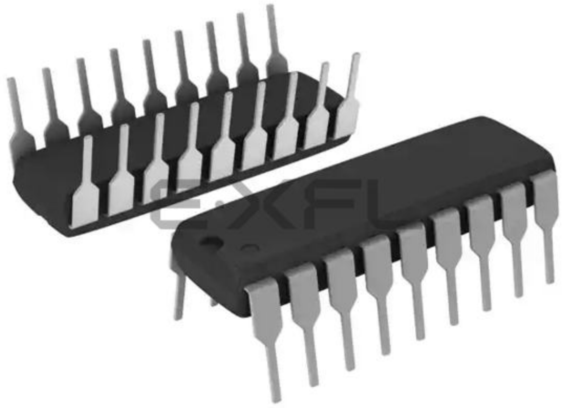
Microchip Technology - PIC18F1230-E/P Datasheet



Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

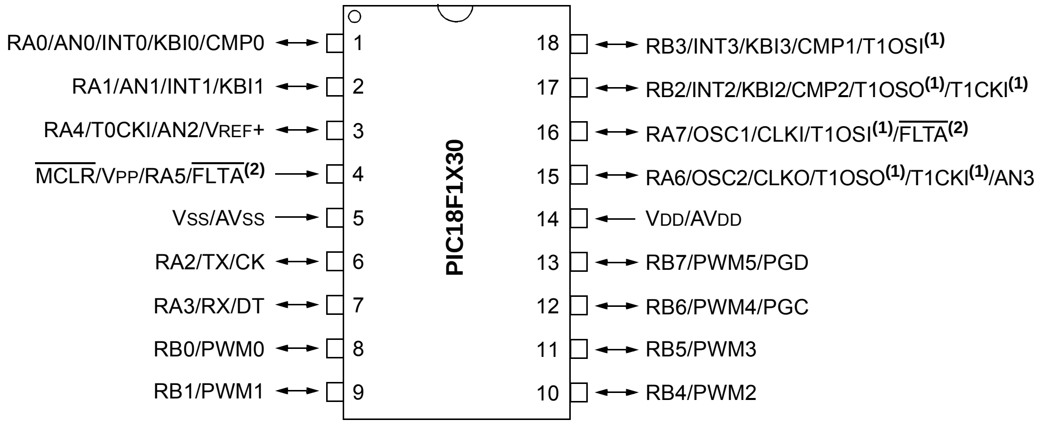
"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

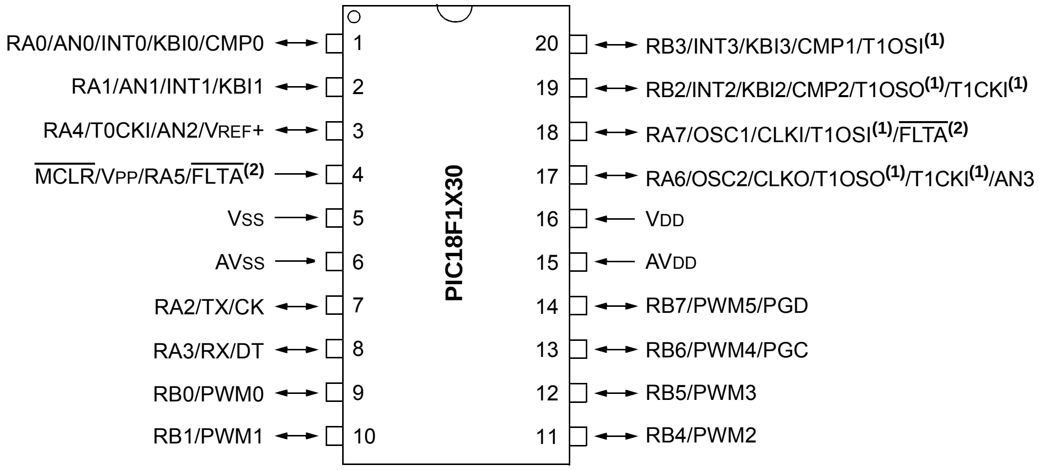
Details	
Product Status	Active
Core Processor	PIC
Core Size	8-Bit
Speed	25MHz
Connectivity	UART/USART
Peripherals	Brown-out Detect/Reset, LVD, POR, PWM, WDT
Number of I/O	16
Program Memory Size	4KB (2K x 16)
Program Memory Type	FLASH
EEPROM Size	128 x 8
RAM Size	256 x 8
Voltage - Supply (Vcc/Vdd)	4.2V ~ 5.5V
Data Converters	A/D 4x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Through Hole
Package / Case	18-DIP (0.300", 7.62mm)
Supplier Device Package	18-PDIP
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f1230-e-p

Pin Diagrams

18-Pin PDIP, SOIC



20-Pin SSOP



Note 1: Placement of T1OSI and T1OSO/T1CKI depends on the value of Configuration bit, T1OSCMX, of CONFIG3H.

2: Placement of $\overline{\text{FLTA}}$ depends on the value of Configuration bit, FLTAMX, of CONFIG3H.

8.7 Operation During Code-Protect

Data EEPROM memory has its own code-protect bits in Configuration Words. External read and write operations are disabled if either of these mechanisms are enabled.

The microcontroller itself can both read and write to the internal data EEPROM, regardless of the state of the code-protect Configuration bit. Refer to **Section 20.0 "Special Features of the CPU"** for additional information.

8.8 Using the Data EEPROM

The data EEPROM is a high-endurance, byte addressable array that has been optimized for the storage of frequently changing information (e.g., program variables or other data that are updated often). Frequently changing values will typically be updated more often than specification D124. If this is not the case, an array refresh must be performed. For this reason, variables that change infrequently (such as constants, IDs, calibration, etc.) should be stored in Flash program memory.

A simple data EEPROM refresh routine is shown in Example 8-3.

Note:	If data EEPROM is only used to store constants and/or data that changes rarely, an array refresh is likely not required. See specification D124.

EXAMPLE 8-3: DATA EEPROM REFRESH ROUTINE

```
           CLRF    EEADR                          ; Start at address 0
           BCF     EECON1, CFGS                   ; Set for memory
           BCF     EECON1, EEPGD                  ; Set for Data EEPROM
           BCF     INTCON, GIE                    ; Disable interrupts
           BSF     EECON1, WREN                   ; Enable writes
    LOOP                                      ; Loop to refresh array
           BSF     EECON1, RD                     ; Read current address
           MOVLW   55h                            ;
Required   MOVWF   EECON2                         ; Write 55h
Sequence   MOVLW   0AAh                           ;
           MOVWF   EECON2                         ; Write 0AAh
           BSF     EECON1, WR                     ; Set WR bit to begin write
           BTFSC   EECON1, WR                     ; Wait for write to complete
           BRA     $-2
           INCFSZ  EEADR, F                       ; Increment address
           BRA     LOOP                           ; Not zero, do it again

           BCF     EECON1, WREN                   ; Disable writes
           BSF     INTCON, GIE                    ; Enable interrupts
```

TABLE 8-1: REGISTERS ASSOCIATED WITH DATA EEPROM MEMORY

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	47
EEADR	EEPROM Address Register								49
EEDATA	EEPROM Data Register								49
EECON2	EEPROM Control Register 2 (not a physical register)								49
EECON1	EEPGD	CFGS	—	FREE	WRERR	WREN	WR	RD	49
IPR2	OSCFIP	—	—	EEIP	—	LVDIP	—	—	49
PIR2	OSCFIF	—	—	EEIF	—	LVDIF	—	—	49
PIE2	OSCFIE	—	—	EEIE	—	LVDIE	—	—	49

Legend: — = unimplemented, read as '0'. Shaded cells are not used during Flash/EEPROM access.

Example 9-3 shows the sequence to do a 16 x 16 unsigned multiplication. Equation 9-1 shows the algorithm that is used. The 32-bit result is stored in four registers (RES3:RES0).

EQUATION 9-1: 16 x 16 UNSIGNED MULTIPLICATION ALGORITHM

$$\begin{aligned}
\mathrm{RES3{:}RES0} &= \mathrm{ARG1H{:}ARG1L} \bullet \mathrm{ARG2H{:}ARG2L} \\
&= (\mathrm{ARG1H} \bullet \mathrm{ARG2H} \bullet 2^{16}) + \\
&\quad (\mathrm{ARG1H} \bullet \mathrm{ARG2L} \bullet 2^{8}) + \\
&\quad (\mathrm{ARG1L} \bullet \mathrm{ARG2H} \bullet 2^{8}) + \\
&\quad (\mathrm{ARG1L} \bullet \mathrm{ARG2L})
\end{aligned}$$

EXAMPLE 9-3: 16 x 16 UNSIGNED MULTIPLY ROUTINE

```
    MOVF    ARG1L, W
    MULWF   ARG2L       ; ARG1L * ARG2L->
                        ; PRODH:PRODL
    MOVFF   PRODH, RES1 ;
    MOVFF   PRODL, RES0 ;
;
    MOVF    ARG1H, W
    MULWF   ARG2H       ; ARG1H * ARG2H->
                        ; PRODH:PRODL
    MOVFF   PRODH, RES3 ;
    MOVFF   PRODL, RES2 ;
;
    MOVF    ARG1L, W
    MULWF   ARG2H       ; ARG1L * ARG2H->
                        ; PRODH:PRODL
    MOVF    PRODL, W    ;
    ADDWF   RES1, F     ; Add cross
    MOVF    PRODH, W    ; products
    ADDWFC  RES2, F     ;
    CLRF    WREG        ;
    ADDWFC  RES3, F     ;
;
    MOVF    ARG1H, W    ;
    MULWF   ARG2L       ; ARG1H * ARG2L->
                        ; PRODH:PRODL
    MOVF    PRODL, W    ;
    ADDWF   RES1, F     ; Add cross
    MOVF    PRODH, W    ; products
    ADDWFC  RES2, F     ;
    CLRF    WREG        ;
    ADDWFC  RES3, F     ;
```

Example 9-4 shows the sequence to do a 16 x 16 signed multiply. Equation 9-2 shows the algorithm used. The 32-bit result is stored in four registers (RES3:RES0). To account for the sign bits of the arguments, the MSb for each argument pair is tested and the appropriate subtractions are done.

EQUATION 9-2: 16 x 16 SIGNED MULTIPLICATION ALGORITHM

$$\begin{aligned}
\mathrm{RES3{:}RES0} &= \mathrm{ARG1H{:}ARG1L} \bullet \mathrm{ARG2H{:}ARG2L} \\
&= (\mathrm{ARG1H} \bullet \mathrm{ARG2H} \bullet 2^{16}) + \\
&\quad (\mathrm{ARG1H} \bullet \mathrm{ARG2L} \bullet 2^{8}) + \\
&\quad (\mathrm{ARG1L} \bullet \mathrm{ARG2H} \bullet 2^{8}) + \\
&\quad (\mathrm{ARG1L} \bullet \mathrm{ARG2L}) + \\
&\quad (-1 \bullet \mathrm{ARG2H}{<}7{>} \bullet \mathrm{ARG1H{:}ARG1L} \bullet 2^{16}) + \\
&\quad (-1 \bullet \mathrm{ARG1H}{<}7{>} \bullet \mathrm{ARG2H{:}ARG2L} \bullet 2^{16})
\end{aligned}$$

EXAMPLE 9-4: 16 x 16 SIGNED MULTIPLY ROUTINE

```
    MOVF    ARG1L, W
    MULWF   ARG2L       ; ARG1L * ARG2L ->
                        ; PRODH:PRODL
    MOVFF   PRODH, RES1 ;
    MOVFF   PRODL, RES0 ;
;
    MOVF    ARG1H, W
    MULWF   ARG2H       ; ARG1H * ARG2H ->
                        ; PRODH:PRODL
    MOVFF   PRODH, RES3 ;
    MOVFF   PRODL, RES2 ;
;
    MOVF    ARG1L, W
    MULWF   ARG2H       ; ARG1L * ARG2H ->
                        ; PRODH:PRODL
    MOVF    PRODL, W    ;
    ADDWF   RES1, F     ; Add cross
    MOVF    PRODH, W    ; products
    ADDWFC  RES2, F     ;
    CLRF    WREG        ;
    ADDWFC  RES3, F     ;
;
    MOVF    ARG1H, W    ;
    MULWF   ARG2L       ; ARG1H * ARG2L ->
                        ; PRODH:PRODL
    MOVF    PRODL, W    ;
    ADDWF   RES1, F     ; Add cross
    MOVF    PRODH, W    ; products
    ADDWFC  RES2, F     ;
    CLRF    WREG        ;
    ADDWFC  RES3, F     ;
;
    BTFSS   ARG2H, 7    ; ARG2H:ARG2L neg?
    BRA     SIGN_ARG1   ; no, check ARG1
    MOVF    ARG1L, W    ;
    SUBWF   RES2        ;
    MOVF    ARG1H, W    ;
    SUBWFB  RES3
;
SIGN_ARG1
    BTFSS   ARG1H, 7    ; ARG1H:ARG1L neg?
    BRA     CONT_CODE   ; no, done
    MOVF    ARG2L, W    ;
    SUBWF   RES2        ;
    MOVF    ARG2H, W    ;
    SUBWFB  RES3
;
CONT_CODE
   :
```

TABLE 10-2: SUMMARY OF REGISTERS ASSOCIATED WITH PORTA

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
PORTA	RA7(1)	RA6(1)	RA5	RA4	RA3	RA2	RA1	RA0	50
LATA	LATA7(1)	LATA6(1)	PORTA Output Latch Register (Read and Write to Data Latch)						49
TRISA	TRISA7(1)	TRISA6(1)	PORTA Data Direction Control Register						49
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	47
INTCON2	$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	RBIP	47
ADCON1	—	—	—	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	48
CMCON	C2OUT	C1OUT	C0OUT	—	—	CMEN2	CMEN1	CMEN0	48
CVRCON	CVREN	—	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	48

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

Note 1: RA7:RA6 and their associated latch and data direction bits are enabled as I/O pins based on oscillator configuration; otherwise, they are read as ‘0’.

11.0 INTERRUPTS

The PIC18F1230/1330 devices have multiple interrupt sources and an interrupt priority feature that allows most interrupt sources to be assigned a high-priority level or a low-priority level. The high-priority interrupt vector is at 0008h and the low-priority interrupt vector is at 0018h. High-priority interrupt events will interrupt any low-priority interrupts that may be in progress.

There are thirteen registers which are used to control interrupt operation. These registers are:

- RCON
- INTCON
- INTCON2
- INTCON3
- PIR1, PIR2, PIR3
- PIE1, PIE2, PIE3
- IPR1, IPR2, IPR3

It is recommended that the Microchip header files supplied with MPLAB® IDE be used for the symbolic bit names in these registers. This allows the assembler/compiler to automatically take care of the placement of these bits within the specified register.

In general, interrupt sources have three bits to control their operation. They are:

- **Flag bit** to indicate that an interrupt event occurred
- **Enable bit** that allows program execution to branch to the interrupt vector address when the flag bit is set
- **Priority bit** to select high priority or low priority

The interrupt priority feature is enabled by setting the IPEN bit (RCON<7>). When interrupt priority is enabled, there are two bits which enable interrupts globally. Setting the GIEH bit (INTCON<7>) enables all interrupts that have the priority bit set (high priority). Setting the GIEL bit (INTCON<6>) enables all interrupts that have the priority bit cleared (low priority). When the interrupt flag, enable bit and appropriate global interrupt enable bit are set, the interrupt will vector immediately to address 0008h or 0018h, depending on the priority bit setting. Individual interrupts can be disabled through their corresponding enable bits.

When the IPEN bit is cleared (default state), the interrupt priority feature is disabled and interrupts are compatible with PIC® mid-range devices. In Compatibility mode, the interrupt priority bits for each source have no effect. INTCON<6> is the PEIE bit, which enables/disables all peripheral interrupt sources. INTCON<7> is the GIE bit, which enables/disables all interrupt sources. All interrupts branch to address 0008h in Compatibility mode.

When an interrupt is responded to, the global interrupt enable bit is cleared to disable further interrupts. If the IPEN bit is cleared, this is the GIE bit. If interrupt priority levels are used, this will be either the GIEH or GIEL bit. High-priority interrupt sources can interrupt a low-priority interrupt. Low-priority interrupts are not processed while high-priority interrupts are in progress.

The return address is pushed onto the stack and the PC is loaded with the interrupt vector address (0008h or 0018h). Once in the Interrupt Service Routine, the source(s) of the interrupt can be determined by polling the interrupt flag bits. The interrupt flag bits must be cleared in software before re-enabling interrupts to avoid recursive interrupts.

The “return from interrupt” instruction, `RETFIE`, exits the interrupt routine and sets the GIE bit (GIEH or GIEL if priority levels are used), which re-enables interrupts.

For external interrupt events, such as the INT pins or the PORTB input change interrupt, the interrupt latency will be three to four instruction cycles. The exact latency is the same for one or two-cycle instructions. Individual interrupt flag bits are set, regardless of the status of their corresponding enable bit or the GIE bit.

Note:	Do not use the `MOVFF` instruction to modify any of the interrupt control registers while **any** interrupt is enabled. Doing so may cause erratic microcontroller behavior.

REGISTER 11-8: PIE2: PERIPHERAL INTERRUPT ENABLE REGISTER 2

R/W-0	U-0	U-0	R/W-0	U-0	R/W-0	U-0	U-0
OSCFIE	—	—	EEIE	—	LVDIE	—	—
bit 7							bit 0

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

bit 7 **OSCFIE:** Oscillator Fail Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 6-5 **Unimplemented:** Read as ‘0’

bit 4 **EEIE:** Data EEPROM/Flash Write Operation Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 3 **Unimplemented:** Read as ‘0’

bit 2 **LVDIE:** Low-Voltage Detect Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 1-0 **Unimplemented:** Read as ‘0’

REGISTER 11-9: PIE3: PERIPHERAL INTERRUPT ENABLE REGISTER 3

U-0	U-0	U-0	R/W-0	U-0	U-0	U-0	U-0
—	—	—	PTIE	—	—	—	—
bit 7							bit 0

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

bit 7-5 **Unimplemented:** Read as ‘0’

bit 4 **PTIE:** PWM Time Base Interrupt Enable bit
1 = PWM enabled
0 = PWM disabled

bit 3-0 **Unimplemented:** Read as ‘0’

15.2.2 EUSART ASYNCHRONOUS RECEIVER

The receiver block diagram is shown in Figure 15-6. The data is received on the RX pin and drives the data recovery block. The data recovery block is actually a high-speed shifter operating at x16 times the baud rate, whereas the main receive serial shifter operates at the bit rate or at FOSC. This mode would typically be used in RS-232 systems.

To set up an Asynchronous Reception:

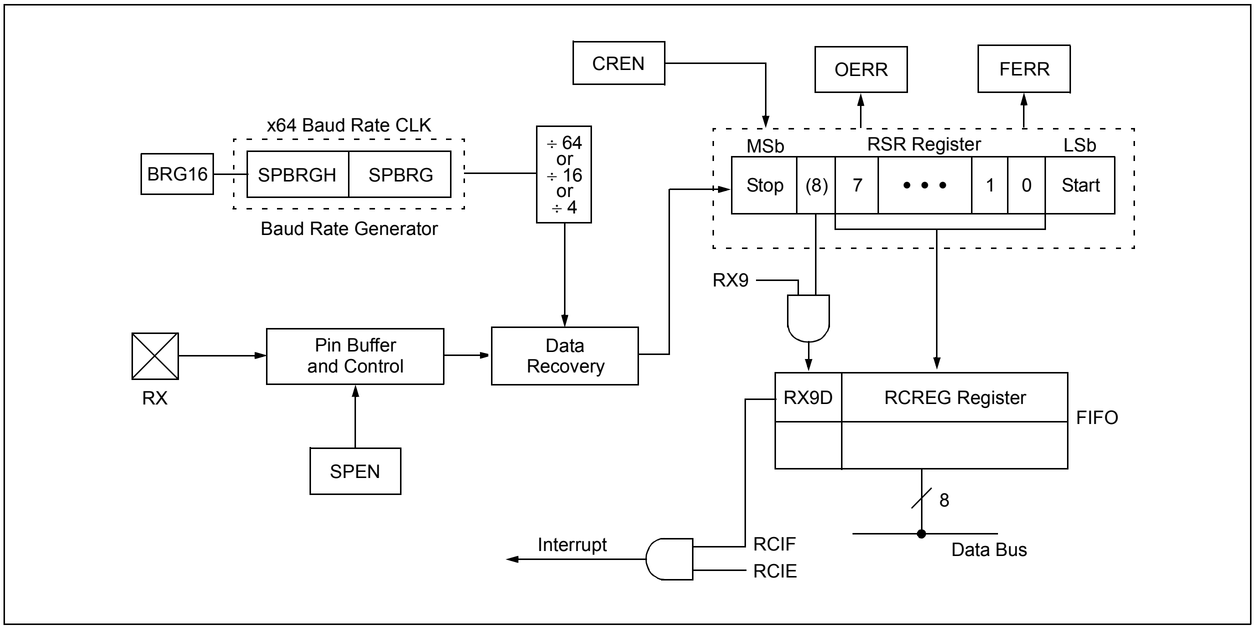
1. Initialize the SPBRGH:SPBRG registers for the appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate.
2. Enable the asynchronous serial port by clearing bit SYNC and setting bit SPEN.
3. If interrupts are desired, set enable bit RCIE.
4. If 9-bit reception is desired, set bit RX9.
5. Enable the reception by setting bit CREN.
6. Flag bit, RCIF, will be set when reception is complete and an interrupt will be generated if enable bit, RCIE, was set.
7. Read the RCSTA register to get the 9th bit (if enabled) and determine if any error occurred during reception.
8. Read the 8-bit received data by reading the RCREG register.
9. If any error occurred, clear the error by clearing enable bit CREN.
10. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

15.2.3 SETTING UP 9-BIT MODE WITH ADDRESS DETECT

This mode would typically be used in RS-485 systems. To set up an Asynchronous Reception with Address Detect Enable:

1. Initialize the SPBRGH:SPBRG registers for the appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate.
2. Enable the asynchronous serial port by clearing the SYNC bit and setting the SPEN bit.
3. If interrupts are required, set the RCEN bit and select the desired priority level with the RCIP bit.
4. Set the RX9 bit to enable 9-bit reception.
5. Set the ADDEN bit to enable address detect.
6. Enable reception by setting the CREN bit.
7. The RCIF bit will be set when reception is complete. The interrupt will be Acknowledged if the RCIE and GIE bits are set.
8. Read the RCSTA register to determine if any error occurred during reception, as well as read bit 9 of data (if applicable).
9. Read RCREG to determine if the device is being addressed.
10. If any error occurred, clear the CREN bit.
11. If the device has been addressed, clear the ADDEN bit to allow all received data into the receive buffer and interrupt the CPU.

FIGURE 15-6: EUSART RECEIVE BLOCK DIAGRAM



18.0 COMPARATOR VOLTAGE REFERENCE MODULE

The comparator voltage reference is a 16-tap resistor ladder network that provides a selectable reference voltage. Its purpose is to provide a reference for the analog comparators.

A block diagram of the module is shown in Figure 18-1. The resistor ladder is segmented to provide two ranges of CVREF values and has a power-down function to conserve power when the reference is not being used. The module's supply reference can be provided from either device VDD/VSS or an external voltage reference.

18.1 Configuring the Comparator Voltage Reference

The voltage reference module is controlled through the CVRCON register (Register 18-1). 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 (CVR3:CVR0), with one range offering finer resolution. The equations used to calculate the output of the comparator voltage reference are as follows:

If CVRR = 1:
CVREF = ((CVR3:CVR0)/24) x CVRSRC

If CVRR = 0:
CVREF = (CVRSRC x 1/4) + (((CVR3:CVR0)/32) x CVRSRC)

The comparator reference supply voltage can come from either AVDD or AVSS, or the external VREF+ that is multiplexed with RA4 and AVSS. The voltage source is selected by the CVRSS bit (CVRCON<4>).

Additionally, the voltage reference can select the unscaled VREF+ input for use by the comparators, bypassing the CVREF module. (See Table 18-1 and Figure 18-1.)

The settling time of the comparator voltage reference must be considered when changing the CVREF output (see Table 23-3 in **Section 23.0 "Electrical Characteristics"**).

TABLE 18-1: VOLTAGE REFERENCE OUTPUT

CVREN	CVRSS	CVREF	Comparator Input
0	0	Disabled	No reference
0	1	Disabled	From VREF (CVREF bypassed)
1	0	Enabled	From CVREF
1	1	Enabled	From CVREF

REGISTER 18-1: CVRCON: COMPARATOR VOLTAGE REFERENCE CONTROL REGISTER

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CVREN	—	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0
bit 7							bit 0

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

bit 7 **CVREN**: Comparator Voltage Reference Enable bit

1 = CVREF circuit powered on
0 = CVREF circuit powered down

bit 6 **Unimplemented:** Read as ‘0’

bit 5 **CVRR**: Comparator VREF Range Selection bit

1 = 0 to 0.667 CVRSRC, with CVRSRC/24 step size (low range)
0 = 0.25 CVRSRC to 0.75 CVRSRC, with CVRSRC/32 step size (high range)

bit 4 **CVRSS**: Comparator VREF Source Selection bit

When CVRR = 1
1 = Comparator reference source, CVRSRC = (VREF+) – (AVSS)
0 = Comparator reference source, CVRSRC = AVDD – AVSS
When CVRR = 0
1 = VREF+ input used directly, comparator voltage reference bypassed
0 = No reference is provided

bit 3-0 **CVR3:CVR0:** Comparator VREF Value Selection bits (0 ≤ (CVR3:CVR0) ≤ 15)

When CVRR = 1:
CVREF = ((CVR3:CVR0)/24) • (CVRSRC)
When CVRR = 0:
CVREF = (CVRSRC/4) + ((CVR3:CVR0)/32) • (CVRSRC)

REGISTER 20-11: CONFIG7L: CONFIGURATION REGISTER 7 LOW (BYTE ADDRESS 30000Ch)

U-0	U-0	U-0	U-0	U-0	U-0	R/C-1	R/C-1
—	—	—	—	—	—	EBTR1[(1)]	EBTR0[(1)]
bit 7							bit 0

Legend:		
R = Readable bit	C = Clearable bit	U = Unimplemented bit, read as ‘0’
-n = Value when device is unprogrammed		u = Unchanged from programmed state

bit 7-2 **Unimplemented:** Read as ‘0’

bit 1 **EBTR1:** Table Read Protection bit (Block 1 Code Memory Area)
1 = Block 1 is not protected from table reads executed in other blocks
0 = Block 1 is protected from table reads executed in other blocks

bit 0 **EBTR0:** Table Read Protection bit (Block 0 Code Memory Area)
1 = Block 0 is not protected from table reads executed in other blocks
0 = Block 0 is protected from table reads executed in other blocks

Note 1: It is recommended to enable the corresponding CPx bit to protect block from external read operations.

REGISTER 20-12: CONFIG7H: CONFIGURATION REGISTER 7 HIGH (BYTE ADDRESS 30000Dh)

U-0	R/C-1	U-0	U-0	U-0	U-0	U-0	U-0
—	EBTRB[(1)]	—	—	—	—	—	—
bit 7							bit 0

Legend:		
R = Readable bit	C = Clearable bit	U = Unimplemented bit, read as ‘0’
-n = Value when device is unprogrammed		u = Unchanged from programmed state

bit 7 **Unimplemented:** Read as ‘0’

bit 6 **EBTRB:** Table Read Protection bit (Boot Block Memory Area)
1 = Boot Block is not protected from table reads executed in other blocks
0 = Boot Block is protected from table reads executed in other blocks

bit 5-0 **Unimplemented:** Read as ‘0’

Note 1: It is recommended to enable the corresponding CPx bit to protect block from external read operations.

FIGURE 22-1: GENERAL FORMAT FOR INSTRUCTIONS

Example Instruction

Byte-oriented file register operations

15 10	9	8	7 0
OPCODE	d	a	f (FILE #)

ADDWF MYREG, W, B

d = 0 for result destination to be WREG register
d = 1 for result destination to be file register (f)
a = 0 to force Access Bank
a = 1 for BSR to select bank
f = 8-bit file register address

Byte to Byte move operations (2-word)

15 12	11 0
OPCODE	f (Source FILE #)
15 12	11 0
1111	f (Destination FILE #)

MOVFF MYREG1, MYREG2

f = 12-bit file register address

Bit-oriented file register operations

15 12	11 9	8	7 0
OPCODE	b (BIT #)	a	f (FILE #)

BSF MYREG, bit, B

b = 3-bit position of bit in file register (f)
a = 0 to force Access Bank
a = 1 for BSR to select bank
f = 8-bit file register address

Literal operations

15 8	7 0
OPCODE	k (literal)

MOVLW 7Fh

k = 8-bit immediate value

Control operations

CALL, GOTO and Branch operations

15 8	7 0
OPCODE	n<7:0> (literal)
15 12	11 0
1111	n<19:8> (literal)

GOTO Label

n = 20-bit immediate value

15	8	7 0
OPCODE	S	n<7:0> (literal)
15 12		11 0
1111		n<19:8> (literal)

CALL MYFUNC

S = Fast bit

15 11	10 0
OPCODE	n<10:0> (literal)

BRA MYFUNC

15 8	7 0
OPCODE	n<7:0> (literal)

BC MYFUNC

DAW Decimal Adjust W Register

Syntax:	DAW
Operands:	None
Operation:	If [W<3:0> > 9] or [DC = 1] then, (W<3:0>) + 6 → W<3:0>; else, (W<3:0>) → W<3:0> If [W<7:4> + DC > 9] or [C = 1] then, (W<7:4>) + 6 + DC → W<7:4> ; else, (W<7:4>) + DC → W<7:4>
Status Affected:	C
Encoding:	0000 0000 0000 0111
Description:	DAW adjusts the eight-bit value in W resulting from the earlier addition of two variables (each in packed BCD format) and produces a correct packed BCD result.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register W	Process Data	Write W

Example 1:

DAW

Before Instruction

W = A5h
C = 0
DC = 0

After Instruction

W = 05h
C = 1
DC = 0

Example 2:

Before Instruction

W = CEh
C = 0
DC = 0

After Instruction

W = 34h
C = 1
DC = 0

DECF Decrement f

Syntax:	DECF f {,d {,a}}
Operands:	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
Operation:	(f) – 1 → dest
Status Affected:	C, DC, N, OV, Z
Encoding:	0000 01da ffff ffff
Description:	Decrement register 'f'. If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register 'f'. If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank. If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 22.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.
Words:	1
Cycles:	1

Q Cycle Activity:

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

Example: DECF CNT, 1, 0

Before Instruction

CNT = 01h
Z = 0

After Instruction

CNT = 00h
Z = 1

GOTO Unconditional Branch

Syntax:	GOTO k
Operands:	$0 \le k \le 1048575$
Operation:	k → PC<20:1>
Status Affected:	None

Encoding:				
1st word (k<7:0>)	1110	1111	$k_7$kkk	kkkk$_0$
2nd word(k<19:8>)	1111	k_{19}kkk	kkkk	kkkk$_8$

Description: `GOTO` allows an unconditional branch anywhere within entire 2-Mbyte memory range. The 20-bit value 'k' is loaded into PC<20:1>. `GOTO` is always a two-cycle instruction.

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'<7:0>,	No operation	Read literal 'k'<19:8>, Write to PC
No operation	No operation	No operation	No operation

Example: `GOTO THERE`

After Instruction
PC = Address (THERE)

INCF Increment f

Syntax:	INCF f {,d {,a}}
Operands:	$0 \le f \le 255$ $d \in [0,1]$ $a \in [0,1]$
Operation:	(f) + 1 → dest
Status Affected:	C, DC, N, OV, Z

Encoding:				
	0010	10da	ffff	ffff

Description: The contents of register 'f' are incremented. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f'.
If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.
If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 22.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: `INCF CNT, 1, 0`

Before Instruction

CNT	=	FFh
Z	=	0
C	=	?
DC	=	?

After Instruction

CNT	=	00h
Z	=	1
C	=	1
DC	=	1

23.1 DC Characteristics: Supply Voltage
PIC18F1230/1330 (Industrial)
PIC18LF1230/1330 (Industrial)

PIC18LF1230/1330 (Industrial)			**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial				
PIC18F1230/1330 (Industrial, Extended)			**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**
D001	VDD	**Supply Voltage**					
		PIC18LF1230/1330	2.0	—	5.5	V	HS, XT, RC and LP Oscillator modes
		PIC18F1230/1330	4.2	—	5.5	V	
D001C	AVDD	**Analog Supply Voltage**	VDD - 0.3	—	VDD + 0.3	V	
D001D	AVSS	**Analog Ground Voltage**	VSS - 0.3	—	VSS + 0.3	V	
D002	VDR	**RAM Data Retention Voltage**[1]	1.5	—	—	V	
D003	VPOR	**VDD Start Voltage** to ensure internal Power-on Reset signal	—	—	0.7	V	See section on Power-on Reset for details
D004	SVDD	**VDD Rise Rate** to ensure internal Power-on Reset signal	0.05	—	—	V/ms	See section on Power-on Reset for details
	VBOR	**Brown-out Reset Voltage**					
D005		PIC18LF1230/1330					
		BORV1:BORV0 = 11	2.00	2.05	2.16	V	
		BORV1:BORV0 = 10	2.65	2.79	2.93	V	
D005		All devices					
		BORV1:BORV0 = 01	4.11[2]	4.33	4.55	V	
		BORV1:BORV0 = 00	4.36	4.59	4.82	V	

Legend: Shading of rows is to assist in readability of the table.

Note 1: This is the limit to which VDD can be lowered in Sleep mode, or during a device Reset, without losing RAM data.

2: With BOR enabled, full-speed operation (FOSC = 40 MHz) is supported until a BOR occurs. This is valid although VDD may be below the minimum voltage for this frequency.

23.2 DC Characteristics: Power-Down and Supply Current PIC18F1230/1330 (Industrial) PIC18LF1230/1330 (Industrial) (Continued)

PIC18LF1230/1330 (Industrial)		**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ T_A ≤ +85°C for industrial					
PIC18F1230/1330 (Industrial, Extended)		**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ T_A ≤ +85°C for industrial -40°C ≤ T_A ≤ +125°C for extended					
Param No.	**Device**	**Typ**	**Max**	**Units**	**Conditions**		
	Module Differential Currents (ΔI_{WDT}, ΔI_{BOR}, ΔI_{LVD}, ΔI_{OSCB}, ΔI_{AD})						
D026 (ΔI_{AD})	**A/D Converter**	1.0	1.6	μA	-40°C to +85°C	V_{DD} = 2.0V	A/D on, not converting
		1.0	1.6	μA	-40°C to +85°C	V_{DD} = 3.0V	
		1.0	1.6	μA	-40°C to +85°C	V_{DD} = 5.0V	
		2.0	7.6	μA	-40°C to +125°C		

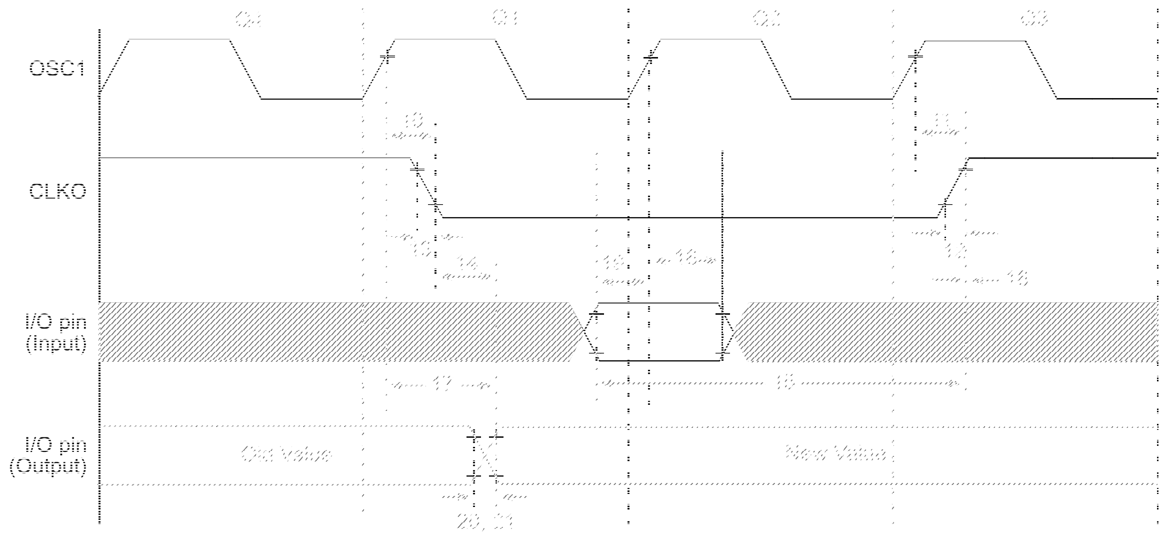
Note 1: The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to V_{DD} or V_{SS} and all features that add delta current disabled (such as WDT, Timer1 oscillator, BOR, etc.).

2: The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.
The test conditions for all I_{DD} measurements in active operation mode are:
OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to V_{DD} or V_{SS};
$\overline{MCLR}$ = V_{DD}; WDT enabled/disabled as specified.

3: Low-power Timer1 oscillator selected.

4: BOR and LVD enable internal band gap reference. With both modules enabled, current consumption will be less than the sum of both specifications.

FIGURE 23-7: CLKO AND I/O TIMING



Note: Refer to Figure 23-5 for load conditions.

TABLE 23-9: CLKO AND I/O TIMING REQUIREMENTS

Param No.	Symbol	Characteristic		Min	Typ	Max	Units	Conditions
10	TosH2ckL	OSC1 ↑ to CLKO ↓		—	75	200	ns	**(Note 1)**
11	TosH2ckH	OSC1 ↑ to CLKO ↑		—	75	200	ns	**(Note 1)**
12	TckR	CLKO Rise Time		—	35	100	ns	**(Note 1)**
13	TckF	CLKO Fall Time		—	35	100	ns	**(Note 1)**
14	TckL2ioV	CLKO ↓ to Port Out Valid		—	—	0.5 TCY + 20	ns	**(Note 1)**
15	TioV2ckH	Port In Valid before CLKO ↑		0.25 TCY + 25	—	—	ns	**(Note 1)**
16	TckH2ioI	Port In Hold after CLKO ↑		0	—	—	ns	**(Note 1)**
17	TosH2ioV	OSC1 ↑ (Q1 cycle) to Port Out Valid		—	50	150	ns	
18	TosH2ioI	OSC1 ↑ (Q2 cycle) to Port Input Invalid (I/O in hold time)	PIC18**F**XXXX	100	—	—	ns	
18A			PIC18**LF**XXXX	200	—	—	ns	VDD = 2.0V
19	TioV2osH	Port Input Valid to OSC1 ↑ (I/O in setup time)		0	—	—	ns	
20	TioR	Port Output Rise Time	PIC18**F**XXXX	—	10	25	ns	
20A			PIC18**LF**XXXX	—	—	60	ns	VDD = 2.0V
21	TioF	Port Output Fall Time	PIC18**F**XXXX	—	10	25	ns	
21A			PIC18**LF**XXXX	—	—	60	ns	VDD = 2.0V
22†	TINP	INTx Pin High or Low Time		TCY	—	—	ns	
23†	TRBP	RB7:RB4 Change INTx High or Low Time		TCY	—	—	ns	

† These parameters are asynchronous events not related to any internal clock edges.

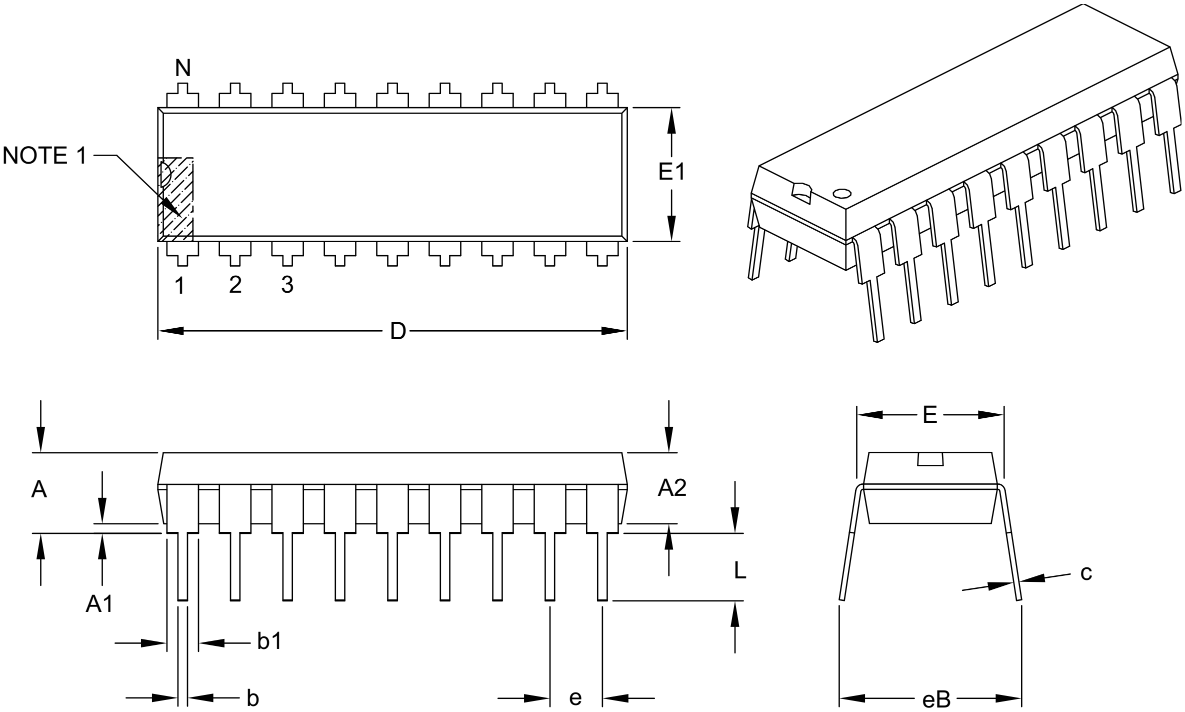
Note 1: Measurements are taken in RC mode, where CLKO output is 4 x TOSC.

24.2 Package Details

The following sections give the technical details of the packages.

18-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	18		
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	.880	.900	.920
Tip to Seating Plane	L	.115	.130	.150
Lead Thickness	c	.008	.010	.014
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 C04-007B

APPENDIX B: DEVICE DIFFERENCES

The differences between the devices listed in this data sheet are shown in Table B-1.

TABLE B-1: DEVICE DIFFERENCES

Features	PIC18F1230	PIC18F1330
Program Memory (Bytes)	4096	8192
Program Memory (Instructions)	2048	4096
Packages	18-Pin PDIP 18-Pin SOIC 20-Pin SSOP 28-Pin QFN	18-Pin PDIP 18-Pin SOIC 20-Pin SSOP 28-Pin QFN

APPENDIX E: MIGRATION FROM MID-RANGE TO ENHANCED DEVICES

A detailed discussion of the differences between the mid-range MCU devices (i.e., PIC16CXXX) and the Enhanced devices (i.e., PIC18FXXX) is provided in *AN716, "Migrating Designs from PIC16C74A/74B to PIC18C442"*. The changes discussed, while device specific, are generally applicable to all mid-range to Enhanced device migrations.

This Application Note is available as Literature Number DS00716.

APPENDIX F: MIGRATION FROM HIGH-END TO ENHANCED DEVICES

A detailed discussion of the migration pathway and differences between the high-end MCU devices (i.e., PIC17CXXX) and the Enhanced devices (i.e., PIC18FXXX) is provided in *AN726, "PIC17CXXX to PIC18CXXX Migration"*.

This Application Note is available as Literature Number DS00726.