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
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	18
Program Memory Size	8KB (2.75K x 24)
Program Memory Type	FLASH
EEPROM Size	512 x 8
RAM Size	1.5K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 3.6V
Data Converters	A/D 9x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Through Hole
Package / Case	20-DIP (0.300", 7.62mm)
Supplier Device Package	20-PDIP
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic24f08ka101-i-p

2.6 External Oscillator Pins

Many microcontrollers have options for at least two oscillators: a high-frequency primary oscillator and a low-frequency secondary oscillator (refer to **Section 9.0 "Oscillator Configuration"** for details).

The oscillator circuit should be placed on the same side of the board as the device. Place the oscillator circuit close to the respective oscillator pins with no more than 0.5 inch (12 mm) between the circuit components and the pins. The load capacitors should be placed next to the oscillator itself, on the same side of the board.

Use a grounded copper pour around the oscillator circuit to isolate it from surrounding circuits. The grounded copper pour should be routed directly to the MCU ground. Do not run any signal traces or power traces inside the ground pour. Also, if using a two-sided board, avoid any traces on the other side of the board where the crystal is placed.

Layout suggestions are shown in Figure 2-5. In-line packages may be handled with a single-sided layout that completely encompasses the oscillator pins. With fine-pitch packages, it is not always possible to completely surround the pins and components. A suitable solution is to tie the broken guard sections to a mirrored ground layer. In all cases, the guard trace(s) must be returned to ground.

In planning the application's routing and I/O assignments, ensure that adjacent port pins and other signals, in close proximity to the oscillator, are benign (i.e., free of high frequencies, short rise and fall times, and other similar noise).

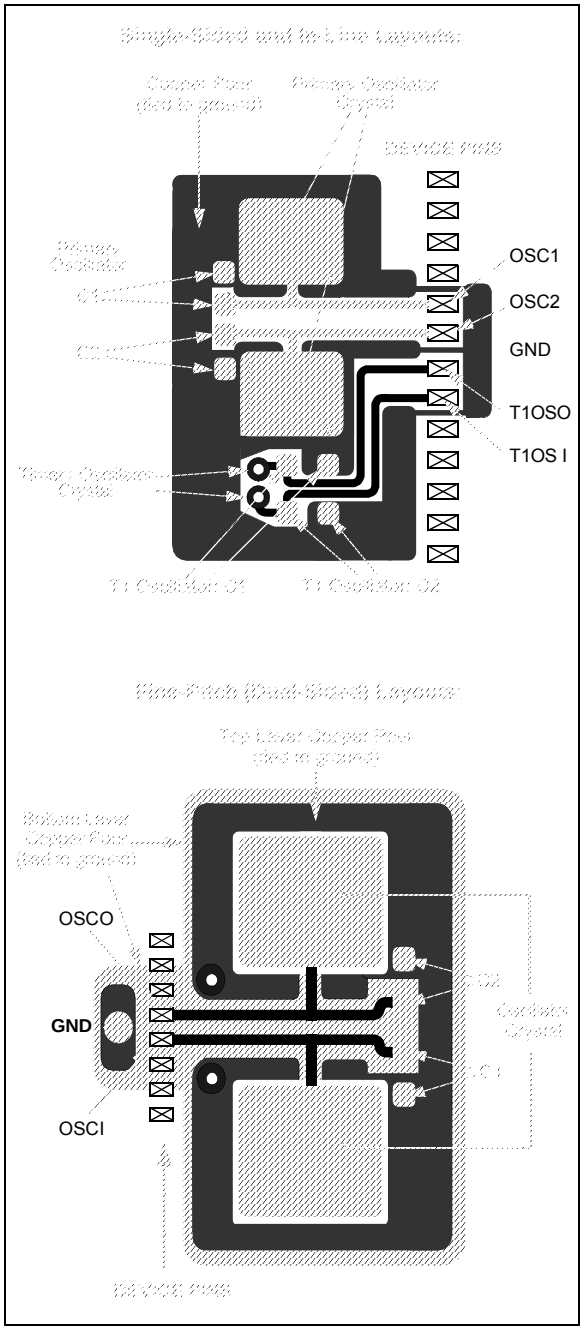
For additional information and design guidance on oscillator circuits, please refer to these Microchip Application Notes, available at the corporate web site (www.microchip.com):

- *AN826, "Crystal Oscillator Basics and Crystal Selection for rfPIC™ and PICmicro® Devices"*
- *AN849, "Basic PICmicro® Oscillator Design"*
- *AN943, "Practical PICmicro® Oscillator Analysis and Design"*
- *AN949, "Making Your Oscillator Work"*

2.7 Unused I/Os

Unused I/O pins should be configured as outputs and driven to a logic low state. Alternatively, connect a 1 kΩ to 10 kΩ resistor to VSS on unused pins and drive the output to logic low.

FIGURE 2-5: SUGGESTED PLACEMENT OF THE OSCILLATOR CIRCUIT



3.3.2 DIVIDER

The divide block supports 32-bit/16-bit and 16-bit/16-bit signed and unsigned integer divide operations with the following data sizes:

1. 32-bit signed/16-bit signed divide
2. 32-bit unsigned/16-bit unsigned divide
3. 16-bit signed/16-bit signed divide
4. 16-bit unsigned/16-bit unsigned divide

The quotient for all divide instructions ends up in W0 and the remainder in W1. Sixteen-bit signed and unsigned DIV instructions can specify any W register for both the 16-bit divisor (Wn), and any W register (aligned) pair (W(m + 1):Wm) for the 32-bit dividend. The divide algorithm takes one cycle per bit of divisor, so both 32-bit/16-bit and 16-bit/16-bit instructions take the same number of cycles to execute.

3.3.3 MULTI-BIT SHIFT SUPPORT

The PIC24F ALU supports both single bit and single-cycle, multi-bit arithmetic and logic shifts. Multi-bit shifts are implemented using a shifter block, capable of performing up to a 15-bit arithmetic right shift, or up to a 15-bit left shift, in a single cycle. All multi-bit shift instructions only support Register Direct Addressing for both the operand source and result destination.

A full summary of instructions that use the shift operation is provided below in Table 3-2.

TABLE 3-2: INSTRUCTIONS THAT USE THE SINGLE AND MULTI-BIT SHIFT OPERATION

Instruction	Description
ASR	Arithmetic shift right source register by one or more bits.
SL	Shift left source register by one or more bits.
LSR	Logical shift right source register by one or more bits.

TABLE 4-15: A/D 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
ADC1BUF0	0300	A/D Data Buffer 0																xxxx
ADC1BUF1	0302	A/D Data Buffer 1																xxxx
ADC1BUF2	0304	A/D Data Buffer 2																xxxx
ADC1BUF3	0306	A/D Data Buffer 3																xxxx
ADC1BUF4	0308	A/D Data Buffer 4																xxxx
ADC1BUF5	030A	A/D Data Buffer 5																xxxx
ADC1BUF6	030C	A/D Data Buffer 6																xxxx
ADC1BUF7	030E	A/D Data Buffer 7																xxxx
ADC1BUF8	0310	A/D Data Buffer 8																xxxx
ADC1BUF9	0312	A/D Data Buffer 9																xxxx
ADC1BUFA	0314	A/D Data Buffer 10																xxxx
ADC1BUFB	0316	A/D Data Buffer 11																xxxx
ADC1BUFC	0318	A/D Data Buffer 12																xxxx
ADC1BUFD	031A	A/D Data Buffer 13																xxxx
ADC1BUFE	031C	A/D Data Buffer 14																xxxx
ADC1BUFF	031E	A/D Data Buffer 15																xxxx
AD1CON1	0320	ADON	—	ADSIDL	—	—	—	FORM1	FORM0	SSRC2	SSRC1	SSRC0	—	—	ASAM	SAMP	DONE	0000
AD1CON2	0322	VCFG2	VCFG1	VCFG0	OFFCAL	—	CSCNA	—	—	BUFS	—	SMPI3	SMPI2	SMPI1	SMPI0	BUFM	ALTS	0000
AD1CON3	0324	ADRC	—	—	SAMC4	SAMC3	SAMC2	SAMC1	SAMC0	—	—	ADCS5	ADCS4	ADCS3	ADCS2	ADCS1	ADCS0	0000
AD1CHS	0328	CH0NB	—	—	—	CH0SB3	CH0SB2	CH0SB1	CH0SB0	CH0NA	—	—	CH0SA4	CH0SA3	CH0SA2	CH0SA1	CH0SA0	0000
AD1PCFG	032C	—	—	—	PCFG12	PCFG11	PCFG10	—	—	—	—	PCFG5	PCFG4	PCFG3	PCFG2	PCFG1	PCFG0	0000
AD1CSSL	0330	—	—	—	CSSL12	CSSL11	CSSL10	—	—	—	—	CSSL5	CSSL4	CSSL3	CSSL2	CSSL1	CSSL0	0000

Legend: — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

TABLE 4-16: CTMU 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
CTMUCON	033C	CTMUEN	—	CTMUSIDL	TGEN	EDGEN	EDGSEQEN	IDISSEN	CTTRIG	EDG2POL	EDG2SEL1	EDG2SEL0	EDG1POL	EDG1SEL1	EDG1SEL0	EDG2STAT	EDG1STAT	0000
CTMUICON	033E	ITRIM5	ITRIM4	ITRIM3	ITRIM2	ITRIM1	ITRIM0	IRNG1	IRNG0	—	—	—	—	—	—	—	—	0000

Legend: — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

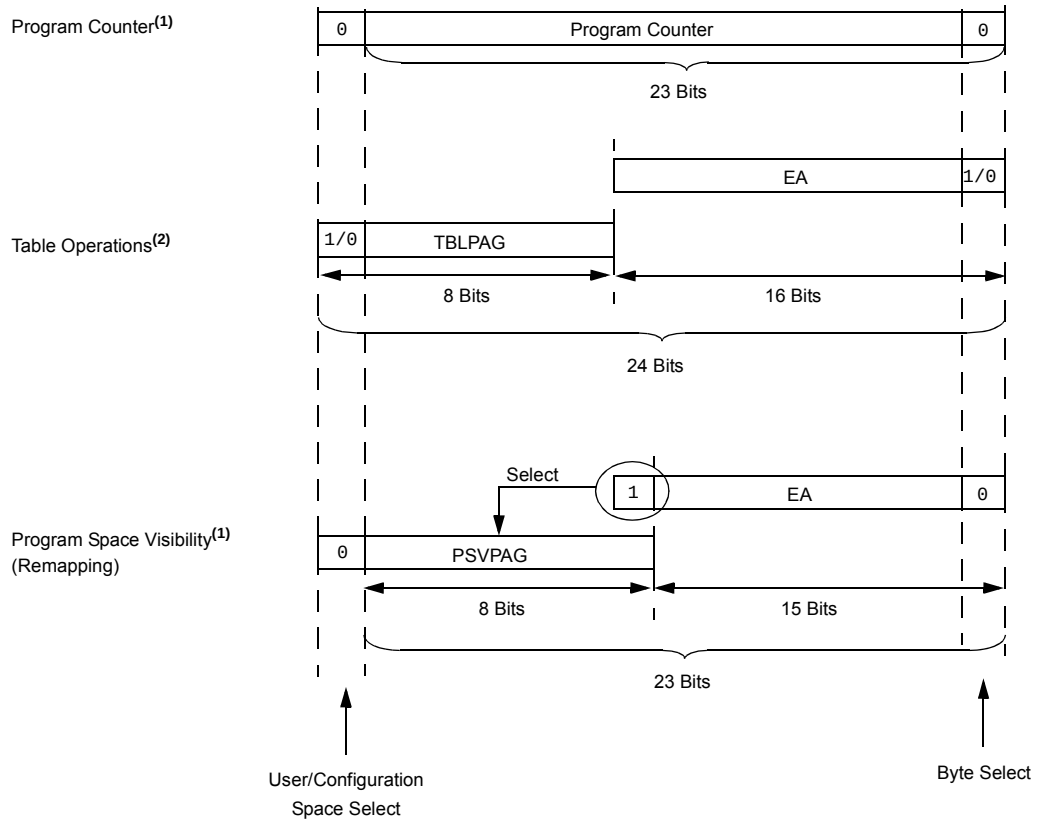
TABLE 4-24: PROGRAM SPACE ADDRESS CONSTRUCTION

Access Type	Access Space	Program Space Address				
		<23>	<22:16>	<15>	<14:1>	<0>
Instruction Access (Code Execution)	User	0	PC<22:1>			0
		0xx xxxx xxxx xxxx xxxx xxx0				
TBLRD/TBLWT (Byte/Word Read/Write)	User	TBLPAG<7:0>		Data EA<15:0>		
		0xxx xxxx		xxxx xxxx xxxx xxxx		
	Configuration	TBLPAG<7:0>		Data EA<15:0>		
		1xxx xxxx		xxxx xxxx xxxx xxxx		
Program Space Visibility (Block Remap/Read)	User	0	PSVPAG<7:0>(2)		Data EA<14:0>(1)	
		0	xxxx xxxx		xxx xxxx xxxx xxxx	

Note 1: Data EA<15> is always '1' in this case, but is not used in calculating the program space address. Bit 15 of the address is PSVPAG<0>.

2: PSVPAG can have only two values ('00' to access program memory and FF to access data EEPROM) on the PIC24F16KA102 family.

FIGURE 4-5: DATA ACCESS FROM PROGRAM SPACE ADDRESS GENERATION



Note 1: The LSb of program space addresses is always fixed as '0' in order to maintain word alignment of data in the program and data spaces.

2: Table operations are not required to be word-aligned. Table read operations are permitted in the configuration memory space.

6.4.1.1 Data EEPROM Bulk Erase

To erase the entire data EEPROM (bulk erase), the address registers do not need to be configured because this operation affects the entire data EEPROM. The following sequence helps in performing bulk erase:

1. Configure NVMCON to Bulk Erase mode.
2. Clear NVMIF status bit and enable NVM interrupt (optional).
3. Write the key sequence to NVMKEY.
4. Set the WR bit to begin erase cycle.
5. Either poll the WR bit or wait for the NVM interrupt (NVMIF is set).

A typical bulk erase sequence is provided in Example 6-3.

6.4.2 SINGLE-WORD WRITE

To write a single word in the data EEPROM, the following sequence must be followed:

1. Erase one data EEPROM word (as mentioned in **Section 6.4.1 "Erase Data EEPROM"**) if the PGMONLY bit (NVMCON<12>) is set to '1'.
2. Write the data word into the data EEPROM latch.
3. Program the data word into the EEPROM:
   - Configure the NVMCON register to program one EEPROM word (NVMCON<5:0> = 0001xx).
   - Clear NVMIF status bit and enable NVM interrupt (optional).
   - Write the key sequence to NVMKEY.
   - Set the WR bit to begin erase cycle.
   - Either poll the WR bit or wait for the NVM interrupt (NVMIF is set).
   - To get cleared, wait until NVMIF is set.

A typical single-word write sequence is provided in Example 6-4.

EXAMPLE 6-3: DATA EEPROM BULK ERASE

```
// Set up NVMCON to bulk erase the data EEPROM
NVMCON = 0x4050;

// Disable Interrupts For 5 Instructions
asm volatile ("disi #5");

// Issue Unlock Sequence and Start Erase Cycle
__builtin_write_NVM();
```

EXAMPLE 6-4: SINGLE-WORD WRITE TO DATA EEPROM

```
int __attribute__ ((space(eedata))) eeData = 0x1234;  // Variable located in EEPROM,declared as a
                                                         global variable.
   int newData;                                       // New data to write to EEPROM
   unsigned int offset;

   // Set up NVMCON to erase one word of data EEPROM
   NVMCON = 0x4004;

   // Set up a pointer to the EEPROM location to be erased
   TBLPAG = __builtin_tblpage(&eeData);               // Initialize EE Data page pointer
   offset = __builtin_tbloffset(&eeData);             // Initizlize lower word of address
   __builtin_tblwtl(offset, newData);                 // Write EEPROM data to write latch

   asm volatile ("disi #5");                          // Disable Interrupts For 5 Instructions
   __builtin_write_NVM();                             // Issue Unlock Sequence & Start Write Cycle
```

REGISTER 8-12: IEC4: INTERRUPT ENABLE CONTROL REGISTER 4

U-0	U-0	R/W-0	U-0	U-0	U-0	U-0	R/W-0
—	—	CTMUIE	—	—	—	—	HLVDIE
bit 15							bit 8

U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	U-0
—	—	—	—	CRCIE	U2ERIE	U1ERIE	—
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 **CTMUIE:** CTMU Interrupt Enable bit
1 = Interrupt request is enabled
0 = Interrupt request is not enabled

bit 12-9 **Unimplemented:** Read as ‘0’

bit 8 **HLVDIE:** High/Low-Voltage Detect Interrupt Enable bit
1 = Interrupt request is enabled
0 = Interrupt request is not enabled

bit 7-4 **Unimplemented:** Read as ‘0’

bit 3 **CRCIE:** CRC Generator Interrupt Enable bit
1 = Interrupt request is enabled
0 = Interrupt request is not enabled

bit 2 **U2ERIE:** UART2 Error Interrupt Enable bit
1 = Interrupt request is enabled
0 = Interrupt request is not enabled

bit 1 **U1ERIE:** UART1 Error Interrupt Enable bit
1 = Interrupt request is enabled
0 = Interrupt request is not enabled

bit 0 **Unimplemented:** Read as ‘0’

REGISTER 8-18: IPC5: INTERRUPT PRIORITY CONTROL REGISTER 5

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

U-0	U-0	U-0	U-0	U-0	R/W-1	R/W-0	R/W-0
—	—	—	—	—	INT1IP2	INT1IP1	INT1IP0
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 15-3 **Unimplemented:** Read as ‘0’

bit 2-0 **INT1IP<2:0>:** External Interrupt 1 Priority bits

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

REGISTER 9-2: CLKDIV: CLOCK DIVIDER REGISTER

R/W-0	R/W-0	R/W-1	R/W-1	R/W-0	R/W-0	R/W-0	R/W-1
ROI	DOZE2	DOZE1	DOZE0	DOZEN(1)	RCDIV2	RCDIV1	RCDIV0
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 **ROI:** Recover on Interrupt bit
1 = Interrupts clear the DOZEN bit and reset the CPU and peripheral clock ratio to 1:1
0 = Interrupts have no effect on the DOZEN bit

bit 14-12 **DOZE<2:0>:** CPU and Peripheral Clock Ratio Select bits
111 = 1:128
110 = 1:64
101 = 1:32
100 = 1:16
011 = 1:8
010 = 1:4
001 = 1:2
000 = 1:1

bit 11 **DOZEN:** DOZE Enable bit(1)
1 = DOZE<2:0> bits specify the CPU and peripheral clock ratio
0 = CPU and peripheral clock ratio are set to 1:1

bit 10-8 **RCDIV<2:0>:** FRC Postscaler Select bits
When OSCCON (COSC<2:0>) = 111:
111 = 31.25 kHz (divide by 256)
110 = 125 kHz (divide by 64)
101 = 250 kHz (divide by 32)
100 = 500 kHz (divide by 16)
011 = 1 MHz (divide by 8)
010 = 2 MHz (divide by 4)
001 = 4 MHz (divide by 2) (default)
000 = 8 MHz (divide by 1)
When OSCCON (COSC<2:0>) = 110:
111 = 1.95 kHz (divide by 256)
110 = 7.81 kHz (divide by 64)
101 = 15.62 kHz (divide by 32)
100 = 31.25 kHz (divide by 16)
011 = 62.5 kHz (divide by 8)
010 = 125 kHz (divide by 4)
001 = 250 kHz (divide by 2) (default)
000 = 500 kHz (divide by 1)

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

Note 1: This bit is automatically cleared when the ROI bit is set and an interrupt occurs.

14.0 INPUT CAPTURE

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 on Input Capture, refer to the *"PIC24F Family Reference Manual"*, **Section 15. "Input Capture"** (DS39701).

The input capture module is used to capture a timer value from one of two selectable time bases upon an event on an input pin.

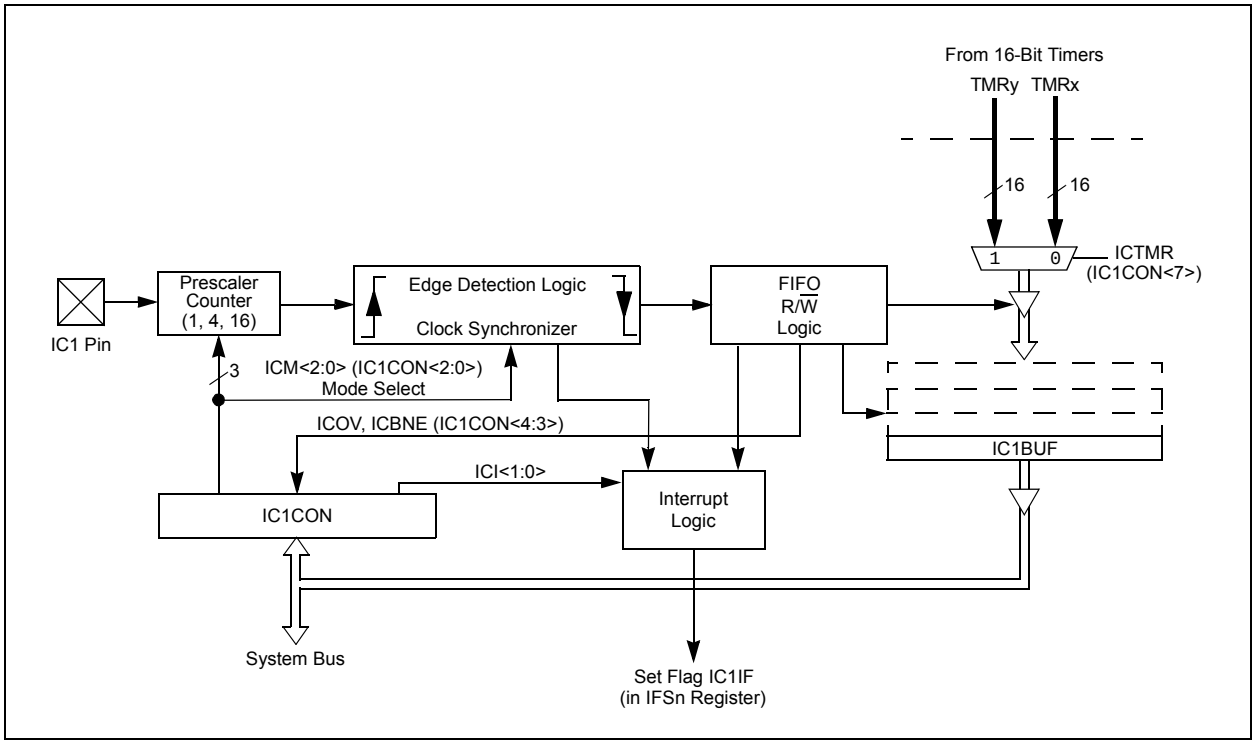
The input capture features are quite useful in applications requiring frequency (Time Period) and pulse measurement. Figure 14-1 depicts a simplified block diagram of the input capture module.

The PIC24F16KA102 family devices have one input capture channel. The input capture module has multiple operating modes, which are selected via the IC1CON register. The operating modes include:

- Capture timer value on every falling edge of input applied at the IC1 pin
- Capture timer value on every rising edge of input applied at the IC1 pin
- Capture timer value on every 4th rising edge of input applied at the IC1 pin
- Capture timer value on every 16th rising edge of input applied at the IC1 pin
- Capture timer value on every rising and every falling edge of input applied at the IC1 pin
- Device wake-up from capture pin during CPU Sleep and Idle modes

The input capture module has a four-level FIFO buffer. The number of capture events required to generate a CPU interrupt can be selected by the user.

FIGURE 14-1: INPUT CAPTURE BLOCK DIAGRAM



REGISTER 17-1: I2C1CON: I2C1 CONTROL REGISTER (CONTINUED)

bit 5 **ACKDT:** Acknowledge Data bit (when operating as I^2C master; applicable during master receive)
Value that will be transmitted when the software initiates an Acknowledge sequence.
1 = Sends NACK during Acknowledge
0 = Sends ACK during Acknowledge

bit 4 **ACKEN:** Acknowledge Sequence Enable bit
(when operating as I^2C master; applicable during master receive)
1 = Initiates Acknowledge sequence on SDA1 and SCL1 pins and transmits ACKDT data bit; hardware is clear at the end of the master Acknowledge sequence
0 = Acknowledge sequence is not in progress

bit 3 **RCEN:** Receive Enable bit (when operating as I^2C master)
1 = Enables Receive mode for I^2C; hardware is clear at the end of eighth bit of master receive data byte
0 = Receive sequence not in progress

bit 2 **PEN:** Stop Condition Enable bit (when operating as I^2C master)
1 = Initiates Stop condition on SDA1 and SCL1 pins; hardware is clear at end of master Stop sequence
0 = Stop condition is not in progress

bit 1 **RSEN:** Repeated Start Condition Enable bit (when operating as I^2C master)
1 = Initiates Repeated Start condition on SDA1 and SCL1 pins; hardware is clear at end of master Repeated Start sequence
0 = Repeated Start condition is not in progress

bit 0 **SEN:** Start Condition Enable bit (when operating as I^2C master)
1 = Initiates Start condition on SDA1 and SCL1 pins; hardware is clear at end of master Start sequence
0 = Start condition is not in progress

18.0 UNIVERSAL ASYNCHRONOUS RECEIVER TRANSMITTER (UART)

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 on the Universal Asynchronous Receiver Transmitter, refer to the *"PIC24F Family Reference Manual"*, **Section 21. "UART"** (DS39708).

The Universal Asynchronous Receiver Transmitter (UART) module is one of the serial I/O modules available in this PIC24F device family. The UART is a full-duplex asynchronous system that can communicate with peripheral devices, such as personal computers, LIN, RS-232 and RS-485 interfaces. This module also supports a hardware flow control option with the $\overline{\text{UxCTS}}$ and $\overline{\text{UxRTS}}$ pins, and also includes an IrDA® encoder and decoder.

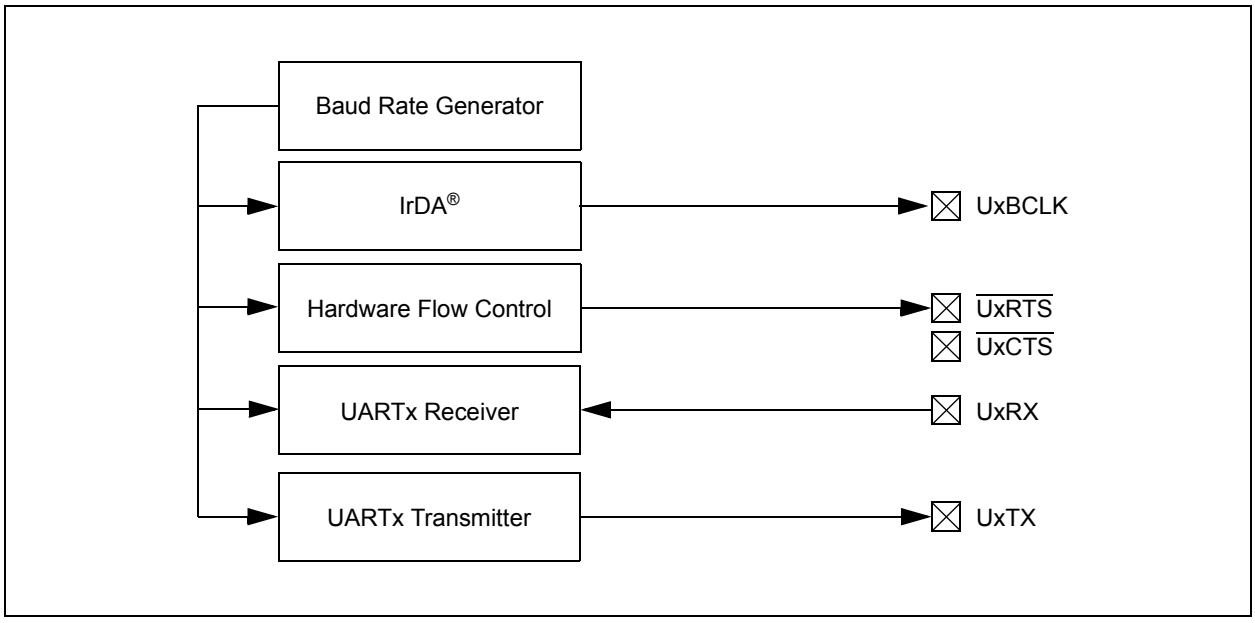
The primary features of the UART module are:

- Full-Duplex, 8-Bit or 9-Bit Data Transmission through the UxTX and UxRX Pins
- Even, Odd or No Parity Options (for 8-bit data)
- One or Two Stop bits
- Hardware Flow Control Option with $\overline{\text{UxCTS}}$ and $\overline{\text{UxRTS}}$ pins
- Fully Integrated Baud Rate Generator (IBRG) with 16-Bit Prescaler
- Baud Rates Ranging from 1 Mbps to 15 bps at 16 MIPS
- 4-Deep, First-In-First-Out (FIFO) Transmit Data Buffer
- 4-Deep FIFO Receive Data Buffer
- Parity, Framing and Buffer Overrun Error Detection
- Support for 9-Bit mode with Address Detect (9th bit = 1)
- Transmit and Receive Interrupts
- Loopback mode for Diagnostic Support
- Support for Sync and Break Characters
- Supports Automatic Baud Rate Detection
- IrDA Encoder and Decoder Logic
- 16x Baud Clock Output for IrDA Support

A simplified block diagram of the UART is displayed in Figure 18-1. The UART module consists of these important hardware elements:

- Baud Rate Generator
- Asynchronous Transmitter
- Asynchronous Receiver

FIGURE 18-1: UART SIMPLIFIED BLOCK DIAGRAM



18.1 UART Baud Rate Generator (BRG)

The UART module includes a dedicated 16-bit Baud Rate Generator (BRG). The UxBRG register controls the period of a free-running, 16-bit timer. Equation 18-1 provides the formula for computation of the baud rate with BRGH = 0.

EQUATION 18-1: UART BAUD RATE WITH BRGH = 0(1)

$$\text{Baud Rate} = \frac{FCY}{16 \bullet (UxBRG + 1)}$$

$$UxBRG = \frac{FCY}{16 \bullet \text{Baud Rate}} - 1$$

Note 1: Based on FCY = FOSC/2; Doze mode and PLL are disabled.

Example 18-1 provides the calculation of the baud rate error for the following conditions:

- FCY = 4 MHz
- Desired Baud Rate = 9600

The maximum baud rate (BRGH = 0) possible is FCY/16 (for UxBRG = 0) and the minimum baud rate possible is FCY/(16 * 65536).

Equation 18-2 provides the formula for computation of the baud rate with BRGH = 1.

EQUATION 18-2: UART BAUD RATE WITH BRGH = 1(1)

$$\text{Baud Rate} = \frac{FCY}{4 \bullet (UxBRG + 1)}$$

$$UxBRG = \frac{FCY}{4 \bullet \text{Baud Rate}} - 1$$

Note 1: Based on FCY = FOSC/2; Doze mode and PLL are disabled.

The maximum baud rate (BRGH = 1) possible is FCY/4 (for UxBRG = 0) and the minimum baud rate possible is FCY/(4 * 65536).

Writing a new value to the UxBRG register causes the BRG timer to be reset (cleared). This ensures the BRG does not wait for a timer overflow before generating the new baud rate.

EXAMPLE 18-1: BAUD RATE ERROR CALCULATION (BRGH = 0)(1)

Desired Baud Rate = FCY/(16 (UxBRG + 1))

Solving for UxBRG value:

UxBRG = ((FCY/Desired Baud Rate)/16) – 1
UxBRG = ((4000000/9600)/16) – 1
UxBRG = 25

Calculated Baud Rate = 4000000/(16 (25 + 1))
= 9615

Error = (Calculated Baud Rate – Desired Baud Rate) / Desired Baud Rate
= (9615 – 9600)/9600
= 0.16%

Note 1: Based on FCY = FOSC/2; Doze mode and PLL are disabled.

REGISTER 18-1: UxMODE: UARTx MODE REGISTER (CONTINUED)

bit 3 **BRGH:** High Baud Rate Enable bit
1 = BRG generates 4 clocks per bit period (4x baud clock, High-Speed mode)
0 = BRG generates 16 clocks per bit period (16x baud clock, Standard mode)

bit 2-1 **PDSEL<1:0>:** Parity and Data Selection bits
11 = 9-bit data, no parity
10 = 8-bit data, odd parity
01 = 8-bit data, even parity
00 = 8-bit data, no parity

bit 0 **STSEL:** Stop Bit Selection bit
1 = Two Stop bits
0 = One Stop bit

Note 1: This feature is only available for the 16x BRG mode (BRGH = 0).
2: Bit availability depends on pin availability.

REGISTER 18-3: UxTXREG: UARTx TRANSMIT REGISTER

U-x	U-x	U-x	U-x	U-x	U-x	U-x	W-x
—	—	—	—	—	—	—	UTX8
bit 15							bit 8

W-x	W-x	W-x	W-x	W-x	W-x	W-x	W-x
UTX7	UTX6	UTX5	UTX4	UTX3	UTX2	UTX1	UTX0
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 15-9 **Unimplemented:** Read as ‘0’

bit 8 **UTX8:** Data of the Transmitted Character bit (in 9-bit mode)

bit 7-0 **UTX<7:0>:** Data of the Transmitted Character bits

REGISTER 18-4: UxRXREG: UARTx RECEIVE REGISTER

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R-0, HSC
—	—	—	—	—	—	—	URX8
bit 15							bit 8

R-0, HSC	R-0, HSC	R-0, HSC	R-0, HSC	R-0, HSC	R-0, HSC	R-0, HSC	R-0, HSC
URX7	URX6	URX5	URX4	URX3	URX2	URX1	URX0
bit 7							bit 0

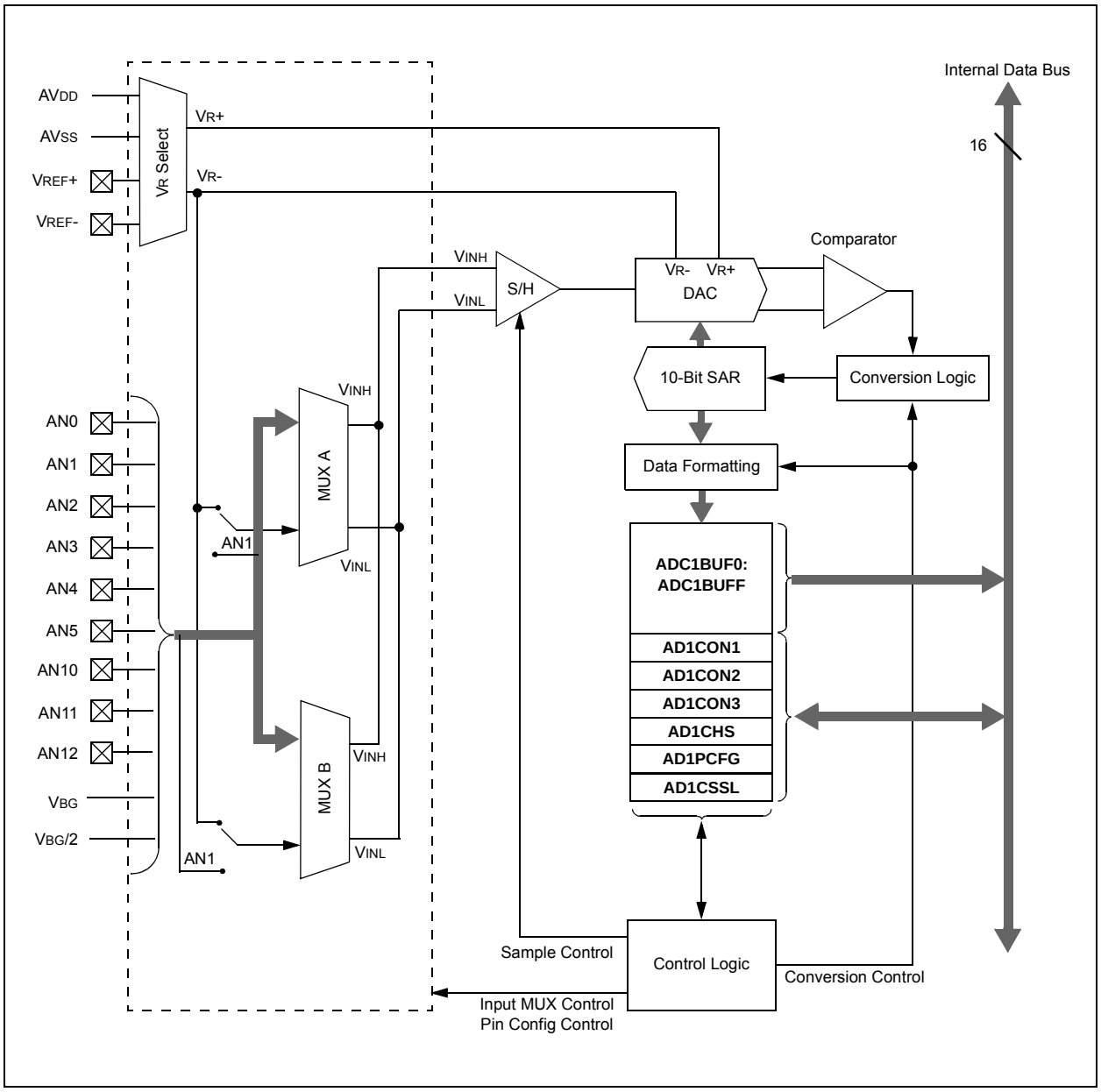
Legend:	HSC = Hardware Settable/Clearable bit		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 15-9 **Unimplemented:** Read as ‘0’

bit 8 **URX8:** Data of the Received Character bit (in 9-bit mode)

bit 7-0 **URX<7:0>:** Data of the Received Character bits

FIGURE 22-1: 10-BIT HIGH-SPEED A/D CONVERTER BLOCK DIAGRAM



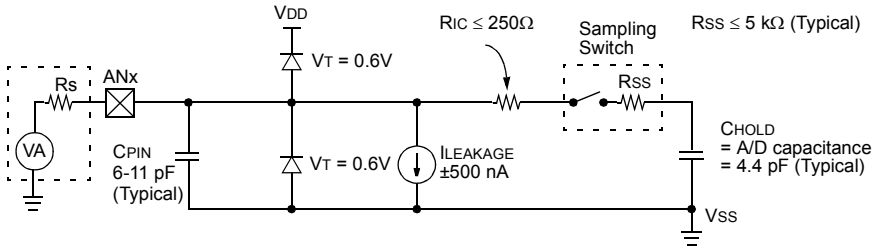
EQUATION 22-1: A/D CONVERSION CLOCK PERIOD(1)

$$ADCS = \frac{TAD}{TCY} - 1$$

$$TAD = TCY \bullet (ADCS + 1)$$

Note 1: Based on TCY = 2 * TOSC; Doze mode and PLL are disabled.

FIGURE 22-2: 10-BIT A/D CONVERTER ANALOG INPUT MODEL



Legend:		
	CPIN	= Input Capacitance
	VT	= Threshold Voltage
	ILEAKAGE	= Leakage Current at the pin due to various junctions
	RIC	= Interconnect Resistance
	RSS	= Sampling Switch Resistance
	CHOLD	= Sample/Hold Capacitance (from A/D)

Note: CPIN value depends on device package and is not tested. Effect of CPIN negligible if Rs ≤ 5 kΩ.

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 on the Comparator Voltage Reference, refer to the *"PIC24F Family Reference Manual"*, **Section 20. "Comparator Voltage Reference Module"** (DS39709).

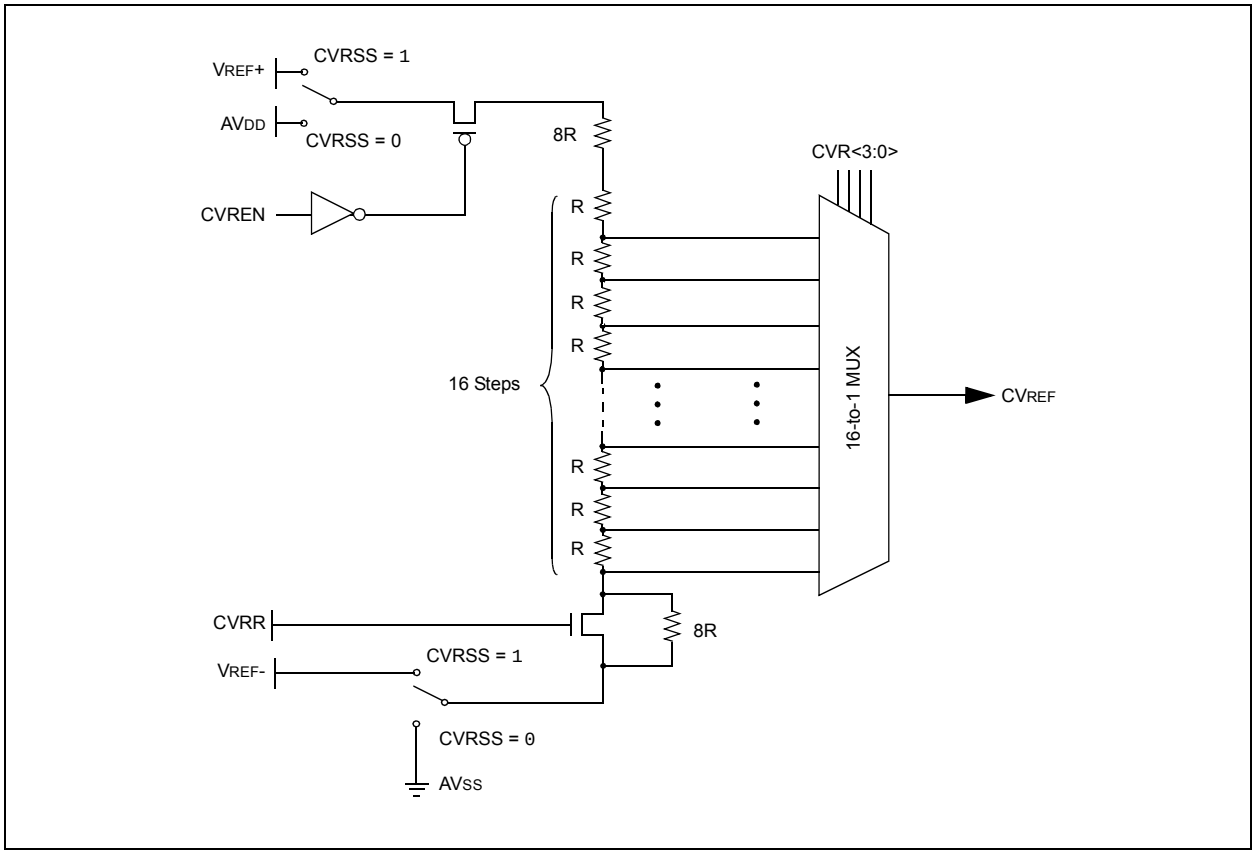
24.1 Configuring the Comparator Voltage Reference

The comparator 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



27.11 PICkit 2 Development Programmer/Debugger and PICkit 2 Debug Express

The PICkit™ 2 Development Programmer/Debugger is a low-cost development tool with an easy to use interface for programming and debugging Microchip's Flash families of microcontrollers. The full featured Windows® programming interface supports baseline (PIC10F, PIC12F5xx, PIC16F5xx), midrange (PIC12F6xx, PIC16F), PIC18F, PIC24, dsPIC30, dsPIC33, and PIC32 families of 8-bit, 16-bit, and 32-bit microcontrollers, and many Microchip Serial EEPROM products. With Microchip's powerful MPLAB Integrated Development Environment (IDE) the PICkit™ 2 enables in-circuit debugging on most PIC® microcontrollers. In-Circuit-Debugging runs, halts and single steps the program while the PIC microcontroller is embedded in the application. When halted at a breakpoint, the file registers can be examined and modified.

The PICkit 2 Debug Express include the PICkit 2, demo board and microcontroller, hookup cables and CDROM with user's guide, lessons, tutorial, compiler and MPLAB IDE software.

27.12 MPLAB PM3 Device Programmer

The MPLAB PM3 Device Programmer is a universal, CE compliant device programmer with programmable voltage verification at V_{DDMIN} and V_{DDMAX} for maximum reliability. It features a large LCD display (128 x 64) for menus and error messages and a modular, detachable socket assembly to support various package types. The ICSP™ cable assembly is included as a standard item. In Stand-Alone mode, the MPLAB PM3 Device Programmer can read, verify and program PIC devices without a PC connection. It can also set code protection in this mode. The MPLAB PM3 connects to the host PC via an RS-232 or USB cable. The MPLAB PM3 has high-speed communications and optimized algorithms for quick programming of large memory devices and incorporates an MMC card for file storage and data applications.

27.13 Demonstration/Development Boards, Evaluation Kits, and Starter Kits

A wide variety of demonstration, development and evaluation boards for various PIC MCUs and dsPIC DSCs allows quick application development on fully functional systems. Most boards include prototyping areas for adding custom circuitry and provide application firmware and source code for examination and modification.

The boards support a variety of features, including LEDs, temperature sensors, switches, speakers, RS-232 interfaces, LCD displays, potentiometers and additional EEPROM memory.

The demonstration and development boards can be used in teaching environments, for prototyping custom circuits and for learning about various microcontroller applications.

In addition to the PICDEM™ and dsPICDEM™ demonstration/development board series of circuits, Microchip has a line of evaluation kits and demonstration software for analog filter design, KEELOQ® security ICs, CAN, IrDA®, PowerSmart battery management, SEEVAL® evaluation system, Sigma-Delta ADC, flow rate sensing, plus many more.

Also available are starter kits that contain everything needed to experience the specified device. This usually includes a single application and debug capability, all on one board.

Check the Microchip web page (www.microchip.com) for the complete list of demonstration, development and evaluation kits.

TABLE 28-2: INSTRUCTION SET OVERVIEW (CONTINUED)

Assembly Mnemonic	Assembly Syntax		Description	# of Words	# of Cycles	Status Flags Affected
BTSS	BTSS	f,#bit4	Bit Test f, Skip if Set	1	1 (2 or 3)	None
	BTSS	Ws,#bit4	Bit Test Ws, Skip if Set	1	1 (2 or 3)	None
BTST	BTST	f,#bit4	Bit Test f	1	1	Z
	BTST.C	Ws,#bit4	Bit Test Ws to C	1	1	C
	BTST.Z	Ws,#bit4	Bit Test Ws to Z	1	1	Z
	BTST.C	Ws,Wb	Bit Test Ws<Wb> to C	1	1	C
	BTST.Z	Ws,Wb	Bit Test Ws<Wb> to Z	1	1	Z
BTSTS	BTSTS	f,#bit4	Bit Test then Set f	1	1	Z
	BTSTS.C	Ws,#bit4	Bit Test Ws to C, then Set	1	1	C
	BTSTS.Z	Ws,#bit4	Bit Test Ws to Z, then Set	1	1	Z
CALL	CALL	lit23	Call Subroutine	2	2	None
	CALL	Wn	Call Indirect Subroutine	1	2	None
CLR	CLR	f	f = 0x0000	1	1	None
	CLR	WREG	WREG = 0x0000	1	1	None
	CLR	Ws	Ws = 0x0000	1	1	None
CLRWDT	CLRWDT		Clear Watchdog Timer	1	1	WDTO, Sleep
COM	COM	f	$f = \overline{f}$	1	1	N, Z
	COM	f,WREG	$WREG = \overline{f}$	1	1	N, Z
	COM	Ws,Wd	$Wd = \overline{Ws}$	1	1	N, Z
CP	CP	f	Compare f with WREG	1	1	C, DC, N, OV, Z
	CP	Wb,#lit5	Compare Wb with lit5	1	1	C, DC, N, OV, Z
	CP	Wb,Ws	Compare Wb with Ws (Wb – Ws)	1	1	C, DC, N, OV, Z
CP0	CP0	f	Compare f with 0x0000	1	1	C, DC, N, OV, Z
	CP0	Ws	Compare Ws with 0x0000	1	1	C, DC, N, OV, Z
CPB	CPB	f	Compare f with WREG, with Borrow	1	1	C, DC, N, OV, Z
	CPB	Wb,#lit5	Compare Wb with lit5, with Borrow	1	1	C, DC, N, OV, Z
	CPB	Wb,Ws	Compare Wb with Ws, with Borrow ($Wb – Ws – \overline{C}$)	1	1	C, DC, N, OV, Z
CPSEQ	CPSEQ	Wb,Wn	Compare Wb with Wn, Skip if =	1	1 (2 or 3)	None
CPSGT	CPSGT	Wb,Wn	Compare Wb with Wn, Skip if >	1	1 (2 or 3)	None
CPSLT	CPSLT	Wb,Wn	Compare Wb with Wn, Skip if <	1	1 (2 or 3)	None
CPSNE	CPSNE	Wb,Wn	Compare Wb with Wn, Skip if ≠	1	1 (2 or 3)	None
DAW	DAW	Wn	Wn = Decimal Adjust Wn	1	1	C
DEC	DEC	f	f = f –1	1	1	C, DC, N, OV, Z
	DEC	f,WREG	WREG = f –1	1	1	C, DC, N, OV, Z
	DEC	Ws,Wd	Wd = Ws – 1	1	1	C, DC, N, OV, Z
DEC2	DEC2	f	f = f – 2	1	1	C, DC, N, OV, Z
	DEC2	f,WREG	WREG = f – 2	1	1	C, DC, N, OV, Z
	DEC2	Ws,Wd	Wd = Ws – 2	1	1	C, DC, N, OV, Z
DISI	DISI	#lit14	Disable Interrupts for k Instruction Cycles	1	1	None
DIV	DIV.SW	Wm,Wn	Signed 16/16-bit Integer Divide	1	18	N, Z, C, OV
	DIV.SD	Wm,Wn	Signed 32/16-bit Integer Divide	1	18	N, Z, C, OV
	DIV.UW	Wm,Wn	Unsigned 16/16-bit Integer Divide	1	18	N, Z, C, OV
	DIV.UD	Wm,Wn	Unsigned 32/16-bit Integer Divide	1	18	N, Z, C, OV
EXCH	EXCH	Wns,Wnd	Swap Wns with Wnd	1	1	None
FF1L	FF1L	Ws,Wnd	Find First One from Left (MSb) Side	1	1	C
FF1R	FF1R	Ws,Wnd	Find First One from Right (LSb) Side	1	1	C

TABLE 29-8: DC CHARACTERISTICS: POWER-DOWN CURRENT (IPD)

DC CHARACTERISTICS			Standard Operating Conditions: 1.8V to 3.6V (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended		
Parameter No.	**Typical(1)**	**Max**	**Units**	**Conditions**	
Power-Down Current (IPD): PMD Bits are Set, PMSLP Bit is '0'(2)					
DC60	0.025	0.200	µA	-40°C	1.8V	Base Power-Down Current (Sleep)(3)
DC60a		0.200		+25°C		
DC60b		0.870		+60°C		
DC60c		1.350		+85°C		
DC60d		10.00		+125°C		
DC60e	0.105	0.540	µA	-40°C	3.3V	
DC60f		0.540		+25°C		
DC60g		1.680		+60°C		
DC60h		2.450		+85°C		
DC60i		11.00		+125°C		
DC70	0.020	0.150	µA	-40°C	1.8V	Base Deep Sleep Current
DC70a		0.150		+25°C		
DC70b		0.430		+60°C		
DC70c		0.630		+85°C		
DC70d		3.00		+125°C		
DC70e	0.035	0.300	µA	-40°C	3.3V	
DC70f		0.300		+25°C		
DC70g		0.700		+60°C		
DC70h		0.980		+85°C		
DC70i		5.00		+125°C		
DC61	0.55	0.65	µA	-40°C	1.8V	Watchdog Timer Current (WDT)(3,4)
DC61a		0.65		+25°C		
DC61b		0.65		+60°C		
DC61c		0.65		+85°C		
DC61d		1.20		+125°C		
DC61e	0.87	0.95	µA	-40°C	3.3V	
DC61f		0.95		+25°C		
DC61g		0.95		+60°C		
DC61h		0.95		+85°C		
DC61i		1.50		+125°C		

Note 1: Data in the "Typical" column is at 3.3V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

2: Base IPD is measured with all peripherals and clocks shut down. All I/Os are configured as outputs and set low. WDT, etc., are all switched off.

3: The Δ current is the additional current consumed when the module is enabled. This current should be added to the base IPD current.

4: Current applies to Sleep only.

5: Current applies to Sleep and Deep Sleep.

6: Current applies to Deep Sleep only.