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	16-Bit
Speed	32MHz
Connectivity	I²C, IrDA, SPI, UART/USART, USB OTG
Peripherals	Brown-out Detect/Reset, LVD, POR, PWM, WDT
Number of I/O	33
Program Memory Size	64KB (22K x 24)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	8K x 8
Voltage - Supply (Vcc/Vdd)	2V ~ 3.6V
Data Converters	A/D 13x10b
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/pic24fj64gb004-i-ml

TABLE 4-18:   USB OTG REGISTER MAP (CONTINUED)

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
U1EP0	04AA	—	—	—	—	—	—	—	—	LSPD[1]	RETRYDIS[1]	—	EPCONDIS	EPRXEN	EPTXEN	EPSTALL	EPHSHK	0000
U1EP1	04AC	—	—	—	—	—	—	—	—	—	—	—	EPCONDIS	EPRXEN	EPTXEN	EPSTALL	EPHSHK	0000
U1EP2	04AE	—	—	—	—	—	—	—	—	—	—	—	EPCONDIS	EPRXEN	EPTXEN	EPSTALL	EPHSHK	0000
U1EP3	04B0	—	—	—	—	—	—	—	—	—	—	—	EPCONDIS	EPRXEN	EPTXEN	EPSTALL	EPHSHK	0000
U1EP4	04B2	—	—	—	—	—	—	—	—	—	—	—	EPCONDIS	EPRXEN	EPTXEN	EPSTALL	EPHSHK	0000
U1EP5	04B4	—	—	—	—	—	—	—	—	—	—	—	EPCONDIS	EPRXEN	EPTXEN	EPSTALL	EPHSHK	0000
U1EP6	04B6	—	—	—	—	—	—	—	—	—	—	—	EPCONDIS	EPRXEN	EPTXEN	EPSTALL	EPHSHK	0000
U1EP7	04B8	—	—	—	—	—	—	—	—	—	—	—	EPCONDIS	EPRXEN	EPTXEN	EPSTALL	EPHSHK	0000
U1EP8	04BA	—	—	—	—	—	—	—	—	—	—	—	EPCONDIS	EPRXEN	EPTXEN	EPSTALL	EPHSHK	0000
U1EP9	04BC	—	—	—	—	—	—	—	—	—	—	—	EPCONDIS	EPRXEN	EPTXEN	EPSTALL	EPHSHK	0000
U1EP10	04BE	—	—	—	—	—	—	—	—	—	—	—	EPCONDIS	EPRXEN	EPTXEN	EPSTALL	EPHSHK	0000
U1EP11	04C0	—	—	—	—	—	—	—	—	—	—	—	EPCONDIS	EPRXEN	EPTXEN	EPSTALL	EPHSHK	0000
U1EP12	04C2	—	—	—	—	—	—	—	—	—	—	—	EPCONDIS	EPRXEN	EPTXEN	EPSTALL	EPHSHK	0000
U1EP13	04C4	—	—	—	—	—	—	—	—	—	—	—	EPCONDIS	EPRXEN	EPTXEN	EPSTALL	EPHSHK	0000
U1EP14	04C6	—	—	—	—	—	—	—	—	—	—	—	EPCONDIS	EPRXEN	EPTXEN	EPSTALL	EPHSHK	0000
U1EP15	04C8	—	—	—	—	—	—	—	—	—	—	—	EPCONDIS	EPRXEN	EPTXEN	EPSTALL	EPHSHK	0000
U1PWMRRS	04CC	USB Power Supply PWM Duty Cycle Register								USB Power Supply PWM Period Register								0000
U1PWMCON	04CE	PWMEN	—	—	—	—	—	—	PWMPOL	CNTEN	—	—	—	—	—	—	—	0000

Legend: — = unimplemented, read as '0'. Reset values are shown in hexadecimal.
Note   1:   Alternate register or bit definitions when the module is operating in Host mode.
         2:   This register is available in Host mode only.

TABLE 4-19:   PARALLEL MASTER/SLAVE PORT REGISTER MAP

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
PMCON	0600	PMPEN	—	PSIDL	ADRMUX1	ADRMUX0	PTBEEN	PTWREN	PTRDEN	CSF1	CSF0	ALP	—	CS1P	BEP	WRSP	RDSP	0000
PMMODE	0602	BUSY	IRQM1	IRQM0	INCM1	INCM0	MODE16	MODE1	MODE0	WAITB1	WAITB0	WAITM3	WAITM2	WAITM1	WAITM0	WAITE1	WAITE0	0000
PMADDR	0604	—	CS1	—	—	—	ADDR10[1]	ADDR9[1]	ADDR8[1]	ADDR7[1]	ADDR6[1]	ADDR5[1]	ADDR4[1]	ADDR3[1]	ADDR2[1]	ADDR1	ADDR0	0000
PMDOUT1		Parallel Port Data Out Register 1 (Buffers 0 and 1)																0000
PMDOUT2	0606	Parallel Port Data Out Register 2 (Buffers 2 and 3)																0000
PMDIN1	0608	Parallel Port Data In Register 1 (Buffers 0 and 1)																0000
PMDIN2	060A	Parallel Port Data In Register 2 (Buffers 2 and 3)																0000
PMAEN	060C	—	PTEN14	—	—	—	PTEN10[1]	PTEN9[1]	PTEN8[1]	PTEN7[1]	PTEN6[1]	PTEN5[1]	PTEN4[1]	PTEN3[1]	PTEN2[1]	PTEN1	PTEN0	0000
PMSTAT	060E	IBF	IBOV	—	—	IB3F	IB2F	IB1F	IB0F	OBE	OBUF	—	—	OB3E	OB2E	OB1E	OB0E	0000

Legend: — = unimplemented, read as '0'. Reset values are shown in hexadecimal.
Note   1:   Bits are not available on 28-pin devices; read as '0'.

PIC24FJ64GB004 FAMILY

5.2 RTSP Operation

The PIC24F Flash program memory array is organized into rows of 64 instructions or 192 bytes. RTSP allows the user to erase blocks of eight rows (512 instructions) at a time and to program one row at a time. It is also possible to program single words.

The 8-row erase blocks and single row write blocks are edge-aligned, from the beginning of program memory, on boundaries of 1536 bytes and 192 bytes, respectively.

When data is written to program memory using TBLWT instructions, the data is not written directly to memory. Instead, data written using table writes is stored in holding latches until the programming sequence is executed.

Any number of TBLWT instructions can be executed and a write will be successfully performed. However, 64 TBLWT instructions are required to write the full row of memory.

To ensure that no data is corrupted during a write, any unused addresses should be programmed with FFFFFFh. This is because the holding latches reset to an unknown state, so if the addresses are left in the Reset state, they may overwrite the locations on rows which were not rewritten.

The basic sequence for RTSP programming is to set up a Table Pointer, then do a series of TBLWT instructions to load the buffers. Programming is performed by setting the control bits in the NVMCON register.

Data can be loaded in any order and the holding registers can be written to multiple times before performing a write operation. Subsequent writes, however, will wipe out any previous writes.

> **Note:** Writing to a location multiple times without erasing is *not* recommended.

All of the table write operations are single-word writes (2 instruction cycles) because only the buffers are written. A programming cycle is required for programming each row.

5.3 JTAG Operation

The PIC24F family supports JTAG boundary scan. Boundary scan can improve the manufacturing process by verifying pin to PCB connectivity.

5.4 Enhanced In-Circuit Serial Programming

Enhanced In-Circuit Serial Programming uses an on-board bootloader, known as the program executive, to manage the programming process. Using an SPI data frame format, the program executive can erase, program and verify program memory. For more information on Enhanced ICSP, see the device programming specification.

5.5 Control Registers

There are two SFRs used to read and write the program Flash memory: NVMCON and NVMKEY.

The NVMCON register (Register 5-1) controls which blocks are to be erased, which memory type is to be programmed and when the programming cycle starts.

NVMKEY is a write-only register that is used for write protection. To start a programming or erase sequence, the user must consecutively write 55h and AAh to the NVMKEY register. Refer to **Section 5.6 "Programming Operations"** for further details.

5.6 Programming Operations

A complete programming sequence is necessary for programming or erasing the internal Flash in RTSP mode. During a programming or erase operation, the processor stalls (waits) until the operation is finished. Setting the WR bit (NVMCON<15>) starts the operation and the WR bit is automatically cleared when the operation is finished.

EXAMPLE 5-2: **ERASING A PROGRAM MEMORY BLOCK – 'C' LANGUAGE CODE**

```
// C example using MPLAB C30
    unsigned long progAddr = 0xXXXXXX;      // Address of row to write
    unsigned int offset;

//Set up pointer to the first memory location to be written
    TBLPAG = progAddr>>16;                  // Initialize PM Page Boundary SFR
    offset = progAddr & 0xFFFF;             // Initialize lower word of address

    __builtin_tblwtl(offset, 0x0000);      // Set base address of erase block
                                           // with dummy latch write

    NVMCON = 0x4042;                        // Initialize NVMCON

    asm("DISI #5");                         // Block all interrupts with priority <7
                                           // for next 5 instructions
    __builtin_write_NVM();                  // C30 function to perform unlock
                                           // sequence and set WR
```

EXAMPLE 5-3: **LOADING THE WRITE BUFFERS – ASSEMBLY LANGUAGE CODE**

```
; Set up NVMCON for row programming operations
        MOV    #0x4001, W0               ;
        MOV    W0, NVMCON                ; Initialize NVMCON
; Set up a pointer to the first program memory location to be written
; program memory selected, and writes enabled
        MOV    #0x0000, W0               ;
        MOV    W0, TBLPAG                ; Initialize PM Page Boundary SFR
        MOV    #0x6000, W0               ; An example program memory address
; Perform the TBLWT instructions to write the latches
; 0th_program_word
        MOV    #LOW_WORD_0, W2           ;
        MOV    #HIGH_BYTE_0, W3          ;
        TBLWTL W2, [W0]                  ; Write PM low word into program latch
        TBLWTH W3, [W0++]                ; Write PM high byte into program latch
; 1st_program_word
        MOV    #LOW_WORD_1, W2           ;
        MOV    #HIGH_BYTE_1, W3          ;
        TBLWTL W2, [W0]                  ; Write PM low word into program latch
        TBLWTH W3, [W0++]                ; Write PM high byte into program latch
;   2nd_program_word
        MOV    #LOW_WORD_2, W2           ;
        MOV    #HIGH_BYTE_2, W3          ;
        TBLWTL W2, [W0]                  ; Write PM low word into program latch
        TBLWTH W3, [W0++]                ; Write PM high byte into program latch
        •
        •
        •
; 63rd_program_word
        MOV    #LOW_WORD_31, W2          ;
        MOV    #HIGH_BYTE_31, W3         ;
        TBLWTL W2, [W0]                  ; Write PM low word into program latch
        TBLWTH W3, [W0]                  ; Write PM high byte into program latch
```

REGISTER 7-9: IFS4: INTERRUPT FLAG STATUS REGISTER 4

U-0	U-0	R/W-0	U-0	U-0	U-0	U-0	R/W-0
—	—	CTMUIF	—	—	—	—	LVDIF
bit 15							bit 8

U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	U-0
—	—	—	—	CRCIF	U2ERIF	U1ERIF	—
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 15-14 **Unimplemented:** Read as '0'

bit 13 **CTMUIF:** CTMU Interrupt Flag Status bit

1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 12-9 **Unimplemented:** Read as '0'

bit 8 **LVDIF:** Low-Voltage Detect Interrupt Flag Status bit

1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 7-4 **Unimplemented:** Read as '0'

bit 3 **CRCIF:** CRC Generator Interrupt Flag Status bit

1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 2 **U2ERIF:** UART2 Error Interrupt Flag Status bit

1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 1 **U1ERIF:** UART1 Error Interrupt Flag Status bit

1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 0 **Unimplemented:** Read as '0'

PIC24FJ64GB004 FAMILY

REGISTER 7-13: IEC2: INTERRUPT ENABLE CONTROL REGISTER 2

U-0	U-0	R/W-0	U-0	U-0	U-0	R/W-0	U-0
—	—	PMPIE	—	—	—	OC5IE	—
bit 15							bit 8

R/W-0	R/W-0	R/W-0	U-0	U-0	U-0	R/W-0	R/W-0
IC5IE	IC4IE	IC3IE	—	—	—	SPI2IE	SPF2IE
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 15-14 **Unimplemented:** Read as '0'

bit 13 **PMPIE:** Parallel Master Port Interrupt Enable bit
1 = Interrupt request is enabled
0 = Interrupt request is not enabled

bit 12-10 **Unimplemented:** Read as '0'

bit 9 **OC5IE:** Output Compare Channel 5 Interrupt Enable bit
1 = Interrupt request is enabled
0 = Interrupt request is not enabled

bit 8 **Unimplemented:** Read as '0'

bit 7 **IC5IE:** Input Capture Channel 5 Interrupt Enable bit
1 = Interrupt request is enabled
0 = Interrupt request is not enabled

bit 6 **IC4IE:** Input Capture Channel 4 Interrupt Enable bit
1 = Interrupt request is enabled
0 = Interrupt request is not enabled

bit 5 **IC3IE:** Input Capture Channel 3 Interrupt Enable bit
1 = Interrupt request is enabled
0 = Interrupt request is not enabled

bit 4-2 **Unimplemented:** Read as '0'

bit 1 **SPI2IE:** SPI2 Event Interrupt Enable bit
1 = Interrupt request is enabled
0 = Interrupt request is not enabled

bit 0 **SPF2IE:** SPI2 Fault Interrupt Enable bit
1 = Interrupt request is enabled
0 = Interrupt request is not enabled

REGISTER 7-28:   IPC11: INTERRUPT PRIORITY CONTROL REGISTER 11

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

U-0	R/W-1	R/W-0	R/W-0	U-0	U-0	U-0	U-0
—	PMPIP2	PMPIP1	PMPIP0	—	—	—	—
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 15-7   **Unimplemented:** Read as '0'

bit 6-4   **PMPIP<2:0>:** Parallel Master Port Interrupt Priority bits

111 = Interrupt is Priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is Priority 1
000 = Interrupt source is disabled

bit 3-0   **Unimplemented:** Read as '0'

REGISTER 7-30: IPC15: INTERRUPT PRIORITY CONTROL REGISTER 15

U-0	U-0	U-0	U-0	U-0	R/W-1	R/W-0	R/W-0
—	—	—	—	—	RTCIP2	RTCIP1	RTCIP0
bit 15							bit 8

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
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 15-11   **Unimplemented:** Read as '0'

bit 10-8   **RTCIP<2:0>:** Real-Time Clock/Calendar Interrupt Priority bits

111 = Interrupt is Priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is Priority 1
000 = Interrupt source is disabled

bit 7-0   **Unimplemented:** Read as '0'

8.1 CPU Clocking Scheme

The system clock source can be provided by one of four sources:

• Primary Oscillator (POSC) on the OSCI and OSCO pins
• Secondary Oscillator (SOSC) on the SOSCI and SOSCO pins
• Fast Internal RC (FRC) Oscillator
• Low-Power Internal RC (LPRC) Oscillator

The primary oscillator and FRC sources have the option of using the internal USB PLL block, which generates both the USB module clock and a separate system clock from the 96 MHz PLL. Refer to **Section 8.5 "Oscillator Modes and USB Operation"** for additional information.

The Fast Internal RC (FRC) provides an 8 MHz clock source. It can optionally be reduced by the programmable clock divider to provide a range of system clock frequencies.

The selected clock source generates the processor and peripheral clock sources. The processor clock source is divided by two to produce the internal instruction cycle clock, F_{CY}. In this document, the instruction cycle clock is also denoted by $F_{OSC}/2$. The internal instruction cycle clock, $F_{OSC}/2$, can be provided on the OSCO I/O pin for some operating modes of the primary oscillator.

8.2 Initial Configuration on POR

The oscillator source (and operating mode) that is used at a device Power-on Reset event is selected using Configuration bit settings. The oscillator Configuration bit settings are located in the Configuration registers in the program memory (refer to **Section 26.1 "Configuration Bits"** for further details). The Primary Oscillator Configuration bits, POSCMD<1:0> (Configuration Word 2<1:0>), and the Initial Oscillator Select Configuration bits, FNOSC<2:0> (Configuration Word 2<10:8>), select the oscillator source that is used at a Power-on Reset. The FRC Primary Oscillator with Postscaler (FRCDIV) is the default (unprogrammed) selection. The secondary oscillator, or one of the internal oscillators, may be chosen by programming these bit locations.

The Configuration bits allow users to choose between the various clock modes, shown in Table 8-1.

8.2.1 CLOCK SWITCHING MODE CONFIGURATION BITS

The FCKSM Configuration bits (Configuration Word 2<7:6>) are used to jointly configure device clock switching and the Fail-Safe Clock Monitor (FSCM). Clock switching is enabled only when FCKSM1 is programmed ('0'). The FSCM is enabled only when the FCKSM<1:0> bits are both programmed ('00').

TABLE 8-1: CONFIGURATION BIT VALUES FOR CLOCK SELECTION

Oscillator Mode	Oscillator Source	POSCMD<1:0>	FNOSC<2:0>	Notes
Fast RC Oscillator with Postscaler (FRCDIV)	Internal	11	111	**1, 2**
(Reserved)	Internal	xx	110	**1**
Low-Power RC Oscillator (LPRC)	Internal	11	101	**1**
Secondary (Timer1) Oscillator (SOSC)	Secondary	11	100	**1**
Primary Oscillator (XT) with PLL Module (XTPLL)	Primary	01	011	
Primary Oscillator (EC) with PLL Module (ECPLL)	Primary	00	011	
Primary Oscillator (HS)	Primary	10	010	
Primary Oscillator (XT)	Primary	01	010	
Primary Oscillator (EC)	Primary	00	010	
Fast RC Oscillator with PLL Module (FRCPLL)	Internal	11	001	**1**
Fast RC Oscillator (FRC)	Internal	11	000	**1**

Note 1: OSCO pin function is determined by the OSCIOFCN Configuration bit.
2: This is the default oscillator mode for an unprogrammed (erased) device.

9.2.2 IDLE MODE

Idle mode has these features:

- The CPU will stop executing instructions.
- The WDT is automatically cleared.
- The system clock source remains active. By default, all peripheral modules continue to operate normally from the system clock source, but can also be selectively disabled (see **Section 9.4 "Selective Peripheral Module Control"**).
- If the WDT or FSCM is enabled, the LPRC will also remain active.

The device will wake from Idle mode on any of these events:

- Any interrupt that is individually enabled
- Any device Reset
- A WDT time-out

On wake-up from Idle mode, the clock is reapplied to the CPU and instruction execution begins immediately, starting with the instruction following the PWRSAV instruction or the first instruction in the ISR.

9.2.3 INTERRUPTS COINCIDENT WITH POWER SAVE INSTRUCTIONS

Any interrupt that coincides with the execution of a PWRSAV instruction (except for Deep Sleep mode) will be held off until entry into Sleep or Idle mode has completed. The device will then wake-up from Sleep or Idle mode.

9.2.4 DEEP SLEEP MODE

In PIC24FJ64GB004 family devices, Deep Sleep mode is intended to provide the lowest levels of power consumption available, without requiring the use of external switches to completely remove all power from the device. Entry into Deep Sleep mode is completely under software control. Exit from Deep Sleep mode can be triggered from any of the following events:

- POR event
- $\overline{\text{MCLR}}$ event
- RTCC alarm (If the RTCC is present)
- External Interrupt 0
- Deep Sleep Watchdog Timer (DSWDT) time-out

In Deep Sleep mode, it is possible to keep the device Real-Time Clock and Calendar (RTCC) running without the loss of clock cycles.

The device has a dedicated Deep Sleep Brown-out Reset (DSBOR) and a Deep Sleep Watchdog Timer Reset (DSWDT) for monitoring voltage and time-out events. The DSBOR and DSWDT are independent of the standard BOR and WDT used with other power-managed modes (Sleep, Idle and Doze).

Note:	Since Deep Sleep mode powers down the microcontroller by turning off the on-chip V_{DDCORE} voltage regulator, Deep Sleep capability is available only when operating with the internal regulator enabled.

9.2.4.1 Entering Deep Sleep Mode

Deep Sleep mode is entered by setting the DSEN bit in the DSCON register, and then executing a SLEEP instruction (PWRSAV #SLEEP_MODE) within one to three instruction cycles to minimize the chance that Deep Sleep will be spuriously entered.

If the PWRSAV command is not given within three instruction cycles, the DSEN bit will be cleared by the hardware and must be set again by the software before entering Deep Sleep mode. The DSEN bit is also automatically cleared when exiting the Deep Sleep mode.

Note:	To re-enter Deep Sleep after a Deep Sleep wake-up, allow a delay of at least 3 T_{CY} after clearing the RELEASE bit.

The sequence to enter Deep Sleep mode is:

1. If the application requires the Deep Sleep WDT, enable it and configure its clock source (see **Section 9.2.4.7 "Deep Sleep WDT"** for details).
2. If the application requires Deep Sleep BOR, enable it by programming the DSBOREN Configuration bit (CW4<6>).
3. If the application requires wake-up from Deep Sleep on RTCC alarm, enable and configure the RTCC module (see **Section 20.0 "Real-Time Clock and Calendar (RTCC)"** for more information).
4. If needed, save any critical application context data by writing it to the DSGPR0 and DSGPR1 registers (optional).
5. Enable Deep Sleep mode by setting the DSEN bit (DSCON<15>).
6. Enter Deep Sleep mode by immediately issuing a PWRSAV #0 instruction.

Any time the DSEN bit is set, all bits in the DSWAKE register will be automatically cleared.

10.4.3 CONTROLLING PERIPHERAL PIN SELECT

Peripheral Pin Select features are controlled through two sets of Special Function Registers: one to map peripheral inputs and one to map outputs. Because they are separately controlled, a particular peripheral's input and output (if the peripheral has both) can be placed on any selectable function pin without constraint.

The association of a peripheral to a peripheral-selectable pin is handled in two different ways, depending on if an input or an output is being mapped.

10.4.3.1 Input Mapping

The inputs of the Peripheral Pin Select options are mapped on the basis of the peripheral; that is, a control register associated with a peripheral dictates the pin it will be mapped to. The RPINRx registers are used to configure peripheral input mapping (see Register 10-1 through Register 10-14). Each register contains up to two sets of 5-bit fields, with each set associated with one of the pin-selectable peripherals. Programming a given peripheral's bit field with an appropriate 6-bit value maps the RPn pin with that value to that peripheral. For any given device, the valid range of values for any of the bit fields corresponds to the maximum number of Peripheral Pin Select options supported by the device.

TABLE 10-2: SELECTABLE INPUT SOURCES (MAPS INPUT TO FUNCTION)[1]

Input Name	Function Name	Register	Function Mapping Bits
External Interrupt 1	INT1	RPINR0	INT1R<5:0>
External Interrupt 2	INT2	RPINR1	INT2R<5:0>
Input Capture 1	IC1	RPINR7	IC1R<5:0>
Input Capture 2	IC2	RPINR7	IC2R<5:0>
Input Capture 3	IC3	RPINR8	IC3R<5:0>
Input Capture 4	IC4	RPINR8	IC4R<5:0>
Input Capture 5	IC5	RPINR9	IC5R<5:0>
Output Compare Fault A	OCFA	RPINR11	OCFAR<5:0>
Output Compare Fault B	OCFB	RPINR11	OCFBR<5:0>
SPI1 Clock Input	SCK1IN	RPINR20	SCK1R<5:0>
SPI1 Data Input	SDI1	RPINR20	SDI1R<5:0>
SPI1 Slave Select Input	SS1IN	RPINR21	SS1R<5:0>
SPI2 Clock Input	SCK2IN	RPINR22	SCK2R<5:0>
SPI2 Data Input	SDI2	RPINR22	SDI2R<5:0>
SPI2 Slave Select Input	SS2IN	RPINR23	SS2R<5:0>
Timer2 External Clock	T2CK	RPINR3	T2CKR<5:0>
Timer3 External Clock	T3CK	RPINR3	T3CKR<5:0>
Timer4 External Clock	T4CK	RPINR4	T4CKR<5:0>
Timer5 External Clock	T5CK	RPINR4	T5CKR<5:0>
UART1 Clear To Send	U1CTS	RPINR18	U1CTSR<5:0>
UART1 Receive	U1RX	RPINR18	U1RXR<5:0>
UART2 Clear To Send	U2CTS	RPINR19	U2CTSR<5:0>
UART2 Receive	U2RX	RPINR19	U2RXR<5:0>

Note 1: Unless otherwise noted, all inputs use the Schmitt Trigger input buffers.

REGISTER 11-1: T1CON: TIMER1 CONTROL REGISTER[1]

R/W-0	U-0	R/W-0	U-0	U-0	U-0	U-0	U-0
TON	—	TSIDL	—	—	—	—	—
bit 15							bit 8

U-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	U-0
—	TGATE	TCKPS1	TCKPS0	—	TSYNC	TCS	—
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 15 **TON:** Timer1 On bit
1 = Starts 16-bit Timer1
0 = Stops 16-bit Timer1

bit 14 **Unimplemented:** Read as '0'

bit 13 **TSIDL:** Stop in Idle Mode bit
1 = Discontinue module operation when device enters Idle mode
0 = Continue module operation in Idle mode

bit 12-7 **Unimplemented:** Read as '0'

bit 6 **TGATE:** Timer1 Gated Time Accumulation Enable bit
<u>When TCS = 1:</u>
This bit is ignored.
<u>When TCS = 0:</u>
1 = Gated time accumulation is enabled
0 = Gated time accumulation is disabled

bit 5-4 **TCKPS<1:0>:** Timer1 Input Clock Prescale Select bits
11 = 1:256
10 = 1:64
01 = 1:8
00 = 1:1

bit 3 **Unimplemented:** Read as '0'

bit 2 **TSYNC:** Timer1 External Clock Input Synchronization Select bit
<u>When TCS = 1:</u>
1 = Synchronize external clock input
0 = Do not synchronize external clock input
<u>When TCS = 0:</u>
This bit is ignored.

bit 1 **TCS:** Timer1 Clock Source Select bit
1 = External clock from T1CK pin (on the rising edge)
0 = Internal clock (F$_{OSC}$/2)

bit 0 **Unimplemented:** Read as '0'

Note 1: Changing the value of TxCON while the timer is running (TON = 1) causes the timer prescale counter to reset and is not recommended.

REGISTER 16-2: I2CxSTAT: I2Cx STATUS REGISTER (CONTINUED)

bit 4 **P:** Stop bit

1 = Indicates that a Stop bit has been detected last
0 = Stop bit was not detected last
Hardware set or clear when Start, Repeated Start or Stop is detected.

bit 3 **S:** Start bit

1 = Indicates that a Start (or Repeated Start) bit has been detected last
0 = Start bit was not detected last
Hardware set or clear when Start, Repeated Start or Stop is detected.

bit 2 **R/$\overline{\text{W}}$:** Read/$\overline{\text{Write}}$ Information bit (when operating as I^{2}C slave)

1 = Read – indicates data transfer is output from slave
0 = Write – indicates data transfer is input to slave
Hardware set or clear after reception of I^{2}C device address byte.

bit 1 **RBF:** Receive Buffer Full Status bit

1 = Receive complete, I2CxRCV is full
0 = Receive not complete, I2CxRCV is empty
Hardware set when I2CxRCV is written with received byte. Hardware clear when software reads I2CxRCV.

bit 0 **TBF:** Transmit Buffer Full Status bit

1 = Transmit in progress, I2CxTRN is full
0 = Transmit complete, I2CxTRN is empty
Hardware set when software writes I2CxTRN. Hardware clear at completion of data transmission.

BDs have a fixed relationship to a particular endpoint, depending on the buffering configuration. Table 18-2 provides the mapping of BDs to endpoints. This relationship also means that gaps may occur in the BDT if endpoints are not enabled contiguously. This theoretically means that the BDs for disabled endpoints could be used as buffer space. In practice, users should avoid using such spaces in the BDT unless a method of validating BD addresses is implemented.

18.2.1 BUFFER OWNERSHIP

Because the buffers and their BDs are shared between the CPU and the USB module, a simple semaphore mechanism is used to distinguish which is allowed to update the BD and associated buffers in memory. This is done by using the UOWN bit as a semaphore to distinguish which is allowed to update the BD and associated buffers in memory. UOWN is the only bit that is shared between the two configurations of BDnSTAT.

When UOWN is clear, the BD entry is "owned" by the microcontroller core. When the UOWN bit is set, the BD entry and the buffer memory are "owned" by the USB peripheral. The core should not modify the BD or its corresponding data buffer during this time. Note that the microcontroller core can still read BDnSTAT while the SIE owns the buffer and vice versa.

The buffer descriptors have a different meaning based on the source of the register update. Register 18-1 and Register 18-2 show the differences in BDnSTAT depending on its current "ownership".

When UOWN is set, the user can no longer depend on the values that were written to the BDs. From this point, the USB module updates the BDs as necessary, overwriting the original BD values. The BDnSTAT register is updated by the SIE with the token PID and the transfer count is updated.

18.2.2 DMA INTERFACE

The USB OTG module uses a dedicated DMA to access both the BDT and the endpoint data buffers. Since part of the address space of the DMA is dedicated to the Buffer Descriptors, a portion of the memory connected to the DMA must comprise a contiguous address space properly mapped for the access by the module.

TABLE 18-2: 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

REGISTER 18-15: U1OTGIE: USB OTG INTERRUPT ENABLE REGISTER (HOST MODE ONLY)

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0
IDIE	T1MSECIE	LSTATEIE	ACTVIE	SESVDIE	SESENDIE	—	VBUSVDIE
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 15-8 **Unimplemented:** Read as '0'

bit 7 **IDIE:** ID Interrupt Enable bit

1 = Interrupt is enabled
0 = Interrupt is disabled

bit 6 **T1MSECIE:** 1 Millisecond Timer Interrupt Enable bit

1 = Interrupt is enabled
0 = Interrupt is disabled

bit 5 **LSTATEIE:** Line State Stable Interrupt Enable bit

1 = Interrupt is enabled
0 = Interrupt is disabled

bit 4 **ACTVIE:** Bus Activity Interrupt Enable bit

1 = Interrupt is enabled
0 = Interrupt is disabled

bit 3 **SESVDIE:** Session Valid Interrupt Enable bit

1 = Interrupt is enabled
0 = Interrupt is disabled

bit 2 **SESENDIE:** B-Device Session End Interrupt Enable bit

1 = Interrupt is enabled
0 = Interrupt is disabled

bit 1 **Unimplemented:** Read as '0'

bit 0 **VBUSVDIE:** A-Device V_{BUS} Valid Interrupt Enable bit

1 = Interrupt is enabled
0 = Interrupt is disabled

NOTES:

24.0 COMPARATOR VOLTAGE REFERENCE

> **Note:** This data sheet summarizes the features of this group of PIC24F devices. It is not intended to be a comprehensive reference source. For more information, refer to the *"PIC24F Family Reference Manual"*, **Section 20. "Comparator Voltage Reference Module"** (DS39709).

### 24.1 Configuring the Comparator Voltage Reference

The voltage reference module is controlled through the CVRCON register (Register 24-1). The comparator voltage reference provides two ranges of output voltage, each with 16 distinct levels. The range to be used is selected by the CVRR bit (CVRCON<5>). The primary difference between the ranges is the size of the steps selected by the CVREF Selection bits (CVR<3:0>), with one range offering finer resolution.

The comparator reference supply voltage can come from either VDD and VSS, or the external VREF+ and VREF-. The voltage source is selected by the CVRSS bit (CVRCON<4>).

The settling time of the comparator voltage reference must be considered when changing the CVREF output.

FIGURE 24-1: COMPARATOR VOLTAGE REFERENCE BLOCK DIAGRAM
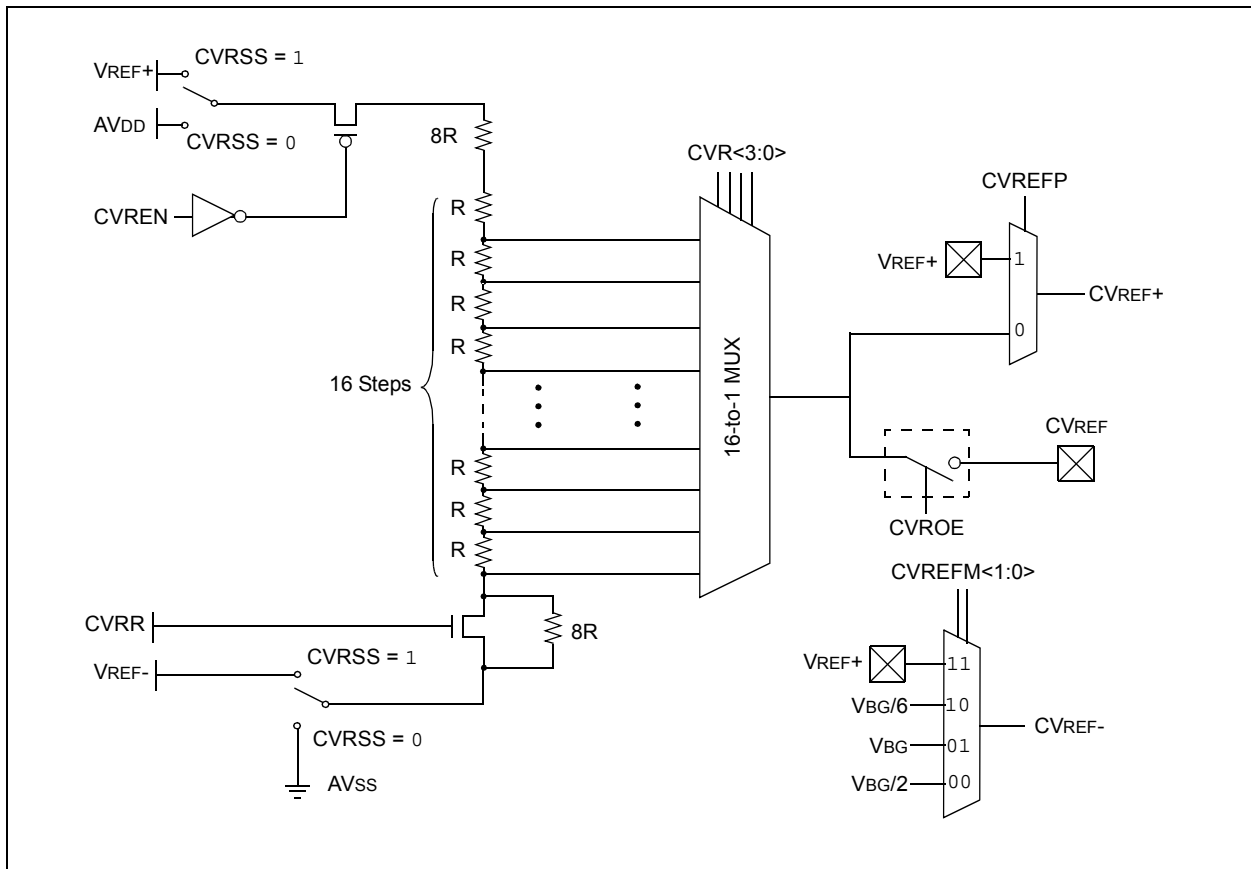
TABLE 28-2: INSTRUCTION SET OVERVIEW (CONTINUED)

| Assembly Mnemonic	Assembly Syntax		Description	# of Words	# of Cycles	Status Flags Affected
PWRSAV	PWRSAV	#lit1	Go into Sleep or Idle mode	1	1	WDTO, Sleep
RCALL	RCALL	Expr	Relative Call	1	2	None
	RCALL	Wn	Computed Call	1	2	None
REPEAT	REPEAT	#lit14	Repeat Next Instruction lit14 + 1 times	1	1	None
	REPEAT	Wn	Repeat Next Instruction (Wn) + 1 times	1	1	None
RESET	RESET		Software Device Reset	1	1	None
RETFIE	RETFIE		Return from Interrupt	1	3 (2)	None
RETLW	RETLW	#lit10,Wn	Return with Literal in Wn	1	3 (2)	None
RETURN	RETURN		Return from Subroutine	1	3 (2)	None
RLC	RLC	f	f = Rotate Left through Carry f	1	1	C, N, Z
	RLC	f,WREG	WREG = Rotate Left through Carry f	1	1	C, N, Z
	RLC	Ws,Wd	Wd = Rotate Left through Carry Ws	1	1	C, N, Z
RLNC	RLNC	f	f = Rotate Left (No Carry) f	1	1	N, Z
	RLNC	f,WREG	WREG = Rotate Left (No Carry) f	1	1	N, Z
	RLNC	Ws,Wd	Wd = Rotate Left (No Carry) Ws	1	1	N, Z
RRC	RRC	f	f = Rotate Right through Carry f	1	1	C, N, Z
	RRC	f,WREG	WREG = Rotate Right through Carry f	1	1	C, N, Z
	RRC	Ws,Wd	Wd = Rotate Right through Carry Ws	1	1	C, N, Z
RRNC	RRNC	f	f = Rotate Right (No Carry) f	1	1	N, Z
	RRNC	f,WREG	WREG = Rotate Right (No Carry) f	1	1	N, Z
	RRNC	Ws,Wd	Wd = Rotate Right (No Carry) Ws	1	1	N, Z
SE	SE	Ws,Wnd	Wnd = Sign-Extended Ws	1	1	C, N, Z
SETM	SETM	f	f = FFFFh	1	1	None
	SETM	WREG	WREG = FFFFh	1	1	None
	SETM	Ws	Ws = FFFFh	1	1	None
SL	SL	f	f = Left Shift f	1	1	C, N, OV, Z
	SL	f,WREG	WREG = Left Shift f	1	1	C, N, OV, Z
	SL	Ws,Wd	Wd = Left Shift Ws	1	1	C, N, OV, Z
	SL	Wb,Wns,Wnd	Wnd = Left Shift Wb by Wns	1	1	N, Z
	SL	Wb,#lit5,Wnd	Wnd = Left Shift Wb by lit5	1	1	N, Z
SUB	SUB	f	f = f – WREG	1	1	C, DC, N, OV, Z
	SUB	f,WREG	WREG = f – WREG	1	1	C, DC, N, OV, Z
	SUB	#lit10,Wn	Wn = Wn – lit10	1	1	C, DC, N, OV, Z
	SUB	Wb,Ws,Wd	Wd = Wb – Ws	1	1	C, DC, N, OV, Z
	SUB	Wb,#lit5,Wd	Wd = Wb – lit5	1	1	C, DC, N, OV, Z
SUBB	SUBB	f	f = f – WREG – ($\overline{C}$)	1	1	C, DC, N, OV, Z
	SUBB	f,WREG	WREG = f – WREG – ($\overline{C}$)	1	1	C, DC, N, OV, Z
	SUBB	#lit10,Wn	Wn = Wn – lit10 – ($\overline{C}$)	1	1	C, DC, N, OV, Z
	SUBB	Wb,Ws,Wd	Wd = Wb – Ws – ($\overline{C}$)	1	1	C, DC, N, OV, Z
	SUBB	Wb,#lit5,Wd	Wd = Wb – lit5 – ($\overline{C}$)	1	1	C, DC, N, OV, Z
SUBR	SUBR	f	f = WREG – f	1	1	C, DC, N, OV, Z
	SUBR	f,WREG	WREG = WREG – f	1	1	C, DC, N, OV, Z
	SUBR	Wb,Ws,Wd	Wd = Ws – Wb	1	1	C, DC, N, OV, Z
	SUBR	Wb,#lit5,Wd	Wd = lit5 – Wb	1	1	C, DC, N, OV, Z
SUBBR	SUBBR	f	f = WREG – f – ($\overline{C}$)	1	1	C, DC, N, OV, Z
	SUBBR	f,WREG	WREG = WREG – f – ($\overline{C}$)	1	1	C, DC, N, OV, Z
	SUBBR	Wb,Ws,Wd	Wd = Ws – Wb – ($\overline{C}$)	1	1	C, DC, N, OV, Z
	SUBBR	Wb,#lit5,Wd	Wd = lit5 – Wb – ($\overline{C}$)	1	1	C, DC, N, OV, Z
SWAP	SWAP.b	Wn	Wn = Nibble Swap Wn	1	1	None
	SWAP	Wn	Wn = Byte Swap Wn	1	1	None

30.2 Package Details

The following sections give the technical details of the packages.

28-Lead Plastic Quad Flat, No Lead Package (ML) – 6x6 mm Body [QFN] with 0.55 mm Contact Length

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

TOP VIEW

BOTTOM VIEW
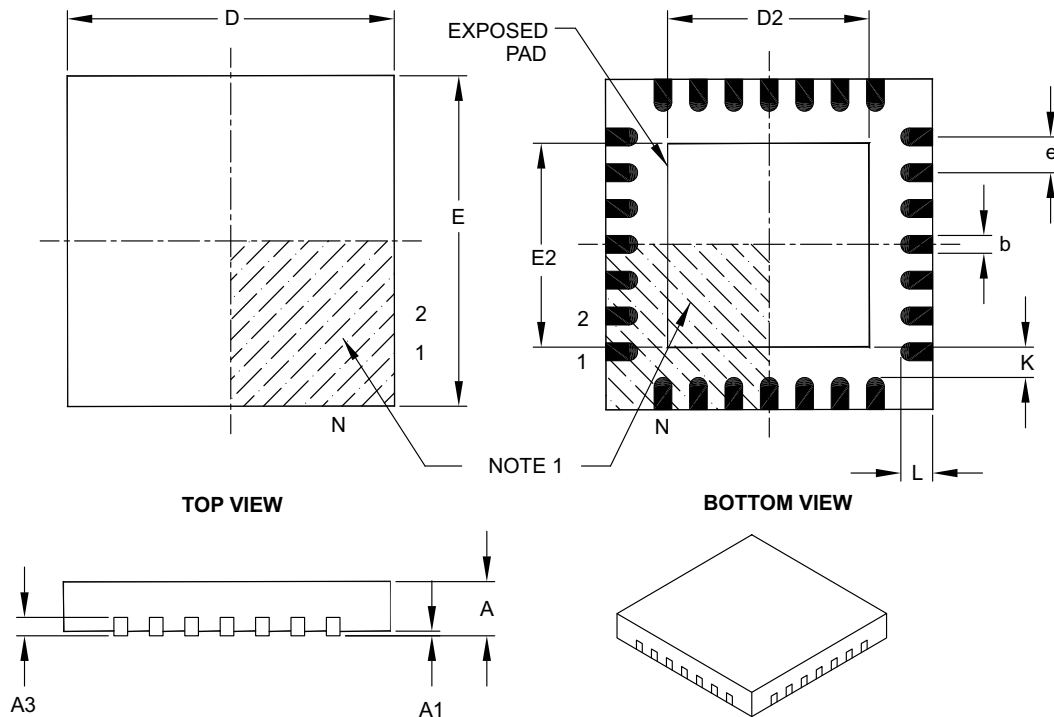
Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N		28	
Pitch	e		0.65 BSC	
Overall Height	A	0.80	0.90	1.00
Standoff	A1	0.00	0.02	0.05
Contact Thickness	A3		0.20 REF	
Overall Width	E		6.00 BSC	
Exposed Pad Width	E2	3.65	3.70	4.20
Overall Length	D		6.00 BSC	
Exposed Pad Length	D2	3.65	3.70	4.20
Contact Width	b	0.23	0.30	0.35
Contact Length	L	0.50	0.55	0.70
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 C04-105B

28-Lead Skinny Plastic Dual In-Line (SP) – 300 mil Body [SPDIP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging
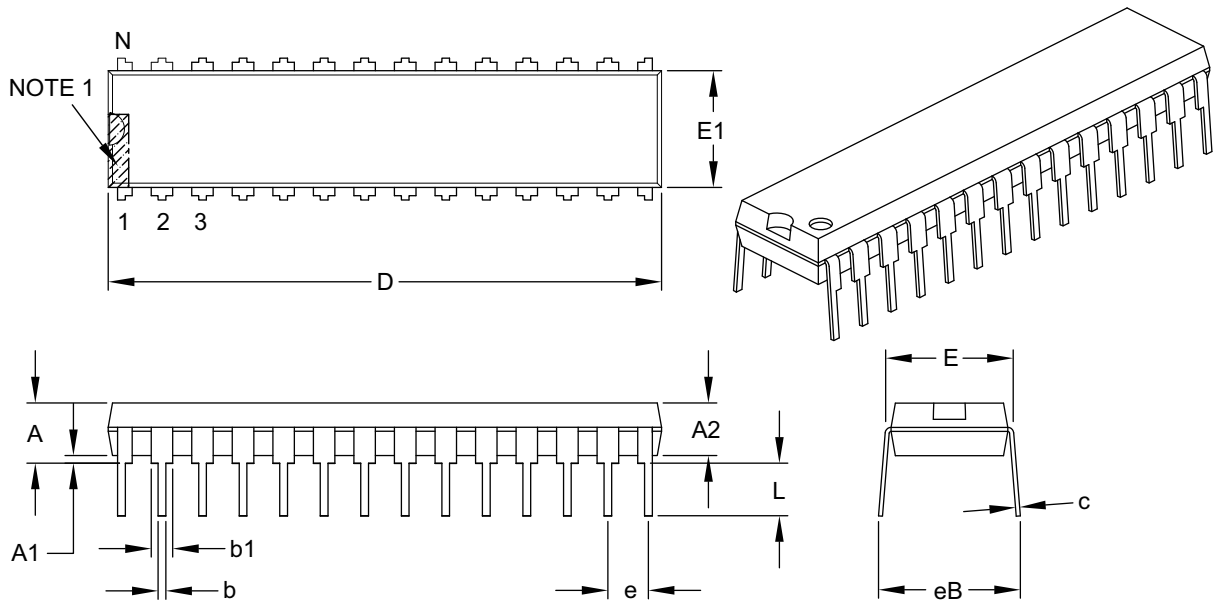
	Units		INCHES	
Dimension Limits		MIN	NOM	MAX
Number of Pins	N		28	
Pitch	e		.100 BSC	
Top to Seating Plane	A	–	–	.200
Molded Package Thickness	A2	.120	.135	.150
Base to Seating Plane	A1	.015	–	–
Shoulder to Shoulder Width	E	.290	.310	.335
Molded Package Width	E1	.240	.285	.295
Overall Length	D	1.345	1.365	1.400
Tip to Seating Plane	L	.110	.130	.150
Lead Thickness	c	.008	.010	.015
Upper Lead Width	b1	.040	.050	.070
Lower Lead Width	b	.014	.018	.022
Overall Row Spacing §	eB	–	–	.430

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. § Significant Characteristic.
3. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" per side.
4. Dimensioning and tolerancing per ASME Y14.5M.
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-070B

PRODUCT IDENTIFICATION SYSTEM

To order or obtain information, e.g., on pricing or delivery, refer to the factory or the listed sales office.

PIC 24 FJ 64 GB0 04 T - I / PT - XXX

Microchip Trademark
Architecture
Flash Memory Family
Program Memory Size (KB)
Product Group
Pin Count
Tape and Reel Flag (if applicable)
Temperature Range
Package
Pattern

Examples:

a) PIC24FJ64GB004-I/PT:
PIC24F device with USB On-The-Go, 64-Kbyte program memory, 44-pin, Industrial temp.,TQFP package.

b) PIC24FJ32GB002-I/ML:
PIC24F device with USB On-The-Go, 32-Kbyte program memory, 28-pin, Industrial temp.,QFN package.

Architecture	24	= 16-bit modified Harvard without DSP
Flash Memory Family	FJ	= Flash program memory
Product Group	GB0 =	General purpose microcontrollers with USB On-The-Go
Pin Count	02	= 28-pin
	04	= 44-pin
Temperature Range	I	= -40°C to +85°C (Industrial)
	E	= -40°C to +125°C (Extended)
Package	ML	= 28-lead (6x6 mm) or 44-lead (8x8 mm) QFN (Quad Flat)
	PT	= 44-lead (10x10x1 mm) TQFP (Thin Quad Flatpack)
	SO	= 28-lead 7.50 mm wide) SOIC (Small Outline)
	SP	= 28-lead (300 mil) SPDIP (Skinny Plastic Dual In-Line)
	SS	= 28-lead (530 mm) SSOP (Plastic Shrink Small)
Pattern		Three-digit QTP, SQTP, Code or Special Requirements (blank otherwise)
	ES	= Engineering Sample