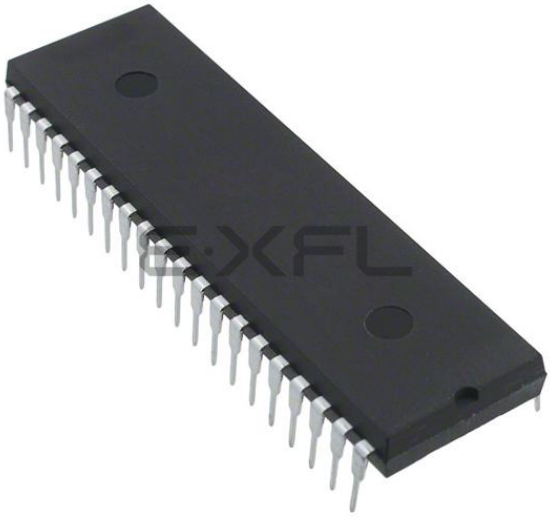


Welcome to E-XFL.COM

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, SPI, UART/USART, USB
Peripherals	Brown-out Detect/Reset, HLVD, POR, PWM, WDT
Number of I/O	35
Program Memory Size	32KB (16K x 16)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	2K x 8
Voltage - Supply (Vcc/Vdd)	4.2V ~ 5.5V
Data Converters	A/D 13x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Through Hole
Package / Case	40-DIP (0.600", 15.24mm)
Supplier Device Package	40-PDIP
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f4550-i-p

2.0 OSCILLATOR CONFIGURATIONS

2.1 Overview

Devices in the PIC18F2455/2550/4455/4550 family incorporate a different oscillator and microcontroller clock system than previous PIC18F devices. The addition of the USB module, with its unique requirements for a stable clock source, make it necessary to provide a separate clock source that is compliant with both USB low-speed and full-speed specifications.

To accommodate these requirements, PIC18F2455/2550/4455/4550 devices include a new clock branch to provide a 48 MHz clock for full-speed USB operation. Since it is driven from the primary clock source, an additional system of prescalers and postscalers has been added to accommodate a wide range of oscillator frequencies. An overview of the oscillator structure is shown in Figure 2-1.

Other oscillator features used in PIC18 enhanced microcontrollers, such as the internal oscillator block and clock switching, remain the same. They are discussed later in this chapter.

2.1.1 OSCILLATOR CONTROL

The operation of the oscillator in PIC18F2455/2550/4455/4550 devices is controlled through two Configuration registers and two control registers. Configuration registers, CONFIG1L and CONFIG1H, select the oscillator mode and USB prescaler/postscaler options. As Configuration bits, these are set when the device is programmed and left in that configuration until the device is reprogrammed.

The OSCCON register (Register 2-2) selects the Active Clock mode; it is primarily used in controlling clock switching in power-managed modes. Its use is discussed in **Section 2.4.1 "Oscillator Control Register"**.

The OSCTUNE register (Register 2-1) is used to trim the INTRC frequency source, as well as select the low-frequency clock source that drives several special features. Its use is described in **Section 2.2.5.2 "OSCTUNE Register"**.

2.2 Oscillator Types

PIC18F2455/2550/4455/4550 devices can be operated in twelve distinct oscillator modes. In contrast with previous PIC18 enhanced microcontrollers, four of these modes involve the use of two oscillator types at once. Users can program the FOSC3:FOSC0 Configuration bits to select one of these modes:

1. XT Crystal/Resonator
2. HS High-Speed Crystal/Resonator
3. HSPLL High-Speed Crystal/Resonator with PLL Enabled
4. EC External Clock with FOSC/4 Output
5. ECIO External Clock with I/O on RA6
6. ECPLL External Clock with PLL Enabled and FOSC/4 Output on RA6
7. ECPIO External Clock with PLL Enabled, I/O on RA6
8. INTHS Internal Oscillator used as Microcontroller Clock Source, HS Oscillator used as USB Clock Source
9. INTIO Internal Oscillator used as Microcontroller Clock Source, EC Oscillator used as USB Clock Source, Digital I/O on RA6
10. INTCKO Internal Oscillator used as Microcontroller Clock Source, EC Oscillator used as USB Clock Source, FOSC/4 Output on RA6

2.2.1 OSCILLATOR MODES AND USB OPERATION

Because of the unique requirements of the USB module, a different approach to clock operation is necessary. In previous PIC® devices, all core and peripheral clocks were driven by a single oscillator source; the usual sources were primary, secondary or the internal oscillator. With PIC18F2455/2550/4455/4550 devices, the primary oscillator becomes part of the USB module and cannot be associated to any other clock source. Thus, the USB module must be clocked from the primary clock source; however, the microcontroller core and other peripherals can be separately clocked from the secondary or internal oscillators as before.

Because of the timing requirements imposed by USB, an internal clock of either 6 MHz or 48 MHz is required while the USB module is enabled. Fortunately, the microcontroller and other peripherals are not required to run at this clock speed when using the primary oscillator. There are numerous options to achieve the USB module clock requirement and still provide flexibility for clocking the rest of the device from the primary oscillator source. These are detailed in **Section 2.3 "Oscillator Settings for USB"**.

2.2.5.4 Compensating for INTOSC Drift

It is possible to adjust the INTOSC frequency by modifying the value in the OSCTUNE register. This has no effect on the INTRC clock source frequency.

Tuning the INTOSC source requires knowing when to make the adjustment, in which direction it should be made and in some cases, how large a change is needed. When using the EUSART, for example, an adjustment may be required when it begins to generate framing errors or receives data with errors while in Asynchronous mode. Framing errors indicate that the device clock frequency is too high; to adjust for this, decrement the value in OSCTUNE to reduce the clock frequency. On the other hand, errors in data may suggest that the clock speed is too low; to compensate, increment OSCTUNE to increase the clock frequency.

It is also possible to verify device clock speed against a reference clock. Two timers may be used: one timer is clocked by the peripheral clock, while the other is clocked by a fixed reference source, such as the Timer1 oscillator. Both timers are cleared but the timer clocked by the reference generates interrupts. When an interrupt occurs, the internally clocked timer is read and both timers are cleared. If the internally clocked timer value is greater than expected, then the internal oscillator block is running too fast. To adjust for this, decrement the OSCTUNE register.

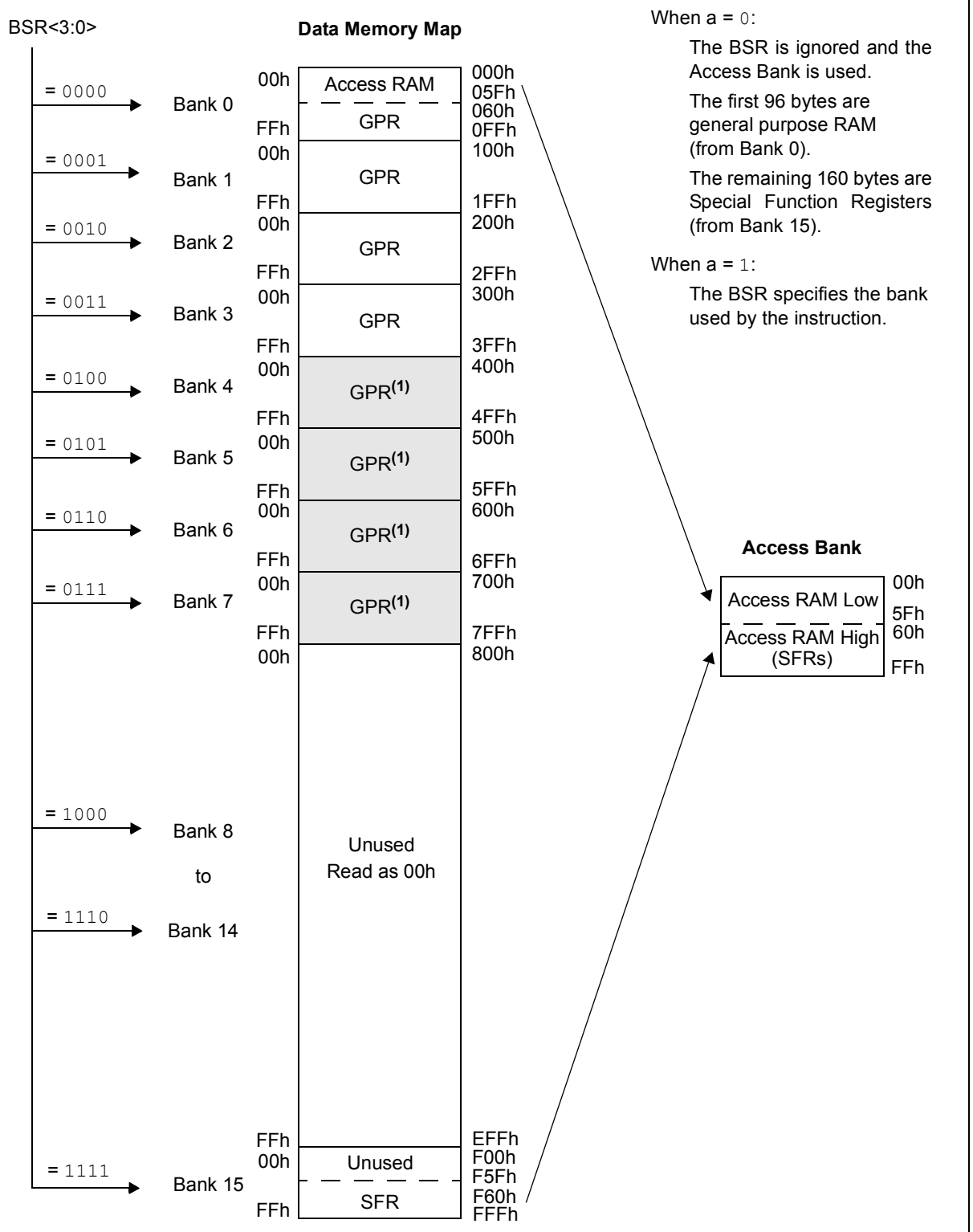
Finally, a CCP module can use free-running Timer1 (or Timer3), clocked by the internal oscillator block and an external event with a known period (i.e., AC power frequency). The time of the first event is captured in the CCPRxH:CCPRxL registers and is recorded for use later. When the second event causes a capture, the time of the first event is subtracted from the time of the second event. Since the period of the external event is known, the time difference between events can be calculated.

If the measured time is much greater than the calculated time, the internal oscillator block is running too fast; to compensate, decrement the OSCTUNE register. If the measured time is much less than the calculated time, the internal oscillator block is running too slow; to compensate, increment the OSCTUNE register.

NOTES:

NOTES:

FIGURE 5-5: DATA MEMORY MAP



Note 1: These banks also serve as RAM buffer for USB operation. See **Section 5.3.1 "USB RAM"** for more information.

9.7 INTx Pin Interrupts

External interrupts on the RB0/AN12/INT0/FLT0/SDI/SDA, RB1/AN10/INT1/SCK/SCL and RB2/AN8/INT2/VMO pins are edge-triggered. If the corresponding INTEDGx bit in the INTCON2 register is set (= 1), the interrupt is triggered by a rising edge; if the bit is clear, the trigger is on the falling edge. When a valid edge appears on the RBx/INTx pin, the corresponding flag bit, INTxIF, is set. This interrupt can be disabled by clearing the corresponding enable bit, INTxIE. Flag bit, INTxIF, must be cleared in software in the Interrupt Service Routine before re-enabling the interrupt.

All external interrupts (INT0, INT1 and INT2) can wake-up the processor from the power-managed modes if bit, INTxIE, was set prior to going into the power-managed modes. If the Global Interrupt Enable bit, GIE, is set, the processor will branch to the interrupt vector following wake-up.

Interrupt priority for INT1 and INT2 is determined by the value contained in the interrupt priority bits, INT1IP (INTCON3<6>) and INT2IP (INTCON3<7>). There is no priority bit associated with INT0. It is always a high-priority interrupt source.

9.8 TMR0 Interrupt

In 8-bit mode (which is the default), an overflow in the TMR0 register (FFh → 00h) will set flag bit, TMR0IF. In 16-bit mode, an overflow in the TMR0H:TMR0L register pair (FFFFh → 0000h) will set TMR0IF. The interrupt can be enabled/disabled by setting/clearing enable bit, TMR0IE (INTCON<5>). Interrupt priority for Timer0 is determined by the value contained in the interrupt priority bit, TMR0IP (INTCON2<2>). See **Section 11.0 "Timer0 Module"** for further details on the Timer0 module.

9.9 PORTB Interrupt-on-Change

An input change on PORTB<7:4> sets flag bit, RBIF (INTCON<0>). The interrupt can be enabled/disabled by setting/clearing enable bit, RBIE (INTCON<3>). Interrupt priority for PORTB interrupt-on-change is determined by the value contained in the interrupt priority bit, RBIP (INTCON2<0>).

9.10 Context Saving During Interrupts

During interrupts, the return PC address is saved on the stack. Additionally, the WREG, STATUS and BSR registers are saved on the Fast Return Stack. If a fast return from interrupt is not used (see **Section 5.3 "Data Memory Organization"**), the user may need to save the WREG, STATUS and BSR registers on entry to the Interrupt Service Routine. Depending on the user's application, other registers may also need to be saved. Example 9-1 saves and restores the WREG, STATUS and BSR registers during an Interrupt Service Routine.

EXAMPLE 9-1: SAVING STATUS, WREG AND BSR REGISTERS IN RAM

```
MOVWF     W_TEMP                    ; W_TEMP is in virtual bank
MOVFF     STATUS, STATUS_TEMP       ; STATUS_TEMP located anywhere
MOVFF     BSR, BSR_TEMP             ; BSR_TMEP located anywhere
;
; USER ISR CODE
;
MOVFF     BSR_TEMP, BSR             ; Restore BSR
MOVF      W_TEMP, W                 ; Restore WREG
MOVFF     STATUS_TEMP, STATUS       ; Restore STATUS
```

11.3 Prescaler

An 8-bit counter is available as a prescaler for the Timer0 module. The prescaler is not directly readable or writable; its value is set by the PSA and T0PS2:T0PS0 bits (T0CON<3:0>) which determine the prescaler assignment and prescale ratio.

Clearing the PSA bit assigns the prescaler to the Timer0 module. When it is assigned, prescale values from 1:2 through 1:256, in power-of-2 increments, are selectable.

When assigned to the Timer0 module, all instructions writing to the TMR0 register (e.g., `CLRF TMR0`, `MOVWF TMR0`, `BSF TMR0`, etc.) clear the prescaler count.

Note:	Writing to TMR0 when the prescaler is assigned to Timer0 will clear the prescaler count but will not change the prescaler assignment.

11.3.1 SWITCHING PRESCALER ASSIGNMENT

The prescaler assignment is fully under software control and can be changed “on-the-fly” during program execution.

11.4 Timer0 Interrupt

The TMR0 interrupt is generated when the TMR0 register overflows from FFh to 00h in 8-bit mode, or from FFFFh to 0000h in 16-bit mode. This overflow sets the TMR0IF flag bit. The interrupt can be masked by clearing the TMR0IE bit (INTCON<5>). Before re-enabling the interrupt, the TMR0IF bit must be cleared in software by the Interrupt Service Routine.

Since Timer0 is shut down in Sleep mode, the TMR0 interrupt cannot awaken the processor from Sleep.

TABLE 11-1: REGISTERS ASSOCIATED WITH TIMER0

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on page
TMR0L	Timer0 Register Low Byte								54
TMR0H	Timer0 Register High Byte								54
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	53
INTCON2	$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	—	TMR0IP	—	RBIP	53
T0CON	TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0	54
TRISA	—	TRISA6(1)	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	56

Legend: — = unimplemented locations, read as ‘0’. Shaded cells are not used by Timer0.

Note 1: RA6 is configured as a port pin based on various primary oscillator modes. When the port pin is disabled, all of the associated bits read ‘0’.

13.0 TIMER2 MODULE

The Timer2 module timer incorporates the following features:

- 8-bit Timer and Period registers (TMR2 and PR2, respectively)
- Readable and writable (both registers)
- Software programmable prescaler (1:1, 1:4 and 1:16)
- Software programmable postscaler (1:1 through 1:16)
- Interrupt on TMR2 to PR2 match
- Optional use as the shift clock for the MSSP module

The module is controlled through the T2CON register (Register 13-1) which enables or disables the timer and configures the prescaler and postscaler. Timer2 can be shut off by clearing control bit, TMR2ON (T2CON<2>), to minimize power consumption.

A simplified block diagram of the module is shown in Figure 13-1.

13.1 Timer2 Operation

In normal operation, TMR2 is incremented from 00h on each clock (FOSC/4). A 2-bit counter/prescaler on the clock input gives direct input, divide-by-4 and divide-by-16 prescale options. These are selected by the prescaler control bits, T2CKPS1:T2CKPS0 (T2CON<1:0>). The value of TMR2 is compared to that of the Period register, PR2, on each clock cycle. When the two values match, the comparator generates a match signal as the timer output. This signal also resets the value of TMR2 to 00h on the next cycle and drives the output counter/post-scaler (see **Section 13.2 "Timer2 Interrupt"**).

The TMR2 and PR2 registers are both directly readable and writable. The TMR2 register is cleared on any device Reset, while the PR2 register initializes at FFh. Both the prescaler and postscaler counters are cleared on the following events:

- a write to the TMR2 register
- a write to the T2CON register
- any device Reset (Power-on Reset, $\overline{\text{MCLR}}$ Reset, Watchdog Timer Reset or Brown-out Reset)

TMR2 is not cleared when T2CON is written.

REGISTER 13-1: T2CON: TIMER2 CONTROL REGISTER

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0
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 **Unimplemented:** Read as '0'

bit 6-3 **T2OUTPS3:T2OUTPS0**: Timer2 Output Postscale Select bits
0000 = 1:1 Postscale
0001 = 1:2 Postscale
•
•
•
1111 = 1:16 Postscale

bit 2 **TMR2ON**: Timer2 On bit
1 = Timer2 is on
0 = Timer2 is off

bit 1-0 **T2CKPS1:T2CKPS0**: Timer2 Clock Prescale Select bits
00 = Prescaler is 1
01 = Prescaler is 4
1x = Prescaler is 16

TABLE 17-4: ASSIGNMENT OF BUFFER DESCRIPTORS FOR THE DIFFERENT BUFFERING MODES

Endpoint	BDs Assigned to Endpoint							
	Mode 0 (No Ping-Pong)		Mode 1 (Ping-Pong on EP0 OUT)		Mode 2 (Ping-Pong on all EPs)		Mode 3 (Ping-Pong on all other EPs, except EP0)	
	Out	In	Out	In	Out	In	Out	In
0	0	1	0 (E), 1 (O)	2	0 (E), 1 (O)	2 (E), 3 (O)	0	1
1	2	3	3	4	4 (E), 5 (O)	6 (E), 7 (O)	2 (E), 3 (O)	4 (E), 5 (O)
2	4	5	5	6	8 (E), 9 (O)	10 (E), 11 (O)	6 (E), 7 (O)	8 (E), 9 (O)
3	6	7	7	8	12 (E), 13 (O)	14 (E), 15 (O)	10 (E), 11 (O)	12 (E), 13 (O)
4	8	9	9	10	16 (E), 17 (O)	18 (E), 19 (O)	14 (E), 15 (O)	16 (E), 17 (O)
5	10	11	11	12	20 (E), 21 (O)	22 (E), 23 (O)	18 (E), 19 (O)	20 (E), 21 (O)
6	12	13	13	14	24 (E), 25 (O)	26 (E), 27 (O)	22 (E), 23 (O)	24 (E), 25 (O)
7	14	15	15	16	28 (E), 29 (O)	30 (E), 31 (O)	26 (E), 27 (O)	28 (E), 29 (O)
8	16	17	17	18	32 (E), 33 (O)	34 (E), 35 (O)	30 (E), 31 (O)	32 (E), 33 (O)
9	18	19	19	20	36 (E), 37 (O)	38 (E), 39 (O)	34 (E), 35 (O)	36 (E), 37 (O)
10	20	21	21	22	40 (E), 41 (O)	42 (E), 43 (O)	38 (E), 39 (O)	40 (E), 41 (O)
11	22	23	23	24	44 (E), 45 (O)	46 (E), 47 (O)	42 (E), 43 (O)	44 (E), 45 (O)
12	24	25	25	26	48 (E), 49 (O)	50 (E), 51 (O)	46 (E), 47 (O)	48 (E), 49 (O)
13	26	27	27	28	52 (E), 53 (O)	54 (E), 55 (O)	50 (E), 51 (O)	52 (E), 53 (O)
14	28	29	29	30	56 (E), 57 (O)	58 (E), 59 (O)	54 (E), 55 (O)	56 (E), 57 (O)
15	30	31	31	32	60 (E), 61 (O)	62 (E), 63 (O)	58 (E), 59 (O)	60 (E), 61 (O)

Legend: (E) = Even transaction buffer, (O) = Odd transaction buffer

TABLE 17-5: SUMMARY OF USB BUFFER DESCRIPTOR TABLE REGISTERS

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
BDnSTAT(1)	UOWN	DTS(4)	PID3(2) KEN(3)	PID2(2) INCDIS(3)	PID1(2) DTSEN(3)	PID0(2) BSTALL(3)	BC9	BC8
BDnCNT(1)	Byte Count							
BDnADRL(1)	Buffer Address Low							
BDnADRH(1)	Buffer Address High							

Note 1: For buffer descriptor registers, n may have a value of 0 to 63. For the sake of brevity, all 64 registers are shown as one generic prototype. All registers have indeterminate Reset values (xxxx xxxx).

2: Bits 5 through 2 of the BDnSTAT register are used by the SIE to return PID3:PID0 values once the register is turned over to the SIE (UOWN bit is set). Once the registers have been under SIE control, the values written for KEN, INCDIS, DTSEN and BSTALL are no longer valid.

3: Prior to turning the buffer descriptor over to the SIE (UOWN bit is cleared), bits 5 through 2 of the BDnSTAT register are used to configure the KEN, INCDIS, DTSEN and BSTALL settings.

4: This bit is ignored unless DTSEN = 1.

17.6 USB Power Modes

Many USB applications will likely have several different sets of power requirements and configuration. The most common power modes encountered are Bus Power Only, Self-Power Only and Dual Power with Self-Power Dominance. The most common cases are presented here.

17.6.1 BUS POWER ONLY

In Bus Power Only mode, all power for the application is drawn from the USB (Figure 17-10). This is effectively the simplest power method for the device.

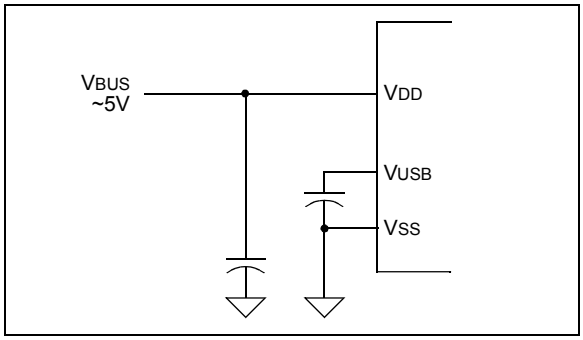
In order to meet the inrush current requirements of the USB 2.0 specifications, the total effective capacitance appearing across VBUS and ground must be no more than 10 μF. If not, some kind of inrush limiting is required. For more details, see Section 7.2.4 of the USB 2.0 specification.

According to the USB 2.0 specification, all USB devices must also support a Low-Power Suspend mode. In the USB Suspend mode, devices must consume no more than 2.5 mA from the 5V VBUS line of the USB cable.

The host signals the USB device to enter the Suspend mode by stopping all USB traffic to that device for more than 3 ms. This condition will cause the IDLEIF bit in the UIR register to become set.

During the USB Suspend mode, the D+ or D- pull-up resistor must remain active, which will consume some of the allowed suspend current: 2.5 mA budget.

FIGURE 17-10: BUS POWER ONLY



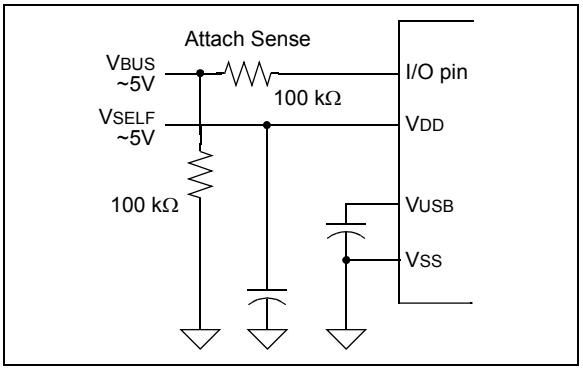
17.6.2 SELF-POWER ONLY

In Self-Power Only mode, the USB application provides its own power, with very little power being pulled from the USB. Figure 17-11 shows an example. Note that an attach indication is added to indicate when the USB has been connected and the host is actively powering VBUS.

In order to meet compliance specifications, the USB module (and the D+ or D- pull-up resistor) should not be enabled until the host actively drives VBUS high. One of the I/O pins may be used for this purpose.

The application should never source any current onto the 5V VBUS pin of the USB cable.

FIGURE 17-11: SELF-POWER ONLY



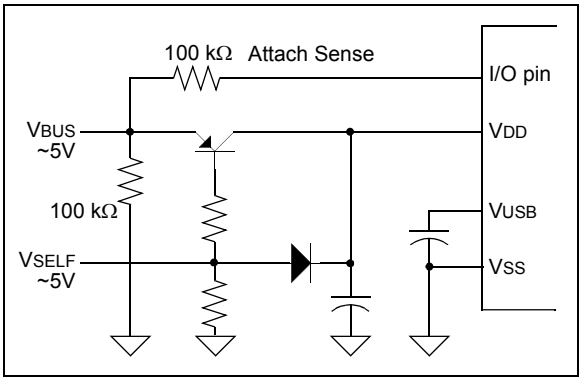
17.6.3 DUAL POWER WITH SELF-POWER DOMINANCE

Some applications may require a dual power option. This allows the application to use internal power primarily, but switch to power from the USB when no internal power is available. Figure 17-12 shows a simple Dual Power with Self-Power Dominance example, which automatically switches between Self-Power Only and USB Bus Power Only modes.

Dual power devices also must meet all of the special requirements for inrush current and Suspend mode current and must not enable the USB module until VBUS is driven high. For descriptions of those requirements, see **Section 17.6.1 "Bus Power Only"** and **Section 17.6.2 "Self-Power Only"**.

Additionally, dual power devices must never source current onto the 5V VBUS pin of the USB cable.

FIGURE 17-12: DUAL POWER EXAMPLE



Note:	Users should keep in mind the limits for devices drawing power from the USB. According to USB specification 2.0, this cannot exceed 100 mA per low-power device or 500 mA per high-power device.

REGISTER 20-1: TXSTA: TRANSMIT STATUS AND CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-1	R/W-0
CSRC	TX9	TXEN(1)	SYNC	SENDB	BRGH	TRMT	TX9D
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 **CSRC:** Clock Source Select bit
Asynchronous mode:
Don't care.
Synchronous mode:
1 = Master mode (clock generated internally from BRG)
0 = Slave mode (clock from external source)

bit 6 **TX9:** 9-Bit Transmit Enable bit
1 = Selects 9-bit transmission
0 = Selects 8-bit transmission

bit 5 **TXEN:** Transmit Enable bit(1)
1 = Transmit enabled
0 = Transmit disabled

bit 4 **SYNC:** EUSART Mode Select bit
1 = Synchronous mode
0 = Asynchronous mode

bit 3 **SENDB:** Send Break Character bit
Asynchronous mode:
1 = Send Sync Break on next transmission (cleared by hardware upon completion)
0 = Sync Break transmission completed
Synchronous mode:
Don't care.

bit 2 **BRGH:** High Baud Rate Select bit
Asynchronous mode:
1 = High speed
0 = Low speed
Synchronous mode:
Unused in this mode.

bit 1 **TRMT:** Transmit Shift Register Status bit
1 = TSR empty
0 = TSR full

bit 0 **TX9D:** 9th bit of Transmit Data
Can be address/data bit or a parity bit.

Note 1: SREN/CREN overrides TXEN in Sync mode with the exception that SREN has no effect in Synchronous Slave mode.

20.4 EUSART Synchronous Slave Mode

Synchronous Slave mode is entered by clearing bit, CSRC (TXSTA<7>). This mode differs from the Synchronous Master mode in that the shift clock is supplied externally at the CK pin (instead of being supplied internally in Master mode). This allows the device to transfer or receive data while in any power-managed mode.

20.4.1 EUSART SYNCHRONOUS SLAVE TRANSMISSION

The operation of the Synchronous Master and Slave modes is identical, except in the case of the Sleep mode.

If two words are written to the TXREG and then the SLEEP instruction is executed, the following will occur:

a) The first word will immediately transfer to the TSR register and transmit.
b) The second word will remain in the TXREG register.
c) Flag bit, TXIF, will not be set.
d) When the first word has been shifted out of TSR, the TXREG register will transfer the second word to the TSR and flag bit, TXIF, will now be set.
e) If enable bit, TXIE, is set, the interrupt will wake the chip from Sleep. If the global interrupt is enabled, the program will branch to the interrupt vector.

To set up a Synchronous Slave Transmission:

1. Enable the synchronous slave serial port by setting bits, SYNC and SPEN, and clearing bit, CSRC.
2. Clear bits, CREN and SREN.
3. If interrupts are desired, set enable bit, TXIE.
4. If the signal from the CK pin is to be inverted, set the TXCKP bit. If the signal from the DT pin is to be inverted, set the RXDTP bit.
5. If 9-bit transmission is desired, set bit, TX9.
6. Enable the transmission by setting enable bit, TXEN.
7. If 9-bit transmission is selected, the ninth bit should be loaded in bit, TX9D.
8. Start transmission by loading data to the TXREG register.
9. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

TABLE 20-9: REGISTERS ASSOCIATED WITH SYNCHRONOUS SLAVE TRANSMISSION

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	53
PIR1	SPPIF(1)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	56
PIE1	SPPIE(1)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	56
IPR1	SPPIP(1)	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	56
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	55
TXREG	EUSART Transmit Register								55
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	55
BAUDCON	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	55
SPBRGH	EUSART Baud Rate Generator Register High Byte								55
SPBRG	EUSART Baud Rate Generator Register Low Byte								55

Legend: — = unimplemented, read as '0'. Shaded cells are not used for synchronous slave transmission.

Note 1: Reserved in 28-pin devices; always maintain these bits clear.

REGISTER 25-2: CONFIG1H: CONFIGURATION REGISTER 1 HIGH (BYTE ADDRESS 300001h)

R/P-0	R/P-0	U-0	U-0	R/P-0	R/P-1	R/P-0	R/P-1
IESO	FCMEN	—	—	FOSC3(1)	FOSC2(1)	FOSC1(1)	FOSC0(1)
bit 7							bit 0

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

bit 7 **IESO:** Internal/External Oscillator Switchover bit
1 = Oscillator Switchover mode enabled
0 = Oscillator Switchover mode disabled

bit 6 **FCMEN:** Fail-Safe Clock Monitor Enable bit
1 = Fail-Safe Clock Monitor enabled
0 = Fail-Safe Clock Monitor disabled

bit 5-4 **Unimplemented:** Read as '0'

bit 3-0 **FOSC3:FOSC0:** Oscillator Selection bits(1)
111x = HS oscillator, PLL enabled (HSPLL)
110x = HS oscillator (HS)
1011 = Internal oscillator, HS oscillator used by USB (INTHS)
1010 = Internal oscillator, XT used by USB (INTXT)
1001 = Internal oscillator, CLKO function on RA6, EC used by USB (INTCKO)
1000 = Internal oscillator, port function on RA6, EC used by USB (INTIO)
0111 = EC oscillator, PLL enabled, CLKO function on RA6 (ECPLL)
0110 = EC oscillator, PLL enabled, port function on RA6 (ECPIO)
0101 = EC oscillator, CLKO function on RA6 (EC)
0100 = EC oscillator, port function on RA6 (ECIO)
001x = XT oscillator, PLL enabled (XTPLL)
000x = XT oscillator (XT)

Note 1: The microcontroller and USB module both use the selected oscillator as their clock source in XT, HS and EC modes. The USB module uses the indicated XT, HS or EC oscillator as its clock source whenever the microcontroller uses the internal oscillator.

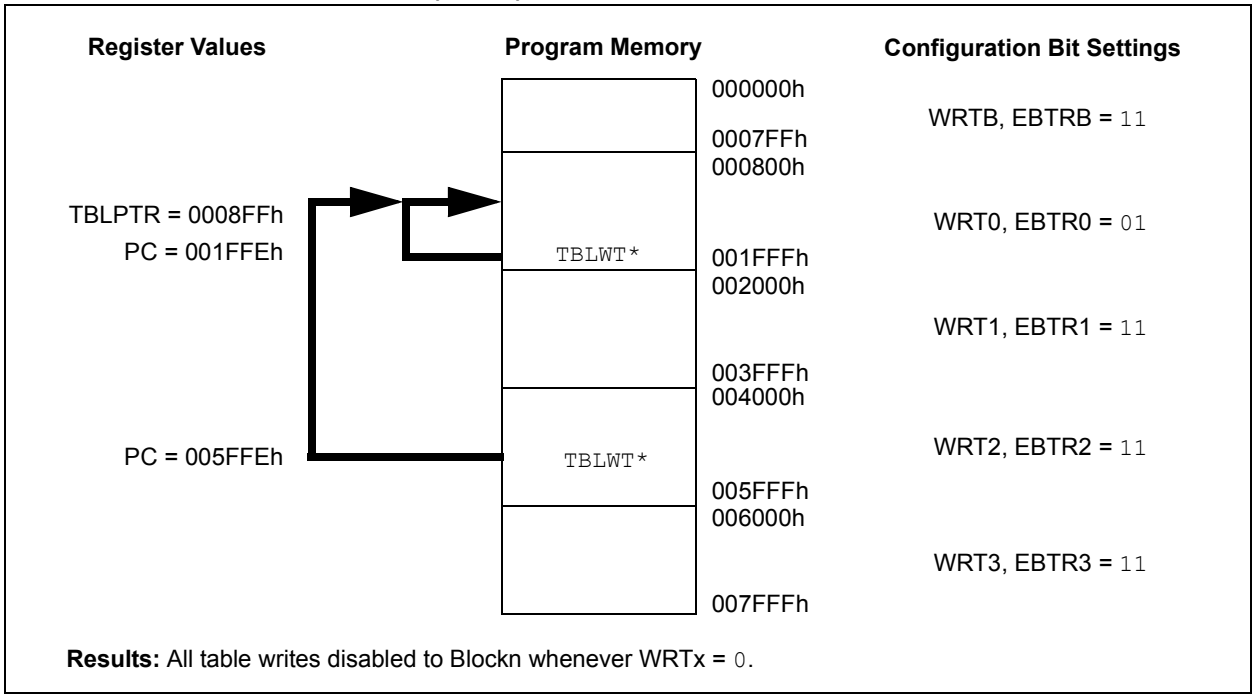
25.5.1 PROGRAM MEMORY CODE PROTECTION

The program memory may be read to or written from any location using the table read and table write instructions. The device ID may be read with table reads. The Configuration registers may be read and written with the table read and table write instructions.

In normal execution mode, the CPx bits have no direct effect. CPx bits inhibit external reads and writes. A block of user memory may be protected from table writes if the WRTx Configuration bit is '0'. The EBTRx bits control table reads. For a block of user memory with the EBTRx bit set to '0', a table read instruction that executes from within that block is allowed to read. A table read instruction that executes from a location outside of that block is not allowed to read and will result in reading '0's. Figures 25-6 through 25-8 illustrate table write and table read protection.

Note:	Code protection bits may only be written to a '0' from a '1' state. It is not possible to write a '1' to a bit in the '0' state. Code protection bits are only set to '1' by a full Chip Erase or Block Erase function. The full Chip Erase and Block Erase functions can only be initiated via ICSP operation or an external programmer.

FIGURE 25-6: TABLE WRITE (WRTx) DISALLOWED



Results: All table writes disabled to Blockn whenever WRTx = 0.

BTG **Bit Toggle f**

Syntax: BTG f, b {,a}

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

Operation: $(\overline{f<b>}) \rightarrow f<b>$

Status Affected: None

Encoding:

0111	bbba	ffff	ffff

Description: Bit 'b' in data memory location 'f' is inverted.
If 'a' is '0', the Access Bank is selected.
If 'a' is '1', the BSR is used to select the GPR bank (default).
If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 26.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: `BTG PORTC, 4, 0`

Before Instruction:
PORTC = 0111 0101 [75h]
After Instruction:
PORTC = 0110 0101 [65h]

BOV **Branch if Overflow**

Syntax: BOV n

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

Operation: if Overflow bit is '1',
$(PC) + 2 + 2n \rightarrow PC$

Status Affected: None

Encoding:

1110	0100	nnnn	nnnn

Description: If the Overflow 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    BOV  Jump
```
Before Instruction
PC = address (HERE)
After Instruction
If Overflow = 1;
PC = address (Jump)
If Overflow = 0;
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

28.3 DC Characteristics: PIC18F2455/2550/4455/4550 (Industrial) PIC18LF2455/2550/4455/4550 (Industrial) (Continued)

DC CHARACTERISTICS			Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial			
Param No.	**Symbol**	**Characteristic**	**Min**	**Max**	**Units**	**Conditions**
	VOL	**Output Low Voltage**				
D080		I/O Ports (except RC4/RC5 in USB mode)	—	0.6	V	IOL = 8.5 mA, VDD = 4.5V, -40°C to +85°C
D083		OSC2/CLKO (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 (except RC4/RC5 in USB mode)	VDD – 0.7	—	V	IOH = -3.0 mA, VDD = 4.5V, -40°C to +85°C
D092		OSC2/CLKO (EC, ECIO, ECPIO 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 and HS 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: 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.

2: Negative current is defined as current sourced by the pin.

3: Parameter is characterized but not tested.

TABLE 28-4: USB MODULE SPECIFICATIONS

Operating Conditions: -40°C < TA < +85°C (unless otherwise stated).							
Param No.	**Sym**	**Characteristic**	**Min**	**Typ**	**Max**	**Units**	**Comments**
D313	VUSB	USB Voltage	3.0	—	3.6	V	Voltage on pin must be in this range for proper USB operation
D314	IIL	Input Leakage on D+ and D- pins	—	—	±1	μA	VSS ≤ VPIN ≤ VDD; pin at high-impedance
D315	VILUSB	Input Low Voltage for USB Buffer	—	—	0.8	V	For VUSB range
D316	VIHUSB	Input High Voltage for USB Buffer	2.0	—	—	V	For VUSB range
D317	VCRS	Crossover Voltage	1.3		2.0	V	Voltage range for D+ and D- crossover to occur
D318	VDIFS	Differential Input Sensitivity	—	—	0.2	V	The difference between D+ and D- must exceed this value while VCM is met
D319	VCM	Differential Common Mode Range	0.8	—	2.5	V	
D320	ZOUT	Driver Output Impedance	28	—	44	Ω	
D321	VOL	Voltage Output Low	0.0	—	0.3	V	1.5 kΩ load connected to 3.6V
D322	VOH	Voltage Output High	2.8	—	3.6	V	15 kΩ load connected to ground

TABLE 28-5: USB INTERNAL VOLTAGE REGULATOR SPECIFICATIONS

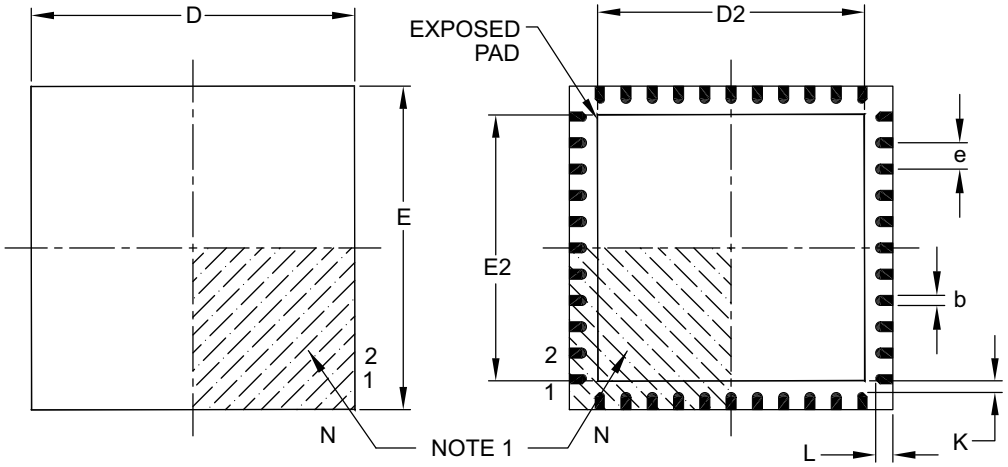
Operating Conditions: -40°C < TA < +85°C (unless otherwise stated).							
Param No.	**Sym**	**Characteristics**	**Min**	**Typ**	**Max**	**Units**	**Comments**
D323	VUSBANA	Regulator Output Voltage	3.0	—	3.6	V	VDD ≥ 4.0V(1)
D324	CUSB	External Filter Capacitor Value (VUSB to VSS)	0.22	0.47	12(2)	μF	Ceramic or other low-ESR capacitor recommended

Note 1: If device VDD is less than 4.0V, the internal USB voltage regulator should be disabled and an external 3.0-3.6V supply should be provided on VUSB if the USB module is used.

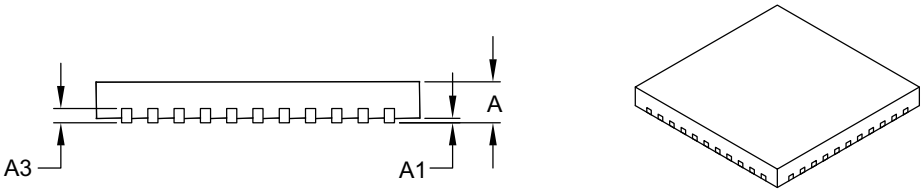
2: This is a recommended maximum for start-up time and in-rush considerations. When the USB regulator is disabled, there is no maximum.

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

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



TOP VIEW **BOTTOM VIEW**



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N		44	
Pitch	e		0.65 BSC	
Overall Height	A	0.80	0.90	1.00
Standoff	A1	0.00	0.02	0.05
Contact Thickness	A3		0.20 REF	
Overall Width	E		8.00 BSC	
Exposed Pad Width	E2	6.30	6.45	6.80
Overall Length	D		8.00 BSC	
Exposed Pad Length	D2	6.30	6.45	6.80
Contact Width	b	0.25	0.30	0.38
Contact Length	L	0.30	0.40	0.50
Contact-to-Exposed Pad	K	0.20	–	–

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Package is saw singulated.
3. Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-103B

C

D