = 1:2 Prescale value
00 = 1:1 Prescale value

bit 3 **T1OSCEN:** Timer1 Oscillator Source Select bit
When TMR1CS<1:0> = 10:
1 = Power up the Timer1 crystal driver and supply the Timer1 clock from the crystal output
0 = Timer1 crystal driver is off, Timer1 clock is from the T1CKI input pin(2)
When TMR1CS<1:0> = 0x:
1 = Power up the Timer1 crystal driver
0 = Timer1 crystal driver is off(2)

bit 2 **T1SYNC:** Timer1 External Clock Input Synchronization Select bit
TMR1CS<1:0> = 10:
1 = Do not synchronize external clock input
0 = Synchronize external clock input
TMR1CS<1:0> = 0x:
This bit is ignored. Timer1 uses the internal clock when TMR1CS<1:0> = 0x.

bit 1 **RD16:** 16-Bit Read/Write Mode Enable bit
1 = Enables register read/write of Timer1 in one 16-bit operation
0 = Enables register read/write of Timer1 in two 8-bit operations

bit 0 **TMR1ON:** Timer1 On bit
1 = Enables Timer1
0 = Stops Timer1

Note 1: The FOSC clock source should not be selected if the timer will be used with the ECCP capture/compare features.
2: The Timer1 oscillator crystal driver is powered whenever T1OSCEN (T1CON<3>) or T3OSCEN (T3CON<3>) = 1. The circuit is enabled by the logical OR of these two bits. When disabled, the inverter and feedback resistor are disabled to eliminate power drain. The TMR1ON and TMR3ON bits do not have to be enabled to power up the crystal driver.

19.2.2 TIMER1/TIMER3 MODE SELECTION

The timers that are to be used with the capture feature (Timer1 and/or Timer3) must be running in Timer mode or Synchronized Counter mode. In Asynchronous Counter mode, the capture operation may not work. The timer to be used with each ECCP module is selected in the CCPTMRS0 register (Register 19-2).

19.2.3 SOFTWARE INTERRUPT

When the Capture mode is changed, a false capture interrupt may be generated. The user should keep the CCPxIE interrupt enable bit clear to avoid false interrupts. The interrupt flag bit, CCPxIF, should also be cleared following any such change in operating mode.

19.2.4 ECCP PRESCALER

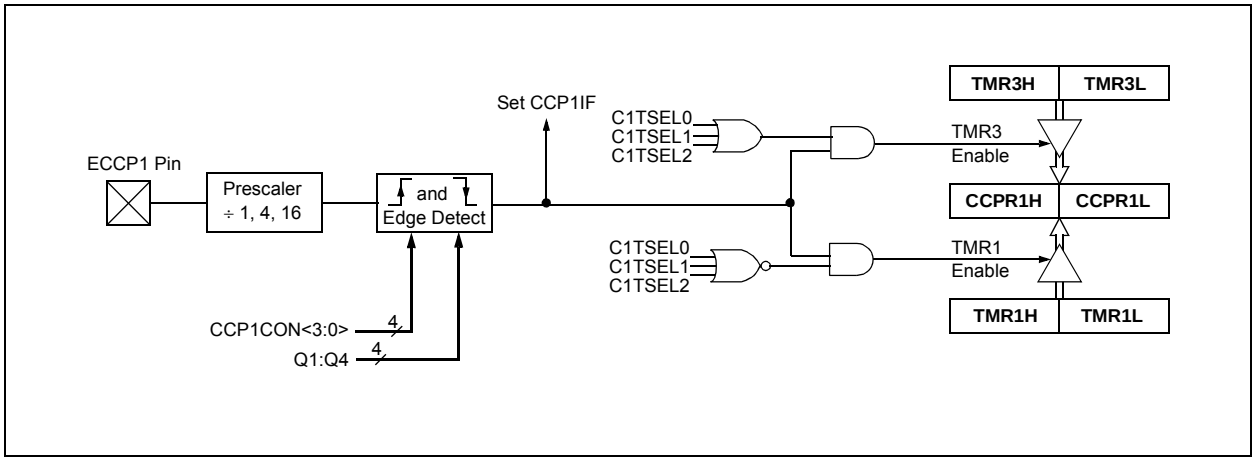
There are four prescaler settings in Capture mode; they are specified as part of the operating mode selected by the mode select bits (CCPxM<3:0>). Whenever the ECCP module is turned off, or Capture mode is disabled, the prescaler counter is cleared. This means that any Reset will clear the prescaler 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 19-1 provides the recommended method for switching between capture prescalers. This example also clears the prescaler counter and will not generate the “false” interrupt.

EXAMPLE 19-1: CHANGING BETWEEN CAPTURE PRESCALERS

```
CLRF   ECCP1CON    ; Turn ECCP module off
MOVLW  NEW_CAPT_PS ; Load WREG with the
                   ; new prescaler mode
                   ; value and ECCP ON
MOVWF  CCP1CON     ; Load ECCP1CON with
                   ; this value
```

FIGURE 19-1: CAPTURE MODE OPERATION BLOCK DIAGRAM

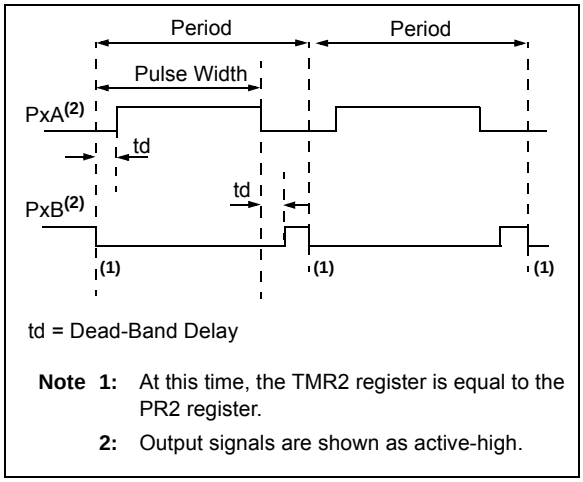


19.4.6 PROGRAMMABLE DEAD-BAND DELAY MODE

In half-bridge applications, where all power switches are modulated at the PWM frequency, the power switches normally require more time to turn off than to turn on. If both the upper and lower power switches are switched at the same time (one turned on and the other turned off), both switches may be on for a short period until one switch completely turns off. During this brief interval, a very high current (shoot-through current) will flow through both power switches, shorting the bridge supply. To avoid this potentially destructive shoot-through current from flowing during switching, turning on either of the power switches is normally delayed to allow the other switch to completely turn off.

In Half-Bridge mode, a digitally programmable, dead-band delay is available to avoid shoot-through current from destroying the bridge power switches. The delay occurs at the signal transition from the non-active state to the active state. For an illustration, see Figure 19-14. The lower seven bits of the associated ECCPxDEL register (Register 19-5) sets the delay period in terms of microcontroller instruction cycles (T_{CY} or 4 T_{OSC}).

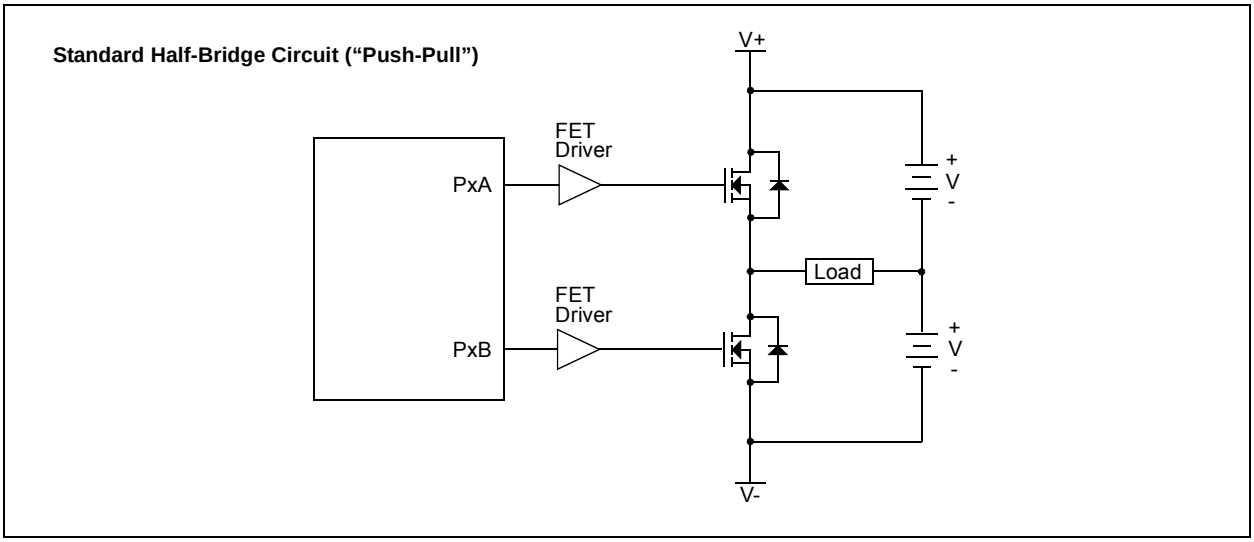
FIGURE 19-14: EXAMPLE OF HALF-BRIDGE PWM OUTPUT



Note 1: At this time, the TMR2 register is equal to the PR2 register.

2: Output signals are shown as active-high.

FIGURE 19-15: EXAMPLE OF HALF-BRIDGE APPLICATIONS



REGISTER 19-5: ECCPxDEL: ECCP1/2/3 ENHANCED PWM CONTROL REGISTER (1, ACCESS FBDh; 2, FB7h; 3, BANKED F18h)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PxRSEN	PxDC6	PxDC5	PxDC4	PxDC3	PxDC2	PxDC1	PxDC0
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 **PxRSEN:** PWM Restart Enable bit
- 1 = Upon auto-shutdown, the ECCPxASE bit clears automatically once the shutdown event goes away; the PWM restarts automatically
- 0 = Upon auto-shutdown, ECCPxASE must be cleared by software to restart the PWM

bit 6-0 **PxDC<6:0>:** PWM Delay Count bits

PxDCn = Number of FOSC/4 (4 * TOSC) cycles between the scheduled time when a PWM signal **should** transition active and the **actual** time it transitions active.

19.4.7 PULSE STEERING MODE

In Single Output mode, pulse steering allows any of the PWM pins to be the modulated signal. Additionally, the same PWM signal can simultaneously be available on multiple pins.

Once the Single Output mode is selected (CCPxM<3:2> = 11 and PxM<1:0> = 00 of the CCPxCON register), the user firmware can bring out the same PWM signal to one, two, three or four output pins by setting the appropriate STR<D:A> bits (PSTRxCON<3:0>), as provided in Table 19-3.

Note:	The associated TRIS bits must be set to output ('0'), to enable the pin output driver, in order to see the PWM signal on the pin.

While the PWM Steering mode is active, the CCPxM<1:0> bits (CCPxCON<1:0>) select the PWM output polarity for the Px<D:A> pins.

The PWM auto-shutdown operation also applies to PWM Steering mode, as described in **Section 19.4.4 "Enhanced PWM Auto-shutdown mode"**. An auto-shutdown event will only affect pins that have PWM outputs enabled.

20.3.7 SLAVE MODE

In Slave mode, the data is transmitted and received as the external clock pulses appear on SCKx. When the last bit is latched, the SSPxIF interrupt flag bit is set.

While in Slave mode, the external clock is supplied by the external clock source on the SCKx pin. This external clock must meet the minimum high and low times as specified in the electrical specifications.

While in Sleep mode, the slave can transmit/receive data. When a byte is received, the device can be configured to wake-up from Sleep.

20.3.8 SLAVE SELECT SYNCHRONIZATION

The $\overline{SSx}$ pin allows a Synchronous Slave mode. The SPI must be in Slave mode with the $\overline{SSx}$ pin control enabled (SSPxCON1<3:0> = 04h). When the $\overline{SSx}$ pin is low, transmission and reception are enabled and the SDOx pin is driven. When the $\overline{SSx}$ pin goes high, the SDOx pin is no longer driven, even if in the middle of a transmitted byte and becomes a floating output. External pull-up/pull-down resistors may be desirable depending on the application.

Note 1:	When the SPI is in Slave mode with the $\overline{SSx}$ pin control enabled (SSPxCON1<3:0> = 0100), the SPI module will reset if the $\overline{SSx}$ pin is set to V_{DD}.
2:	If the SPI is used in Slave mode with CKE set, then the $\overline{SSx}$ pin control must be enabled.

When the SPI module resets, the bit counter is forced to '0'. This can be done by either forcing the $\overline{SSx}$ pin to a high level or clearing the SSPEN bit.

To emulate two-wire communication, the SDOx pin can be connected to the SDIx pin. When the SPI needs to operate as a receiver, the SDOx pin can be configured as an input. This disables transmissions from the SDOx. The SDIx can always be left as an input (SDIx function) since it cannot create a bus conflict.

FIGURE 20-4: SLAVE SYNCHRONIZATION WAVEFORM

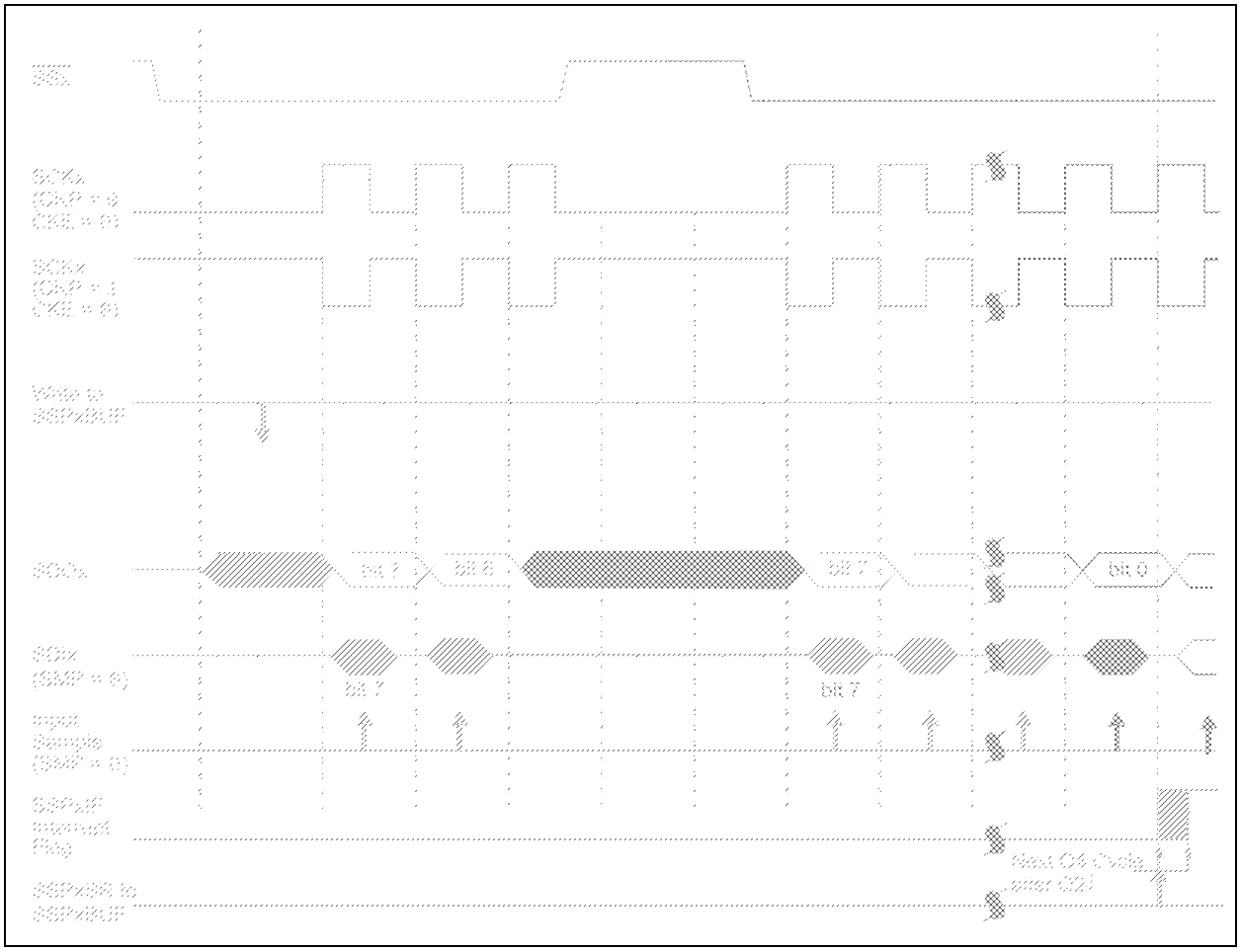


FIGURE 20-15: I²C SLAVE MODE TIMING WITH SEN = 1 (RECEPTION, 7-BIT ADDRESS)

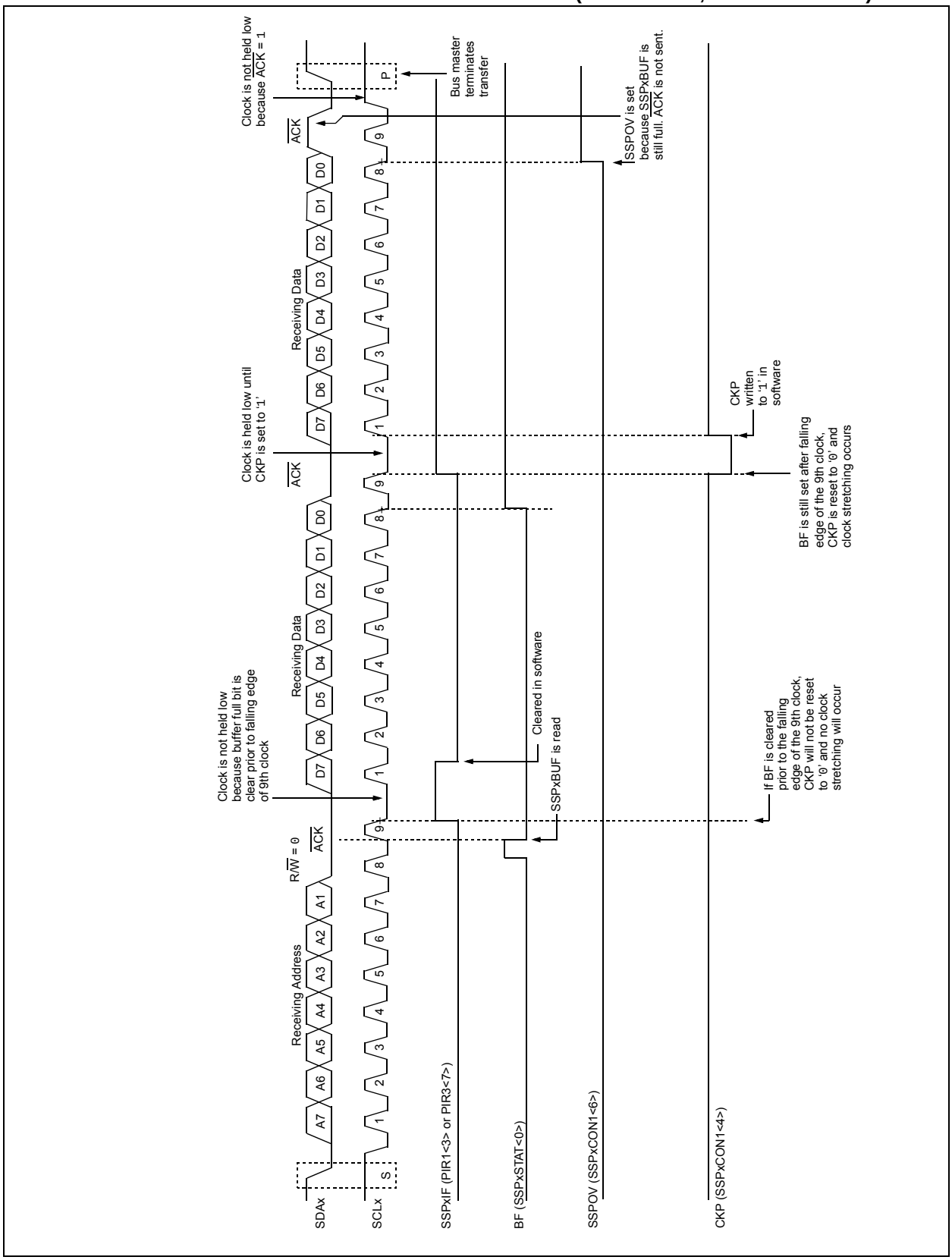
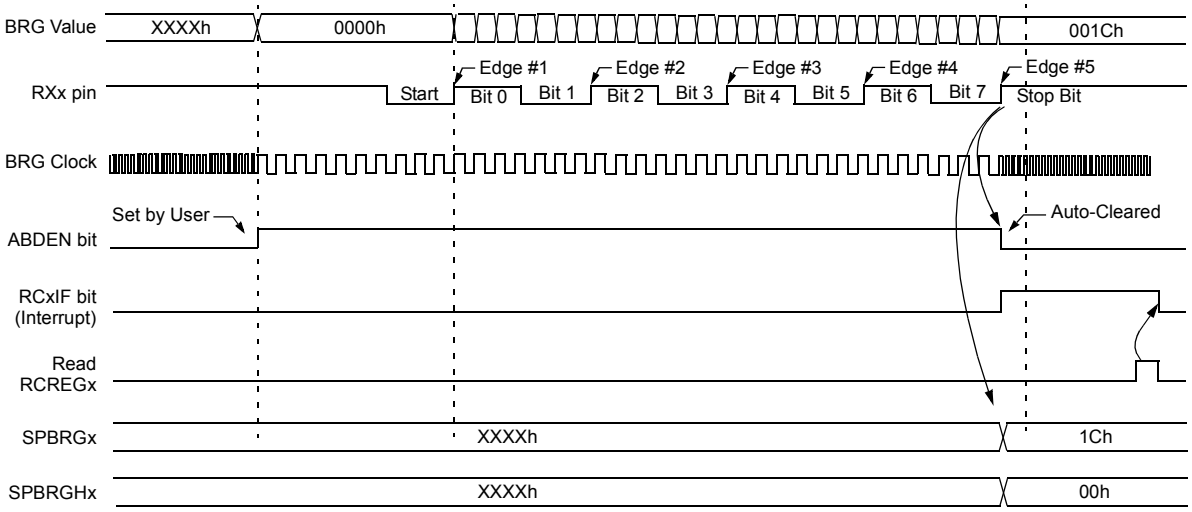
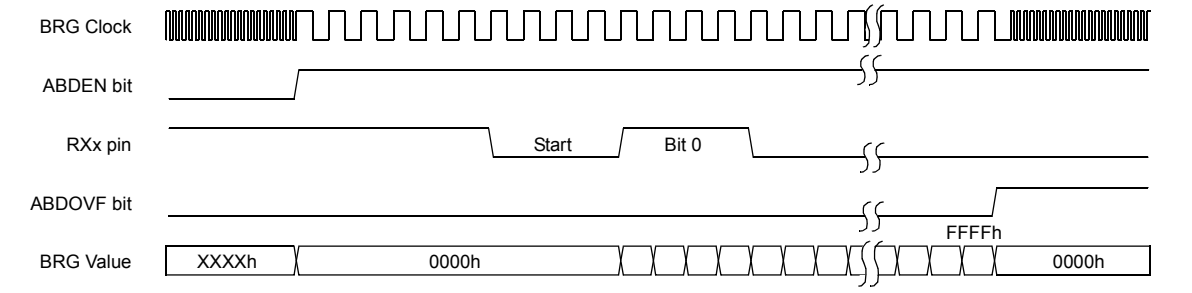


FIGURE 21-1: AUTOMATIC BAUD RATE CALCULATION



Note: The ABD sequence requires the EUSART module to be configured in Asynchronous mode and WUE = 0.

FIGURE 21-2: BRG OVERFLOW SEQUENCE



23.5 USB Interrupts

The USB module can generate multiple interrupt conditions. To accommodate all of these interrupt sources, the module is provided with its own interrupt logic structure, similar to that of the microcontroller. USB interrupts are enabled with one set of control registers and trapped with a separate set of flag registers. All sources are funneled into a single USB interrupt request, USBIF (PIR2<4>), in the microcontroller's interrupt logic.

Figure 23-7 provides the interrupt logic for the USB module. There are two layers of interrupt registers in the USB module. The top level consists of overall USB status interrupts; these are enabled and flagged in the UIE and UIR registers, respectively. The second level consists of USB error conditions, which are enabled and flagged in the UEIR and UEIE registers. An interrupt condition in any of these triggers a USB Error Interrupt Flag (UERRIF) in the top level.

Interrupts may be used to trap routine events in a USB transaction. Figure 23-8 provides some common events within a USB frame and its corresponding interrupts.

FIGURE 23-7: USB INTERRUPT LOGIC FUNNEL

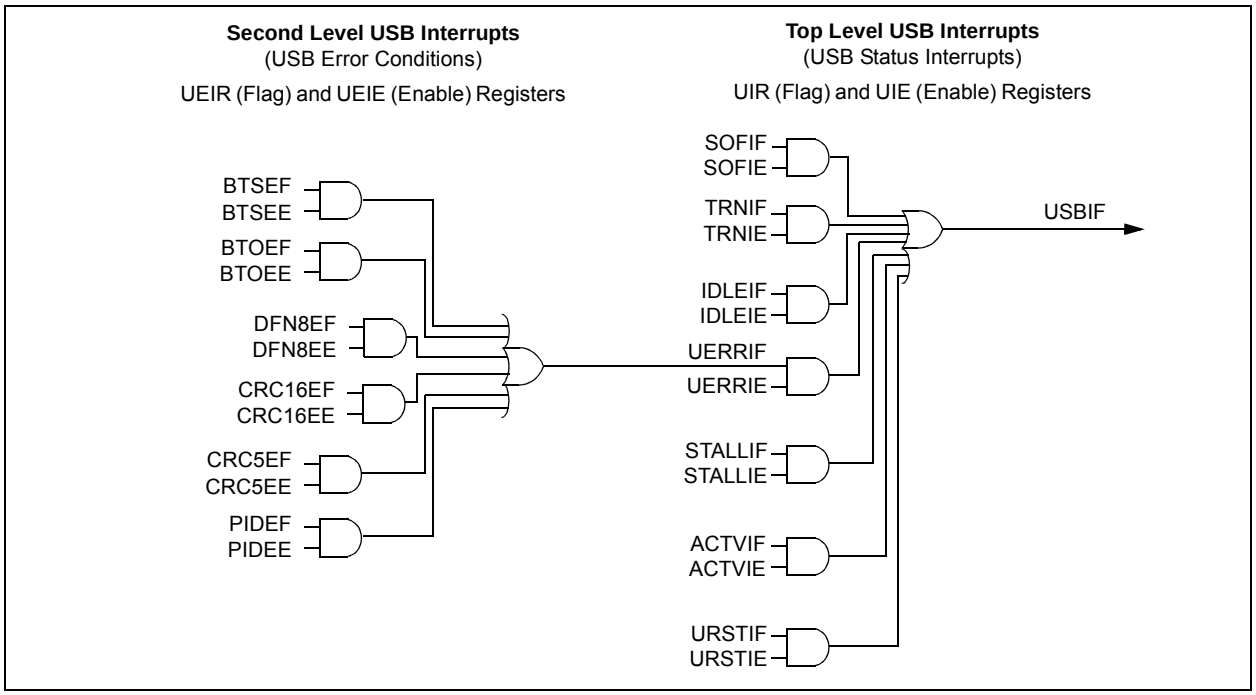
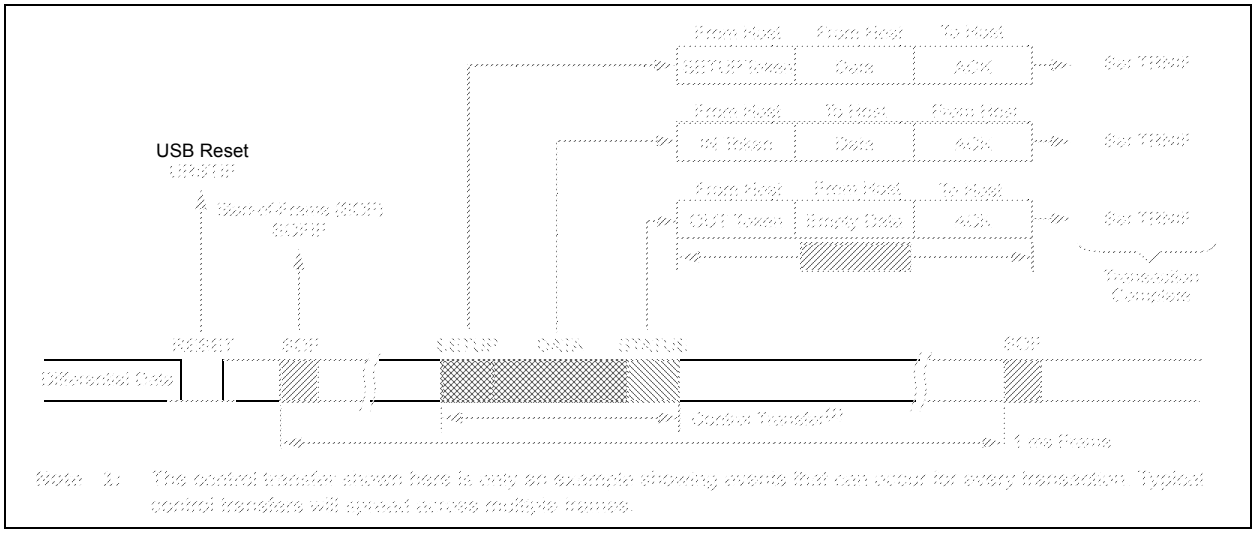


FIGURE 23-8: EXAMPLE OF A USB TRANSACTION AND INTERRUPT EVENTS



REGISTER 24-2: CMSTAT: COMPARATOR STATUS REGISTER (ACCESS F70h)

U-0	U-0	U-0	U-0	U-0	R-1	R-1	R-1
—	—	—	—	—	COUT3	COUT2	COUT1
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 7-3 **Unimplemented:** Read as ‘0’

bit 2-0 **COUT<3:1>:** Comparator x Status bits
(For example, COUT3 gives the status for Comparator 3.)
If CPOL (CMxCON<5>) = 0 non-inverted polarity):
1 = Comparator $V_{IN+} > V_{IN-}$
0 = Comparator $V_{IN+} < V_{IN-}$
If CPOL = 1 (inverted polarity):
1 = Comparator $V_{IN+} < V_{IN-}$
0 = Comparator $V_{IN+} > V_{IN-}$

REGISTER 28-8: CONFIG4H: CONFIGURATION REGISTER 4 HIGH (BYTE ADDRESS 300007h)

U-1	U-1	U-1	U-1	R/WO-1	U-0	R/WO-1	R/WO-1
—	—	—	—	LS48MHZ	—	WPEND	WPDIS
bit 7							bit 0

Legend:			
R = Readable bit	WO = Write-Once bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-4 **Unimplemented:** Program the corresponding Flash Configuration bit to ‘1’

bit 3 **LS48MHZ:** Low-Speed USB Clock Selection
1 = 48-MHz system clock is expected; divide-by-8 generates low-speed USB clock
0 = 24-MHz system clock is expected; divide-by-4 generates low-speed USB clock

bit 2 **Unimplemented:** Read as ‘0’

bit 1 **WPEND:** Write-Protect Disable bit
1 = Flash pages, WPFP<6:0> to (Configuration Words page), are write/erase protected
0 = Flash pages 0 to WPFP<6:0> are write/erase-protected

bit 0 **WPDIS:** Write-Protect Disable bit
1 = WPFP<5:0>, WPEND and WPCFG bits are ignored; all Flash memory may be erased or written
0 = WPFP<5:0>, WPEND and WPCFG bits enabled; erase/write-protect is active for the selected region(s)

REGISTER 28-9: DEVID1: DEVICE ID REGISTER 1 FOR PIC18F47J53 FAMILY DEVICES (BYTE ADDRESS 3FFFFEh)

R	R	R	R	R	R	R	R
DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0
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 **DEV<2:0>:** Device ID bits
These bits are used with DEV<10:3> bits in Device ID Register 2 to identify the part number. See Register 28-10.

bit 4-0 **REV<4:0>:** Revision ID bits
These bits are used to indicate the device revision.

28.4 Two-Speed Start-up

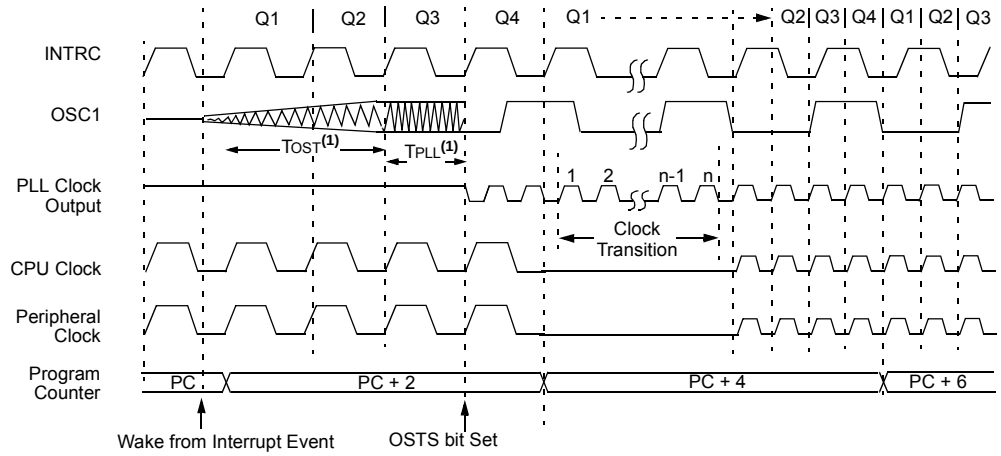
The Two-Speed Start-up feature helps to minimize the latency period, from oscillator start-up to code execution, by allowing the microcontroller to use the INTRC oscillator as a clock source until the primary clock source is available. It is enabled by setting the IESO Configuration bit.

Two-Speed Start-up should be enabled only if the primary oscillator mode is HS or HSPLL (Crystal-Based) modes. Since the EC and ECPLL modes do not require an Oscillator Start-up Timer (OST) delay, Two-Speed Start-up should be disabled.

When enabled, Resets and wake-ups from Sleep mode cause the device to configure itself to run from the internal oscillator block as the clock source, following the time-out of the Power-up Timer after a Power-on Reset is enabled. This allows almost immediate code execution while the primary oscillator starts and the OST is running. Once the OST times out, the device automatically switches to PRI_RUN mode.

In all other power-managed modes, Two-Speed Start-up is not used. The device will be clocked by the currently selected clock source until the primary clock source becomes available. The setting of the IESO bit is ignored.

FIGURE 28-3: TIMING TRANSITION FOR TWO-SPEED START-UP (INTRC TO HSPLL)



Note1: TOST = 1024 TOSC; TPLL = 2 ms (approx). These intervals are not shown to scale.

28.4.1 SPECIAL CONSIDERATIONS FOR USING TWO-SPEED START-UP

While using the INTRC oscillator in Two-Speed Start-up, the device still obeys the normal command sequences for entering power-managed modes, including serial SLEEP instructions (refer to **Section 4.1.4 "Multiple Sleep Commands"**). In practice, this means that user code can change the SCS<1:0> bit settings or issue SLEEP instructions before the OST times out. This would allow an application to briefly wake-up, perform routine "housekeeping" tasks and return to Sleep before the device starts to operate from the primary oscillator.

User code can also check if the primary clock source is currently providing the device clocking by checking the status of the OSTS bit (OSCCON<3>). If the bit is set, the primary oscillator is providing the clock. Otherwise, the internal oscillator block is providing the clock during wake-up from Reset or Sleep mode.

28.5 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 device clock to the internal oscillator block. The FSCM function is enabled by setting the FCMEN Configuration bit.

When FSCM is enabled, the INTRC oscillator runs at all times to monitor clocks to peripherals and provide a backup clock in the event of a clock failure. Clock monitoring (shown in Figure 28-4) 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 device clock and the sample clock are presented as inputs to the clock monitor latch. The clock monitor is set on the falling edge of the device clock source but cleared on the rising edge of the sample clock.

BNOV **Branch if Not Overflow**

Syntax: BNOV n

Operands: -128 ≤ n ≤ 127

Operation: if Overflow bit is '0',
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1110	0101	nnnn	nnnn

Description: If the Overflow bit is '0', 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 2-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 BNOV Jump

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

BNZ **Branch if Not Zero**

Syntax: BNZ n

Operands: -128 ≤ n ≤ 127

Operation: if Zero bit is '0',
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1110	0001	nnnn	nnnn

Description: If the Zero bit is '0', 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 2-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 BNZ Jump

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

CALLW **Subroutine Call Using WREG**

Syntax: CALLW

Operands: None

Operation: (PC + 2) → TOS,
(W) → PCL,
(PCLATH) → PCH,
(PCLATU) → PCU

Status Affected: None

Encoding:

0000	0000	0001	0100

Description: First, the return address (PC + 2) is pushed onto the return stack. Next, the contents of W are written to PCL; the existing value is discarded. Then, the contents of PCLATH and PCLATU are latched into PCH and PCU, respectively. The second cycle is executed as a NOP instruction while the new next instruction is fetched.

Unlike CALL, there is no option to update W, STATUS or BSR.

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read WREG	Push PC to stack	No operation
No operation	No operation	No operation	No operation

Example: HERE CALLW

Before Instruction
PC = address (HERE)
PCLATH = 10h
PCLATU = 00h
W = 06h
After Instruction
PC = 001006h
TOS = address (HERE + 2)
PCLATH = 10h
PCLATU = 00h
W = 06h

MOVSF **Move Indexed to f**

Syntax: MOVSF [z_s], f_d

Operands: $0 \le z_s \le 127$
$0 \le f_d \le 4095$

Operation: ((FSR2) + z_s) → f_d

Status Affected: None

Encoding:

1st word (source)	1110	1011	0zzz	$zzzz_s$
2nd word (destin.)	1111	ffff	ffff	$ffff_d$

Description: The contents of the source register are moved to destination register 'f_d'. The actual address of the source register is determined by adding the 7-bit literal offset 'z_s', in the first word, to the value of FSR2. The address of the destination register is specified by the 12-bit literal 'f_d' in the second word. Both addresses can be anywhere in the 4096-byte data space (000h to FFFh).

The MOVSF instruction cannot use the PCL, TOSU, TOSH or TOSL as the destination register.

If the resultant source address points to an Indirect Addressing register, the value returned will be 00h.

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Determine source addr	Determine source addr	Read source reg
Decode	No operation No dummy read	No operation	Write register 'f' (dest)

Example: MOVSF [05h], REG2

Before Instruction
FSR2 = 80h
Contents of 85h = 33h
REG2 = 11h
After Instruction
FSR2 = 80h
Contents of 85h = 33h
REG2 = 33h

31.2 DC Characteristics: Power-Down and Supply Current PIC18F47J53 Family (Industrial)

PIC18LF47J53 Family		Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial					
PIC18F47J53 Family		**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial					
Param. No.	**Device**	**Typ.**	**Max.**	**Units**	**Conditions**		
	Power-Down Current (IPD)(1) – Sleep mode						
	PIC18LFXXJ53	0.1	1.6	µA	-40°C	VDD = 2.0V, VDDCORE = 2.0V	**Sleep mode**, REGSLP = 1
		0.2	1.6	µA	+25°C		
		0.8	7.0	µA	+60°C		
		2.1	11.5	µA	+85°C		
	PIC18LFXXJ53	0.2	2.0	µA	-40°C	VDD = 2.5V, VDDCORE = 2.5V	
		0.5	2.0	µA	+25°C		
		1.4	9.0	µA	+60°C		
		3.2	15.0	µA	+85°C		
	PIC18FXXJ53	3.0	6.0	µA	-40°C	VDD = 2.15V Vddcore = 10 µF Capacitor	
		3.8	6.0	µA	+25°C		
		4.7	9.0	µA	+60°C		
		6.4	18.5	µA	+85°C		
	PIC18FXXJ53	3.3	9.0	µA	-40°C	VDD = 3.3V Vddcore = 10 µF Capacitor	
		4.2	9.0	µA	+25°C		
		5.5	12.0	µA	+60°C		
		7.8	22.0	µA	+85°C		
	Power-Down Current (IPD)(1) – Deep Sleep mode						
	PIC18FXXJ53	2	25	nA	-40°C	VDD = 2.15V, VDDCORE = 10 µF Capacitor	**Deep Sleep** mode
		9	100	nA	+25°C		
		72	250	nA	+60°C		
		0.26	1.0	µA	+85°C		
	PIC18FXXJ53	17	50	nA	-40°C	VDD = 3.3V, VDDCORE = 10 µF Capacitor	
		53	150	nA	+25°C		
		186	400	nA	+60°C		
		0.50	2.0	µA	+85°C		

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.).

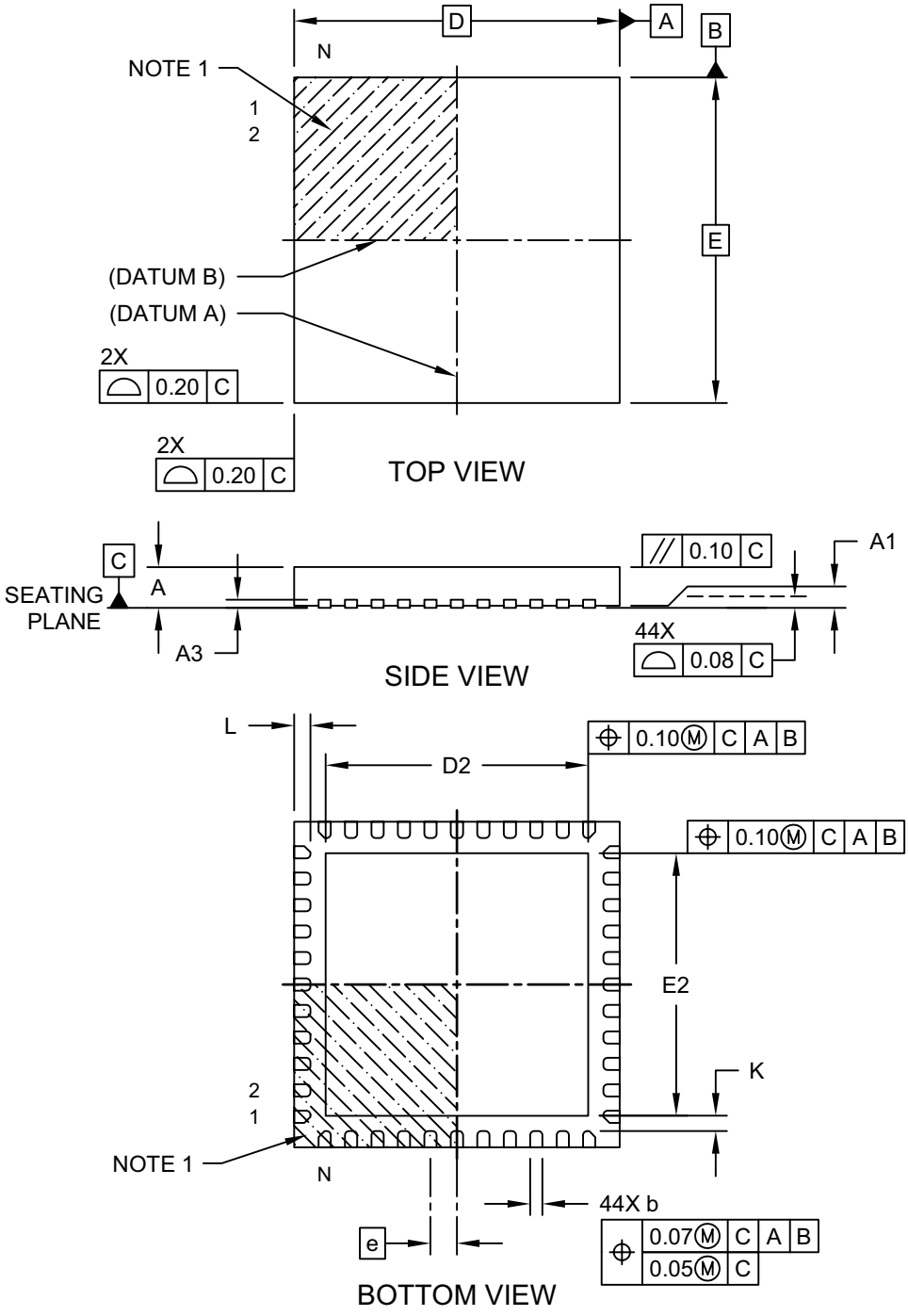
2: The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption. All features that add delta current are disabled (USB module, WDT, etc.). 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/VSS;
$\overline{MCLR}$ = VDD; WDT disabled unless otherwise specified.

3: Low-power Timer1 with 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.

4: This is the module differential current when the USB module is enabled and clocked at 48 MHz, but with no USB cable attached. When the USB cable is attached or data is being transmitted, the current consumption may be much higher (see **Section 23.6.4 "USB Transceiver Current Consumption"**). During USB Suspend mode (USBEN = 1, SUSPND = 1, bus in Idle state), the USB module current will be dominated by the D+ or D- pull-up resistor. The integrated pull-up resistors use "resistor switching" according to the `resistor_ecn` supplement to the USB 2.0 Specifications, and therefore, may be as low as 900Ω during Idle conditions.

44-Lead Plastic Quad Flat, No Lead Package (ML) - 8x8 mm Body [QFN or VQFN]

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



Microchip Technology Drawing C04-103D Sheet 1 of 2