Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

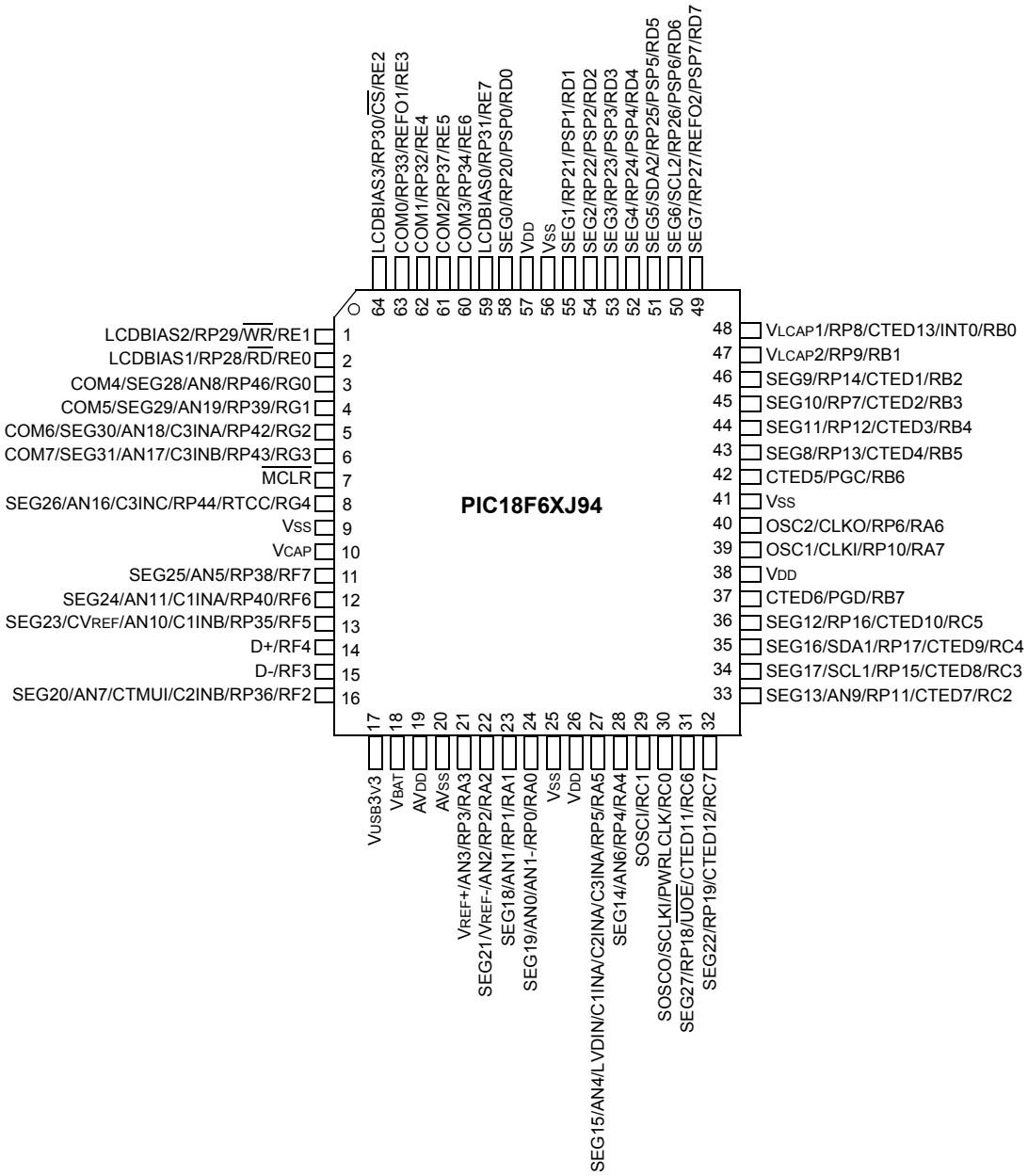
"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" 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	I²C, IrDA, LINbus, SPI, UART/USART, USB
Peripherals	Brown-out Detect/Reset, HLVD, LCD, POR, PWM, WDT
Number of I/O	68
Program Memory Size	128KB (64K x 16)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	4K x 8
Voltage - Supply (Vcc/Vdd)	2V ~ 3.6V
Data Converters	A/D 24x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	80-TQFP
Supplier Device Package	80-TQFP (12x12)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f87j94-i-pt

PIN DIAGRAMS

FIGURE 1: 64-PIN TQFP, QFN DIAGRAM FOR PIC18F6XJ94



Note 1: Pinouts are subject to change.

2: See Table 2 for the pin allocation table.

3.2 Oscillator Configuration

The oscillator source (and operating mode) that is used at a device Power-on Reset (POR) event is selected using Configuration bit settings. The Oscillator Configuration bit settings are in the Configuration registers located in the program memory (refer to **Section 28.1 "Configuration Bits"** for more information). The Primary Oscillator Configuration bits, POSCMD<1:0> (CONFIG3L<1:0>), and Oscillator Configuration bits, FOSC<2:0> (CONFIG2L<2:0>), select the oscillator source that is used at a POR. The FRC Oscillator with Postscaler (FRCDIV) is the default (unprogrammed) selection. The Secondary Oscillator, or one of the internal oscillators, may be chosen by programming these bit locations.

The Configuration bits allow users to choose between 11 different clock modes, as shown in Table 3-1.

TABLE 3-1: CONFIGURATION BIT VALUES FOR CLOCK SELECTION

Oscillator Mode	Oscillator Source	POSCMD<1:0>	FOSC<2:0>	Notes
Fast RC Oscillator with Postscaler (FRCDIV)	Internal	11	111	**1, 2**
Fast RC Oscillator divided by 16 (FRC500kHz)	Internal	11	110	**1**
Low-Power RC Oscillator (LPRC)	Internal	11	101	**1**
Secondary (Timer1) Oscillator (SOSC)	Secondary	11	100	**1**
Primary Oscillator (HS) with PLL Module (HSPLL)	Primary	10	011	
Primary Oscillator (MS) with PLL Module (MSPLL)	Primary	01	011	
Primary Oscillator (EC) with PLL Module (ECPLL)	Primary	00	011	
Primary Oscillator (HS)	Primary	10	010	
Primary Oscillator (MS)	Primary	01	010	
Primary Oscillator (EC)	Primary	00	010	
Fast RC Oscillator with PLL Module (FRCPLL)	Internal	11	001	**1**
Fast RC Oscillator (FRC)	Internal	11	000	**1**

Note 1: OSC2 pin function is determined by the CLKOEN Configuration bit.

2: Default oscillator mode for an unprogrammed (erased) device.

4.4.10 CONTROL BIT SUMMARY FOR SLEEP MODES

Table 4-5 shows the settings for the bits relevant to Deep Sleep modes.

TABLE 4-5: BIT SETTINGS FOR ALL DEEP SLEEP MODES

Instruction-Based Mode	DSEN (DSCONH<7>)	Retention Regulator			DSWDTEN (CONFIG8H<0>)
		$\overline{\text{RETEN}}$ (CONFIG7L<0>)	SRETEN (RCON4<4>)	State	
Retention Deep Sleep	1	0	1	Enabled	0
Deep Sleep	1	1	x	Disabled	x

4.4.11 WAKE-UP DELAYS

The Reset delays associated with wake-up from Deep Sleep and Retention Deep Sleep modes, in different oscillator modes, are provided in Table 4-6 and Table 4-7, respectively.

Note: The PMSLP bit (RCON4<0>) allows the voltage regulator to be maintained during Sleep modes.

TABLE 4-6: DELAY TIMES FOR EXITING FROM DEEP SLEEP MODE

Clock Source		Exit Delay	Oscillator Delay	Notes
EC		TDSWU	—	
ECPLL		TDSWU	TLOCK	**1, 3**
MS, HS		TDSWU	TOST	**1, 2**
MSPLL, HSPLL		TDSWU	TOST + TLOCK	**1, 2, 3**
SOSC	(Off during Sleep)	TDSWU	TOST	**1, 2**
	(On during Sleep)	TDSWU	—	**1**
FRC, FRCDIV		TDSWU	TFRC	**1, 4**
LPRC	(Off during Sleep)	TDSWU	TLPRC	**1, 4**
	(On during Sleep)	TDSWU	—	**1**
FRCPLL		TDSWU	TFRC + TLOCK	**1, 3, 4**

Note 1: TDSWU = Deep Sleep wake-up delay.

2: TOST = Oscillator Start-up Timer; a delay of 1024 oscillator periods before the oscillator clock is released to the system.

3: TLOCK = PLL lock time.

4: TFRC and TLPRC are RC Oscillator start-up times.

TABLE 6-2: REGISTER FILE SUMMARY

File Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
FFFh	TOSU	—	—	—	Top-of-Stack Upper Byte (TOS<20:16>)				
FFEh	TOSH	Top-of-Stack High Byte (TOS<15:8>)							
FFDh	TOSL	Top-of-Stack Low Byte (TOS<7:0>)							
FFCh	STKPTR	STKFUL	STKUNF	—	STKPTR				
FFBh	PCLATU	—	—	—	Holding Register for PC<20:16>				
FFAh	PCLATH	Holding Register for PC<15:8>							
FF9h	PCL	PC Low Byte (PC<7:0>)							
FF8h	TBLPTRU	—	—	ACSS	Program Memory Table Pointer Upper Byte (TBLPTR<20:16>)				
FF7h	TBLPTRH	Program Memory Table Pointer High Byte (TBLPTR<15:8>)							
FF6h	TBLPTRL	Program Memory Table Pointer Low Byte (TBLPTR<7:0>)							
FF5h	TABLAT	Program Memory Table Latch							
FF4h	PRODH	Product Register High Byte							
FF3h	PRODL	Product Register Low Byte							
FF2h	INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	IOCIE	TMR0IF	INT0IF	IOCIF
FF1h	INTCON2	$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	IOCIP
FF0h	INTCON3	INT2IP	INT1IP	INT3IE	INT2IE	INT1IE	INT3IF	INT2IF	INT1IF
FEFh	INDF0	Uses contents of FSR0 to address data memory – value of FSR0 not changed (not a physical register)							
FEEh	POSTINC0	Uses contents of FSR0 to address data memory – value of FSR0 post-incremented (not a physical register)							
FEDh	POSTDEC0	Uses contents of FSR0 to address data memory – value of FSR0 post-decremented (not a physical register)							
FECh	PREINC0	Uses contents of FSR0 to address data memory – value of FSR0 pre-incremented (not a physical register)							
FEBh	PLUSW0	Uses contents of FSR0 to address data memory – value of FSR0 pre-incremented (not a physical register) – value of FSR0 offset by W							
FEAh	FSR0H	—	—	—	—	Indirect Data Memory Address Pointer 0 High			
FE9h	FSR0L	Indirect Data Memory Address Pointer 0 Low Byte							
FE8h	WREG	Working Register							
FE7h	INDF1	Uses contents of FSR1 to address data memory – value of FSR1 not changed (not a physical register)							
FE6h	POSTINC1	Uses contents of FSR1 to address data memory – value of FSR1 post-incremented (not a physical register)							
FE5h	POSTDEC1	Uses contents of FSR1 to address data memory – value of FSR1 post-decremented (not a physical register)							
FE4h	PREINC1	Uses contents of FSR1 to address data memory – value of FSR1 pre-incremented (not a physical register)							
FE3h	PLUSW1	Uses contents of FSR1 to address data memory – value of FSR1 pre-incremented (not a physical register) – value of FSR1 offset by W							
FE2h	FSR1H	—	—	—	—	Indirect Data Memory Address Pointer 1 High			
FE1h	FSR1L	Indirect Data Memory Address Pointer 1 Low Byte							
FE0h	BSR	—	—	—	—	Bank Select Register			
FDFh	INDF2	Uses contents of FSR2 to address data memory – value of FSR2 not changed (not a physical register)							
FDEh	POSTINC2	Uses contents of FSR2 to address data memory – value of FSR2 post-incremented (not a physical register)							
FDDh	POSTDEC2	Uses contents of FSR2 to address data memory – value of FSR2 post-decremented (not a physical register)							
FDCh	PREINC2	Uses contents of FSR2 to address data memory – value of FSR2 pre-incremented (not a physical register)							
FDBh	PLUSW2	Uses contents of FSR2 to address data memory – value of FSR2 pre-incremented (not a physical register) – value of FSR2 offset by W							
FDAh	FSR2H	—	—	—	—	Indirect Data Memory Address Pointer 2 High			
FD9h	FSR2L	Indirect Data Memory Address Pointer 2 Low Byte							
FD8h	STATUS	—	—	—	N	OV	Z	DC	C
FD7h	TMR0H	Timer0 Register High Byte							
FD6h	TMR0L	Timer0 Register Low Byte							
FD5h	T0CON	TMR0ON	T08BIT	T0CS1	T0CS0	PSA	T0PS2	T0PS1	T0PS0
FD4h	Unimplemented	—	—	—	—	—	—	—	—
FD3h	OSCCON	IDLEN	COSC2	COSC1	COSC0	—	NOSC2	NOSC1	NOSC0
FD2h	IPR5	—	ACTORSIP	ACTLOCKIP	TMR8IP	—	TMR6IP	TMR5IP	TMR4IP
FD1h	IOCF	IOCF7	IOCF6	IOCF5	IOCF4	IOCF3	IOCF2	IOCF1	IOCF0
FD0h	RCON	IPEN	—	$\overline{\text{CM}}$	$\overline{\text{RI}}$	$\overline{\text{TO}}$	$\overline{\text{PD}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$

Legend: — = unimplemented, read as '0'.

TABLE 6-2: REGISTER FILE SUMMARY (CONTINUED)

File Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
E36h	RPINR24_25	IOC7R<3:0>				IOC6R<3:0>			
E35h	RPINR22_23	IOC5R<3:0>				IOC4R<3:0>			
E34h	RPINR20_21	IOC3R<3:0>				IOC2R<3:0>			
E33h	RPINR18_19	IOC1R<3:0>				IOC0R<3:0>			
E32h	RPINR16_17	ECCP3R<3:0>				ECCP2R<3:0>			
E31h	RPINR14_15	ECCP1R<3:0>				FLT0R<3:0>			
E30h	RPINR12_13	SS2R<3:0>				SDI2R<3:0>			
E2Fh	RPINR10_11	SCK2R<3:0>				SS1R<3:0>			
E2Eh	RPINR8_9	SDI1R<3:0>				SCK1R<3:0>			
E2Dh	RPINR6_7	U4TXR<3:0>				U4RXR<3:0>			
E2Ch	RPINR4_5	U3TXR<3:0>				U3RXR<3:0>			
E2Bh	RPINR2_3	U2TXR<3:0>				U2RXR<3:0>			
E2Ah	RPINR0_1	U1TXR<3:0>				U1RXR<3:0>			
E29h	RPOR46	—	—	—	—	RPO46R<3:0>			
E28h	RPOR44_45	RPO45R<3:0>				RPO44R<3:0>			
E27h	RPOR42_43	RPO43R<3:0>				RPO42R<3:0>			
E26h	RPOR40_41	RPO41R<3:0>				RPO40R<3:0>			
E25h	RPOR38_39	RPO39R<3:0>				RPO38R<3:0>			
E24h	RPOR36_37	RPO37R<3:0>				RPO36R<3:0>			
E23h	RPOR34_35	RPO35R<3:0>				RPO34R<3:0>			
E22h	RPOR32_33	RPO33R<3:0>				RPO32R<3:0>			
E21h	RPOR30_31	RPO31R<3:0>				RPO30R<3:0>			
E20h	RPOR28_29	RPO29R<3:0>				RPO28R<3:0>			
E1Fh	RPOR26_27	RPO27R<3:0>				RPO26R<3:0>			
E1Eh	RPOR24_25	RPO25R<3:0>				RPO24R<3:0>			
E1Dh	RPOR22_23	RPP23R<3:0>				RPO22R<3:0>			
E1Ch	RPOR20_21	RPO21R<3:0>				RPO20R<3:0>			
E1Bh	RPOR18_19	RPO19R<3:0>				RPO18R<3:0>			
E1Ah	RPOR16_17	RPO17R<3:0>				RPO16R<3:0>			
E19h	RPOR14_15	RPO15R<3:0>				RPO14R<3:0>			
E18h	RPOR12_13	RPO13R<3:0>				RPO12R<3:0>			
E17h	RPOR10_11	RPO11R<3:0>				RPO10R<3:0>			
E16h	RPOR8_9	RPO9R<3:0>				RPO8R<3:0>			
E15h	RPOR6_7	RPO7R<3:0>				RPO6R<3:0>			
E14h	RPOR4_5	RPO5R<3:0>				RPO4R<3:0>			
E13h	RPOR2_3	RPO3R<3:0>				RPO2R<3:0>			
E12h	RPOR0_1	RPO1R<3:0>				RPO0R<3:0>			
E11h	UCFG	UTEYE	UOEMON	—	UPUEN	UTRDIS	FSEN	PPB1	PPB0
E10h	UIE	—	SOFIE	STALLIE	IDLEIE	TRNIE	ACTVIE	UERRIE	URSTIE
E0Fh	UEIE	BTSEE	—	—	BTOEE	DFN8EE	CRC16EE	CRC5EE	PIDEE
E0Eh	UEP15	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
E0Dh	UEP14	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
E0Ch	UEP13	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
E0Bh	UEP12	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
E0Ah	UEP11	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
E09h	UEP10	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
E08h	UEP9	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
E07h	UEP8	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
E06h	UEP7	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
E05h	UEP6	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
E04h	UEP5	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL

Legend: — = unimplemented, read as '0'.

7.3 Reading the Flash Program Memory

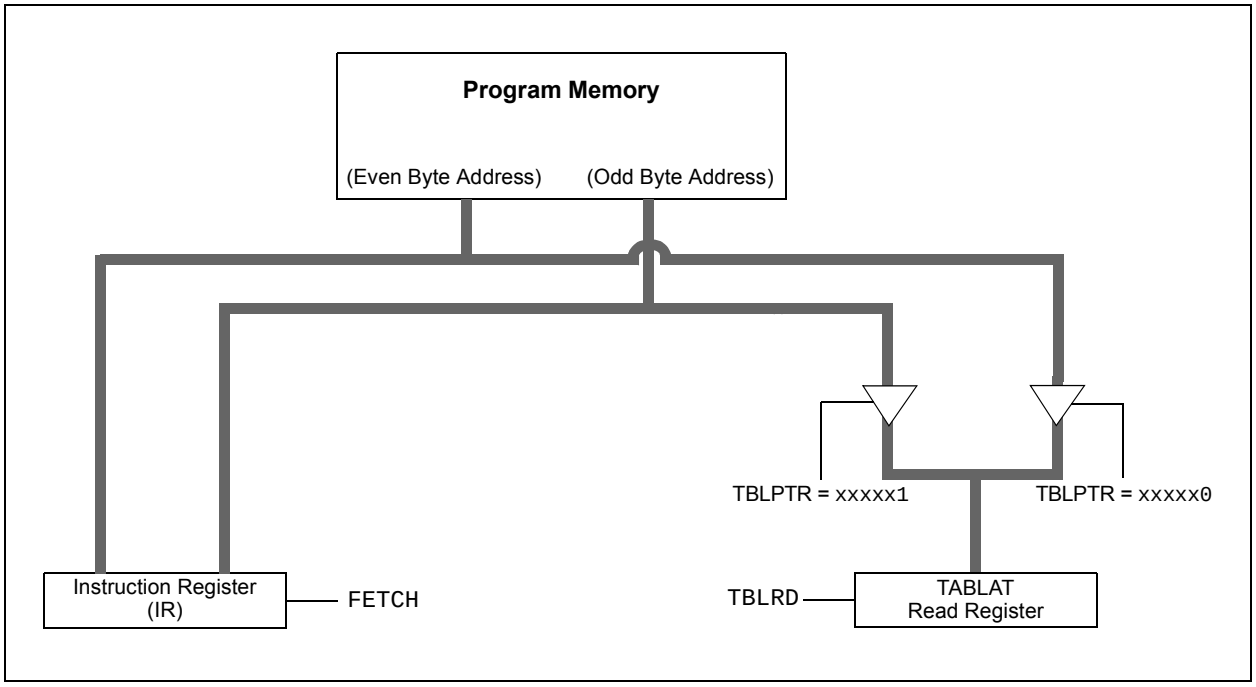
The TBLRD instruction is used to retrieve data from program memory and places it into data RAM. Table reads from program memory are performed one byte at a time.

The TBLPTR points to a byte address in program space. Executing TBLRD places the byte pointed to into TABLAT. In addition, the 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 7-4 shows the interface between the internal program memory and the TABLAT.

FIGURE 7-4: READS FROM FLASH PROGRAM MEMORY



EXAMPLE 7-1: 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
            MOVWF   WORD_ODD
```

REGISTER 10-2: INTCON2: INTERRUPT CONTROL REGISTER 2

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	IOCIP
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 **$\overline{\text{RBPU}}$:** PORTB Pull-up Enable bit
1 = All PORTB pull-ups are disabled
0 = PORTB pull-ups are enabled by individual port latch values

bit 6 **INTEDG0:** External Interrupt 0 Edge Select bit
1 = Interrupt on rising edge
0 = Interrupt on falling edge

bit 5 **INTEDG1:** External Interrupt 1 Edge Select bit
1 = Interrupt on rising edge
0 = Interrupt on falling edge

bit 4 **INTEDG2:** External Interrupt 2 Edge Select bit
1 = Interrupt on rising edge
0 = Interrupt on falling edge

bit 3 **INTEDG3:** External Interrupt 3 Edge Select bit
1 = Interrupt on rising edge
0 = Interrupt on falling edge

bit 2 **TMR0IP:** TMR0 Overflow Interrupt Priority bit
1 = High priority
0 = Low priority

bit 1 **INT3IP:** INT3 External Interrupt Priority bit
1 = High priority
0 = Low priority

bit 0 **IOCIP:** RB Port Change Interrupt Priority bit
1 = High priority
0 = Low priority

Note: Interrupt flag bits are set when an interrupt condition occurs regardless of the state of its corresponding enable bit or the Global Interrupt Enable bit. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt. This feature allows for software polling.

REGISTER 10-9: PIR6: PERIPHERAL INTERRUPT REQUEST (FLAG) REGISTER 6

R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0
RC4IF	TX4IF	RC3IF	TX3IF	—	CMP3IF	CMP2IF	CMP1IF
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 **RC4IF:** EUSART4 Receive Interrupt Flag bit
1 = The EUSART4 receive buffer is full (cleared by reading RCREG4)
0 = The EUSART4 receive buffer is empty

bit 6 **TX4IF**: EUSART4 Transmit Interrupt Flag bit
1 = The EUSART4 transmit buffer is empty (cleared by writing to TXREG4)
0 = The EUSART4 transmit buffer is full

bit 5 **RC3IF**: EUSART3 Receive Interrupt Flag bit
1 = The EUSART3 receive buffer is full (cleared by reading RCREG3)
0 = The EUSART3 receive buffer is empty

bit 4 **TX3IF:** EUSART3 Transmit Interrupt Flag bit
1 = The EUSART3 transmit buffer is empty (cleared by writing to TXREG3)
0 = The EUSART3 transmit buffer is full

bit 3 **Unimplemented:** Read as ‘0’

bit 2 **CMP3IF:** CMP3 Interrupt Flag bit
1 = CMP3 interrupt occurred (must be cleared in software)
0 = No CMP3 interrupt occurred

bit 1 **CMP2IF:** CMP2 Interrupt Flag bit
1 = CMP2 interrupt occurred (must be cleared in software)
0 = No CMP2 interrupt occurred

bit 0 **CMP1IF:** CM1 Interrupt Flag bit
1 = CMP1 interrupt occurred (must be cleared in software)
0 = No CMP1 interrupt occurred

REGISTER 11-4: PSPCON: PARALLEL SLAVE PORT CONTROL REGISTER

R-0	R-0	R/W-0	R/W-0	U-0	U-0	U-0	U-0
IBF	OBF	IBOV	PSPMODE	—	—	—	—
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 **IBF:** Input Buffer Full Status bit
1 = A word has been received and is waiting to be read by the CPU
0 = No word has been received

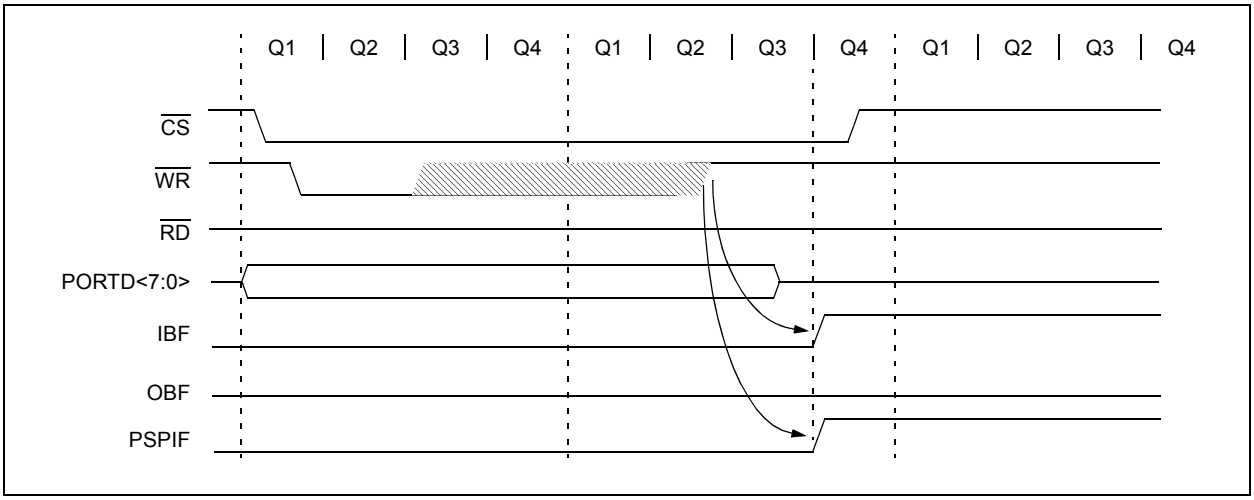
bit 6 **OBF:** Output Buffer Full Status bit
1 = The output buffer still holds a previously written word
0 = The output buffer has been read

bit 5 **IBOV:** Input Buffer Overflow Detect bit
1 = A write occurred when a previously input word had 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 3-0 **Unimplemented:** Read as ‘0’

FIGURE 11-4: PARALLEL SLAVE PORT WRITE WAVEFORMS



REGISTER 13-6: LCDREF: LCD REFERENCE LADDER 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
LCDIRE	—	LCDCST2	LCDCST1	LCDCST0	VLCD3PE	VLCD2PE	VLCD1PE
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 **LCDIRE**: LCD Internal Reference Enable bit
1 = Internal LCD reference is enabled and connected to the internal contrast control circuit
0 = Internal LCD reference is disabled

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

bit 5-3 **LCDCST<2:0>**: LCD Contrast Control bits
Selects the Resistance of the LCD Contrast Control Resistor Ladder:
111 =Resistor ladder is at maximum resistance (minimum contrast)
110 =Resistor ladder is at 6/7th of maximum resistance
101 =Resistor ladder is at 5/7th of maximum resistance
100 =Resistor ladder is at 4/7th of maximum resistance
011 =Resistor ladder is at 3/7th of maximum resistance
010 =Resistor ladder is at 2/7th of maximum resistance
001 =Resistor ladder is at 1/7th of maximum resistance
000 =Minimum resistance (maximum contrast); resistor ladder is shorted

bit 2 **VLCD3PE**: Bias3 Pin Enable bit
1 = BIAS3 level is connected to the external pin, LCDBIAS3
0 = BIAS3 level is internal (internal resistor ladder)

bit 1 **VLCD2PE**: Bias2 Pin Enable bit
1 = BIAS2 level is connected to the external pin, LCDBIAS2
0 = BIAS2 level is internal (internal resistor ladder)

bit 0 **VLCD1PE**: Bias1 Pin Enable bit
1 = BIAS1 level is connected to the external pin, LCDBIAS1
0 = BIAS1 level is internal (internal resistor ladder)

13.13 LCD Interrupts

The LCD timing generation provides an interrupt that defines the LCD frame timing. This interrupt can be used to coordinate the writing of the pixel data with the start of a new frame, which produces a visually crisp transition of the image.

This interrupt can also be used to synchronize external events to the LCD. For example, the interface to an external segment driver can be synchronized for segment data updates to the LCD frame.

A new frame is defined as beginning at the leading edge of the COM0 common signal. The interrupt will be set immediately after the LCD controller completes accessing all pixel data required for a frame. This will occur at a fixed interval before the frame boundary (TFINT), as shown in Figure 13-22.

The LCD controller will begin to access data for the next frame within the interval from the interrupt to when the controller begins accessing data after the interrupt (TFWR). New data must be written within TFWR, as this is when the LCD controller will begin to access the data for the next frame.

When the LCD driver is running with Type-B waveforms, and the LMUX<2:0> bits are not equal to '000', there are some additional issues.

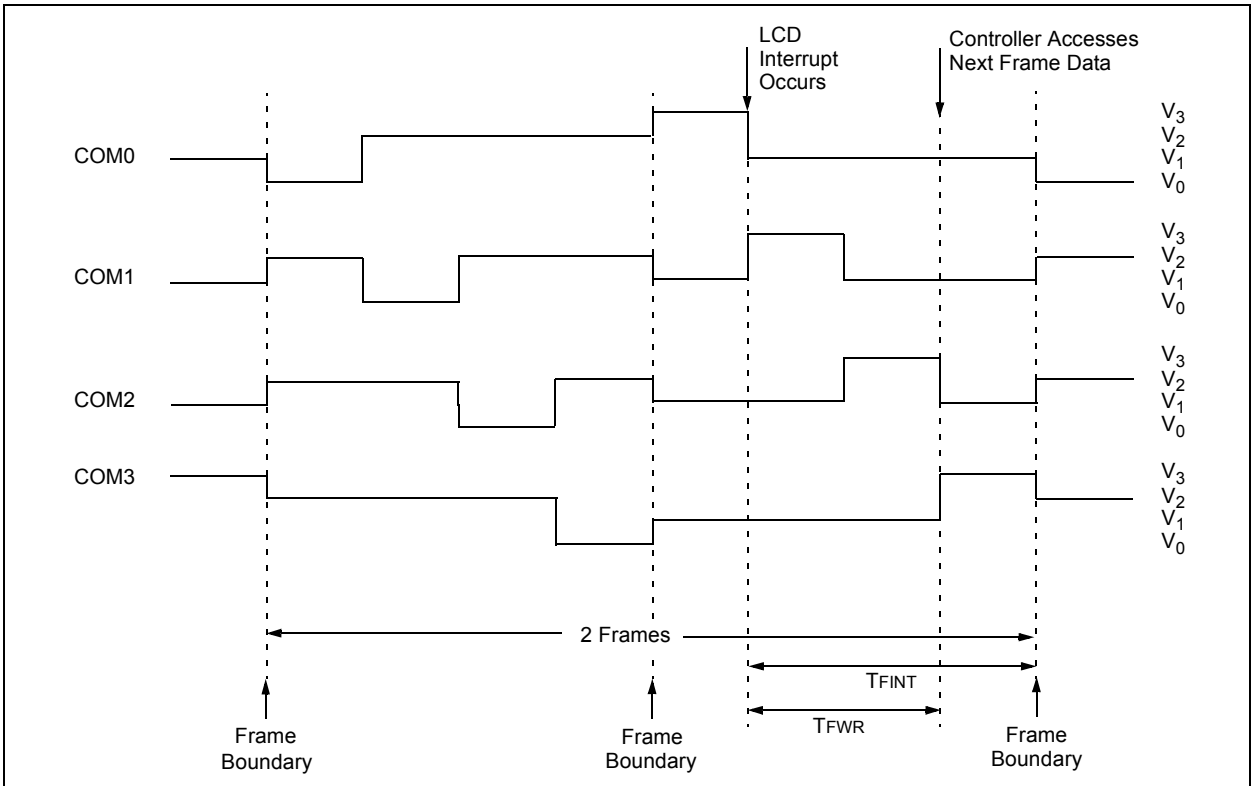
Since the DC voltage on the pixel takes two frames to maintain 0V, the pixel data must not change between subsequent frames. If the pixel data were allowed to change, the waveform for the odd frames would not necessarily be the complement of the waveform generated in the even frames and a DC component would be introduced into the panel.

Because of this, using Type-B waveforms requires synchronizing the LCD pixel updates to occur within a subframe after the frame interrupt.

To correctly sequence writing in Type-B, the interrupt only occurs on complete phase intervals. If the user attempts to write when the write is disabled, the WERR bit (LCDCON<5>) is set.

Note:	The interrupt is not generated when the Type-A waveform is selected and when the Type-B with no multiplex (static) is selected.

FIGURE 13-22: EXAMPLE WAVEFORMS AND INTERRUPT TIMING IN QUARTER DUTY CYCLE DRIVE



TFWR = TFRAME/2 * (LMUX<2:0> + 1) + TCY/2

TFINT = (TFWR/2 – (2 TCY + 40 ns)) → Minimum = 1.5(TFRAME/4) – (2 TCY + 40 ns)
(TFWR/2 – (1 TCY + 40 ns)) → Maximum = 1.5(TFRAME/4) – (1 TCY + 40 ns)

REGISTER 16-1: TxCON: TIMERx CONTROL REGISTER (TIMER2/4/6/8)

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	TxOUTPS3	TxOUTPS2	TxOUTPS1	TxOUTPS0	TMRxON	TxCKPS1	TxCKPS0
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 **Unimplemented:** Read as '0'

bit 6-3 **TxOUTPS<3:0>:** Timerx Output Postscale Select bits
0000 = 1:1 Postscale
0001 = 1:2 Postscale
•
•
•
1111 = 1:16 Postscale

bit 2 **TMRxON**: Timerx On bit
1 = Timerx is on
0 = Timerx is off

bit 1-0 **TxCKPS<1:0>:** Timerx Clock Prescale Select bits
00 = Prescaler is 1
01 = Prescaler is 4
1x = Prescaler is 16

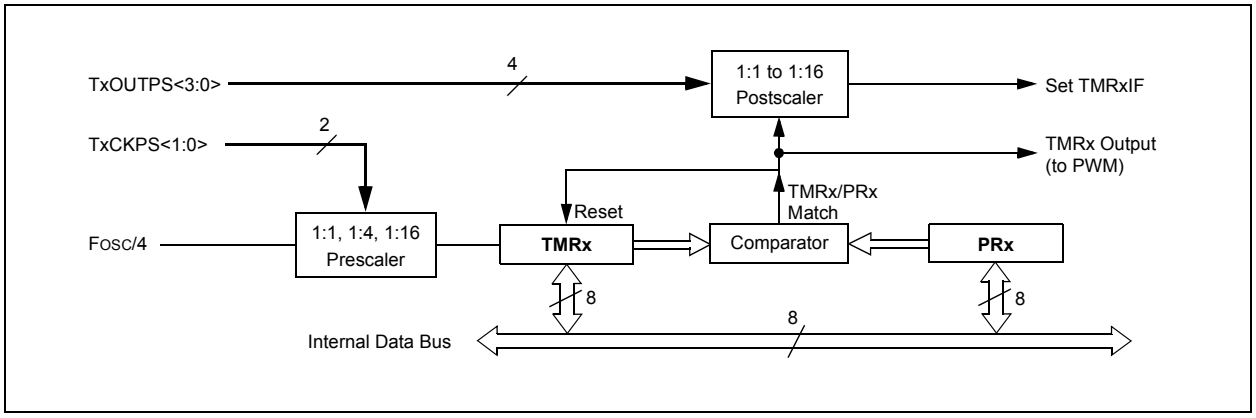
16.2 Timer2/4/6/8 Interrupt

The Timer2/4/6/8 modules have 8-bit Period registers, PRx, that are both readable and writable. Timer2/4/6/8 increment from 00h until they match PR2/4/6/8 and then reset to 00h on the next increment cycle. The PRx registers are initialized to FFh upon Reset.

16.3 Output of TMRx

The outputs of TMRx (before the postscaler) are used only as a PWM time base for the ECCP modules. They are not used as baud rate clocks for the MSSPx modules as is the Timer2 output.

FIGURE 16-1: TIMER2/4/6/8 BLOCK DIAGRAM



The SPI DMA module can write received data to all general purpose memory on the device, including memory used for USB endpoint buffers. The SPI DMA module cannot be used to modify the Special Function Registers contained in Banks 14 and 15.

20.4.5 INTERRUPTS

The SPI DMA module alters the behavior of the SSP1IF interrupt flag. In normal non-DMA modes, the SSP1IF is set once after every single byte is transmitted/received through the MSSP1 module. When MSSP1 is used with the SPI DMA module, the SSP1IF interrupt flag will be set according to the user-selected INTLVL<3:0> value specified in the DMACON2 register. The SSP1IF interrupt condition will also be generated once the SPI DMA transaction has fully completed and the DMAEN bit has been cleared by hardware.

The SSP1IF flag becomes set once the DMA byte count value indicates that the specified INTLVLx has been reached. For example, if DMACON2<3:0> = 0101 (16 bytes remaining), the SSP1IF interrupt flag will become set once DMABC reaches 00Fh. If user firmware then clears the SSP1IF interrupt flag, the flag will not be set again by the hardware until after all bytes have been fully transmitted and the DMA transaction is complete.

Note:	User firmware may modify the INTLVLx bits while a DMA transaction is in progress (DMAEN = 1). If an INTLVLx value is selected which is higher than the actual remaining number of bytes (indicated by DMABC + 1), the SSP1IF interrupt flag will immediately become set.

For example, if DMABC = 00Fh (implying 16 bytes are remaining) and user firmware writes '1111' to INTLVL<3:0> (interrupt when 576 bytes are remaining), the SSP1IF interrupt flag will immediately become set. If user firmware clears this interrupt flag, a new interrupt condition will not be generated until either: user firmware again writes INTLVLx with an interrupt level higher than the actual remaining level, or the DMA transaction completes and the DMAEN bit is cleared.

Note:	If the INTLVLx bits are modified while a DMA transaction is in progress, care should be taken to avoid inadvertently changing the DLYCYC<3:0> value.

22.0 12-BIT A/D CONVERTER WITH THRESHOLD SCAN

The 12-bit A/D Converter has the following key features:

- Successive Approximation Register (SAR) Conversion
- Conversion Speeds of up to 200 ksps at 12 bits and 500 ksps at 10 bits
- Up to 32 Analog Input Channels (internal and external)
- Selectable 10-Bit or 12-Bit (default) Conversion
- Resolution
- Multiple Internal Reference Input Channels
- External Voltage Reference Input Pins
- Unipolar Differential Sample-and-Hold (S/H) Amplifier
- Automated Threshold Scan and Compare Operation to Pre-Evaluate up to 26 Conversion Results
- Selectable Conversion Trigger Source
- Fixed Length (one word per channel), Configurable Conversion Result Buffer
- Four Options for Results Alignment
- Configurable Interrupt Generation
- Operation During CPU Sleep and Idle modes

The 12-bit A/D Converter module is an enhanced version of the 10-bit module offered in some PIC18 devices. Both modules are Successive Approximation Register (SAR) Converters at their cores, surrounded by a range of hardware features for flexible configuration. This version of the module extends functionality by providing 12-bit resolution, a wider range of automatic sampling options, tighter integration with other analog modules, such as the CTMU, and a configurable results buffer. This module also includes a unique Threshold Detect feature that allows the module itself to make simple decisions based on the conversion results.

As before, an internal Sample-and-Hold (S/H) amplifier acquires a sample of an input signal, then holds that value constant during the conversion process. A combination of input multiplexers selects the signal to be converted from up to 32 analog inputs, both external (analog input pins) and internal (e.g., on-chip voltage references and other analog modules). The whole multiplexer path includes provisions for differential analog input, although, with a limited number of negative input pins. The sampled voltage is held and converted to a digital value, which strictly speaking, represents the ratio of that input voltage to a reference voltage. Configuration choices allow connection of an external reference or use of the device power and ground (AV_{DD} and AV_{SS}). Reference and input signal pins are assigned differently depending on the particular device.

An array of timing and control selections allow the user to create flexible scanning sequences. Conversions can be started individually by program control, continuously free-running or triggered by selected hardware events. A single channel may be repeatedly converted. Alternate conversions may be performed on two channels, or any or all of the channels may be sequentially scanned and converted according to a user-defined bit map. The resulting conversion output is a 12-bit digital number, which can be signed or unsigned, left or right justified. (In some devices, a user-selectable resolution of ten bits is available; in other devices, 10-bit resolution is the only option available.)

Conversions are automatically stored in a dedicated buffer, allowing for multiple successive readings to be taken before software service is needed. The buffer can be configured to function as a FIFO buffer or as a channel indexed buffer. In FIFO mode, the buffer can be split into two equal sections for simultaneous conversion and read operations. In Indexed mode, the buffer can use the Threshold Scan feature to determine if a conversion meets specific, user-defined criteria, storing or discarding the converted value as appropriate, and then set semaphore flags to indicate the event. This allows conversions to occur in low-power modes when the CPU is inactive, waking the device only when specific conditions have occurred.

The module sets its interrupt flag after a selectable number of conversions, when the buffer can be read, or after a successful Threshold Detect comparison. After the interrupt, the sequence restarts at the beginning of the buffer. When the interrupt flag is set, according to the earlier selection, scan selections and the Output Buffer Pointer return to their starting positions.

During Sleep or Idle mode, the A/D can wake-up at preconfigured intervals while the device maintains a Low-Power mode. If threshold conditions have not been met on any of the conversions, the module will return to a Low-Power mode.

The A/D module provides configuration to directly interact with the CTMU on specific input channels. This allows the CTMU to automatically turn on only when requested directly by the A/D, even though the rest of the device stays in Sleep mode.

A simplified block diagram for the module is shown in Figure 22-1.

22.1 Registers

The 12-bit A/D converter module uses up to 75 registers for its operation. All registers are mapped in the data memory space.

22.1.1 CONTROL REGISTERS

Depending on the specific device, the module has up to twelve control and STATUS registers:

- ADCON1H/L: A/D Control Registers
- ADCON2H/L: A/D Control Registers
- ADCON3H/L: A/D Control Registers
- ADCON5H/L: A/D Control Registers
- ADCHS0H/L: A/D Input Channel Select Registers
- ADCHITH1H/L and ADCHITH0H/L: A/D Scan Compare Hit Registers
- ADCSS1H/L and ADCSS0H/L: A/D Input Scan Select Registers
- ADCTMUEN1H/L and ADCTMUEN0H/L: CTMU Enable Register

The ADCON1H/L, ADCON2H/L and ADCON3H/L registers control the overall operation of the A/D module. This includes enabling the module, configuring the conversion clock and voltage reference sources, selecting the sampling and conversion triggers, and manually controlling the sample/convert sequences. The ADCON5H/L registers specifically controls features of Threshold Detect operation, including its functioning in power-saving modes.

The ADCHS0H/L registers selects the input channels to be connected to the S/H amplifier. It also allows the choice of input multiplexers and the selection of a reference source for differential sampling.

The ADCHITH1H/L and ADCHITH0H/L registers are semaphore registers used with Threshold Detect operations. The status of individual bits, or bit pairs in some cases, indicate if a match condition has occurred. Their use is described in more detail in Section 22.7 “Threshold Detect Operation”. ADCHITH0H/L is always implemented, whereas ADCHITH1H/L may not be implemented in devices with 16 channels or less. The ADCSS0H/L/L registers select the channels to be included for sequential scanning. The ADCTMUEN1H/L/L registers select the channel(s) to be used by the CTMU during conversions. Selecting a particular channel allows the A/D Converter to control the CTMU (particularly, its current source) and read its data through that channel. ADCTMUEN0H/L is always implemented, whereas ADCTMUEN1H/L may not be implemented in devices with 16 channels or less.

22.1.2 A/D RESULT BUFFERS

The module incorporates a multi-word, dual port RAM, called ADCBUF. The buffer is composed of at least the same number of word locations as there are external analog channels for a particular device, with a maximum number of 26. The number of buffer addresses is always even. Each of the locations is mapped into the data memory space and is separately addressable.The buffer locations are referred to as ADCBUF0H/L through ADCBUFnH/L (up to 26).

The A/D result buffers are both readable and writable. When the module is active (ADCON1H<7> = 1), the buffers are read-only, and store the results of A/D conversions. When the module is inactive (ADCON1H<7> = 0), the buffers are both readable and writable. In this state, writing to a buffer location programs a conversion threshold for Threshold Detect operations, as described in Section 22.7.2, Setting Comparison Thresholds.

REGISTER 22-20: ADCSS0H: A/D INPUT SCAN SELECT REGISTER 0 HIGH (HIGH WORD)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CSS15	CSS14	CSS13	CSS12	CSS11	CSS10	CSS9	CSS8
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 **CSS<15:8>:** A/D Input Scan Selection bits
1 = Includes corresponding channel for input scan
0 = Skips channel for input scan

REGISTER 22-21: ADCSS0L: A/D INPUT SCAN SELECT REGISTER 0 LOW (LOW WORD)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CSS7	CSS6	CSS5	CSS4	CSS3	CSS2	CSS1	CSS0
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

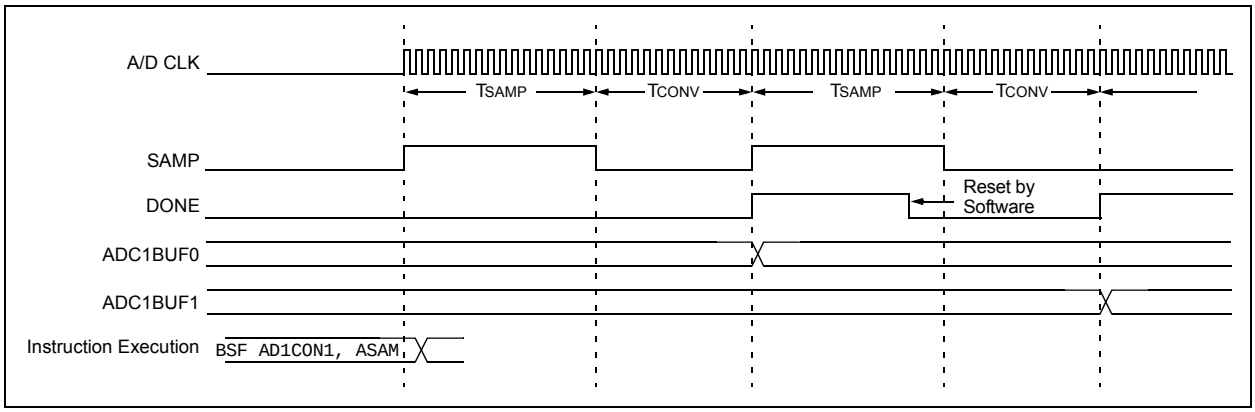
bit 7-0 **CSS<7:0>:** A/D Input Scan Selection bits
1 = Includes corresponding channel for input scan
0 = Skips channel for input scan

22.5.2.1 Free-running Sample Conversion Sequence

Using the Auto-Convert Conversion Trigger mode (SSRC<3:0> = 0111), in combination with the Auto-Sample Start mode (ASAM = 1), allows the A/D module to schedule sample/conversion sequences with no intervention by the user or other device resources. This "Clocked" mode, shown in Figure 22-7, allows continuous data collection after module initialization.

Note that all timing in this mode scales with TAD, either from the A/D internal RC clock or from TCY (as prescaled by the ADCS<7:0> bits). In both cases, the SAMC<4:0> bits set the number of TAD clocks in TSAMP. TCONV is fixed at 12 TAD.

FIGURE 22-7: CONVERTING ONE CHANNEL, AUTO-SAMPLE START, TAD-BASED CONVERSION START



22.5.2.2 Sample Time Considerations Using Clocked Conversion Trigger And Automatic Sampling

The user must ensure the sampling time satisfies the sampling requirements, as outlined in **Section 22.9 "A/D Sampling Requirements"**. Assuming that the module is set for automatic sampling and using a clocked conversion trigger, the sampling interval is specified by the SAMCx bits.

22.5.3 EVENT TRIGGER CONVERSION START

It is often desirable to synchronize the end of sampling and the start of conversion with some other time event. Depending on the device family, the A/D module has up to 16 sources available to use as a conversion trigger event. The event trigger is selected by the SSRC<3:0> bits (ADCON1L<7:4>).

As noted, the available event triggers vary between device families. Refer to the specific device data sheet for specific information. The examples that follow represent trigger sources that are implemented in most devices. Note that the SSRCx bit assignments may vary in some devices.

22.5.3.1 External Int0 Pin Trigger

When SSRC<3:0> = 0001, the A/D conversion is triggered by an active transition on the INT0 pin. The pin may be programmed for either a rising edge input or a falling edge input.

22.5.3.2 Special Event Trigger

When SSRC<3:0> = 0010, the A/D is triggered by a Special Event Trigger. Refer to CCP and ECCP section for more information about Special Event Triggers.

22.5.3.3 Synchronizing A/D Operations To Internal Or External Events

The modes where an external event trigger pulse ends sampling and starts conversion may be used in combination with auto-sampling (ASAM = 1) to cause the A/D to synchronize the sample conversion events to the trigger pulse source. For example, in Figure 22-9, where SSRC<3:0> = 0010 and ASAM = 1, the A/D will always end sampling and start conversions synchronously with the timer compare trigger event. The A/D will have a sample conversion rate that corresponds to the timer comparison event rate.

26.5 Measuring Capacitance with the CTMU

There are two ways to measure capacitance with the CTMU. The absolute method measures the actual capacitance value. The relative method only measures for any change in the capacitance.

26.5.1 ABSOLUTE CAPACITANCE MEASUREMENT

For absolute capacitance measurements, both the current and capacitance calibration steps found in **Section 26.4 "Calibrating the CTMU Module"** should be followed.

To perform these measurements:

1. Initialize the A/D Converter.
2. Initialize the CTMU.
3. Set EDG1STAT.
4. Wait for a fixed delay, T.
5. Clear EDG1STAT.
6. Perform an A/D conversion.
7. Calculate the total capacitance, $C_{TOTAL} = (I * T)/V$, where:
   - I is known from the current source measurement step (**Section 26.4.1 "Current Source Calibration"**)
   - T is a fixed delay
   - V is measured by performing an A/D conversion
8. Subtract the stray and A/D capacitance (C_{OFFSET} from **Section 26.4.2 "Capacitance Calibration"**) from C_{TOTAL} to determine the measured capacitance.

26.5.2 CAPACITIVE TOUCH SENSE USING RELATIVE CHARGE MEASUREMENT

Not all applications require precise capacitance measurements. When detecting a valid press of a capacitance-based switch, only a relative change of capacitance needs to be detected.

In such an application when the switch is open (or not touched), the total capacitance is the capacitance of the combination of the board traces, the A/D Converter and other elements. A larger voltage will be measured by the A/D Converter. When the switch is closed (or touched), the total capacitance is larger due to the addition of the capacitance of the human body to the above listed capacitances and a smaller voltage will be measured by the A/D Converter.

To detect capacitance changes simply:

1. Initialize the A/D Converter and the CTMU.
2. Set EDG1STAT.
3. Wait for a fixed delay.
4. Clear EDG1STAT.
5. Perform an A/D conversion.

The voltage measured by performing the A/D conversion is an indication of the relative capacitance. In this case, no calibration of the current source or circuit capacitance measurement is needed. (For a sample software routine for a capacitive touch switch, see Example 26-4.)

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 2-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.

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 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 register 'f'

Example: BSF FLAG_REG, 7, 1

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

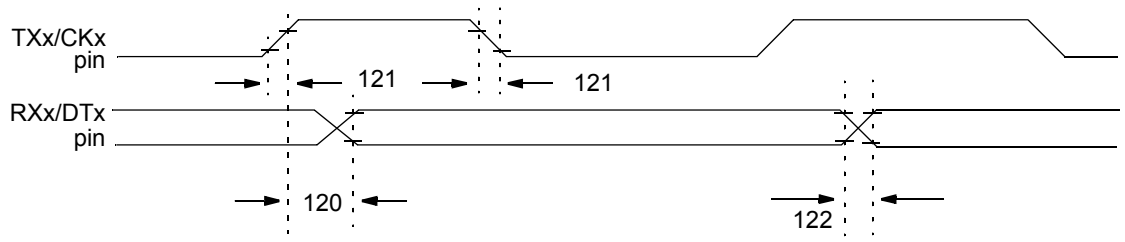
TABLE 30-37: MSSPx I²C BUS DATA REQUIREMENTS

Param. No.	Symbol	Characteristic		Min.	Max.	Units	Conditions
92	TSU:STO	Stop Condition Setup Time	100 kHz mode	2(TOSC)(BRG + 1)	—	—	
			400 kHz mode	2(TOSC)(BRG + 1)	—	—	
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—	—	
109	TAA	Output Valid from Clock	100 kHz mode	—	3500	ns	
			400 kHz mode	—	1000	ns	
			1 MHz mode(1)	—	—	ns	
110	TBUF	Bus Free Time	100 kHz mode	4.7	—	μs	Time the bus must be free before a new transmission can start
			400 kHz mode	1.3	—	μs	
			1 MHz mode(1)	—	—	μs	
D102	CB	Bus Capacitive Loading		—	400	pF	

Note 1: Maximum pin capacitance = 10 pF for all II²C pins.

2: A Fast mode I²C bus device can be used in a Standard mode I²C bus system, but Parameter #107 ≥ 250 ns must then be met. This will automatically be the case if the device does not stretch the LOW period of the SCLx signal. If such a device does stretch the LOW period of the SCLx signal, it must output the next data bit to the SDAx line, Parameter #102 + Parameter #107 = 1000 + 250 = 1250 ns (for 100 kHz mode), before the SCLx line is released.

FIGURE 30-20: EUSARTx SYNCHRONOUS TRANSMISSION (MASTER/SLAVE) TIMING



Note: Refer to Figure 30-2 for load conditions.

TABLE 30-38: EUSARTx/AUSARTx SYNCHRONOUS TRANSMISSION REQUIREMENTS

Param. No.	Symbol	Characteristic	Min.	Max.	Units	Conditions
120	TCKH2DTV	SYNC XMIT (MASTER and SLAVE) Clock High to Data Out Valid	—	40	ns	
121	TCKRF	Clock Out Rise Time and Fall Time (Master mode)	—	20	ns	
122	TDTRF	Data Out Rise Time and Fall Time	—	20	ns	