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	64MHz
Connectivity	ECANbus, I²C, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, LVD, POR, PWM, WDT
Number of I/O	24
Program Memory Size	32KB (16K x 16)
Program Memory Type	FLASH
EEPROM Size	1K x 8
RAM Size	3.6K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 3.6V
Data Converters	A/D 8x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	28-SSOP (0.209", 5.30mm Width)
Supplier Device Package	28-SSOP
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18lf25k80-i-ss

REGISTER 4-3: PMD0: PERIPHERAL MODULE DISABLE REGISTER 0

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CCP5MD	CCP4MD	CCP3MD	CCP2MD	CCP1MD	UART2MD	UART1MD	SSPMD
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 **CCP5MD:** CCP5 Module Disable bit
1 = The CCP5 module is disabled; all CCP5 registers are held in Reset and are not writable
0 = The CCP5 module is enabled

bit 6 **CCP4MD:** CCP4 Module Disable bit
1 = The CCP4 module is disabled; all CCP4 registers are held in Reset and are not writable
0 = The CCP4 module is enabled

bit 5 **CCP3MD:** CCP3 Module Disable bit
1 = The CCP3 module is disabled; all CCP3 registers are held in Reset and are not writable
0 = The CCP3 module is enabled

bit 4 **CCP2MD:** CCP2 Module Disable bit
1 = The CCP2 module is disabled; all CCP2 registers are held in Reset and are not writable
0 = The CCP2 module is enabled

bit 3 **CCP1MD:** ECCP1 Module Disable bit
1 = The ECCP1 module is disabled; all ECCP1 registers are held in Reset and are not writable
0 = The ECCP1 module is enabled

bit 2 **UART2MD:** EUSART2 Module Disable bit
1 = The USART2 module is disabled; all USART2 registers are held in Reset and are not writable
0 = The USART2 module is enabled

bit 1 **UART1MD:** EUSART1 Module Disable bit
1 = The USART1 module is disabled; all USART1 registers are held in Reset and are not writable
0 = The USART1 module is enabled

bit 0 **SSPMD:** MSSP Module Disable bit
1 = The MSSP module is disabled; all SSP registers are held in Reset and are not writable
0 = The MSSP module is enabled

6.1 Program Memory Organization

PIC18 microcontrollers implement a 21-bit Program Counter (PC) that is capable of addressing a 2-Mbyte program memory space. Accessing a location between the upper boundary of the physically implemented memory and the 2-Mbyte address will return all '0's (a NOP instruction).

The entire PIC18F66K80 family offers a range of on-chip Flash program memory sizes, from 32 Kbytes (16,384 single-word instructions) to 64 Kbytes (32,768 single-word instructions).

- PIC18F25K80, PIC18F45K80 and PIC18F65K80 – 32 Kbytes of Flash memory, storing up to 16,384 single-word instructions
- PIC18F26K80, PIC18F46K80 and PIC18F66K80 – 64 Kbytes of Flash memory, storing up to 32,768 single-word instructions

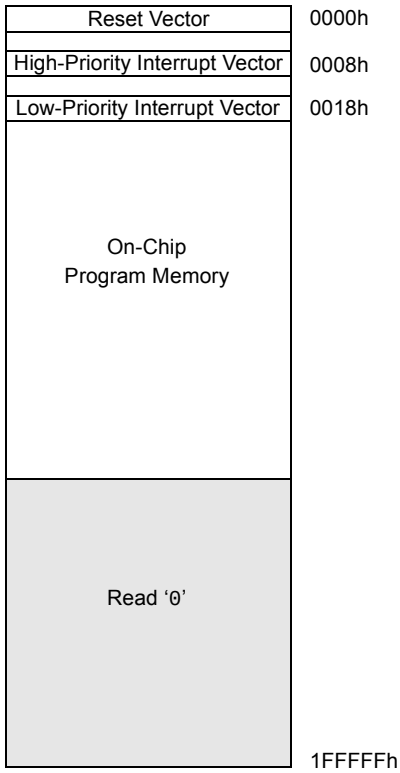
The program memory maps for individual family members are shown in Figure 6-1.

6.1.1 HARD MEMORY VECTORS

All PIC18 devices have a total of three hard-coded return vectors in their program memory space. The Reset vector address is the default value to which the Program Counter returns on all device Resets. It is located at 0000h.

PIC18 devices also have two interrupt vector addresses for handling high-priority and low-priority interrupts. The high-priority interrupt vector is located at 0008h and the low-priority interrupt vector is at 0018h. The locations of these vectors are shown, in relation to the program memory map, in Figure 6-2.

FIGURE 6-2: HARD VECTOR FOR PIC18F66K80 FAMILY DEVICES



Legend: (Top of Memory) represents upper boundary of on-chip program memory space (see Figure 6-1 for device-specific values). Shaded area represents unimplemented memory. Areas are not shown to scale.

10.9 Context Saving During Interrupts

During interrupts, the return PC address is saved on the stack. Additionally, the WREG, STATUS and BSR registers are saved on the Fast Return Stack.

If a fast return from interrupt is not used (see **Section 6.3 "Data Memory Organization"**), the user may need to save the WREG, STATUS and BSR registers on entry to the Interrupt Service Routine (ISR). Depending on the user's application, other registers also may need to be saved.

Example 10-1 saves and restores the WREG, STATUS and BSR registers during an Interrupt Service Routine.

EXAMPLE 10-1: SAVING STATUS, WREG AND BSR REGISTERS IN RAM

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

TABLE 10-1: SUMMARY OF REGISTERS ASSOCIATED WITH INTERRUPTS

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
INTCON2	$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	RBIP
INTCON3	INT2IP	INT1IP	INT3IE	INT2IE	INT1IE	INT3IF	INT2IF	INT1IF
PIR1	PSPIP	ADIF	RC1IF	TX1IF	SSPIF	TMR1GIF	TMR2IF	TMR1IF
PIR2	OSCFIF	—	—	—	BCLIF	HLVDIF	TMR3IF	TMR3GIF
PIR3	—	—	RC2IF	TX2IF	CTMUIF	CCP2IF	CCP1IF	—
PIR4	TMR4IF	EEIF	CMP2IF	CMP1IF	—	CCP5IF	CCP4IF	CCP3IF
PIR5	IRXIF	WAKIF	ERRIF	TXB2IF	TXB1IF	TXB0IF	RXB1IF	RXB0IF
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSPIE	TMR1GIE	TMR2IE	TMR1IE
PIE2	OSCFIE	—	—	—	BCLIE	HLVDIE	TMR3IE	TMR3GIE
PIE3	—	—	RC2IE	TX2IE	CTMUIE	CCP2IE	CCP1IE	—
PIE4	TMR4IE	EEIE	CCP2IE	CMP1IE	—	CCP5IE	CCP4IE	CCP3IE
PIE5	IRXIE	WAKIE	ERRIE	TXB2IE	TXB1IE	TXB0IE	RXB1IE	RXB0IE
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSPIP	TMR1GIP	TMR2IP	TMR1IP
IPR2	OSCFIP	—	—	—	BCLIP	HLVDIP	TMR3IP	TMR3GIP
IPR3	—	—	RC2IP	TX2IP	CTMUIP	CCP2IP	CCP1IP	—
IPR4	TMR4IP	EEIP	CMP2IP	CMP1IP	—	CCP5IP	CCP4IP	CCP3IP
IPR5	IRXIP	WAKIP	ERRIP	TXB2IP	TXB1IP	TXB0IP	RXB1IP	RXB0IP
RCON	IPEN	SBOREN	$\overline{\text{CM}}$	$\overline{\text{RI}}$	$\overline{\text{TO}}$	$\overline{\text{PD}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$

Legend: Shaded cells are not used by the interrupts.

13.1 Timer0 Operation

Timer0 can operate as either a timer or a counter. The mode is selected with the T0CS bit (T0CON<5>). In Timer mode (T0CS = 0), the module increments on every clock by default unless a different prescaler value is selected (see **Section 13.3 "Prescaler"**). If the TMR0 register is written to, the increment is inhibited for the following two instruction cycles. The user can work around this by writing an adjusted value to the TMR0 register.

The Counter mode is selected by setting the T0CS bit (= 1). In this mode, Timer0 increments either on every rising edge or falling edge of the T0CKI pin. The incrementing edge is determined by the Timer0 Source Edge Select bit, T0SE (T0CON<4>); clearing this bit selects the rising edge. Restrictions on the external clock input are discussed below.

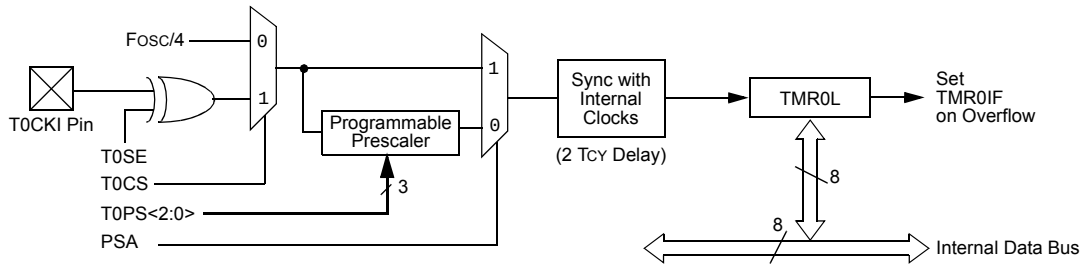
An external clock source can be used to drive Timer0; however, it must meet certain requirements to ensure that the external clock can be synchronized with the internal phase clock (TOSC). There is a delay between synchronization and the onset of incrementing the timer/counter.

13.2 Timer0 Reads and Writes in 16-Bit Mode

TMR0H is not the actual high byte of Timer0 in 16-bit mode. It is actually a buffered version of the real high byte of Timer0, which is not directly readable nor writable. (See Figure 13-2.) TMR0H is updated with the contents of the high byte of Timer0 during a read of TMR0L. This provides the ability to read all 16 bits of Timer0 without having to verify that the read of the high and low byte were valid, due to a rollover between successive reads of the high and low byte.

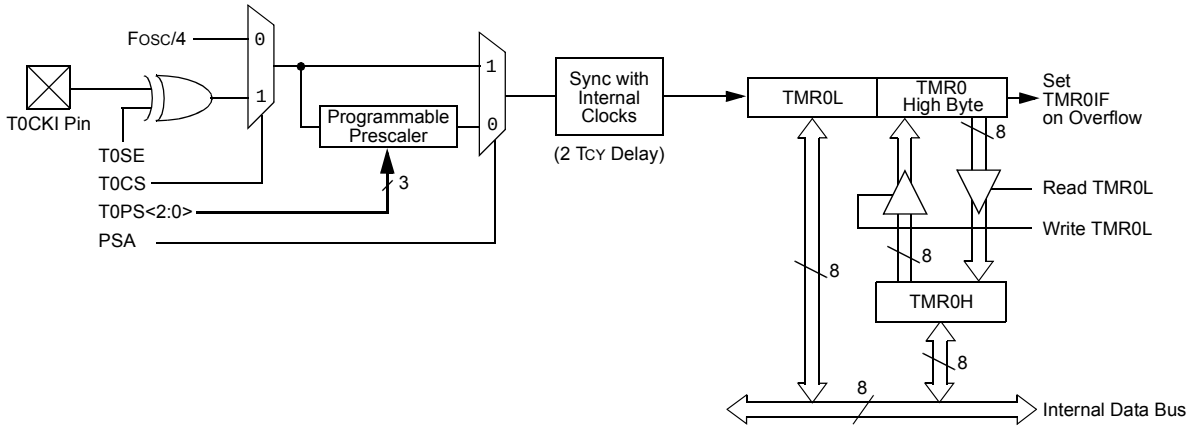
Similarly, a write to the high byte of Timer0 must also take place through the TMR0H Buffer register. The high byte is updated with the contents of TMR0H when a write occurs to TMR0L. This allows all 16 bits of Timer0 to be updated at once.

FIGURE 13-1: TIMER0 BLOCK DIAGRAM (8-BIT MODE)



Note: Upon Reset, Timer0 is enabled in 8-bit mode with clock input from T0CKI max. prescale.

FIGURE 13-2: TIMER0 BLOCK DIAGRAM (16-BIT MODE)



Note: Upon Reset, Timer0 is enabled in 8-bit mode with clock input from T0CKI max. prescale.

16.5.2 TIMER3 GATE SOURCE SELECTION

The Timer3 gate source can be selected from one of four different sources. Source selection is controlled by the T3GSS<1:0> bits (T3GCON<1:0>). The polarity for each available source is also selectable and is controlled by the T3GPOL bit (T3GCON<6>).

TABLE 16-2: TIMER3 GATE SOURCES

T3GSS<1:0>	Timer3 Gate Source
00	Timerx Gate Pin
01	TMR4 to Match PR4 (TMR4 increments to match PR4)
10	Comparator 1 Output (comparator logic high output)
11	Comparator 2 Output (comparator logic high output)

16.5.2.1 T3G Pin Gate Operation

The T3G pin is one source for Timer3 gate control. It can be used to supply an external source to the Timerx gate circuitry.

16.5.2.2 Timer4 Match Gate Operation

The TMR4 register will increment until it matches the value in the PR4 register. On the very next increment cycle, TMR4 will be reset to 00h. When this Reset occurs, a low-to-high pulse will automatically be generated and internally supplied to the Timerx gate circuitry. The pulse will remain high for one instruction cycle and will return back to a low state until the next match.

Depending on T3GPOL, Timerx increments differently when TMR4 matches PR4. When T3GPOL = 1, Timer3 increments for a single instruction cycle following a TMR4 match with PR4. When T3GPOL = 0, Timer3 increments continuously, except for the cycle following the match, when the gate signal goes from low-to-high.

16.5.2.3 Comparator 1 Output Gate Operation

The output of Comparator 1 can be internally supplied to the Timer3 gate circuitry. After setting up Comparator 1 with the CM1CON register, Timer3 will increment depending on the transitions of the CMP1OUT (CMSTAT<6>) bit.

16.5.2.4 Comparator 2 Output Gate Operation

The output of Comparator 2 can be internally supplied to the Timer3 gate circuitry. After setting up Comparator 2 with the CM2CON register, Timer3 will increment depending on the transitions of the CMP2OUT (CMSTAT<7>) bit.

16.5.3 TIMER3 GATE TOGGLE MODE

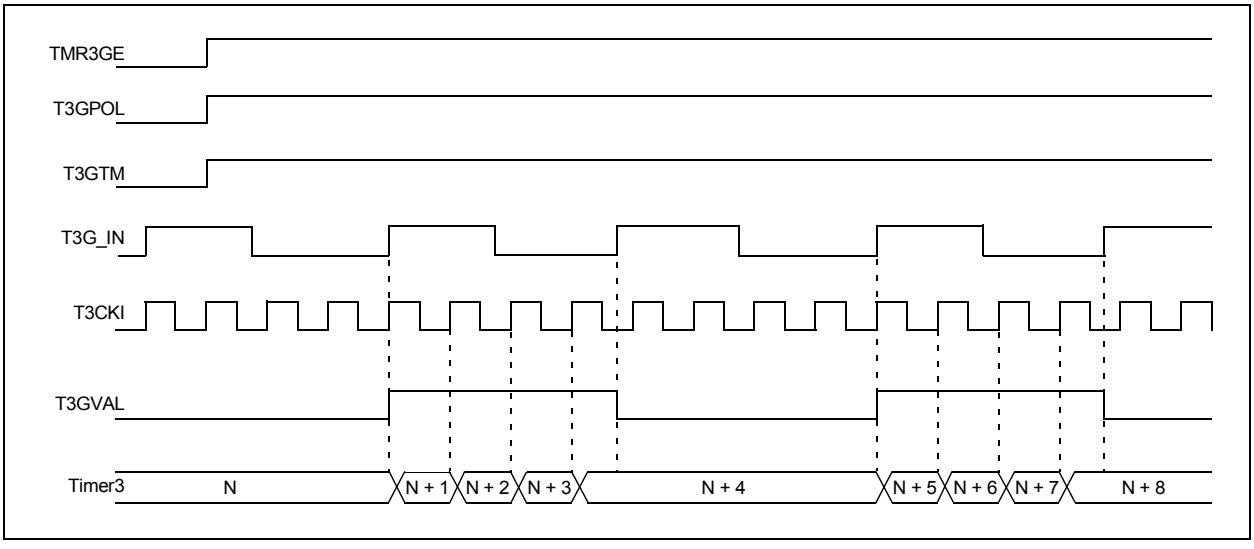
When Timer3 Gate Toggle mode is enabled, it is possible to measure the full cycle length of a Timer3 gate signal, as opposed to the duration of a single level pulse.

The Timer3 gate source is routed through a flip-flop that changes state on every incrementing edge of the signal. (For timing details, see Figure 16-3.)

The T3GVAL bit will indicate when the Toggled mode is active and the timer is counting.

Timer3 Gate Toggle mode is enabled by setting the T3GTM bit (T3GCON<5>). When the T3GTM bit is cleared, the flip-flop is cleared and held clear. This is necessary in order to control which edge is measured.

FIGURE 16-3: TIMER3 GATE TOGGLE MODE



REGISTER 18-3: CTMUICON: CTMU CURRENT CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ITRIM5	ITRIM4	ITRIM3	ITRIM2	ITRIM1	ITRIM0	IRNG1	IRNG0
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 7-2 **ITRIM<5:0>:** Current Source Trim bits

011111 = Maximum positive change (+62% typ.) from nominal current
011110
.
.
.
000001 = Minimum positive change (+2% typ.) from nominal current
000000 = Nominal current output specified by IRNG<1:0>
111111 = Minimum negative change (-2% typ.) from nominal current
.
.
.
100010
100001 = Maximum negative change (-62% typ.) from nominal current

bit 1-0 **IRNG<1:0>:** Current Source Range Select bits

11 = 100 x Base Current
10 = 10 x Base Current
01 = Base Current level (0.55 μA nominal)
00 = Current source is disabled

EXAMPLE 18-3: CAPACITANCE CALIBRATION ROUTINE

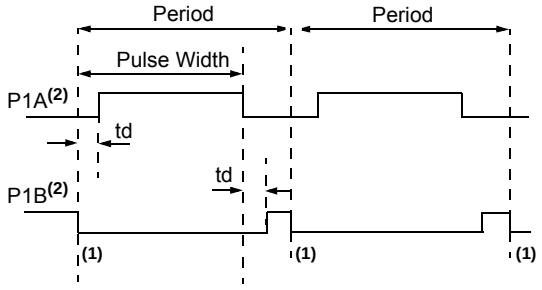
```
#include "p18cxxx.h"

#define COUNT 25                        //@ 8MHz INTFRC = 62.5 us.
#define ETIME COUNT*2.5                 //time in uS
#define DELAY for(i=0;i<COUNT;i++)
#define ADSCALE 1023                    //for unsigned conversion 10 sig bits
#define ADREF 3.3                       //Vdd connected to A/D Vr+
#define RCAL .027                       //R value is 4200000 (4.2M)
                                        //scaled so that result is in
                                        //1/100th of uA

int main(void)
{
    int i;
    int j = 0;                          //index for loop
    unsigned int Vread = 0;
    float CTMUISrc, CTMUCap, Vavg, VTot, Vcal;

//assume CTMU and A/D have been setup correctly
//see Example 25-1 for CTMU & A/D setup
setup();

CTMUCONHbits.CTMUEN = 1;                //Enable the CTMU
    for(j=0;j<10;j++)
    {
        CTMUCONHbits.IDISSEN = 1;       //drain charge on the circuit
        DELAY;                          //wait 125us
        CTMUCONHbits.IDISSEN = 0;       //end drain of circuit

        CTMUCONLbits.EDG1STAT = 1;      //Begin charging the circuit
                                        //using CTMU current source
        DELAY;                          //wait for 125us
        CTMUCONLbits.EDG1STAT = 0;      //Stop charging circuit

        PIR1bits.ADIF = 0;              //make sure A/D Int not set
        ADCON0bits.GO=1;                //and begin A/D conv.
        while(!PIR1bits.ADIF);          //Wait for A/D convert complete


        Vread = ADRES;                  //Get the value from the A/D
        PIR1bits.ADIF = 0;              //Clear A/D Interrupt Flag
        VTot += Vread;                  //Add the reading to the total
    }

    Vavg = (float)(VTot/10.000);        //Average of 10 readings
    Vcal = (float)(Vavg/ADSCALE*ADREF);
    CTMUISrc = Vcal/RCAL;               //CTMUISrc is in 1/100ths of uA
    CTMUCap = (CTMUISrc*ETIME/Vcal)/100;
}
```

20.4.6 PROGRAMMABLE DEAD-BAND DELAY MODE

In half-bridge applications, where all power switches are modulated at the PWM frequency, the power switches normally require more time to turn off than to turn on. If both the upper and lower power switches are switched at the same time (one turned on and the other turned off), both switches may be on for a short period until one switch completely turns off. During this brief interval, a very high current (shoot-through current) will flow through both power switches, shorting the bridge supply. To avoid this potentially destructive shoot-through current from flowing during switching, turning on either of the power switches is normally delayed to allow the other switch to completely turn off.

In Half-Bridge mode, a digitally programmable dead-band delay is available to avoid shoot-through current from destroying the bridge power switches. The delay occurs at the signal transition from the non-active state to the active state. For an illustration, see Figure 20-14. The lower seven bits of the associated ECCP1DEL register (Register 20-4) set the delay period in terms of microcontroller instruction cycles (TCY or 4 TOSC).

FIGURE 20-14: EXAMPLE OF HALF-BRIDGE PWM OUTPUT

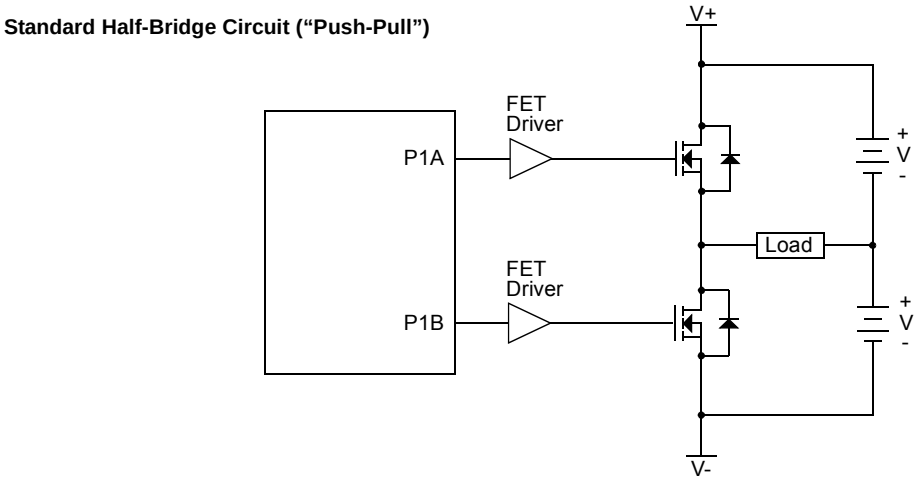


td = Dead-Band Delay

Note 1: At this time, the TMR2 register is equal to the PR2 register.

2: Output signals are shown as active-high.

FIGURE 20-15: EXAMPLE OF HALF-BRIDGE APPLICATIONS



REGISTER 21-2: SSPCON1: MSSP CONTROL REGISTER 1 (SPI MODE)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPOV(1)	SSPEN(2)	CKP	SSPM3(3)	SSPM2(3)	SSPM1(3)	SSPM0(3)
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 **WCOL:** Write Collision Detect bit
- 1 = The SSPBUF register is written while it is still transmitting the previous word (must be cleared in software)
- 0 = No collision

bit 6 **SSPOV:** Receive Overflow Indicator bit(1)

SPI Slave mode:
- 1 = A new byte is received while the SSPBUF register is still holding the previous data. In case of overflow, the data in SSPSR is lost. Overflow can only occur in Slave mode. The user must read the SSPBUF, even if only transmitting data, to avoid setting overflow (must be cleared in software).
- 0 = No overflow

bit 5 **SSPEN:** Master Synchronous Serial Port Enable bit(2)
- 1 = Enables the serial port and configures SCK, SDO, SDI and $\overline{SS}$ as serial port pins
- 0 = Disables the serial port and configures these pins as I/O port pins

bit 4 **CKP:** Clock Polarity Select bit
- 1 = Idle state for clock is a high level
- 0 = Idle state for clock is a low level

bit 3-0 **SSPM<3:0>:** Master Synchronous Serial Port Mode Select bits(3)
- 1010 = SPI Master mode: clock = FOSC/8
- 0101 = SPI Slave mode: clock = SCK pin; $\overline{SS}$ pin control disabled; $\overline{SS}$ can be used as I/O pin
- 0100 = SPI Slave mode: clock = SCK pin; $\overline{SS}$ pin control enabled
- 0011 = SPI Master mode: clock = TMR2 output/2
- 0010 = SPI Master mode: clock = FOSC/64
- 0001 = SPI Master mode: clock = FOSC/16
- 0000 = SPI Master mode: clock = FOSC/4

Note 1: In Master mode, the overflow bit is not set since each new reception (and transmission) is initiated by writing to the SSPBUF register.

2: When enabled, these pins must be properly configured as inputs or outputs.

3: Bit combinations not specifically listed here are either reserved or implemented in I^2C mode only.

21.3.6 MASTER MODE

The master can initiate the data transfer at any time because it controls the SCK. The master determines when the slave (Processor 1, Figure 21-2) is to broadcast data by the software protocol.

In Master mode, the data is transmitted/received as soon as the SSPBUF register is written to. If the SPI is only going to receive, the SDO output could be disabled (programmed as an input). The SSPSR register will continue to shift in the signal present on the SDI pin at the programmed clock rate. As each byte is received, it will be loaded into the SSPBUF register as if a normal received byte (interrupts and status bits appropriately set). This could be useful in receiver applications as a “Line Activity Monitor” mode.

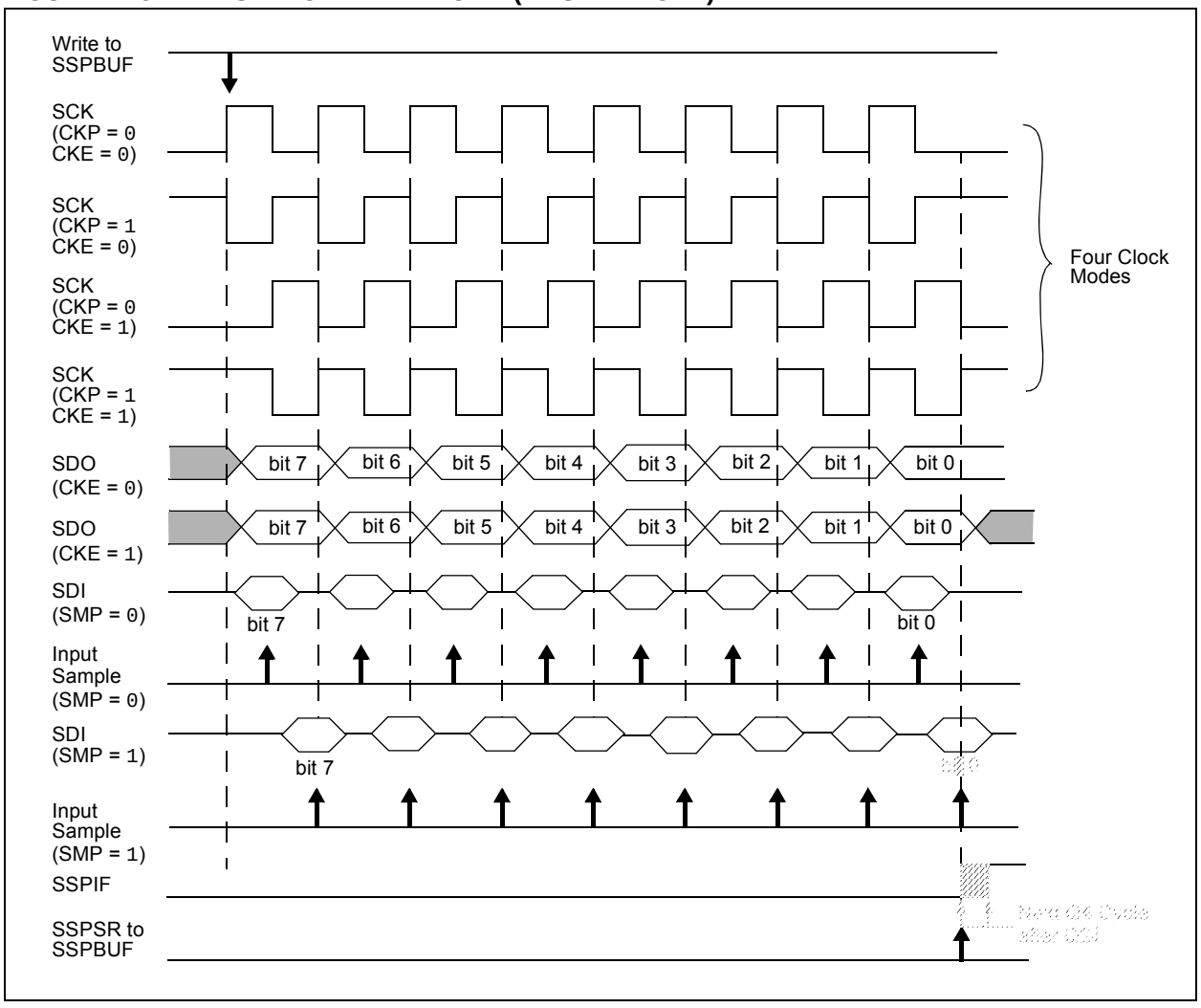
The clock polarity is selected by appropriately programming the CKP bit (SSPCON1<4>). This then, would give waveforms for SPI communication as shown in Figure 21-3, Figure 21-5 and Figure 21-6, where the MSB is transmitted first. In Master mode, the SPI clock rate (bit rate) is user-programmable to be one of the following:

- FOSC/4 (or TCY)
- FOSC/16 (or 4 • TCY)
- FOSC/64 (or 16 • TCY)
- Timer2 output/2

This allows a maximum data rate (at 64 MHz) of 16 Mbps.

Figure 21-3 shows the waveforms for Master mode. When the CKE bit is set, the SDO data is valid before there is a clock edge on SCK. The change of the input sample is shown based on the state of the SMP bit. The time when the SSPBUF is loaded with the received data is shown.

FIGURE 21-3: SPI MODE WAVEFORM (MASTER MODE)



22.0 ENHANCED UNIVERSAL SYNCHRONOUS ASYNCHRONOUS RECEIVER TRANSMITTER (EUSART)

The Enhanced Universal Synchronous Asynchronous Receiver Transmitter (EUSART) module is one of two serial I/O modules. (Generically, the EUSART is also known as a Serial Communications Interface or SCI.)

The EUSART can be configured as a full-duplex, asynchronous system that can communicate with peripheral devices, such as CRT terminals and personal computers. It can also be configured as a half-duplex synchronous system that can communicate with peripheral devices, such as A/D or D/A integrated circuits, serial EEPROMs, etc.

The Enhanced USARTx modules implement additional features, including automatic baud rate detection and calibration, automatic wake-up on Sync Break reception and 12-bit Break character transmit. These make it ideally suited for use in Local Interconnect Network bus (LIN/J2602 bus) systems.

All members of the PIC18F66K80 family are equipped with two independent EUSART modules, referred to as EUSART1 and EUSART2. They can be configured in the following modes:

- Asynchronous (full duplex) with:
  - Auto-wake-up on character reception
  - Auto-baud calibration
  - 12-bit Break character transmission
- Synchronous – Master (half duplex) with selectable clock polarity
- Synchronous – Slave (half duplex) with selectable clock polarity

The pins of EUSART1 and EUSART2 are multiplexed with the functions with the following ports, depending on the device pin count. See Table 22-1.

TABLE 22-1: CONFIGURING EUSARTx PINS(1)

Pin Count	EUSART1		EUSART2	
	Port	Pins	Port	Pins
28-pin	PORTC	RC6/TX1/CK1 and RC7/RX1/DT1	PORTB	RB6/PGC/TX2/CK2/KBI2 and RB7/PGD/T3G/RX2/DT2/KBI3
40/44-pin	PORTC	RC6/TX1/CK1 and RC7/RX1/DT1	PORTD	RD6/TX2/CK2/P1C/PSP6 and RD7/RX2/DT2/P1D/PSP7
64-pin	PORTG	RG3/TX1/CK1 and RG0/RX1/DT1	PORTE	RE7/TX2/CK2 and RE6/RX2/DT2

Note 1: The EUSARTx control will automatically reconfigure the pin from input to output as needed.

In order to configure the pins as an EUSARTx:

- For EUSART1:
  - SPEN (RCSTA1<7>) must be set (= 1)
  - TRISx<x> must be set (= 1)
  - For Asynchronous and Synchronous Master modes, TRISx<x> must be cleared (= 0)
  - For Synchronous Slave mode, TRISx<x> must be set (= 1)
- For EUSART2:
  - SPEN (RCSTA2<7>) must be set (= 1)
  - TRISx<x> must be set (= 1)
  - For Asynchronous and Synchronous Master modes, TRISx<x> must be cleared (= 0)
  - For Synchronous Slave mode, TRISx<x> must be set (= 1)

TABLE 22-4: BAUD RATES FOR ASYNCHRONOUS MODES (CONTINUED)

BAUD RATE (K)	SYNC = 0, BRGH = 0, BRG16 = 1											
	FOSC = 64.000 MHz			FOSC = 40.000 MHz			FOSC = 20.000 MHz			FOSC = 10.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	0.300	0.00	13332	0.300	0.00	8332	0.300	0.02	4165	0.300	0.02	2082
1.2	1.200	0.00	3332	1.200	0.02	2082	1.200	-0.03	1041	1.200	-0.03	520
2.4	2.400	0.00	1666	2.402	0.06	1040	2.399	-0.03	520	2.404	0.16	259
9.6	9.592	-0.08	416	9.615	0.16	259	9.615	0.16	129	9.615	0.16	64
19.2	19.417	1.13	207	19.231	0.16	129	19.231	0.16	64	19.531	1.73	31
57.6	59.701	3.65	68	58.140	0.94	42	56.818	-1.36	21	56.818	-1.36	10
115.2	121.212	5.22	34	113.636	-1.36	21	113.636	-1.36	10	125.000	8.51	4

BAUD RATE (K)	SYNC = 0, BRGH = 0, BRG16 = 1											
	FOSC = 8.000 MHz			FOSC = 4.000 MHz			FOSC = 2.000 MHz			FOSC = 1.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	0.300	-0.04	1665	0.300	0.04	832	0.300	-0.16	415	0.300	-0.16	207
1.2	1.201	-0.16	415	1.202	0.16	207	1.201	-0.16	103	1.201	-0.16	51
2.4	2.403	-0.16	207	2.404	0.16	103	2.403	-0.16	51	2.403	-0.16	25
9.6	9.615	-0.16	51	9.615	0.16	25	9.615	-0.16	12	—	—	—
19.2	19.230	-0.16	25	19.231	0.16	12	—	—	—	—	—	—
57.6	55.555	3.55	8	62.500	8.51	3	—	—	—	—	—	—
115.2	—	—	—	125.000	8.51	1	—	—	—	—	—	—

BAUD RATE (K)	SYNC = 0, BRGH = 1, BRG16 = 1 or SYNC = 1, BRG16 = 1											
	FOSC = 64.000 MHz			FOSC = 40.000 MHz			FOSC = 20.000 MHz			FOSC = 10.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	0.300	0.00	53332	0.300	0.00	33332	0.300	0.00	16665	0.300	0.00	8332
1.2	1.200	0.00	13332	1.200	0.00	8332	1.200	0.02	4165	1.200	0.02	2082
2.4	2.400	0.00	6666	2.400	0.02	4165	2.400	0.02	2082	2.402	0.06	1040
9.6	9.598	-0.02	1666	9.606	0.06	1040	9.596	-0.03	520	9.615	0.16	259
19.2	19.208	0.04	832	19.193	-0.03	520	19.231	0.16	259	19.231	0.16	129
57.6	57.348	-0.44	278	57.803	0.35	172	57.471	-0.22	86	58.140	0.94	42
115.2	115.108	-0.08	138	114.943	-0.22	86	116.279	0.94	42	113.636	-1.36	21

BAUD RATE (K)	SYNC = 0, BRGH = 1, BRG16 = 1 or SYNC = 1, BRG16 = 1											
	FOSC = 8.000 MHz			FOSC = 4.000 MHz			FOSC = 2.000 MHz			FOSC = 1.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	0.300	-0.01	6665	0.300	0.01	3332	0.300	-0.04	1665	0.300	-0.04	832
1.2	1.200	-0.04	1665	1.200	0.04	832	1.201	-0.16	415	1.201	-0.16	207
2.4	2.400	-0.04	832	2.404	0.16	415	2.403	-0.16	207	2.403	-0.16	103
9.6	9.615	-0.16	207	9.615	0.16	103	9.615	-0.16	51	9.615	-0.16	25
19.2	19.230	-0.16	103	19.231	0.16	51	19.230	-0.16	25	19.230	-0.16	12
57.6	57.142	0.79	34	58.824	2.12	16	55.555	3.55	8	—	—	—
115.2	117.647	-2.12	16	111.111	-3.55	8	—	—	—	—	—	—

REGISTER 23-2: ADCON1: A/D CONTROL REGISTER 1

R/W-0	R/W-0	R/W-0	R/W-0	R/W-x	R/W-x	R/W-x	R/W-x
TRIGSEL1	TRIGSEL0	VCFG1	VCFG0	VNCFG	CHSN2	CHSN1	CHSN0
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 7-6 **TRIGSEL<1:0>:** Special Trigger Select bits
11 = Selects the special trigger from the CCP2
10 = Selects the special trigger from the Timer1
01 = Selects the special trigger from the CTMU
00 = Selects the special trigger from the ECCP1

bit 5-4 **VCFG<1:0>:** A/D V_{REF}+ Configuration bits
11 = Internal V_{REF}+ (4.1V)
10 = Internal V_{REF}+ (2.0V)
01 = External V_{REF}+
00 = AV_{DD}

bit 3 **VNCFG:** A/D V_{REF}- Configuration bit
1 = External V_{REF}
0 = AV_{SS}

bit 2-0 **CHSN<2:0>:** Analog Negative Channel Select bits
111 = Channel 07 (AN6)
110 = Channel 06 (AN5)
101 = Channel 05 (AN4)
100 = Channel 04 (AN3)
011 = Channel 03 (AN2)
010 = Channel 02 (AN1)
001 = Channel 01 (AN0)
000 = Channel 00 (AV_{SS})

23.8 Use of the Special Event Triggers

A/D conversion can be started by the Special Event Trigger of any of these modules:

- CCP2 – Requires CCP2M<3:0> bits (CCP2CON<3:0>) set at '1011'(†)
- ECCP1
- CTMU – Requires the setting of the CTTRIG bit (CTMUCONH<0>)
- Timer1

To start an A/D conversion:

- The A/D module must be enabled (ADON = 1)
- The appropriate analog input channel selected
- The minimum acquisition period set one of these ways:
  - Timing provided by the user
  - Selection made of an appropriate TACQ time

With these conditions met, the trigger sets the GO/$\overline{\text{DONE}}$ bit and the A/D acquisition starts.

If the A/D module is not enabled (ADON = 0), the module ignores the Special Event Trigger.

Note:	With an ECCP1 or CCP2 trigger, Timer1 or Timer3 is cleared. The timers reset to automatically repeat the A/D acquisition period with minimal software overhead (moving ADRESH:ADRESL to the desired location). If the A/D module is not enabled, the Special Event Trigger is ignored by the module, but the timer's counter resets.

23.9 Operation in Power-Managed Modes

The selection of the automatic acquisition time and A/D conversion clock is determined, in part, by the clock source and frequency while in a power-managed mode.

If the A/D is expected to operate while the device is in a power-managed mode, the ACQT<2:0> and ADCS<2:0> bits in ADCON2 should be updated in accordance with the power-managed mode clock that will be used.

After the power-managed mode is entered (either of the power-managed Run modes), an A/D acquisition or conversion may be started. Once an acquisition or conversion is started, the device should continue to be clocked by the same power-managed mode clock source until the conversion has been completed. If desired, the device may be placed into the corresponding power-managed Idle mode during the conversion.

If the power-managed mode clock frequency is less than 1 MHz, the A/D RC clock source should be selected.

Operation in Sleep mode requires that the A/D RC clock be selected. If bits, ACQT<2:0>, are set to '000' and a conversion is started, the conversion will be delayed one instruction cycle to allow execution of the SLEEP instruction and entry into Sleep mode. The IDLEN and SCS<1:0> bits in the OSCCON register must have already been cleared prior to starting the conversion.

27.2 CAN Module Registers

Note:	Not all CAN registers are available in the Access Bank.

There are many control and data registers associated with the CAN module. For convenience, their descriptions have been grouped into the following sections:

- Control and Status Registers
- Dedicated Transmit Buffer Registers
- Dedicated Receive Buffer Registers
- Programmable TX/RX and Auto RTR Buffers
- Baud Rate Control Registers
- I/O Control Register
- Interrupt Status and Control Registers

Detailed descriptions of each register and their usage are described in the following sections.

27.2.1 CAN CONTROL AND STATUS REGISTERS

The registers described in this section control the overall operation of the CAN module and show its operational status.

REGISTER 27-9: TXBnEIDL: TRANSMIT BUFFER 'n' EXTENDED IDENTIFIER REGISTERS, LOW BYTE [0 ≤ n ≤ 2]

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0
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 7-0 **EID<7:0>:** Extended Identifier bits (not used when transmitting standard identifier message)

REGISTER 27-10: TXBnDm: TRANSMIT BUFFER 'n' DATA FIELD BYTE 'm' REGISTERS [0 ≤ n ≤ 2, 0 ≤ m ≤ 7]

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
TXBnDm7	TXBnDm6	TXBnDm5	TXBnDm4	TXBnDm3	TXBnDm2	TXBnDm1	TXBnDm0
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 7-0 **TXBnDm<7:0>:** Transmit Buffer n Data Field Byte m bits (where 0 ≤ n < 3 and 0 ≤ m < 8)

Each transmit buffer has an array of registers. For example, Transmit Buffer 0 has 7 registers: TXB0D0 to TXB0D7.

REGISTER 27-21: RXERRCNT: RECEIVE ERROR COUNT REGISTER

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
REC7	REC6	REC5	REC4	REC3	REC2	REC1	REC0
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 7-0 **REC<7:0>:** Receive Error Counter bits

This register contains the receive error value as defined by the CAN specifications. When RXERRCNT > 127, the module will go into an error-passive state. RXERRCNT does not have the ability to put the module in "bus-off" state.

EXAMPLE 27-5: READING A CAN MESSAGE

```
    ; Need to read a pending message from RXB0 buffer.
    ; To receive any message, filter, mask and RXM1:RXM0 bits in RXB0CON registers must be
    ; programmed correctly.
    ;
    ; Make sure that there is a message pending in RXB0.
    BTFSS   RXB0CON, RXFUL                ; Does RXB0 contain a message?
    BRA     NoMessage                     ; No. Handle this situation...
    ; We have verified that a message is pending in RXB0 buffer.
    ; If this buffer can receive both Standard or Extended Identifier messages,
    ; identify type of message received.
    BTFSS   RXB0SIDL, EXID                ; Is this Extended Identifier?
    BRA     StandardMessage               ; No. This is Standard Identifier message.
                                          ; Yes. This is Extended Identifier message.
    ; Read all 29-bits of Extended Identifier message.
    ...
    ; Now read all data bytes
    MOVFF   RXB0DO, MY_DATA_BYTE1
    ...
    ; Once entire message is read, mark the RXB0 that it is read and no longer FULL.
    BCF     RXB0CON, RXFUL                ; This will allow CAN Module to load new messages
                                          ; into this buffer.
    ...
```

TABLE 28-4: SUMMARY OF CODE PROTECTION REGISTERS

File Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
300008h	CONFIG5L	—	—	—	—	CP3	CP2	CP1	CP0
300009h	CONFIG5H	CPD	CPB	—	—	—	—	—	—
30000Ah	CONFIG6L	—	—	—	—	WRT3	WRT2	WRT1	WRT0
30000Bh	CONFIG6H	WRTD	WRTB	WRTC	—	—	—	—	—
30000Ch	CONFIG7L	—	—	—	—	EBTR3	EBTR2	EBTR1	EBTR0
30000Dh	CONFIG7H	—	EBTRB	—	—	—	—	—	—

Legend: Shaded cells are unimplemented.

28.6.1 PROGRAM MEMORY CODE PROTECTION

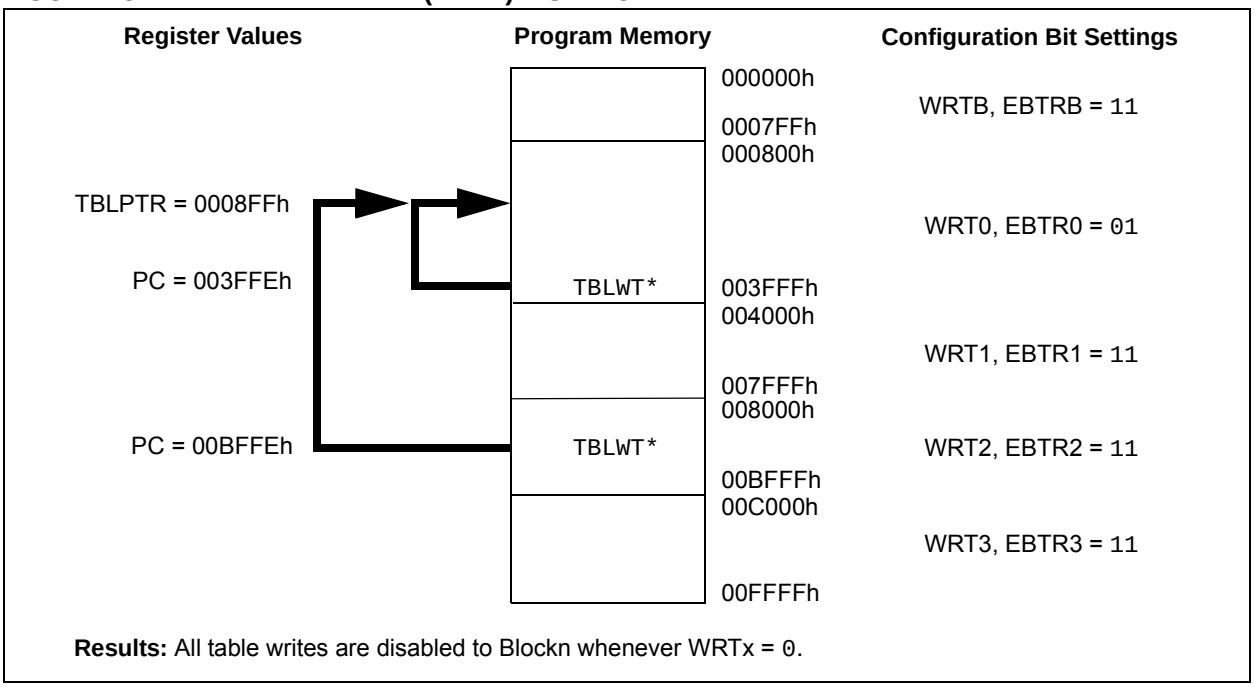
The program memory may be read to, or written from, any location using the table read and table write instructions. The Device ID may be read with table reads. The Configuration registers may be read and written with the table read and table write instructions.

In normal execution mode, the CPx bits have no direct effect. CPx bits inhibit external reads and writes. A block of user memory may be protected from table writes if the WRTx Configuration bit is '0'.

The EBTRx bits control table reads. For a block of user memory with the EBTRx bit set to '0', a table read instruction that executes from within that block is allowed to read. A table read instruction that executes from a location outside of that block is not allowed to read and will result in reading '0's. Figure 28-7 throughFigure 28-9 illustrate table write and table read protection.

Note:	Code protection bits may only be written to a '0' from a '1' state. It is not possible to write a '1' to a bit in the '0' state. Code protection bits are only set to '1' by a full chip erase or block erase function. The full chip erase and block erase functions can only be initiated via ICSP or an external programmer. Refer to the device programming specification for more information.

FIGURE 28-7: TABLE WRITE (WRTx) DISALLOWED



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

COMF **Complement f**

Syntax: COMF f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: $\bar{f} \rightarrow$ dest

Status Affected: N, Z

Encoding:

0001	11da	ffff	ffff

Description: The contents of register 'f' are complemented. If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register 'f' (default).

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 $f \le 95$ (5Fh). See **Section 29.2.3 "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: `COMF REG, 0, 0`

Before Instruction
REG = 13h
After Instruction
REG = 13h
W = ECh

CPFSEQ **Compare f with W, Skip if f = W**

Syntax: CPFSEQ f {,a}

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

Operation: (f) – (W),
skip if (f) = (W)
(unsigned comparison)

Status Affected: None

Encoding:

0110	001a	ffff	ffff

Description: Compares the contents of data memory location 'f' to the contents of W by performing an unsigned subtraction.

If 'f' = W, then the fetched instruction is discarded and a NOP is executed instead, making this a two-cycle instruction.

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 $f \le 95$ (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	No operation

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

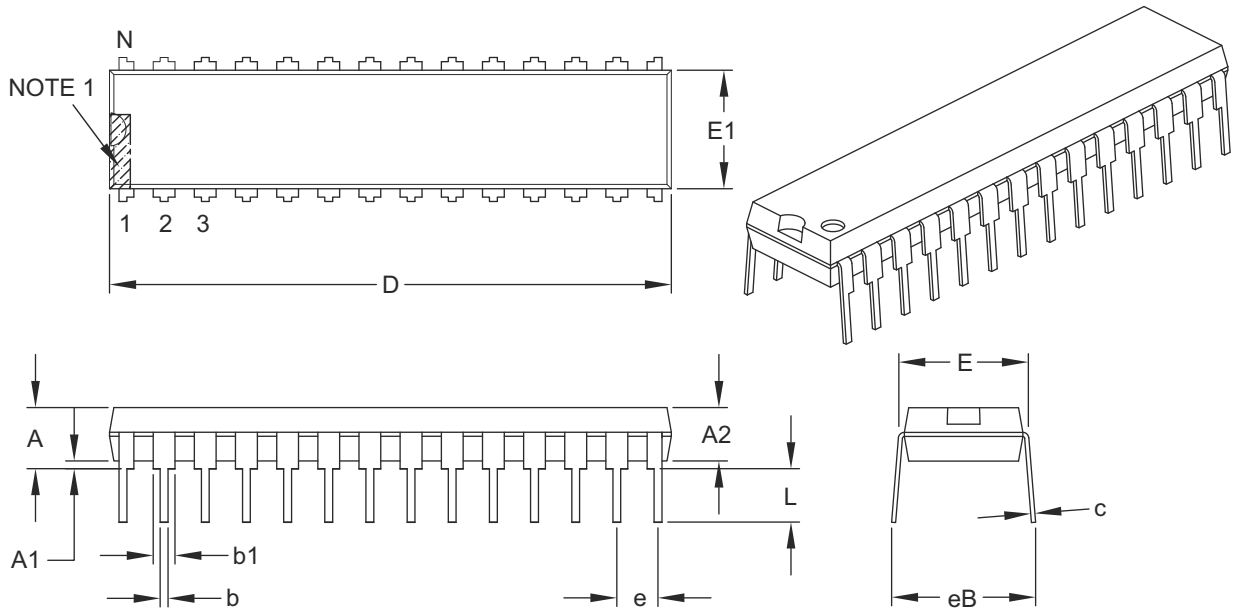
Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE    CPFSEQ REG, 0
NEQUAL  :
EQUAL   :
```

Before Instruction
PC Address = HERE
W = ?
REG = ?
After Instruction
If REG = W;
PC = Address (EQUAL)
If REG ≠ W;
PC = Address (NEQUAL)

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