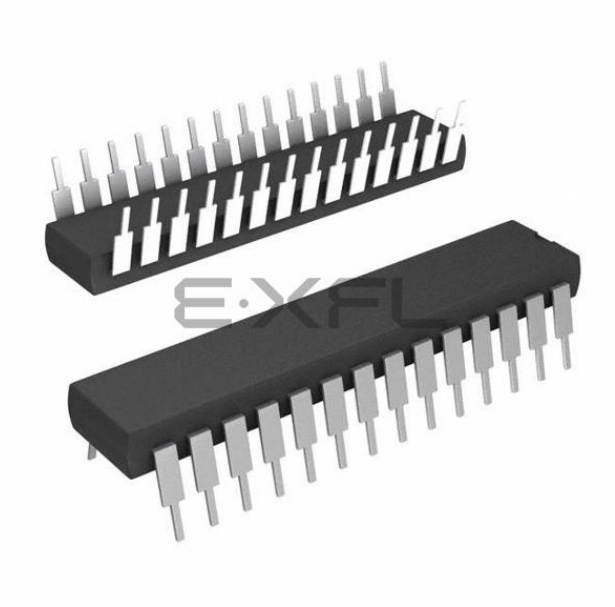


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

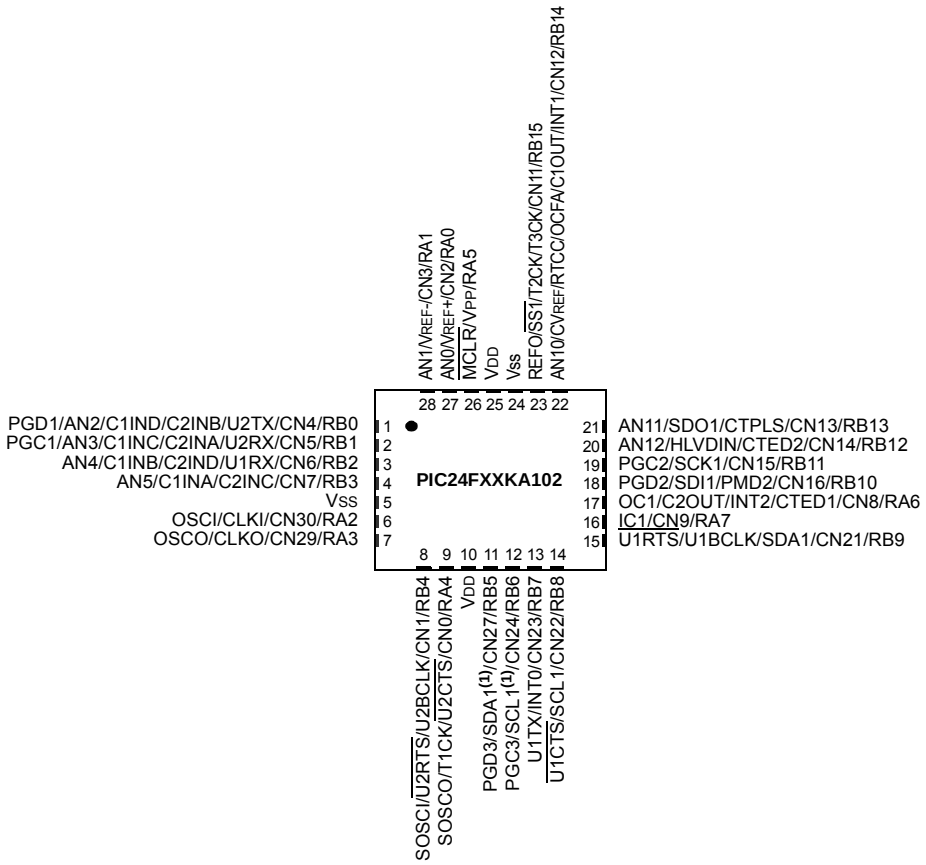
"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" 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	24
Program Memory Size	16KB (5.5K 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 ~ 125°C (TA)
Mounting Type	Through Hole
Package / Case	28-DIP (0.300", 7.62mm)
Supplier Device Package	28-SPDIP
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic24f16ka102-e-sp

Pin Diagrams (Continued)

28-Pin QFN(2,3)



Note 1: Alternative multiplexing for SDA1 and SCL1 when the I2CSEL Configuration bit is set.

2: The bottom pad of the QFN package should be connected to VSS.

3: All device pins have a maximum voltage of 3.6V and are not 5V tolerant.

TABLE 1-2: PIC24F16KA102 FAMILY PINOUT DESCRIPTIONS (CONTINUED)

Function	Pin Number				I/O	Input Buffer	Description
	20-Pin PDIP/SSOP/ SOIC	20-Pin QFN	28-Pin SPDIP/ SSOP/SOIC	28-Pin QFN			
PGC1	5	2	5	2	I/O	ST	In-Circuit Debugger and ICSP™ Programming Clock
PGD1	4	1	4	1	I/O	ST	In-Circuit Debugger and ICSP Programming Data
PGC2	2	19	22	19	I/O	ST	In-Circuit Debugger and ICSP Programming Clock
PGD2	3	20	21	18	I/O	ST	In-Circuit Debugger and ICSP Programming Data
PGC3	10	7	15	12	I/O	ST	In-Circuit Debugger and ICSP Programming Clock
PGD3	9	6	14	11	I/O	ST	In-Circuit Debugger and ICSP Programming Data
RA0	2	19	2	27	I/O	ST	PORTA Digital I/O
RA1	3	20	3	28	I/O	ST	
RA2	7	4	9	6	I/O	ST	
RA3	8	5	10	7	I/O	ST	
RA4	10	7	12	9	I/O	ST	
RA5	1	18	1	26	I/O	ST	
RA6	14	11	20	17	I/O	ST	
RA7	—	—	19	16	I/O	ST	
RB0	4	1	4	1	I/O	ST	PORTB Digital I/O
RB1	5	2	5	2	I/O	ST	
RB2	6	3	6	3	I/O	ST	
RB3	—	—	7	4	I/O	ST	
RB4	9	6	11	8	I/O	ST	
RB5	—	—	14	11	I/O	ST	
RB6	—	—	15	12	I/O	ST	
RB7	11	8	16	13	I/O	ST	
RB8	12	9	17	14	I/O	ST	
RB9	13	10	18	15	I/O	ST	
RB10	—	—	21	18	I/O	ST	
RB11	—	—	22	19	I/O	ST	
RB12	15	12	23	20	I/O	ST	
RB13	16	13	24	21	I/O	ST	
RB14	17	14	25	22	I/O	ST	
RB15	18	15	26	23	I/O	ST	
REFO	18	15	26	23	O	—	Reference Clock Output
RTCC	17	14	25	22	O	—	Real-Time Clock Alarm Output
SCK1	15	12	22	19	I/O	ST	SPI1 Serial Clock Input/Output
SCL1	12	9	17, 15(1)	14, 12 (1)	I/O	I^2C	I2C1 Synchronous Serial Clock Input/Output
SDA1	13	10	18, 14(1)	15, 11(1)	I/O	I^2C	I2C1 Data Input/Output
SDI1	17	14	21	18	I	ST	SPI1 Serial Data Input
SDO1	16	13	24	21	O	—	SPI1 Serial Data Output
SOSCI	9	6	11	8	I	ANA	Secondary Oscillator Input
SOSCO	10	7	12	9	O	ANA	Secondary Oscillator Output
$\overline{SS1}$	18	15	26	23	I/O	ST	Slave Select Input/Frame Select Output (SPI1)

Legend: ST = Schmitt Trigger input buffer, ANA = Analog level input/output, I^2C™ = I^2C/SMBus input buffer

Note 1: Alternative multiplexing when the I2C1SEL Configuration bit is cleared.

REGISTER 3-2: CORCON: CPU CONTROL REGISTER

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	R/C-0, HSC	R/W-0	U-0	U-0
—	—	—	—	IPL3(1)	PSV	—	—
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 15-4 **Unimplemented:** Read as '0'

bit 3 **IPL3:** CPU Interrupt Priority Level Status bit(1)
1 = CPU interrupt priority level is greater than 7
0 = CPU interrupt priority level is 7 or less

bit 2 **PSV:** Program Space Visibility in Data Space Enable bit
1 = Program space is visible in data space
0 = Program space is not visible in data space

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

Note 1: User interrupts are disabled when IPL3 = 1.

3.3 Arithmetic Logic Unit (ALU)

The PIC24F ALU is 16 bits wide and is capable of addition, subtraction, bit shifts and logic operations. Unless otherwise mentioned, arithmetic operations are 2's complement in nature. Depending on the operation, the ALU may affect the values of the Carry (C), Zero (Z), Negative (N), Overflow (OV) and Digit Carry (DC) Status bits in the SR register. The C and DC Status bits operate as Borrow and Digit Borrow bits, respectively, for subtraction operations.

The ALU can perform 8-bit or 16-bit operations, depending on the mode of the instruction that is used. Data for the ALU operation can come from the W register array, or data memory, depending on the addressing mode of the instruction. Likewise, output data from the ALU can be written to the W register array or a data memory location.

The PIC24F CPU incorporates hardware support for both multiplication and division. This includes a dedicated hardware multiplier and support hardware division for 16-bit divisor.

3.3.1 MULTIPLIER

The ALU contains a high-speed, 17-bit x 17-bit multiplier. It supports unsigned, signed or mixed sign operation in several multiplication modes:

- 16-bit x 16-bit signed
- 16-bit x 16-bit unsigned
- 16-bit signed x 5-bit (literal) unsigned
- 16-bit unsigned x 16-bit unsigned
- 16-bit unsigned x 5-bit (literal) unsigned
- 16-bit unsigned x 16-bit signed
- 8-bit unsigned x 8-bit unsigned

4.0 MEMORY ORGANIZATION

As with Harvard architecture devices, the PIC24F microcontrollers feature separate program and data memory space and busing. This architecture also allows the direct access of program memory from the data space during code execution.

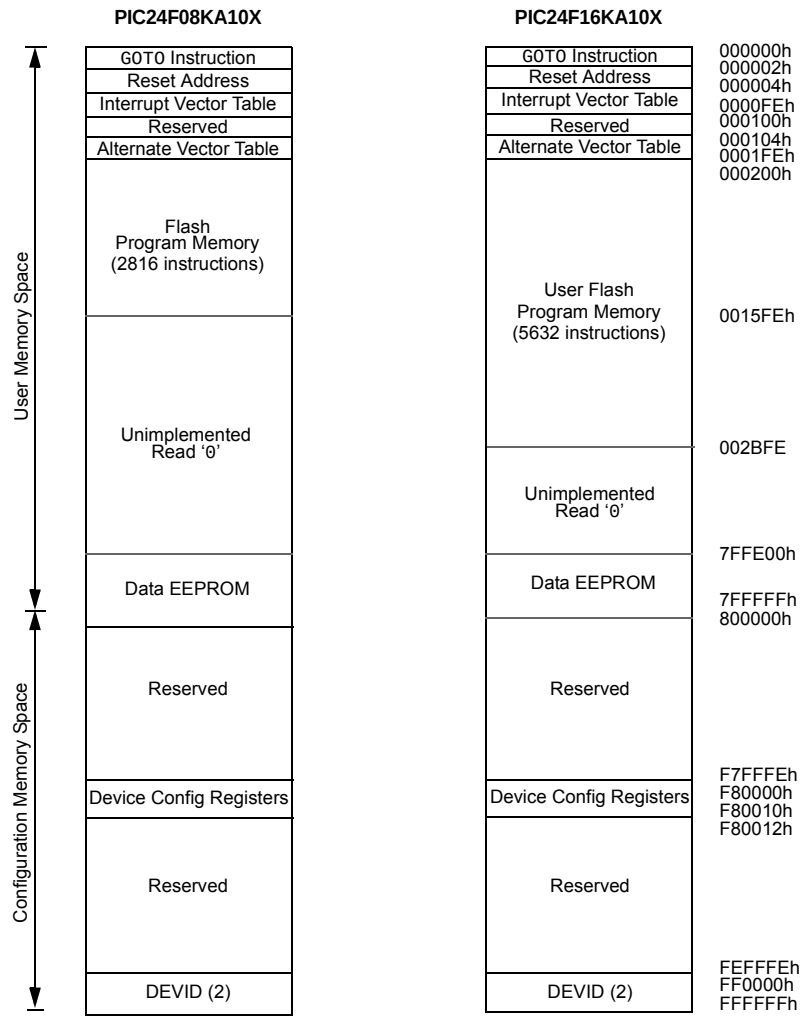
4.1 Program Address Space

The program address memory space of the PIC24F devices is 4M instructions. The space is addressable by a 24-bit value derived from either the 23-bit Program Counter (PC) during program execution, or from a table operation or data space remapping, as described in **Section 4.3 "Interfacing Program and Data Memory Spaces"**.

The user access to the program memory space is restricted to the lower half of the address range (000000h to 7FFFFFh). The exception is the use of TBLRD/TBLWT operations, which use TBLPAG<7> to permit access to the Configuration bits and Device ID sections of the configuration memory space.

Memory maps for the PIC24F16KA102 family of devices are displayed in Figure 4-1.

FIGURE 4-1: PROGRAM SPACE MEMORY MAP FOR PIC24F16KA102 FAMILY DEVICES



Note: Memory areas are not displayed to scale.

TABLE 4-3: CPU CORE REGISTERS 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
WREG0	0000	Working Register 0																0000
WREG1	0002	Working Register 1																0000
WREG2	0004	Working Register 2																0000
WREG3	0006	Working Register 3																0000
WREG4	0008	Working Register 4																0000
WREG5	000A	Working Register 5																0000
WREG6	000C	Working Register 6																0000
WREG7	000E	Working Register 7																0000
WREG8	0010	Working Register 8																0000
WREG9	0012	Working Register 9																0000
WREG10	0014	Working Register 10																0000
WREG11	0016	Working Register 11																0000
WREG12	0018	Working Register 12																0000
WREG13	001A	Working Register 13																0000
WREG14	001C	Working Register 14																0000
WREG15	001E	Working Register 15																0800
SPLIM	0020	Stack Pointer Limit Value Register																xxxx
PCL	002E	Program Counter Low Byte Register																0000
PCH	0030	—	—	—	—	—	—	—	—	Program Counter Register High Byte								0000
TBLPAG	0032	—	—	—	—	—	—	—	—	Table Memory Page Address Register								0000
PSVPAG	0034	—	—	—	—	—	—	—	—	Program Space Visibility Page Address Register								0000
RCOUNT	0036	REPEAT Loop Counter Register																xxxx
SR	0042	—	—	—	—	—	—	—	DC	IPL2	IPL1	IPL0	RA	N	OV	Z	C	0000
CORCON	0044	—	—	—	—	—	—	—	—	—	—	—	—	IPL3	PSV	—	—	0000
DISICNT	0052	—	—	Disable Interrupts Counter Register														xxxx

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

5.5.1 PROGRAMMING ALGORITHM FOR FLASH PROGRAM MEMORY

The user can program one row of Flash program memory at a time by erasing the programmable row. The general process is:

1. Read a row of program memory (32 instructions) and store in data RAM.
2. Update the program data in RAM with the desired new data.
3. Erase a row (see Example 5-1):
   a) Set the NVMOP bits (NVMCON<5:0>) to '011000' to configure for row erase. Set the ERASE (NVMCON<6>) and WREN (NVMCON<14>) bits.
   b) Write the starting address of the block to be erased into the TBLPAG and W registers.
   c) Write 55h to NVMKEY.
   d) Write AAh to NVMKEY.
   e) Set the WR bit (NVMCON<15>). The erase cycle begins and the CPU stalls for the duration of the erase cycle. When the erase is done, the WR bit is cleared automatically.
4. Write the first 32 instructions from data RAM into the program memory buffers (see Example 5-1).
5. Write the program block to Flash memory:
   a) Set the NVMOP bits to '000100' to configure for row programming. Clear the ERASE bit and set the WREN bit.
   b) Write 55h to NVMKEY.
   c) Write AAh to NVMKEY.
   d) Set the WR bit. The programming cycle begins and the CPU stalls for the duration of the write cycle. When the write to Flash memory is done, the WR bit is cleared automatically.

For protection against accidental operations, the write initiate sequence for NVMKEY must be used to allow any erase or program operation to proceed. After the programming command has been executed, the user must wait for the programming time until programming is complete. The two instructions following the start of the programming sequence should be NOPs, as displayed in Example 5-5.

EXAMPLE 5-1: ERASING A PROGRAM MEMORY ROW – ASSEMBLY LANGUAGE CODE

```
; Set up NVMCON for row erase operation
        MOV     #0x4058, W0                 ;
        MOV     W0, NVMCON                  ; Initialize NVMCON
; Init pointer to row to be ERASED
        MOV     #tblpage(PROG_ADDR), W0     ;
        MOV     W0, TBLPAG                  ; Initialize PM Page Boundary SFR
        MOV     #tbloffset(PROG_ADDR), W0   ; Initialize in-page EA[15:0] pointer
        TBLWTL  W0, [W0]                    ; Set base address of erase block
        DISI    #5                          ; Block all interrupts
                                              for next 5 instructions
        MOV     #0x55, W0
        MOV     W0, NVMKEY                  ; Write the 55 key
        MOV     #0xAA, W1                   ;
        MOV     W1, NVMKEY                  ; Write the AA key
        BSET    NVMCON, #WR                 ; Start the erase sequence
        NOP                                 ; Insert two NOPs after the erase
        NOP                                 ; command is asserted
```

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

```
// C example using MPLAB C30

int __attribute__ ((space(auto_psv))) progAddr = 0x1234;    // Variable located in Pgm Memory, declared as a
                                                            // global variable
    unsigned int offset;

//Set up pointer to the first memory location to be written

    TBLPAG = __builtin_tblpage(&progAddr);                  // Initialize PM Page Boundary SFR
    offset = __builtin_tbloffset(&progAddr);                // Initialize lower word of address

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

    NVMCON = 0x4058;                                        // Initialize NVMCON

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

14.1 Input Capture Registers

REGISTER 14-1: IC1CON: INPUT CAPTURE 1 CONTROL REGISTER

U-0	U-0	R/W-0	U-0	U-0	U-0	U-0	U-0
—	—	ICSIDL	—	—	—	—	—
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R-0, HC	R-0, HC	R/W-0	R/W-0	R/W-0
ICTMR	ICI1	ICI0	ICOV	ICBNE	ICM2	ICM1	ICM0
bit 7							bit 0

Legend:	HC = Hardware 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 15-14 **Unimplemented:** Read as '0'

bit 13 **ICSIDL:** Input Capture 1 Module Stop in Idle Control bit
1 = Input capture module will halt in CPU Idle mode
0 = Input capture module will continue to operate in CPU Idle mode

bit 12-8 **Unimplemented:** Read as '0'

bit 7 **ICTMR:** Input Capture 1 Timer Select bit
1 = TMR2 contents are captured on capture event
0 = TMR3 contents are captured on capture event

bit 6-5 **ICI<1:0>:** Select Number of Captures per Interrupt bits
11 = Interrupt on every fourth capture event
10 = Interrupt on every third capture event
01 = Interrupt on every second capture event
00 = Interrupt on every capture event

bit 4 **ICOV:** Input Capture 1 Overflow Status Flag bit (read-only)
1 = Input capture overflow occurred
0 = No input capture overflow occurred

bit 3 **ICBNE:** Input Capture 1 Buffer Empty Status bit (read-only)
1 = Input capture buffer is not empty, at least one more capture value can be read
0 = Input capture buffer is empty

bit 2-0 **ICM<2:0>:** Input Capture 1 Mode Select bits
111 = Input capture functions as interrupt pin only when device is in Sleep or Idle mode (rising edge detect only, all other control bits are not applicable)
110 = Unused (module is disabled)
101 = Capture mode, every 16th rising edge
100 = Capture mode, every 4th rising edge
011 = Capture mode, every rising edge
010 = Capture mode, every falling edge
001 = Capture mode, every edge (rising and falling) – ICI<1:0> bits do not control interrupt generation for this mode
000 = Input capture module is turned off

15.4 Output Compare Register

REGISTER 15-1: OC1CON: OUTPUT COMPARE 1 CONTROL REGISTER

U-0	U-0	R/W-0	U-0	U-0	U-0	U-0	U-0
—	—	OCSIDL	—	—	—	—	—
bit 15							bit 8

U-0	U-0	U-0	R-0, HC	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	OCFLT	OCTSEL	OCM2	OCM1	OCM0
bit 7							bit 0

Legend:	HC = Hardware 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 15-14 **Unimplemented:** Read as '0'

bit 13 **OCSIDL:** Stop Output Compare 1 in Idle Mode Control bit
1 = Output Compare 1 will halt in CPU Idle mode
0 = Output Compare 1 will continue to operate in CPU Idle mode

bit 12-5 **Unimplemented:** Read as '0'

bit 4 **OCFLT:** PWM Fault Condition Status bit
1 = PWM Fault condition has occurred (cleared in HW only)
0 = No PWM Fault condition has occurred (this bit is only used when OCM<2:0> = 111)

bit 3 **OCTSEL:** Output Compare 1 Timer Select bit
1 = Timer3 is the clock source for Output Compare 1
0 = Timer2 is the clock source for Output Compare 1
Refer to the device data sheet for specific time bases available to the output compare module.

bit 2-0 **OCM<2:0>:** Output Compare 1 Mode Select bits
111 = PWM mode on OC1, Fault pin; OCF1 enabled(1)
110 = PWM mode on OC1, Fault pin; OCF1 disabled(1)
101 = Initialize OC1 pin low, generate continuous output pulses on OC1 pin
100 = Initialize OC1 pin low, generate single output pulse on OC1 pin
011 = Compare event toggles OC1 pin
010 = Initialize OC1 pin high, compare event forces OC1 pin low
001 = Initialize OC1 pin low, compare event forces OC1 pin high
000 = Output compare channel is disabled

Note 1: The OCFA pin controls the OC1 channel.

18.2 Transmitting in 8-Bit Data Mode

1. Set up the UART:
   a) Write appropriate values for data, parity and Stop bits.
   b) Write appropriate baud rate value to the UxBRG register.
   c) Set up transmit and receive interrupt enable and priority bits.
2. Enable the UART.
3. Set the UTXEN bit (causes a transmit interrupt two cycles after being set).
4. Write data byte to lower byte of UxTXREG word. The value will be immediately transferred to the Transmit Shift Register (TSR) and the serial bit stream will start shifting out with the next rising edge of the baud clock.
5. Alternately, the data byte may be transferred while UTXEN = 0, and then, the user may set UTXEN. This will cause the serial bit stream to begin immediately because the baud clock will start from a cleared state.
6. A transmit interrupt will be generated as per interrupt control bit, UTXISELx.

18.3 Transmitting in 9-Bit Data Mode

1. Set up the UART (as described in **Section 18.2 "Transmitting in 8-Bit Data Mode"**).
2. Enable the UART.
3. Set the UTXEN bit (causes a transmit interrupt two cycles after being set).
4. Write UxTXREG as a 16-bit value only.
5. A word write to UxTXREG triggers the transfer of the 9-bit data to the TSR. The serial bit stream will start shifting out with the first rising edge of the baud clock.
6. A transmit interrupt will be generated as per the setting of control bit, UTXISELx.

18.4 Break and Sync Transmit Sequence

The following sequence will send a message frame header made up of a Break, followed by an auto-baud Sync byte.

1. Configure the UART for the desired mode.
2. Set UTXEN and UTXBRK – sets up the Break character.
3. Load the UxTXREG with a dummy character to initiate transmission (value is ignored).
4. Write '55h' to UxTXREG – loads the Sync character into the transmit FIFO.
5. After the Break has been sent, the UTXBRK bit is reset by hardware. The Sync character now transmits.

18.5 Receiving in 8-Bit or 9-Bit Data Mode

1. Set up the UART (as described in **Section 18.2 "Transmitting in 8-Bit Data Mode"**).
2. Enable the UART.
3. A receive interrupt will be generated when one or more data characters have been received, as per interrupt control bit, URXISELx.
4. Read the OERR bit to determine if an overrun error has occurred. The OERR bit must be reset in software.
5. Read UxRXREG.

The act of reading the UxRXREG character will move the next character to the top of the receive FIFO, including a new set of PERR and FERR values.

18.6 Operation of $\overline{\text{UxCTS}}$ and $\overline{\text{UxRTS}}$ Control Pins

UARTx Clear to Send ($\overline{\text{UxCTS}}$) and Request to Send ($\overline{\text{UxRTS}}$) are the two hardware-controlled pins that are associated with the UART module. These two pins allow the UART to operate in Simplex and Flow Control modes. They are implemented to control the transmission and reception between the Data Terminal Equipment (DTE). The UEN<1:0> bits in the UxMODE register configure these pins.

18.7 Infrared Support

The UART module provides two types of infrared UART support: one is the IrDA clock output to support an external IrDA encoder and decoder device (legacy module support), and the other is the full implementation of the IrDA encoder and decoder.

As the IrDA modes require a 16x baud clock, they will only work when the BRGH bit (UxMODE<3>) is '0'.

18.7.1 EXTERNAL IrDA SUPPORT – IrDA CLOCK OUTPUT

To support external IrDA encoder and decoder devices, the UxBCLK pin (same as the $\overline{\text{UxRTS}}$ pin) can be configured to generate the 16x baud clock. When UEN<1:0> = 11, the UxBCLK pin will output the 16x baud clock if the UART module is enabled; it can be used to support the IrDA codec chip.

18.7.2 BUILT-IN IrDA ENCODER AND DECODER

The UART has full implementation of the IrDA encoder and decoder as part of the UART module. The built-in IrDA encoder and decoder functionality is enabled using the IREN bit (UxMODE<12>). When enabled (IREN = 1), the receive pin (UxRX) acts as the input from the infrared receiver. The transmit pin (UxTX) acts as the output to the infrared transmitter.

REGISTER 18-2: UxSTA: UARTx STATUS AND CONTROL REGISTER

R/W-0	R/W-0	R/W-0	U-0	R/W-0, HC	R/W-0	R-0, HSC	R-1, HSC
UTXISEL1	UTXINV	UTXISEL0	—	UTXBRK	UTXEN	UTXBF	TRMT
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R-1, HSC	R-0, HSC	R-0, HSC	R/C-0, HS	R-0, HSC
URXISEL1	URXISEL0	ADDEN	RIDLE	PERR	FERR	OERR	URXDA
bit 7							bit 0

Legend:	HC = Hardware Clearable bit		
C = Clearable bit	HS = Hardware Settable bit	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 15,13 **UTXISEL<1:0>:** Transmission Interrupt Mode Selection bits

11 = Reserved; do not use
10 = Interrupt when a character is transferred to the Transmit Shift Register (TSR), and as a result, the transmit buffer becomes empty
01 = Interrupt when the last character is shifted out of the Transmit Shift Register; all transmit operations are completed
00 = Interrupt when a character is transferred to the Transmit Shift Register (this implies there is at least one character open in the transmit buffer)

bit 14 **UTXINV:** IrDA® Encoder Transmit Polarity Inversion bit

If IREN = 0:
1 = UxTX Idle '0'
0 = UxTX Idle '1'

If IREN = 1:
1 = UxTX Idle '1'
0 = UxTX Idle '0'

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

bit 11 **UTXBRK:** Transmit Break bit

1 = Send Sync Break on next transmission – Start bit, followed by twelve '0' bits, followed by Stop bit; cleared by hardware upon completion
0 = Sync Break transmission is disabled or completed

bit 10 **UTXEN:** Transmit Enable bit

1 = Transmit is enabled, UxTX pin is controlled by UARTx
0 = Transmit is disabled, any pending transmission is aborted and buffer is reset. UxTX pin is controlled by the PORT register.

bit 9 **UTXBF:** Transmit Buffer Full Status bit (read-only)

1 = Transmit buffer is full
0 = Transmit buffer is not full, at least one more character can be written

bit 8 **TRMT:** Transmit Shift Register Empty bit (read-only)

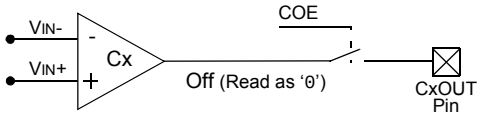
1 = Transmit Shift Register is empty and transmit buffer is empty (the last transmission has completed)
0 = Transmit Shift Register is not empty, a transmission is in progress or queued

bit 7-6 **URXISEL<1:0>:** Receive Interrupt Mode Selection bits

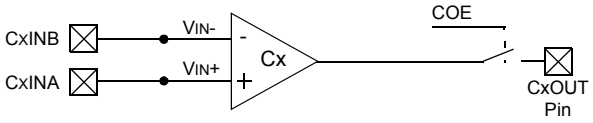
11 = Interrupt is set on RSR transfer, making the receive buffer full (i.e., has 4 data characters)
10 = Interrupt is set on RSR transfer, making the receive buffer 3/4 full (i.e., has 3 data characters)
0x = Interrupt is set when any character is received and transferred from the RSR to the receive buffer; receive buffer has one or more characters

FIGURE 23-2: INDIVIDUAL COMPARATOR CONFIGURATIONS

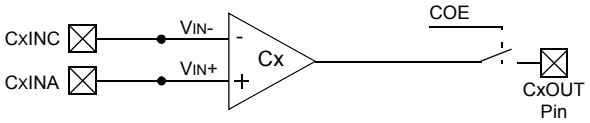
Comparator Off
CON = 0, CREF = x, CCH<1:0> = xx



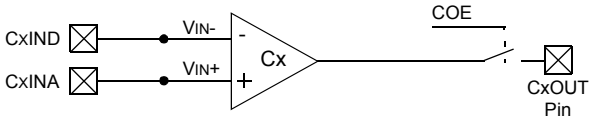
Comparator CxINB < CxINA Compare
CON = 1, CREF = 0, CCH<1:0> = 00



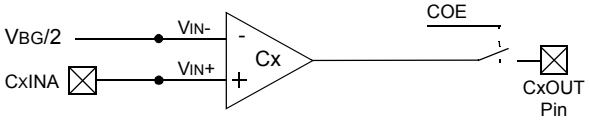
Comparator CxINC < CxINA Compare
CON = 1, CREF = 0, CCH<1:0> = 01



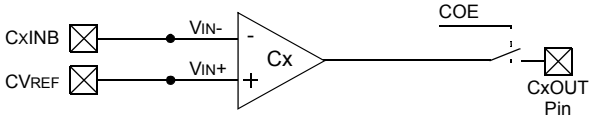
Comparator CxIND < CxINA Compare
CON = 1, CREF = 0, CCH<1:0> = 10



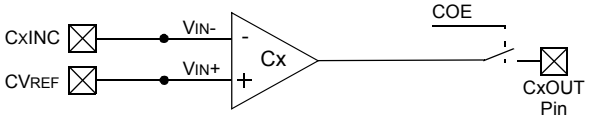
Comparator VBG/2 < CxINA Compare
CON = 1, CREF = 0, CCH<1:0> = 11



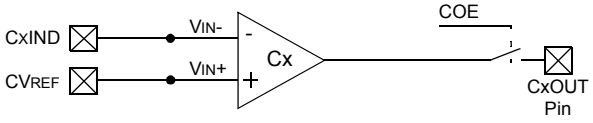
Comparator CxINB < CVREF Compare
CON = 1, CREF = 1, CCH<1:0> = 00



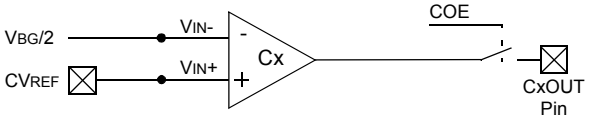
Comparator CxINC < CVREF Compare
CON = 1, CREF = 1, CCH<1:0> = 01



Comparator CxIND < CVREF Compare
CON = 1, CREF = 1, CCH<1:0> = 10



Comparator VBG/2 < CVREF Compare
CON = 1, CREF = 1, CCH<1:0> = 11



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).

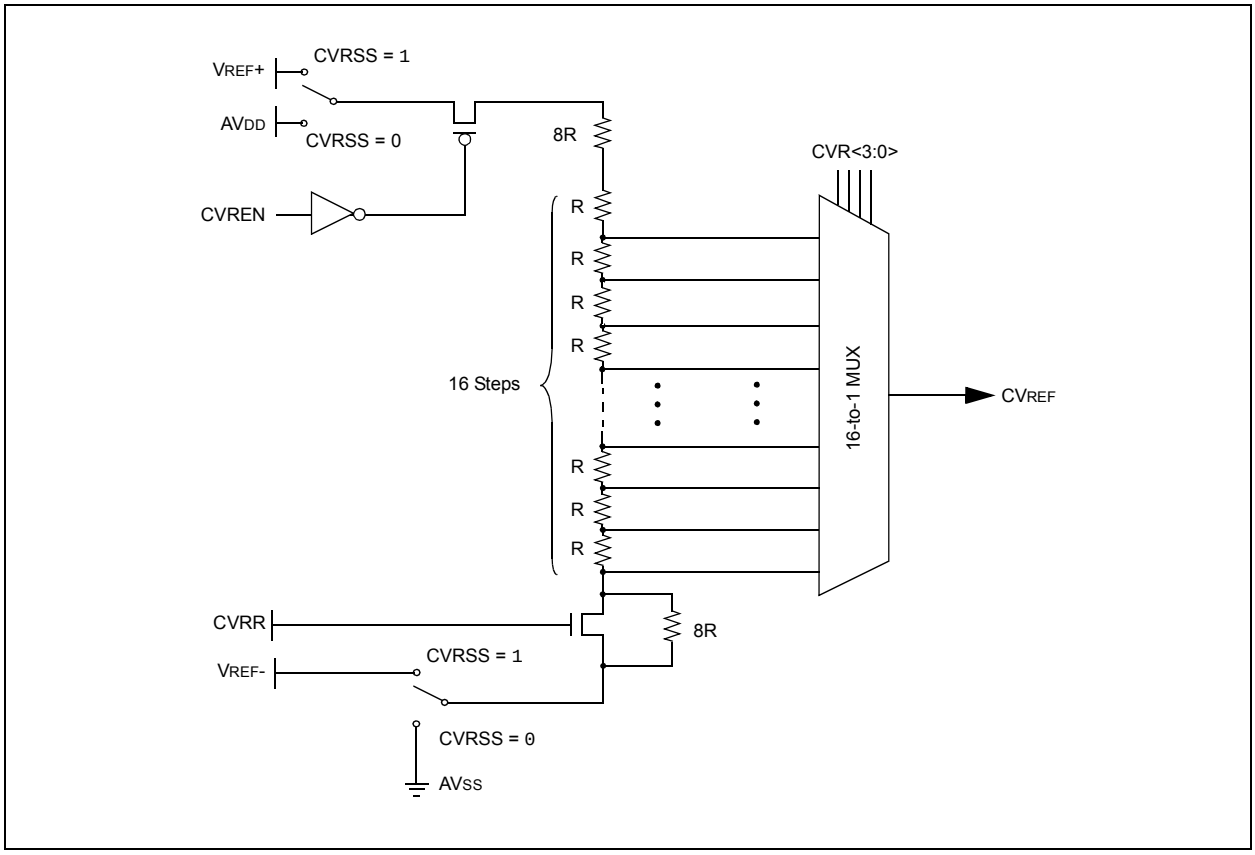
24.1 Configuring the Comparator Voltage Reference

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



REGISTER 26-2: FGS: GENERAL SEGMENT CONFIGURATION REGISTER

U-0	U-0	U-0	U-0	U-0	U-0	R/C-1	R/C-1
—	—	—	—	—	—	GSS0	GWRP
bit 7							bit 0

Legend:			
R = Readable bit	C = Clearable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-2 **Unimplemented:** Read as '0'

bit 1 **GSS0:** General Segment Code Flash Code Protection bit
1 = No protection
0 = Standard security is enabled

bit 0 **GWRP**: General Segment Code Flash Write Protection bit
1 = General segment may be written
0 = General segment is write-protected

REGISTER 26-3: FOSCSEL: OSCILLATOR SELECTION CONFIGURATION REGISTER

R/P-1	U-0	U-0	U-0	U-0	R/P-1	R/P-1	R/P-1
IESO	—	—	—	—	FNOSC2	FNOSC1	FNOSC0
bit 7							bit 0

Legend:			
R = Readable bit	P = Programmable 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 **IESO:** Internal External Switchover bit
1 = Internal External Switchover mode is enabled (Two-Speed Start-up is enabled)
0 = Internal External Switchover mode is disabled (Two-Speed Start-up is disabled)

bit 6-3 **Unimplemented:** Read as '0'

bit 2-0 **FNOSC<2:0>:** Oscillator Selection bits
000 = Fast RC Oscillator (FRC)
001 = Fast RC Oscillator with divide-by-N with PLL module (FRCDIV+PLL)
010 = Primary Oscillator (XT, HS, EC)
011 = Primary Oscillator with PLL module (HS+PLL, EC+PLL)
100 = Secondary Oscillator (SOSC)
101 = Low-Power RC Oscillator (LPRC)
110 = 500 kHz Low-Power FRC Oscillator with divide-by-N (LPFRCDIV)
111 = 8 MHz FRC Oscillator with divide-by-N (FRCDIV)

27.0 DEVELOPMENT SUPPORT

The PIC® microcontrollers and dsPIC® digital signal controllers are supported with a full range of software and hardware development tools:

- Integrated Development Environment
  - MPLAB® IDE Software
- Compilers/Assemblers/Linkers
  - MPLAB C Compiler for Various Device Families
  - HI-TECH C® for Various Device Families
  - MPASM™ Assembler
  - MPLINK™ Object Linker/ MPLIB™ Object Librarian
  - MPLAB Assembler/Linker/Librarian for Various Device Families
- Simulators
  - MPLAB SIM Software Simulator
- Emulators
  - MPLAB REAL ICE™ In-Circuit Emulator
- In-Circuit Debuggers
  - MPLAB ICD 3
  - PICkit™ 3 Debug Express
- Device Programmers
  - PICkit™ 2 Programmer
  - MPLAB PM3 Device Programmer
- Low-Cost Demonstration/Development Boards, Evaluation Kits, and Starter Kits

27.1 MPLAB Integrated Development Environment Software

The MPLAB IDE software brings an ease of software development previously unseen in the 8/16/32-bit microcontroller market. The MPLAB IDE is a Windows® operating system-based application that contains:

- A single graphical interface to all debugging tools
  - Simulator
  - Programmer (sold separately)
  - In-Circuit Emulator (sold separately)
  - In-Circuit Debugger (sold separately)
- A full-featured editor with color-coded context
- A multiple project manager
- Customizable data windows with direct edit of contents
- High-level source code debugging
- Mouse over variable inspection
- Drag and drop variables from source to watch windows
- Extensive on-line help
- Integration of select third party tools, such as IAR C Compilers

The MPLAB IDE allows you to:

- Edit your source files (either C or assembly)
- One-touch compile or assemble, and download to emulator and simulator tools (automatically updates all project information)
- Debug using:
  - Source files (C or assembly)
  - Mixed C and assembly
  - Machine code

MPLAB IDE supports multiple debugging tools in a single development paradigm, from the cost-effective simulators, through low-cost in-circuit debuggers, to full-featured emulators. This eliminates the learning curve when upgrading to tools with increased flexibility and power.

28.0 INSTRUCTION SET SUMMARY

Note: This chapter is a brief summary of the PIC24F instruction set architecture and is not intended to be a comprehensive reference source.

The PIC24F instruction set adds many enhancements to the previous PIC® MCU instruction sets, while maintaining an easy migration from previous PIC MCU instruction sets. Most instructions are a single program memory word. Only three instructions require two program memory locations.

Each single-word instruction is a 24-bit word divided into an 8-bit opcode, which specifies the instruction type and one or more operands, which further specify the operation of the instruction. The instruction set is highly orthogonal and is grouped into four basic categories:

- Word or byte-oriented operations
- Bit-oriented operations
- Literal operations
- Control operations

Table 28-1 lists the general symbols used in describing the instructions. The PIC24F instruction set summary in Table 28-2 lists all the instructions, along with the status flags affected by each instruction.

Most word or byte-oriented W register instructions (including barrel shift instructions) have three operands:

- The first source operand, which is typically a register ‘Wb’ without any address modifier
- The second source operand, which is typically a register ‘Ws’ with or without an address modifier
- The destination of the result, which is typically a register ‘Wd’ with or without an address modifier

However, word or byte-oriented file register instructions have two operands:

- The file register, specified by the value, ‘f’
- The destination, which could either be the file register ‘f’ or the W0 register, which is denoted as ‘WREG’

Most bit-oriented instructions (including simple rotate/shift instructions) have two operands:

- The W register (with or without an address modifier) or file register (specified by the value of ‘Ws’ or ‘f’)
- The bit in the W register or file register (specified by a literal value or indirectly by the contents of register ‘Wb’)

The literal instructions that involve data movement may use some of the following operands:

- A literal value to be loaded into a W register or file register (specified by the value of ‘k’)
- The W register or file register, where the literal value is to be loaded (specified by ‘Wb’ or ‘f’)

However, literal instructions that involve arithmetic or logical operations use some of the following operands:

- The first source operand, which is a register ‘Wb’ without any address modifier
- The second source operand, which is a literal value
- The destination of the result (only if not the same as the first source operand), which is typically a register ‘Wd’ with or without an address modifier

The control instructions may use some of the following operands:

- A program memory address
- The mode of the table read and table write instructions

All instructions are a single word, except for certain double-word instructions, which were made double-word instructions so that all of the required information is available in these 48 bits. In the second word, the 8 MSbs are ‘0’s. If this second word is executed as an instruction (by itself), it will execute as a NOP.

Most single-word instructions are executed in a single instruction cycle, unless a conditional test is true or the Program Counter (PC) is changed as a result of the instruction. In these cases, the execution takes two instruction cycles, with the additional instruction cycle(s) executed as a NOP. Notable exceptions are the BRA (unconditional/computed branch), indirect CALL/GOTO, all table reads and writes, and RETURN/RETFIE instructions, which are single-word instructions but take two or three cycles.

Certain instructions that involve skipping over the subsequent instruction require either two or three cycles if the skip is performed, depending on whether the instruction being skipped is a single-word or two-word instruction. Moreover, double-word moves require two cycles. The double-word instructions execute in two instruction cycles.

TABLE 29-3: DC CHARACTERISTICS: TEMPERATURE AND VOLTAGE SPECIFICATIONS

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				
Param No.	**Symbol**	**Characteristic**	**Min**	**Typ(1)**	**Max**	**Units**	**Conditions**
DC10	VDD	**Supply Voltage**	1.8	—	3.6	V	
DC12	VDR	**RAM Data Retention Voltage(2)**	1.5	—	—	V	
DC16	VPOR	**VDD Start Voltage** to Ensure Internal Power-on Reset Signal	VSS	—	0.7	V	
DC17	SVDD	**VDD Rise Rate** to Ensure Internal Power-on Reset Signal	0.05	—	—	V/ms	0-3.3V in 0.1s 0-2.5V in 60 ms

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

2: This is the limit to which VDD can be lowered without losing RAM data.

TABLE 29-4: HIGH/LOW–VOLTAGE DETECT CHARACTERISTICS

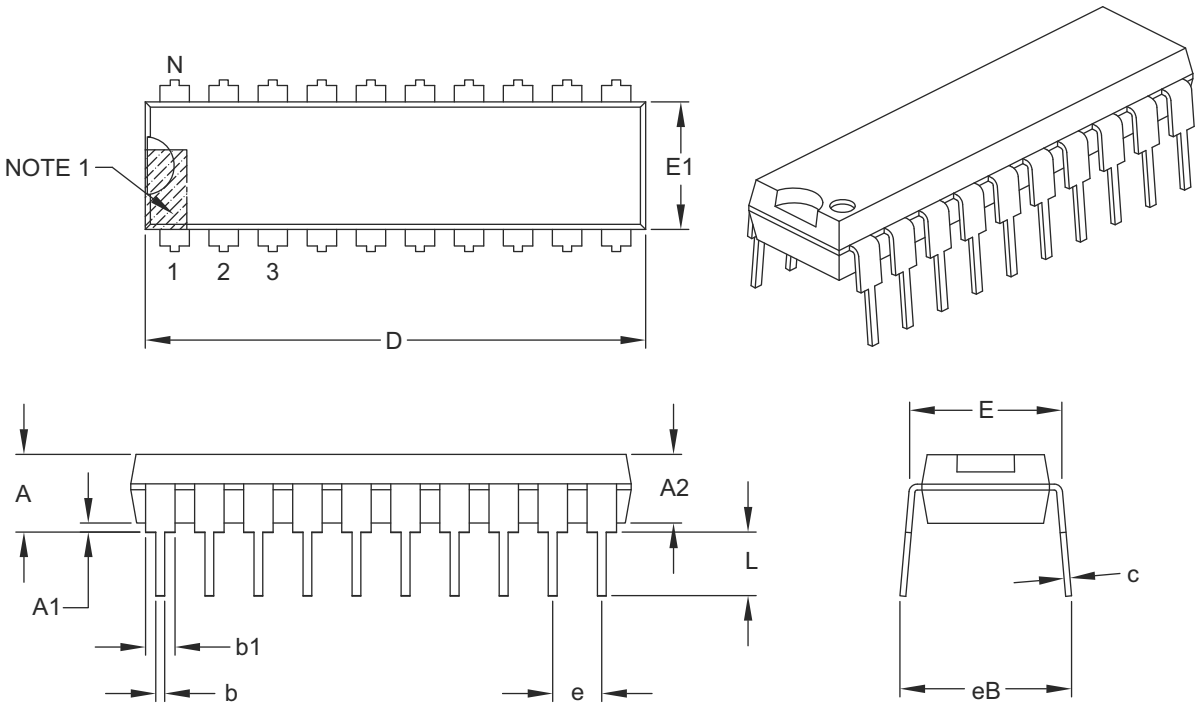
Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for Extended								
Param No.	**Symbol**	**Characteristic**		**Min**	**Typ**	**Max**	**Units**	**Conditions**
DC18	VHLVD	HLVD Voltage on VDD Transition	HLVDL<3:0> = 0000	—	1.85	1.94	V	
			HLVDL<3:0> = 0001	1.81	1.90	2.00	V	
			HLVDL<3:0> = 0010	1.85	1.95	2.05	V	
			HLVDL<3:0> = 0011	1.90	2.00	2.10	V	
			HLVDL<3:0> = 0100	1.95	2.05	2.15	V	
			HLVDL<3:0> = 0101	2.06	2.17	2.28	V	
			HLVDL<3:0> = 0110	2.12	2.23	2.34	V	
			HLVDL<3:0> = 0111	2.24	2.36	2.48	V	
			HLVDL<3:0> = 1000	2.31	2.43	2.55	V	
			HLVDL<3:0> = 1001	2.47	2.60	2.73	V	
			HLVDL<3:0> = 1010	2.64	2.78	2.92	V	
			HLVDL<3:0> = 1011	2.74	2.88	3.02	V	
			HLVDL<3:0> = 1100	2.85	3.00	3.15	V	
			HLVDL<3:0> = 1101	2.96	3.12	3.28	V	
			HLVDL<3:0> = 1110	3.22	3.39	3.56	V	

30.2 Package Details

The following sections give the technical details of the packages.

20-Lead Plastic Dual In-Line (P) – 300 mil Body [PDIP]

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		20	
Pitch	e		.100 BSC	
Top to Seating Plane	A	–	–	.210
Molded Package Thickness	A2	.115	.130	.195
Base to Seating Plane	A1	.015	–	–
Shoulder to Shoulder Width	E	.300	.310	.325
Molded Package Width	E1	.240	.250	.280
Overall Length	D	.980	1.030	1.060
Tip to Seating Plane	L	.115	.130	.150
Lead Thickness	c	.008	.010	.015
Upper Lead Width	b1	.045	.060	.070
Lower Lead Width	b	.014	.018	.022
Overall Row Spacing §	eB	–	–	.430

Notes:

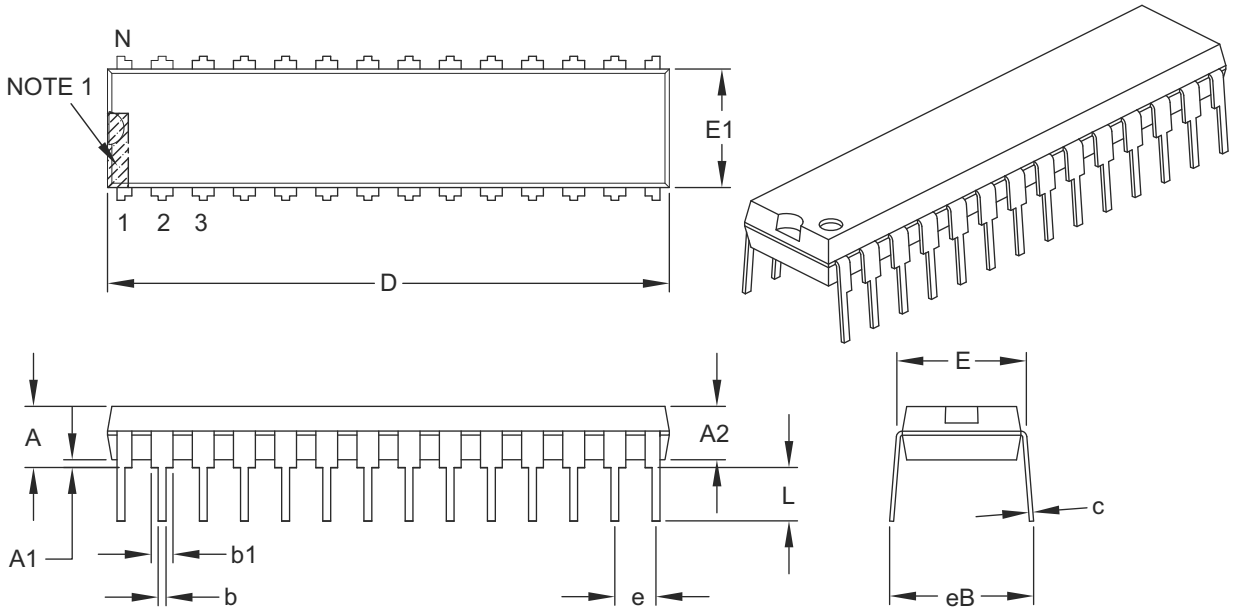
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. § Significant Characteristic.
3. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-019B

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

M

Microchip Internet Web Site 275
MPLAB ASM30 Assembler, Linker, Librarian 204
MPLAB Integrated Development
    Environment Software 203
MPLAB PM3 Device Programmer 206
MPLAB REAL ICE In-Circuit Emulator System 205
MPLINK Object Linker/MPLIB Object Librarian 204

N

Near Data Space 32

O

Oscillator Configuration
    Bit Values for Clock Selection 92
    Clock Switching 97
        Sequence 97
    CPU Clocking Scheme 92
    Initial Configuration on POR 92
    Reference Clock Output 98
Output Compare
    Continuous Output Pulse Generation 125
    Single Output Pulse Generation 125

P

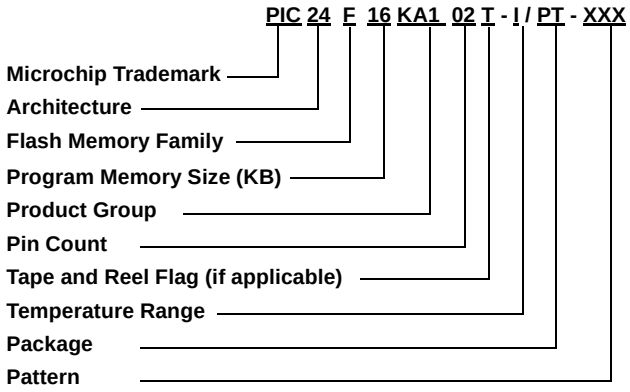
Packaging
    Details 253
    Marking 251
Pinout Descriptions 13–16
Power-Saving Features 101
    Clock Frequency, Clock Switching 101
    Instruction-Based Modes 101
        Deep Sleep 102
        Idle 102
        Sleep 101
Product Identification System 277
Program and Data Memory
    Access Using Table Instructions 43
    Program Space Visibility 44
Program and Data Memory Spaces
    Interfacing, Addressing 41
Program Memory
    Address Space 29
    Configuration Word Addresses 30
    Memory Map 29
Program Verification 202
Programmable Cyclic Redundancy Check (CRC)
    Generator 167
Pulse-Width Modulation. *See* PWM.

R

Reader Response 276
Register Maps
    A/D 38
    Clock Control 40
    CPU Core 33
    CRC 39
    CTMU 38
    Deep Sleep 40
    Dual Comparator 39
    I^2C 36
    ICN 34
    Input Capture 35
    Interrupt Controller 34
    NVM 40
    Output Compare 35
    Pad Configuration 37
    PMD 40
    PORTA 37
    PORTB 37
    RTCC 39
    SPI 36
    Timer 35
    UART 36
Registers
    AD1CHS (A/D Input Select) 178
    AD1CON1 (A/D Control 1) 175
    AD1CON2 (A/D Control 2) 176
    AD1CON3 (A/D Control 3) 177
    AD1CSSL (A/D Input Scan Select, Low) 180
    AD1PCFG (A/D Port Configuration) 179
    ALCFGRPT (Alarm Configuration) 159
    ALMINSEC (Alarm Minutes and
        Seconds Value) 163
    ALMTHDY (Alarm Month and Day Value) 162
    ALWDHR (Alarm Weekday and Hours Value) 162
    CLKDIV (Clock Divider) 95
    CMSTAT (Comparator Status) 186
    CMxCON (Comparator x Control) 185
    CORCON (CPU Control) 27, 68
    CRCCON (CRC Control) 169
    CRCXOR (CRC XOR Polynomial) 170
    CTMUCON (CTMU Control) 191
    CTMUICON (CTMU Current Control) 192
    CVRCON (Comparator Voltage
        Reference Control) 188
    DEVID (Device ID) 200
    DEVREV (Device Revision) 200
    DSCON (Deep Sleep Control) 105
    DSWAKE (Deep Sleep Wake-up Source) 106
    FBS (Boot Segment Configuration) 193
    FDS (Deep Sleep Configuration) 199
    FGS (General Segment Configuration) 194
    FICD (In-Circuit Debugger Configuration) 198
    FOSC (Oscillator Configuration) 195
    FOSCSEL (Oscillator Selection Configuration) 194
    FPOR (Reset Configuration) 197
    FWDT (Watchdog Timer Configuration) 196
    HLVDCON (High/Low-Voltage Detect Control) 172
    I2C1CON (I2C1 Control) 142
    I2C1MSK (I2C1 Slave Mode Address Mask) 146
    I2C1STAT (I2C1 Status) 144
    IC1CON (Input Capture 1 Control) 124
    IEC0 (Interrupt Enable Control 0) 75
    IEC1 (Interrupt Enable Control 1) 76
    IEC3 (Interrupt Enable Control 3) 77
    IEC4 (Interrupt Enable Control 4) 78
    IFS0 (Interrupt Flag Status 0) 71
    IFS1 (Interrupt Flag Status 1) 72
    IFS3 (Interrupt Flag Status 3) 73
    IFS4 (Interrupt Flag Status 4) 74
    INTCON1 (Interrupt Control 1) 69
    INTCON2 (Interrupt Control 2) 70
    INTTREG Interrupt Control and Status 89
    IPC0 (Interrupt Priority Control 0) 79
    IPC1 (Interrupt Priority Control 1) 80
    IPC15 (Interrupt Priority Control 15) 86
    IPC16 (Interrupt Priority Control 16) 87
    IPC18 (Interrupt Priority Control 18) 88
    IPC19 (Interrupt Priority Control 19) 88

PRODUCT IDENTIFICATION SYSTEM

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



Architecture	24	= 16-bit modified Harvard without DSP
Flash Memory Family	F	= Flash program memory
Product Group	KA1	= General purpose microcontrollers
Pin Count	01 02	= 20-pin = 28-pin
Temperature Range	I E	= -40°C to +85°C (Industrial) = -40°C to +125°C (Extended)
Package	SP SO SS ML P	= SPDIP = SOIC = SSOP = QFN = PDIP
Pattern	Three-digit QTP, SQTP, Code or Special Requirements (blank otherwise) ES = Engineering Sample	

Examples:

a) PIC24F16KA102-I/ML: General purpose, 16-Kbyte program memory, 28-pin, Industrial temp., QFN package.