Welcome to E-XFL.COM

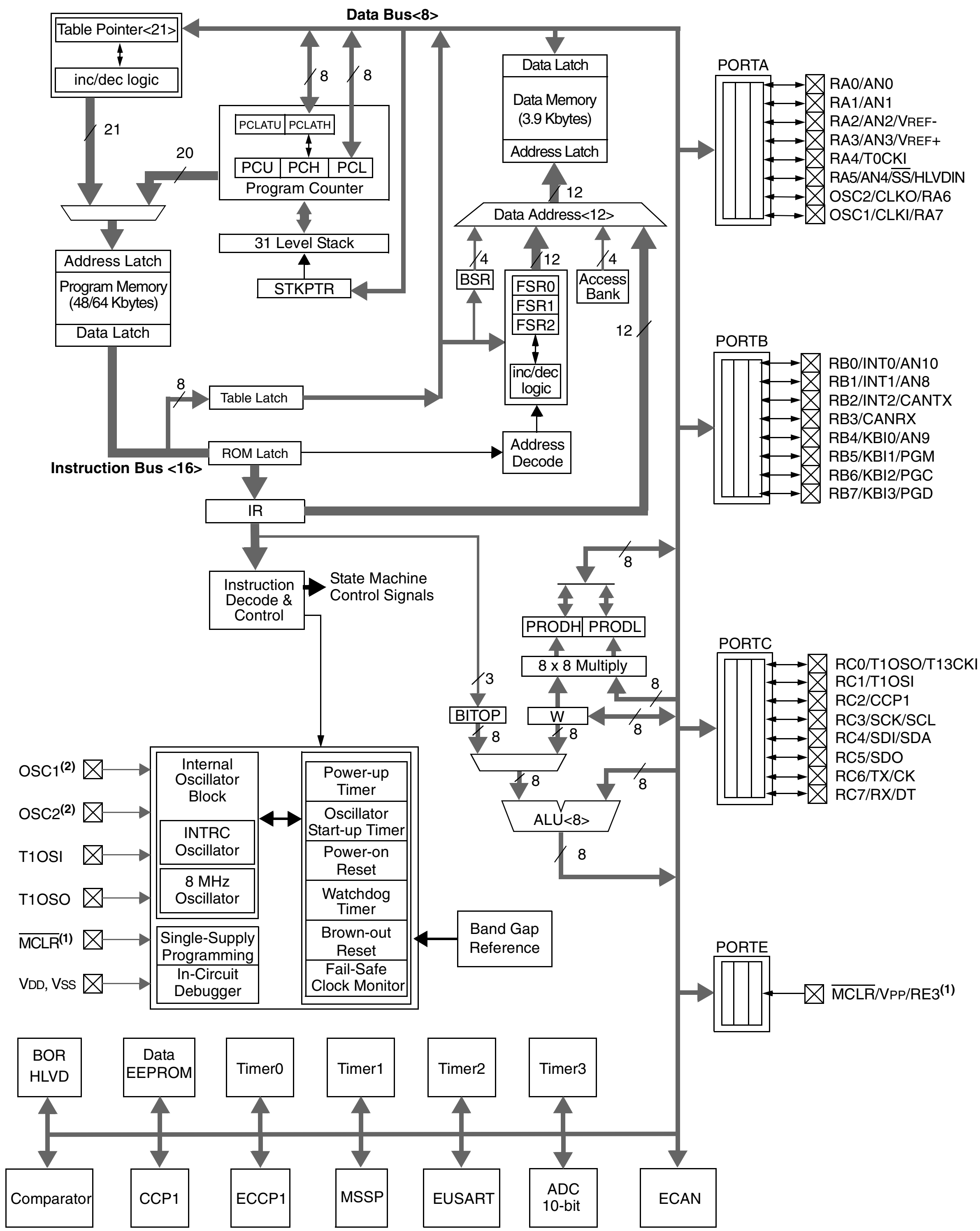
What is "Embedded - Microcontrollers"?

"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	40MHz
Connectivity	CANbus, I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, HLVD, POR, PWM, WDT
Number of I/O	25
Program Memory Size	64KB (32K x 16)
Program Memory Type	FLASH
EEPROM Size	1K x 8
RAM Size	3.25K x 8
Voltage - Supply (Vcc/Vdd)	2V ~ 5.5V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°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/pic18lf2680-i-sp

FIGURE 1-1: PIC18F2585/2680 (28-PIN) BLOCK DIAGRAM



Note 1: RE3 is multiplexed with MCLR and is only available when the MCLR Resets are disabled.

2: OSC1/CLKI and OSC2/CLKO are only available in select oscillator modes and when these pins are not being used as digital I/O. Refer to **Section 2.0 "Oscillator Configurations"** for additional information.

TABLE 5-2: REGISTER FILE SUMMARY (PIC18F2585/2680/4585/4680) (CONTINUED)

File Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Details on page:
PORTE(3)	—	—	—	—	RE3(5)	RE2(3)	RE1(3)	RE0(3)	---- xxxx	52, 145
PORTD(3)	Read PORTD pins, Write PORTD Data Latch								xxxx xxxx	52, 138
PORTC	Read PORTC pins, Write PORTC Data Latch								xxxx xxxx	52, 135
PORTB	Read PORTB pins, Write PORTB Data Latch								xxxx xxxx	52, 132
PORTA	RA7(6)	RA6(6)	Read PORTA pins, Write PORTA Data Latch						xx00 0000	52, 129
ECANCON	MDSEL1	MDSEL0	FIFOWM	EWIN4	EWIN3	EWIN2	EWIN1	EWIN0	0001 000	52, 280
TXERRCNT	TEC7	TEC6	TEC5	TEC4	TEC3	TEC2	TEC1	TEC0	0000 0000	52, 285
RXERRCNT	REC7	REC6	REC5	REC4	REC3	REC2	REC1	REC0	0000 0000	52, 293
COMSTAT Mode 0	RXB0OVFL	RXB1OVFL	TXBO	TXBP	RXBP	TXWARN	RXWARN	EWARN	0000 0000	52, 281
COMSTAT Mode 1	—	RXBnOVFL	TXBO	TXBP	RXBP	TXWARN	RXWARN	EWARN	-000 0000	52, 281
COMSTAT Mode 2	$\overline{\text{FIFOEMPTY}}$	RXBnOVFL	TXBO	TXBP	RXBP	TXWARN	RXWARN	EWARN	0000 0000	52, 281
CIOCON	—	—	ENDRHI	CANCAP	—	—	—	—	--00 ----	52, 314
BRGCON3	WAKDIS	WAKFIL	—	—	—	SEG2PH2	SEG2PH1	SEG2PH0	00-- -000	52, 313
BRGCON2	SEG2PHTS	SAM	SEG1PH2	SEG1PH1	SEG1PH0	PRSEG2	PRSEG1	PRSEG0	0000 0000	52, 312
BRGCON1	SJW1	SJW0	BRP5	BRP4	BRP3	BRP2	BRP1	BRP0	0000 0000	52, 311
CANCON Mode 0	REQOP2	REQOP1	REQOP0	ABAT	WIN2(7)	WIN1(7)	WIN0(7)	—(7)	1000 000-	53, 276
CANCON Mode 1	REQOP2	REQOP1	REQOP0	ABAT	—(7)	—(7)	—(7)	—(7)	1000 ----	53, 276
CANCON Mode 2	REQOP2	REQOP1	REQOP0	ABAT	FP3(7)	FP2(7)	FP1(7)	FP0(7)	1000 0000	53, 276
CANSTAT Mode 0	OPMODE2	OPMODE1	OPMODE0	—(7)	ICODE3(7)	ICODE2(7)	ICODE1(7)	—(7)	000- 0000	53, 277
CANSTAT Modes 1, 2	OPMODE2	OPMODE1	OPMODE0	EICODE4(7)	EICODE3(7)	EICODE2(7)	EICODE1(7)	EICODE0(7)	0000 0000	53, 277
RXB0D7	RXB0D77	RXB0D76	RXB0D75	RXB0D74	RXB0D73	RXB0D72	RXB0D71	RXB0D70	xxxx xxxx	53, 292
RXB0D6	RXB0D67	RXB0D66	RXB0D65	RXB0D64	RXB0D63	RXB0D62	RXB0D61	RXB0D60	xxxx xxxx	53, 292
RXB0D5	RXB0D57	RXB0D56	RXB0D55	RXB0D54	RXB0D53	RXB0D52	RXB0D51	RXB0D50	xxxx xxxx	53, 292
RXB0D4	RXB0D47	RXB0D46	RXB0D45	RXB0D44	RXB0D43	RXB0D42	RXB0D41	RXB0D40	xxxx xxxx	53, 292
RXB0D3	RXB0D37	RXB0D36	RXB0D35	RXB0D34	RXB0D33	RXB0D32	RXB0D31	RXB0D30	xxxx xxxx	53, 292
RXB0D2	RXB0D27	RXB0D26	RXB0D25	RXB0D24	RXB0D23	RXB0D22	RXB0D21	RXB0D20	xxxx xxxx	53, 292
RXB0D1	RXB0D17	RXB0D16	RXB0D15	RXB0D14	RXB0D13	RXB0D12	RXB0D11	RXB0D10	xxxx xxxx	53, 292
RXB0D0	RXB0D07	RXB0D06	RXB0D05	RXB0D04	RXB0D03	RXB0D02	RXB0D01	RXB0D00	xxxx xxxx	53, 292
RXB0DLC	—	RXRTR	RB1	RB0	DLC3	DLC2	DLC1	DLC0	-xxx xxxx	53, 292
RXB0EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	xxxx xxxx	53, 291
RXB0EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	xxxx xxxx	53, 291
RXB0SIDL	SID2	SID1	SID0	SRR	EXID	—	EID17	EID16	xxxx x-xx	53, 291
RXB0SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	xxxx xxxx	53, 290
RXB0CON Mode 0	RXFUL	RXM1	RXM0(7)	—(7)	RXRTRRO(7)	RXBODBEN(7)	JTOFF(7)	FILHIT0(7)	000- 0000	53, 287
RXB0CON Mode 1, 2	RXFUL	RXM1	RTRRO	FILHIT4	FILHIT3	FILHIT2	FILHIT1	FILHIT0	0000 0000	53, 287

Legend: x = unknown, u = unchanged, - = unimplemented, q = value depends on condition

Note 1: Bit 21 of the PC is only available in Test mode and Serial Programming modes.

2: The SBOREN bit is only available when CONFIG2L<1:0> = 01; otherwise, it is disabled and reads as '0'. See **Section 4.4 "Brown-out Reset (BOR)"**.

3: These registers and/or bits are not implemented on PIC18F2X8X devices and are read as '0'. Reset values are shown for PIC18F4X8X devices; individual unimplemented bits should be interpreted as '—'.

4: The PLLEN bit is only available in specific oscillator configuration; otherwise, it is disabled and reads as '0'. See **Section 2.6.4 "PLL in INTOSC Modes"**.

5: The RE3 bit is only available when Master Clear Reset is disabled (CONFIG3H<7> = 0); otherwise, RE3 reads as '0'. This bit is read-only.

6: RA6/RA7 and their associated latch and direction bits are individually configured as port pins based on various primary oscillator modes. When disabled, these bits read as '0'.

7: CAN bits have multiple functions depending on the selected mode of the CAN module.

8: This register reads all '0's until the ECAN™ technology is set up in Mode 1 or Mode 2.

9: These registers are available on PIC18F4X8X devices only.

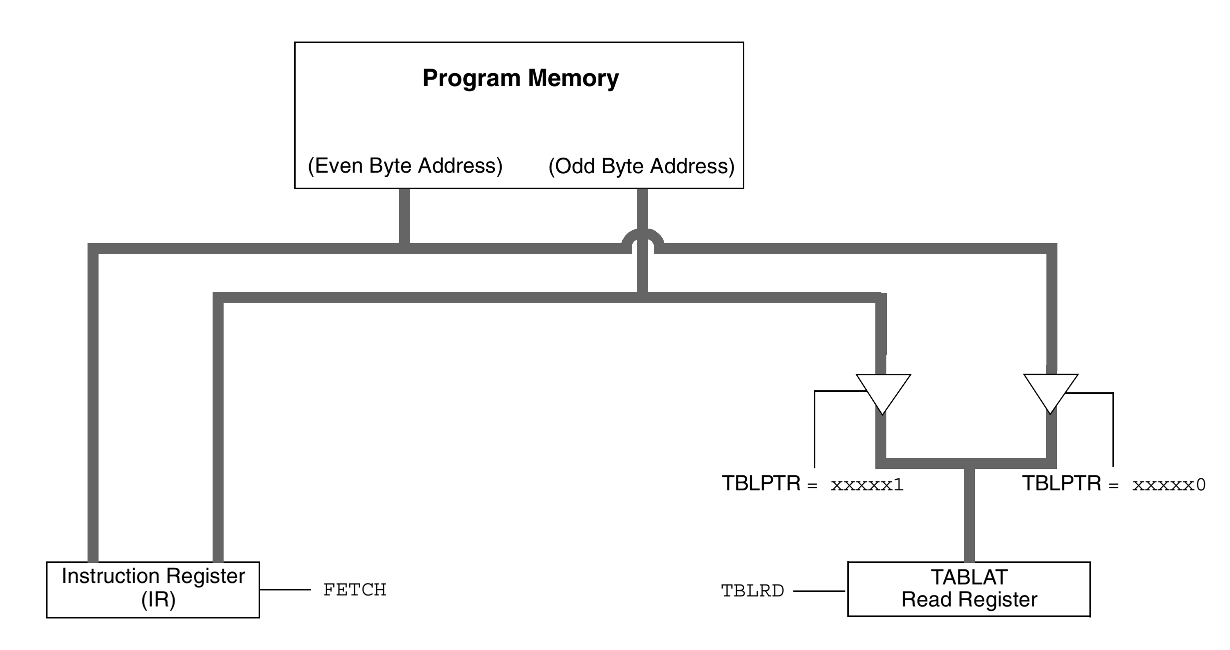
6.3 Reading the Flash Program Memory

The TBLRD instruction is used to retrieve data from program memory and places it into data RAM. Table reads from program memory are performed one byte at a time.

TBLPTR points to a byte address in program space. Executing TBLRD places the byte pointed to into TABLAT. In addition, TBLPTR can be modified automatically for the next table read operation.

The internal program memory is typically organized by words. The Least Significant bit of the address selects between the high and low bytes of the word. Figure 6-4 shows the interface between the internal program memory and the TABLAT.

FIGURE 6-4: READS FROM FLASH PROGRAM MEMORY



EXAMPLE 6-1: READING A FLASH PROGRAM MEMORY WORD

```
            MOVLW     CODE_ADDR_UPPER            ; Load TBLPTR with the base
            MOVWF     TBLPTRU                    ; address of the word
            MOVLW     CODE_ADDR_HIGH
            MOVWF     TBLPTRH
            MOVLW     CODE_ADDR_LOW
            MOVWF     TBLPTRL
READ_WORD
            TBLRD*+                              ; read into TABLAT and increment
            MOVF      TABLAT, W                  ; get data
            MOVWF     WORD_EVEN
            TBLRD*+                              ; read into TABLAT and increment
            MOVF      TABLAT, W                  ; get data
            MOVF      WORD_ODD
```

9.6 INTn Pin Interrupts

External interrupts on the RB0/INT0, RB1/INT1 and RB2/INT2 pins are edge-triggered. If the corresponding INTEDGx bit in the INTCON2 register is set (= `1`), the interrupt is triggered by a rising edge; if the bit is clear, the trigger is on the falling edge. When a valid edge appears on the RBx/INTx pin, the corresponding flag bit INTxF is set. This interrupt can be disabled by clearing the corresponding enable bit INTxE. Flag bit INTxF must be cleared in software in the Interrupt Service Routine before re-enabling the interrupt.

All external interrupts (INT0, INT1 and INT2) can wake-up the processor from the power managed modes, if bit INTxE was set prior to going into power managed modes. If the Global Interrupt Enable bit, GIE, is set, the processor will branch to the interrupt vector following wake-up.

Interrupt priority for INT1 and INT2 is determined by the value contained in the interrupt priority bits, INT1IP (INTCON3<6>) and INT2IP (INTCON3<7>). There is no priority bit associated with INT0. It is always a high priority interrupt source.

9.7 TMR0 Interrupt

In 8-bit mode (which is the default), an overflow in the TMR0 register (FFh $\rightarrow$ 00h) will set flag bit TMR0IF. In 16-bit mode, an overflow in the TMR0H:TMR0L register pair (FFFFh $\rightarrow$ 0000h) will set TMR0IF. The interrupt can be enabled/disabled by setting/clearing enable bit TMR0IE (INTCON<5>). Interrupt priority for Timer0 is determined by the value contained in the interrupt priority bit, TMR0IP (INTCON2<2>). See **Section 11.0 "Timer0 Module"** for further details on the Timer0 module.

9.8 PORTB Interrupt-on-Change

An input change on PORTB<7:4> sets flag bit, RBIF (INTCON<0>). The interrupt can be enabled/disabled by setting/clearing enable bit, RBIE (INTCON<3>). Interrupt priority for PORTB interrupt-on-change is determined by the value contained in the interrupt priority bit, RBIP (INTCON2<0>).

9.9 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 5.3 "Data Memory Organization"**), the user may need to save the WREG, STATUS and BSR registers on entry to the Interrupt Service Routine. Depending on the user's application, other registers may also need to be saved. Example 9-1 saves and restores the WREG, STATUS and BSR registers during an Interrupt Service Routine.

EXAMPLE 9-1: 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
```

REGISTER 10-1: TRISE REGISTER (PIC18F4X8X DEVICES ONLY)

R-0	R-0	R/W-0	R/W-0	U-0	R/W-1	R/W-1	R/W-1
IBF	OBF	IBOV	PSPMODE	—	TRISE2	TRISE1	TRISE0
bit 7							bit 0

bit 7 **IBF:** Input Buffer Full Status bit
1 = A word has been received and waiting to be read by the CPU
0 = No word has been received

bit 6 **OBF:** Output Buffer Full Status bit
1 = The output buffer still holds a previously written word
0 = The output buffer has been read

bit 5 **IBOV:** Input Buffer Overflow Detect bit (in Microprocessor mode)
1 = A write occurred when a previously input word has not been read (must be cleared in software)
0 = No overflow occurred

bit 4 **PSPMODE:** Parallel Slave Port Mode Select bit
1 = Parallel Slave Port mode
0 = General Purpose I/O mode

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

bit 2 **TRISE2:** RE2 Direction Control bit
1 = Input
0 = Output

bit 1 **TRISE1:** RE1 Direction Control bit
1 = Input
0 = Output

bit 0 **TRISE0:** RE0 Direction Control bit
1 = Input
0 = Output

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

14.0 TIMER3 MODULE

The Timer3 module timer/counter incorporates these features:

- Software selectable operation as a 16-bit timer or counter
- Readable and writable 8-bit registers (TMR3H and TMR3L)
- Selectable clock source (internal or external) with device clock or Timer1 oscillator internal options
- Interrupt-on-overflow
- Module Reset on CCP1 special event trigger

A simplified block diagram of the Timer3 module is shown in Figure 14-1. A block diagram of the module's operation in Read/Write mode is shown in Figure 14-2.

The Timer3 module is controlled through the T3CON register (Register 14-1). It also selects the clock source options for the CCP1 modules (see **Section 15.1.1 "CCP1 Modules and Timer Resources"** for more information).

REGISTER 14-1: T3CON: TIMER3 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
RD16	T3ECCP1[(1)]	T3CKPS1	T3CKPS0	T3CCP1[(1)]	$\overline{\text{T3SYNC}}$	TMR3CS	TMR3ON
bit 7							bit 0

bit 7 **RD16:** 16-bit Read/Write Mode Enable bit
1 = Enables register read/write of Timer3 in one 16-bit operation
0 = Enables register read/write of Timer3 in two 8-bit operations

bit 6,3 **T3ECCP1:T3CCP1:** Timer3 and Timer1 to ECCP1/CCP1 Enable bits[(1)]
1x = Timer3 is the capture/compare clock source for both CCP1 and ECCP1 modules
01 = Timer3 is the capture/compare clock source for ECCP1;
Timer1 is the capture/compare clock source for CCP1
00 = Timer1 is the capture/compare clock source for both CCP1 and ECCP1 modules

Note 1: These bits are available on PIC18F4X8X devices only.

bit 5-4 **T3CKPS1:T3CKPS0**: Timer3 Input Clock Prescale Select bits
11 = 1:8 Prescale value
10 = 1:4 Prescale value
01 = 1:2 Prescale value
00 = 1:1 Prescale value

bit 2 **$\overline{\text{T3SYNC}}$:** Timer3 External Clock Input Synchronization Control bit
(Not usable if the device clock comes from Timer1/Timer3.)
When TMR3CS = 1:
1 = Do not synchronize external clock input
0 = Synchronize external clock input
When TMR3CS = 0:
This bit is ignored. Timer3 uses the internal clock when TMR3CS = 0.

bit 1 **TMR3CS:** Timer3 Clock Source Select bit
1 = External clock input from Timer1 oscillator or T13CKI (on the rising edge after the first falling edge)
0 = Internal clock (FOSC/4)

bit 0 **TMR3ON:** Timer3 On bit
1 = Enables Timer3
0 = Stops Timer3

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

NOTES:

The ECCPR1H register and a 2-bit internal latch are used to double-buffer the PWM duty cycle. This double-buffering is essential for glitchless PWM operation.

When the ECCPR1H and 2-bit latch match TMR2, concatenated with an internal 2-bit Q clock or 2 bits of the TMR2 prescaler, the CCP1 pin is cleared.

The maximum PWM resolution (bits) for a given PWM frequency is given by the equation.

EQUATION 15-3:

$$\text{PWM Resolution (max)} = \frac{\log\left(\frac{F_{OSC}}{F_{PWM}}\right)}{\log(2)} \text{ bits}$$

Note:	If the PWM duty cycle value is longer than the PWM period, the ECCP1 pin will not be cleared.

TABLE 15-4: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS AT 40 MHz

PWM Frequency	2.44 kHz	9.77 kHz	39.06 kHz	156.25 kHz	312.50 kHz	416.67 kHz
Timer Prescaler (1, 4, 16)	16	4	1	1	1	1
PR2 Value	FFh	FFh	FFh	3Fh	1Fh	17h
Maximum Resolution (bits)	14	12	10	8	7	6.58

15.4.3 PWM AUTO-SHUTDOWN (ECCP1 ONLY)

The PWM auto-shutdown features of the Enhanced CCP1 module are available to ECCP1 in PIC18F4585/4680 (40/44-pin) devices. The operation of this feature is discussed in detail in **Section 16.4.7 “Enhanced PWM Auto-Shutdown”**.

Auto-shutdown features are not available for CCP1.

15.4.4 SETUP FOR PWM OPERATION

The following steps should be taken when configuring the CCP1 module for PWM operation:

1. Set the PWM period by writing to the PR2 register.
2. Set the PWM duty cycle by writing to the CCPR1L register and CCP1CON<5:4> bits.
3. Make the CCP1 pin an output by clearing the appropriate TRIS bit.
4. Set the TMR2 prescale value, then enable Timer2 by writing to T2CON.
5. Configure the CCP1 module for PWM operation.

17.3.8 OPERATION IN POWER MANAGED MODES

In SPI Master mode, module clocks may be operating at a different speed than when in full power mode; in the case of the Sleep mode, all clocks are halted.

In most power managed modes, a clock is provided to the peripherals. That clock should be from the primary clock source, the secondary clock (Timer1 oscillator at 32.768 kHz) or the INTOSC source. See **Section 2.7 "Clock Sources and Oscillator Switching"** for additional information.

In most cases, the speed that the master clocks SPI data is not important; however, this should be evaluated for each system.

If MSSP interrupts are enabled, they can wake the controller from Sleep mode, or one of the Idle modes, when the master completes sending data. If an exit from Sleep or Idle mode is not desired, MSSP interrupts should be disabled.

If the Sleep mode is selected, all module clocks are halted and the transmission/reception will remain in that state until the devices wakes. After the device returns to Run mode, the module will resume transmitting and receiving data.

In SPI Slave mode, the SPI Transmit/Receive Shift register operates asynchronously to the device. This allows the device to be placed in any power managed mode and data to be shifted into the SPI Transmit/ Receive Shift register. When all 8 bits have been received, the MSSP interrupt flag bit will be set and if enabled, will wake the device.

17.3.9 EFFECTS OF A RESET

A Reset disables the MSSP module and terminates the current transfer.

17.3.10 BUS MODE COMPATIBILITY

Table 17-1 shows the compatibility between the standard SPI modes and the states of the CKP and CKE control bits.

TABLE 17-1: SPI BUS MODES

Standard SPI Mode Terminology	Control Bits State	
	CKP	CKE
0, 0	0	1
0, 1	0	0
1, 0	1	1
1, 1	1	0

There is also a SMP bit which controls when the data is sampled.

TABLE 17-2: REGISTERS ASSOCIATED WITH SPI OPERATION

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	49
PIR1	PSPIF(1)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	52
PIE1	PSPIE(1)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	52
IPR1	PSPIP(1)	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	52
TRISA	PORTA Data Direction Register								52
TRISC	PORTC Data Direction Register								52
SSPBUF	Synchronous Serial Port Receive Buffer/Transmit Register								50
SSPCON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	50
SSPSTAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	50

Legend: — = unimplemented, read as '0'. Shaded cells are not used by the MSSP in SPI mode.

Note 1: These bits are unimplemented in PIC18F2X8X devices; always maintain these bits clear.

REGISTER 23-24: BnSIDH: TX/RX BUFFER n STANDARD IDENTIFIER REGISTERS, HIGH BYTE IN RECEIVE MODE [0 ≤ n ≤ 5, TXnEN (BSEL0<n>) = 0][(1)]

R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3
bit 7							bit 0

bit 7-0 **SID10:SID3:** Standard Identifier bits (if EXIDE (BnSIDL<3>) = 0)
Extended Identifier bits EID28:EID21 (if EXIDE = 1).

Note 1: These registers are available in Mode 1 and 2 only.

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

REGISTER 23-25: BnSIDH: TX/RX BUFFER n STANDARD IDENTIFIER REGISTERS, HIGH BYTE IN TRANSMIT MODE [0 ≤ n ≤ 5, TXnEN (BSEL0<n>) = 1][(1)]

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3
bit 7							bit 0

bit 7-0 **SID10:SID3:** Standard Identifier bits (if EXIDE (BnSIDL<3>) = 0)
Extended Identifier bits EID28:EID21 (if EXIDE = 1).

Note 1: These registers are available in Mode 1 and 2 only.

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

REGISTER 23-35: BnDLC: TX/RX BUFFER n DATA LENGTH CODE REGISTERS IN TRANSMIT MODE [$0 \le n \le 5$, TXnEN (BSEL<n>) = 1][(1)]

U-0	R/W-x	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x
—	TXRTR	—	—	DLC3	DLC2	DLC1	DLC0
bit 7							bit 0

bit 7 **Unimplemented:** Read as ‘0’

bit 6 **TXRTR:** Transmitter Remote Transmission Request bit
1 = Transmitted message will have RTR bit set
0 = Transmitted message will have RTR bit cleared

bit 5-4 **Unimplemented:** Read as ‘0’

bit 3-0 **DLC3:DLC0:** Data Length Code bits
1111-1001 = Reserved
1000 = Data length = 8 bytes
0111 = Data length = 7 bytes
0110 = Data length = 6 bytes
0101 = Data length = 5 bytes
0100 = Data length = 4 bytes
0011 = Data length = 3 bytes
0010 = Data length = 2 bytes
0001 = Data length = 1 bytes
0000 = Data length = 0 bytes

Note 1: These registers are available in Mode 1 and 2 only.

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

REGISTER 23-36: BSEL0: BUFFER SELECT REGISTER 0[(1)]

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	U-0
B5TXEN	B4TXEN	B3TXEN	B2TXEN	B1TXEN	B0TXEN	—	—
bit 7							bit 0

bit 7-2 **B5TXEN:B0TXEN:** Buffer 5 to Buffer 0 Transmit Enable bit
1 = Buffer is configured in Transmit mode
0 = Buffer is configured in Receive mode

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

Note 1: This register is available in Mode 1 and 2 only.

.

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

REGISTER 23-57: PIE3: PERIPHERAL INTERRUPT ENABLE REGISTER

	bit 7							bit 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
Mode 0	IRXIE	WAKIE	ERRIE	TXB2IE	TXB1IE(1)	TXB0IE(1)	RXB1IE	RXB0IE
	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
Mode 1, 2	IRXIE	WAKIE	ERRIE	TXBnIE	TXB1IE(1)	TXB0IE(1)	RXBnIE	FIFOWMIE

bit 7 **IRXIE:** CAN Invalid Received Message Interrupt Enable bit
1 = Enable invalid message received interrupt
0 = Disable invalid message received interrupt

bit 6 **WAKIE:** CAN bus Activity Wake-up Interrupt Enable bit
1 = Enable bus activity wake-up interrupt
0 = Disable bus activity wake-up interrupt

bit 5 **ERRIE:** CAN bus Error Interrupt Enable bit
1 = Enable CAN bus error interrupt
0 = Disable CAN bus error interrupt

bit 4 When CAN is in Mode 0:
TXB2IE: CAN Transmit Buffer 2 Interrupt Enable bit
1 = Enable Transmit Buffer 2 interrupt
0 = Disable Transmit Buffer 2 interrupt
When CAN is in Mode 1 or 2:
TXBnIE: CAN Transmit Buffer Interrupts Enable bit
1 = Enable transmit buffer interrupt; individual interrupt is enabled by TXBIE and BIE0
0 = Disable all transmit buffer interrupts

bit 3 **TXB1IE:** CAN Transmit Buffer 1 Interrupt Enable bit(1)
1 = Enable Transmit Buffer 1 interrupt
0 = Disable Transmit Buffer 1 interrupt

bit 2 **TXB0IE:** CAN Transmit Buffer 0 Interrupt Enable bit(1)
1 = Enable Transmit Buffer 0 interrupt
0 = Disable Transmit Buffer 0 interrupt

bit 1 When CAN is in Mode 0:
RXB1IE: CAN Receive Buffer 1 Interrupt Enable bit
1 = Enable Receive Buffer 1 interrupt
0 = Disable Receive Buffer 1 interrupt
When CAN is in Mode 1 or 2:
RXBnIE: CAN Receive Buffer Interrupts Enable bit
1 = Enable receive buffer interrupt; individual interrupt is enabled by BIE0
0 = Disable all receive buffer interrupts

bit 0 When CAN is in Mode 0:
RXB0IE: CAN Receive Buffer 0 Interrupt Enable bit
1 = Enable Receive Buffer 0 interrupt
0 = Disable Receive Buffer 0 interrupt
When CAN is in Mode 1:
Unimplemented: Read as '0'
When CAN is in Mode 2:
FIFOWMIE: FIFO Watermark Interrupt Enable bit
1 = Enable FIFO watermark interrupt
0 = Disable FIFO watermark interrupt

Note 1: In CAN Mode 1 and 2, this bit is forced to '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

23.5.4 PROGRAMMABLE AUTO-RTR BUFFERS

In Mode 1 and 2, any of six programmable transmit/receive buffers may be programmed to automatically respond to predefined RTR messages without user firmware intervention. Automatic RTR handling is enabled by setting the TXnEN bit in the BSEL0 register and the RTREN bit in the BnCON register. After this setup, when an RTR request is received, the TXREQ bit is automatically set and the current buffer content is automatically queued for transmission as a RTR response. As with all transmit buffers, once the TXREQ bit is set, buffer registers become read-only and any writes to them will be ignored.

The following outlines the steps required to automatically handle RTR messages:

1. Set buffer to Transmit mode by setting TXnEN bit to '1' in BSEL0 register.
2. At least one acceptance filter must be associated with this buffer and preloaded with expected RTR identifier.
3. Bit RTREN in BnCON register must be set to '1'.
4. Buffer must be preloaded with the data to be sent as a RTR response.

Normally, user firmware will keep buffer data registers up to date. If firmware attempts to update the buffer while an automatic RTR response is in the process of transmission, all writes to buffers are ignored.

23.6 CAN Message Transmission

23.6.1 INITIATING TRANSMISSION

For the MCU to have write access to the message buffer, the TXREQ bit must be clear, indicating that the message buffer is clear of any pending message to be transmitted. At a minimum, the SIDH, SIDL and DLC registers must be loaded. If data bytes are present in the message, the data registers must also be loaded. If the message is to use extended identifiers, the EIDH:EIDL registers must also be loaded and the EXIDE bit set.

To initiate message transmission, the TXREQ bit must be set for each buffer to be transmitted. When TXREQ is set, the TXABT, TXLARB and TXERR bits will be cleared. To successfully complete the transmission, there must be at least one node with matching baud rate on the network.

Setting the TXREQ bit does not initiate a message transmission; it merely flags a message buffer as ready for transmission. Transmission will start when the device detects that the bus is available. The device will then begin transmission of the highest priority message that is ready.

When the transmission has completed successfully, the TXREQ bit will be cleared, the TXBnIF bit will be set and an interrupt will be generated if the TXBnIE bit is set.

If the message transmission fails, the TXREQ will remain set, indicating that the message is still pending for transmission and one of the following condition flags will be set. If the message started to transmit but encountered an error condition, the TXERR and the IRXIF bits will be set and an interrupt will be generated. If the message lost arbitration, the TXLARB bit will be set.

23.6.2 ABORTING TRANSMISSION

The MCU can request to abort a message by clearing the TXREQ bit associated with the corresponding message buffer (TXBnCON<3> or BnCON<3>). Setting the ABAT bit (CANCON<4>) will request an abort of all pending messages. If the message has not yet started transmission, or if the message started but is interrupted by loss of arbitration or an error, the abort will be processed. The abort is indicated when the module sets the TXABT bit for the corresponding buffer (TXBnCON<6> or BnCON<6>). If the message has started to transmit, it will attempt to transmit the current message fully. If the current message is transmitted fully and is not lost to arbitration or an error, the TXABT bit will not be set because the message was transmitted successfully. Likewise, if a message is being transmitted during an abort request and the message is lost to arbitration or an error, the message will not be retransmitted and the TXABT bit will be set, indicating that the message was successfully aborted.

Once an abort is requested by setting the ABAT or TXABT bits, it cannot be cleared to cancel the abort request. Only CAN module hardware or a POR condition can clear it.

In Mode 1 and 2, there are an additional 10 acceptance filters, RXF6-RXF15, creating a total of 16 available filters. RXF15 can be used either as an acceptance filter or acceptance mask register. Each of these acceptance filters can be individually enabled or disabled by setting or clearing the RXFENn bit in the RXFCONn register. Any of these 16 acceptance filters can be dynamically associated with any of the receive buffers. Actual association is made by setting appropriate bits in the RXFBCONn register. Each RXFBCONn register contains a nibble for each filter. This nibble can be used to associate a specific filter to any of available receive buffers. User firmware may associate more than one filter to any one specific receive buffer.

In addition to dynamic filter to buffer association, in Mode 1 and 2, each filter can also be dynamically associated to available acceptance mask registers. The FILn_m bits in the MSELn register can be used to link a specific acceptance filter to an acceptance mask register. As with filter to buffer association, one can also associate more than one mask to a specific acceptance filter.

When a filter matches and a message is loaded into the receive buffer, the filter number that enabled the message reception is loaded into the FILHIT bit(s). In Mode 0 for RXB1, the RXB1CON register contains the FILHIT<2:0> bits. They are coded as follows:

- `101` = Acceptance Filter 5 (RXF5)
- `100` = Acceptance Filter 4 (RXF4)
- `011` = Acceptance Filter 3 (RXF3)
- `010` = Acceptance Filter 2 (RXF2)
- `001` = Acceptance Filter 1 (RXF1)
- `000` = Acceptance Filter 0 (RXF0)

Note:	'`000`' and '`001`' can only occur if the RXB0DBEN bit is set in the RXB0CON register, allowing RXB0 messages to rollover into RXB1.

The coding of the RXB0DBEN bit enables these three bits to be used similarly to the FILHIT bits and to distinguish a hit on filter RXF0 and RXF1, in either RXB0 or after a rollover into RXB1.

- `111` = Acceptance Filter 1 (RXF1)
- `110` = Acceptance Filter 0 (RXF0)
- `001` = Acceptance Filter 1 (RXF1)
- `000` = Acceptance Filter 0 (RXF0)

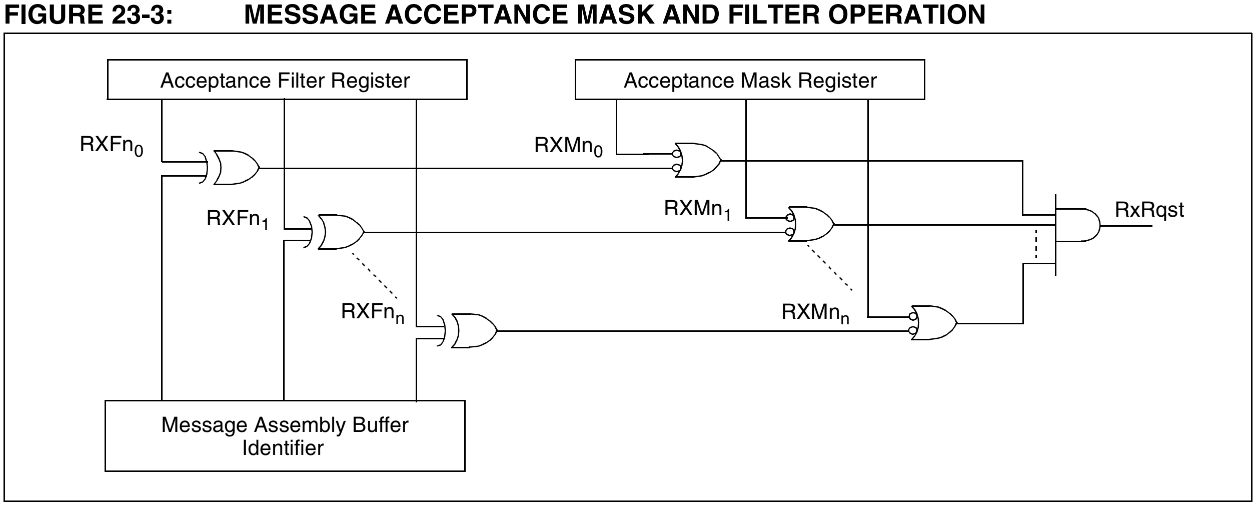
If the RXB0DBEN bit is clear, there are six codes corresponding to the six filters. If the RXB0DBEN bit is set, there are six codes corresponding to the six filters, plus two additional codes corresponding to RXF0 and RXF1 filters, that rollover into RXB1.

In Mode 1 and 2, each buffer control register contains 5 bits of filter hit bits (FILHIT<4:0>). A binary value of '`0`' indicates a hit from RXF0 and 15 indicates RXF15.

If more than one acceptance filter matches, the FILHIT bits will encode the binary value of the lowest numbered filter that matched. In other words, if filter RXF2 and filter RXF4 match, FILHIT will be loaded with the value for RXF2. This essentially prioritizes the acceptance filters with a lower number filter having higher priority. Messages are compared to filters in ascending order of filter number.

The mask and filter registers can only be modified when the PIC18F2585/2680/4585/4680 devices are in Configuration mode.

FIGURE 23-3: MESSAGE ACCEPTANCE MASK AND FILTER OPERATION



BRA — Unconditional Branch

Syntax: BRA n

Operands: $-1024 \leq n \leq 1023$

Operation: $(PC) + 2 + 2n \rightarrow PC$

Status Affected: None

Encoding:

`1101`	`0nnn`	`nnnn`	`nnnn`

Description: Add the 2's complement number '2n' to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is a two-cycle instruction.

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

Example: `HERE BRA Jump`

Before Instruction
PC = address `(HERE)`
After Instruction
PC = address `(Jump)`

BSF — Bit Set f

Syntax: BSF f, b {,a}

Operands: $0 \leq f \leq 255$
$0 \leq b \leq 7$
$a \in [0,1]$

Operation: $1 \rightarrow f<b>$

Status Affected: None

Encoding:

`1000`	`bbba`	`ffff`	`ffff`

Description: Bit 'b' in register 'f' is set.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank (default).

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh). See **Section 25.2.3 "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 register 'f'

Example: `BSF FLAG_REG, 7, 1`

Before Instruction
FLAG_REG = 0Ah
After Instruction
FLAG_REG = 8Ah

MULLW — Multiply Literal with W

Syntax:	MULLW k
Operands:	$0 \le k \le 255$
Operation:	(W) x k → PRODH:PRODL
Status Affected:	None
Encoding:	`0000` `1101` `kkkk` `kkkk`
Description:	An unsigned multiplication is carried out between the contents of W and the 8-bit literal 'k'. The 16-bit result is placed in the PRODH:PRODL register pair. PRODH contains the high byte. W is unchanged. None of the Status flags are affected. Note that neither overflow nor carry is possible in this operation. A zero result is possible but not detected.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write registers PRODH: PRODL

Example: `MULLW 0C4h`

Before Instruction
- W = E2h
- PRODH = ?
- PRODL = ?

After Instruction
- W = E2h
- PRODH = ADh
- PRODL = 08h

MULWF — Multiply W with f

Syntax:	MULWF f {,a}
Operands:	$0 \le f \le 255$ $a \in [0,1]$
Operation:	(W) x (f) → PRODH:PRODL
Status Affected:	None
Encoding:	`0000` `001a` `ffff` `ffff`
Description:	An unsigned multiplication is carried out between the contents of W and the register file location 'f'. The 16-bit result is stored in the PRODH:PRODL register pair. PRODH contains the high byte. Both W and 'f' are unchanged. None of the Status flags are affected. Note that neither overflow nor carry is possible in this operation. A zero result is possible but not detected. If 'a' is '`0`', the Access Bank is selected. If 'a' is '`1`', the BSR is used to select the GPR bank (default). 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 25.2.3 "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 registers PRODH: PRODL

Example: `MULWF REG, 1`

Before Instruction
- W = C4h
- REG = B5h
- PRODH = ?
- PRODL = ?

After Instruction
- W = C4h
- REG = B5h
- PRODH = 8Ah
- PRODL = 94h

RETFIE — Return from Interrupt

Syntax: RETFIE {s}

Operands: s ∈ [0,1]

Operation: (TOS) → PC,
1 → GIE/GIEH or PEIE/GIEL,
if s = 1
(WS) → W,
(STATUSS) → STATUS,
(BSRS) → BSR,
PCLATU, PCLATH are unchanged.

Status Affected: GIE/GIEH, PEIE/GIEL.

Encoding:

0000	0000	0001	000s

Description: Return from Interrupt. Stack is popped and Top-of-Stack (TOS) is loaded into the PC. Interrupts are enabled by setting either the high or low priority global interrupt enable bit. If 's' = 1, the contents of the shadow registers, WS, STATUSS and BSRS, are loaded into their corresponding registers, W, STATUS and BSR. If 's' = 0, no update of these registers occurs (default).

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	No operation	POP PC from stack Set GIEH or GIEL
No operation	No operation	No operation	No operation

Example: `RETFIE 1`

After Interrupt

PC	=	TOS
W	=	WS
BSR	=	BSRS
STATUS	=	STATUSS
GIE/GIEH, PEIE/GIEL	=	1

RETLW — Return Literal to W

Syntax: RETLW k

Operands: 0 ≤ k ≤ 255

Operation: k → W,
(TOS) → PC,
PCLATU, PCLATH are unchanged

Status Affected: None

Encoding:

0000	1100	kkkk	kkkk

Description: W is loaded with the eight-bit literal 'k'. The program counter is loaded from the top of the stack (the return address). The high address latch (PCLATH) remains unchanged.

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	POP PC from stack, Write to W
No operation	No operation	No operation	No operation

Example:

```
    CALL  TABLE  ; W contains table
                 ; offset value
                 ; W now has
                 ; table value
   :
TABLE
    ADDWF PCL    ; W = offset
    RETLW k0     ; Begin table
    RETLW k1     ;
   :
   :
    RETLW kn     ; End of table
```

Before Instruction
W = 07h
After Instruction
W = value of kn

FIGURE 27-3: HIGH/LOW-VOLTAGE DETECT CHARACTERISTICS

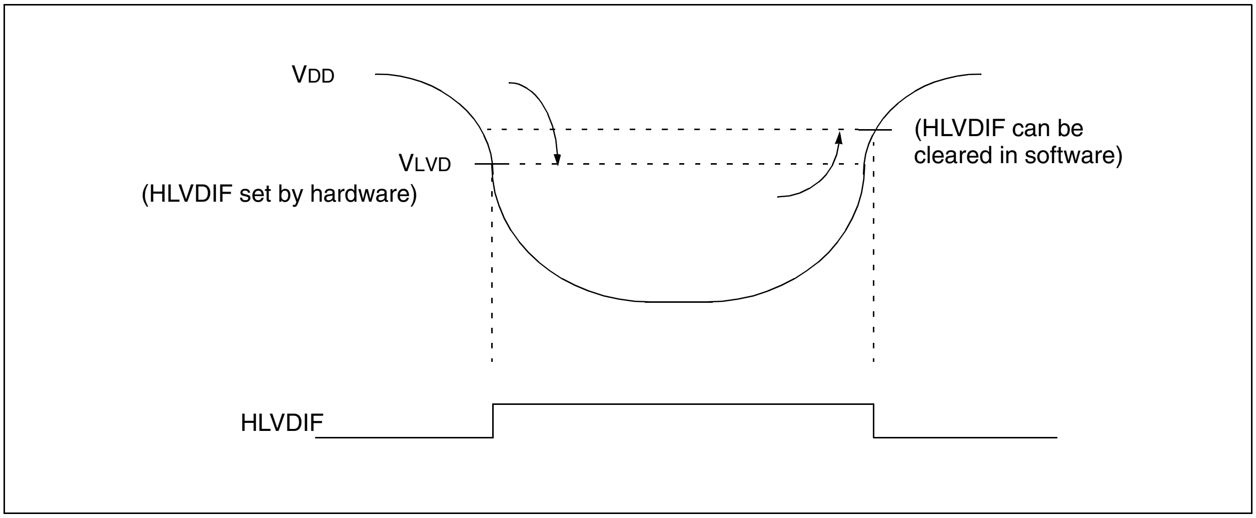


TABLE 27-4: HIGH/LOW-VOLTAGE DETECT CHARACTERISTICS

				Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ T_A ≤ +85°C for industrial				
Param No.	**Symbol**	**Characteristic**		**Min**	**Typ†**	**Max**	**Units**	**Conditions**
D420		HLVD Voltage on VDD Transition High to Low	LVV = 0000	2.12	2.17	2.22	V	
			LVV = 0001	2.18	2.23	2.28	V	
			LVV = 0010	2.31	2.36	2.42	V	
			LVV = 0011	2.38	2.44	2.49	V	
			LVV = 0100	2.54	2.60	2.66	V	
			LVV = 0101	2.72	2.79	2.85	V	
			LVV = 0110	2.82	2.89	2.95	V	
			LVV = 0111	3.05	3.12	3.19	V	
			LVV = 1000	3.31	3.39	3.47	V	
			LVV = 1001	3.46	3.55	3.63	V	
			LVV = 1010	3.63	3.71	3.80	V	
			LVV = 1011	3.81	3.90	3.99	V	
			LVV = 1100	4.01	4.11	4.20	V	
			LVV = 1101	4.23	4.33	4.43	V	
			LVV = 1110	4.48	4.59	4.69	V	
			LVV = 1111	1.14	1.20	1.26	V	

† Production tested at T_{AMB} = 25°C. Specifications over temperature limits ensured by characterization.

28.0 DC AND AC CHARACTERISTICS GRAPHS AND TABLES

Graphs and tables are not available at this time.

C

D

