Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

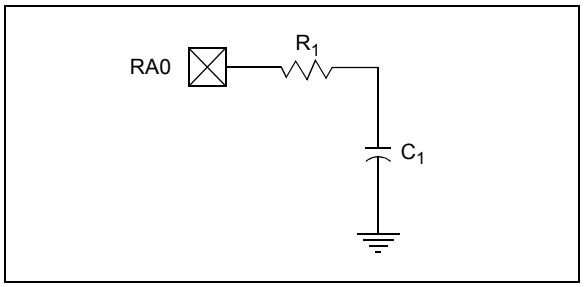
"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" 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	48MHz
Connectivity	I²C, LINbus, SPI, UART/USART, USB
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	22
Program Memory Size	64KB (32K x 16)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	3.8K x 8
Voltage - Supply (Vcc/Vdd)	2V ~ 2.75V
Data Converters	A/D 10x10b/12b
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/pic18lf26j53t-i-ss

A series resistor between RA0 and the external capacitor provides overcurrent protection for the RA0/AN0/C1INA/ULPWU/RP0 pin and can allow for software calibration of the time-out (see Figure 4-9).

FIGURE 4-9: SERIAL RESISTOR



A timer can be used to measure the charge time and discharge time of the capacitor. The charge time can then be adjusted to provide the desired interrupt delay. This technique will compensate for the affects of temperature, voltage and component accuracy. The ULPWU peripheral can also be configured as a simple Programmable Low-Voltage Detect (LVD) or temperature sensor.

Note:	For more information, refer to *AN879, Using the Microchip Ultra Low-Power Wake-up Module* application note (DS00879).

4.8 Peripheral Module Disable

All peripheral modules (except for I/O ports) also have a second control bit that can disable their functionality. These bits, known as the Peripheral Module Disable (PMDISx) bits, are generically named “xxxMD” (using “xxx” as the mnemonic version of the module’s name).

These bits are located in the PMDISx Special Function Registers. In contrast to the module enable bits (generically named “xxxEN” and located in bit position seven of the control registers), the PMDISx bits must be set (= 1) to disable the modules.

While the PMD and module enable bits both disable a peripheral’s functionality, the PMD bit completely shuts down the peripheral, effectively powering down all circuits and removing all clock sources. This has the additional effect of making any of the module’s control and buffer registers, mapped in the SFR space, unavailable for operations. Essentially, the peripheral ceases to exist until the PMD bit is cleared.

This differs from using the module enable bit, which allows the peripheral to be reconfigured and buffer registers preloaded, even when the peripheral’s operations are disabled.

The PMDISx bits are most useful in highly power-sensitive applications. In these cases, the bits can be set before the main body of the application to remove peripherals that will not be needed at all.

TABLE 4-2: LOW-POWER MODE REGISTERS

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR
PMDIS3	CCP10MD	CCP9MD	CCP8MD	CCP7MD	CCP6MD	CCP5MD	CCP4MD	—	0000 000-
PMDIS2	—	TMR8MD	—	TMR6MD	TMR5MD	CMP3MD	CMP2MD	CMP1MD	-0-0 0000
PMDIS1	PSPMD(1)	CTMUMD	RTCCMD	TMR4MD	TMR3MD	TMR2MD	TMR1MD	—	0000 000-
PMDIS0	ECCP3MD	ECCP2MD	ECCP1MD	UART2MD	UART1MD	SPI2MD	SPI1MD	ADCMD	0000 0000

Note 1: Not implemented on 28-pin devices (PIC18F26J53, PIC18F27J53, PIC18LF26J53 and PIC18LF27J53).

9.4 IPR Registers

The IPR registers contain the individual priority bits for the peripheral interrupts. Due to the number of peripheral interrupt sources, there are three Peripheral Interrupt Priority registers (IPR1, IPR2, IPR3). Using the priority bits requires that the Interrupt Priority Enable (IPEN) bit be set.

REGISTER 9-14: IPR1: PERIPHERAL INTERRUPT PRIORITY REGISTER 1 (ACCESS F9Fh)

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
PMPIP(1)	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP
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 **PMPIP:** Parallel Master Port Read/Write Interrupt Priority bit(1)
1 = High priority
0 = Low priority

bit 6 **ADIP:** A/D Converter Interrupt Priority bit
1 = High priority
0 = Low priority

bit 5 **RC1IP:** EUSART1 Receive Interrupt Priority bit
1 = High priority
0 = Low priority

bit 4 **TX1IP:** EUSART1 Transmit Interrupt Priority bit
1 = High priority
0 = Low priority

bit 3 **SSP1IP:** Master Synchronous Serial Port Interrupt Priority bit (MSSP1 module)
1 = High priority
0 = Low priority

bit 2 **CCP1IP:** ECCP1 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 1 **TMR2IP:** TMR2 to PR2 Match Interrupt Priority bit
1 = High priority
0 = Low priority

bit 0 **TMR1IP:** TMR1 Overflow Interrupt Priority bit
1 = High priority
0 = Low priority

Note 1: These bits are unimplemented on 28-pin devices.

TABLE 10-3: PORTA I/O SUMMARY (CONTINUED)

Pin	Function	TRIS Setting	I/O	I/O Type	Description
RA5/AN4/C1INC/ $\overline{SS1}$/HLVDIN/ RCV/RP2	RA5	0	O	DIG	LATA<5> data output; not affected by analog input.
		1	I	TTL	PORTA<5> data input; disabled when analog input is enabled.
	AN4	1	I	ANA	A/D Input Channel 4. Default configuration on POR.
	C1INC	0	O	DIG	Comparator 1 Input C.
	$\overline{SS1}$	1	I	TTL	Slave select input for MSSP1.
	HLVDIN	1	I	ANA	High/Low-Voltage Detect external trip point reference input.
	RCV	1	I	TTL	External USB transceiver RCV input.
	RP2	1	I	ST	Remappable Peripheral Pin 2 input.
		0	O	DIG	Remappable Peripheral Pin 2 output.
OSC2/CLKO/ RA6	OSC2	x	O	ANA	Main oscillator feedback output connection (HS mode).
	CLKO	x	O	DIG	System cycle clock output (FOSC/4) in RC and EC Oscillator modes.
	RA6	1	I	TTL	PORTA<6> data input.
		0	O	DIG	LATA<6> data output.
OSC1/CLKI/RA7	OSC1	1	I	ANA	Main oscillator input connection.
	CLKI	1	I	ANA	Main clock input connection.
	RA7	1	I	TTL	PORTA<6> data input.
		0	O	DIG	LATA<6> data output.

Legend: DIG = Digital level output; TTL = TTL input buffer; ST = Schmitt Trigger input buffer; ANA = Analog level input/output; x = Don't care (TRIS bit does not affect port direction or is overridden for this option)

Note 1: This bit is only available on 44-pin devices (PIC18F46J53, PIC18F47J53, PIC18LF46J53 and PIC18LF47J53).

TABLE 10-4: SUMMARY OF REGISTERS ASSOCIATED WITH PORTA

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PORTA	RA7	RA6	RA5	—	RA3	RA2	RA1	RA0
LATA	LAT7	LAT6	LAT5	—	LAT3	LAT2	LAT1	LAT0
TRISA	TRIS7	TRIS6	TRISA5	—	TRISA3	TRISA2	TRISA1	TRISA0
ANCON0	PCFG7(1)	PCFG6(1)	PCFG5(1)	PCFG4	PCFG3	PCFG2	PCFG1	PCFG0
CMxCON	CON	COE	CPOL	EVPOL1	EVPOL0	CREF	CCH1	CCH0
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0
WDTCON	REGSLP	LVDSTAT	ULPLVL	VBGOE	DS	ULPEN	ULPSINK	SWDTEN
HLVDCON	VDIRMAG	BGVST	IRVST	HLVDEN	HLVDL3	HLVDL2	HLVDL1	HLVDL0

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

Note 1: These bits are only available in 44-pin devices (PIC18F46J53, PIC18F47J53, PIC18LF46J53 and PIC18LF47J53).

11.2.2 WRITE TO SLAVE PORT

When chip select is active and a write strobe occurs (PMCSx = 1 and PMWR = 1), the data from PMD<7:0> is captured into the lower PMDIN1L register. The PMPIF and IBF flag bits are set when the write ends.The timing for the control signals in Write mode is displayed in Figure 11-3. The polarity of the control signals are configurable.

11.2.3 READ FROM SLAVE PORT

When chip select is active and a read strobe occurs (PMCSx = 1 and PMRD = 1), the data from the PMDOUT1L register (PMDOUT1L<7:0>) is presented on to PMD<7:0>. Figure 11-4 provides the timing for the control signals in Read mode.

FIGURE 11-3: PARALLEL SLAVE PORT WRITE WAVEFORMS

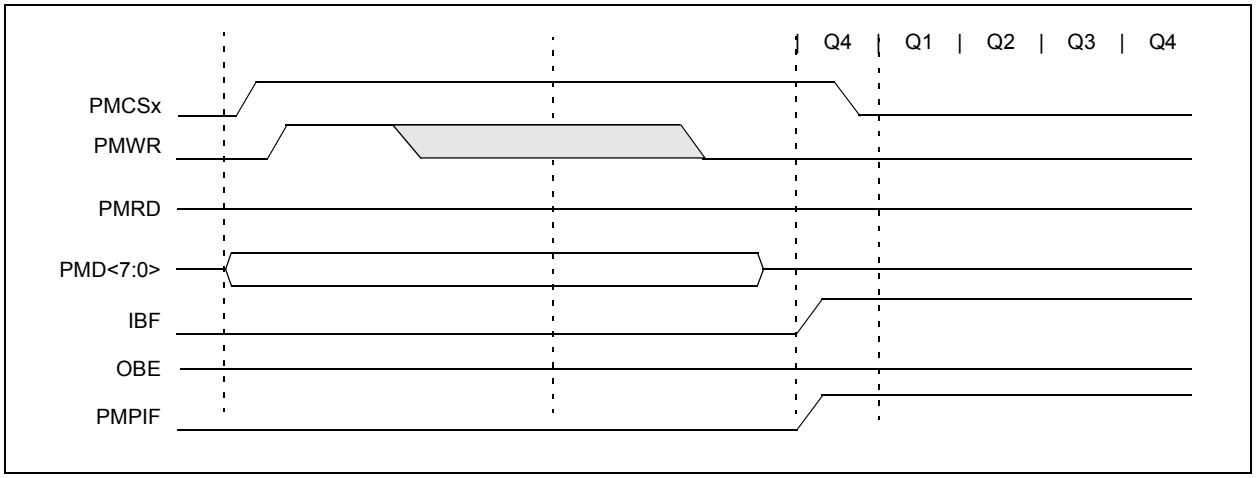
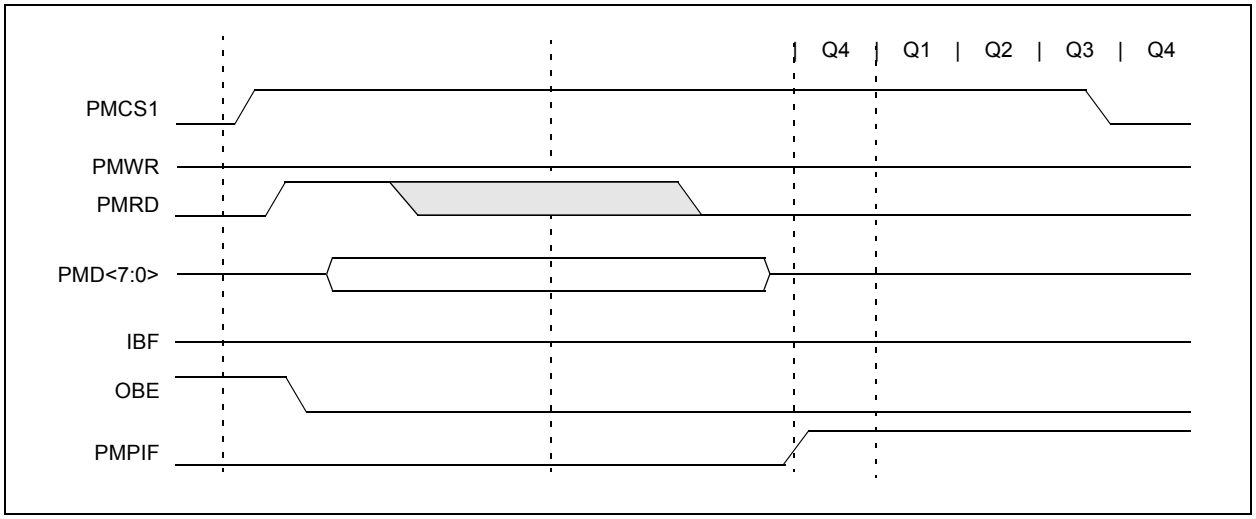


FIGURE 11-4: PARALLEL SLAVE PORT READ WAVEFORMS



12.3 Prescaler

An 8-bit counter is available as a prescaler for the Timer0 module. The prescaler is not directly readable or writable. Its value is set by the PSA and T0PS<2:0> bits (T0CON<3:0>), which determine the prescaler assignment and prescale ratio.

Clearing the PSA bit assigns the prescaler to the Timer0 module. When it is assigned, prescale values from 1:2 through 1:256, in power-of-2 increments, are selectable.

When assigned to the Timer0 module, all instructions writing to the TMR0 register (e.g., `CLRF TMR0`, `MOVWF TMR0`, `BSF TMR0`, etc.) clear the prescaler count.

Note:	Writing to TMR0 when the prescaler is assigned to Timer0 will clear the prescaler count but will not change the prescaler assignment.

12.3.1 SWITCHING PRESCALER ASSIGNMENT

The prescaler assignment is fully under software control and can be changed “on-the-fly” during program execution.

12.4 Timer0 Interrupt

The TMR0 interrupt is generated when the TMR0 register overflows from FFh to 00h in 8-bit mode, or from FFFFh to 0000h in 16-bit mode. This overflow sets the TMR0IF flag bit. The interrupt can be masked by clearing the TMR0IE bit (INTCON<5>). Before re-enabling the interrupt, the TMR0IF bit must be cleared in software by the Interrupt Service Routine (ISR).

Since Timer0 is shut down in Sleep mode, the TMR0 interrupt cannot awaken the processor from Sleep.

TABLE 12-1: REGISTERS ASSOCIATED WITH TIMER0

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TMR0L	Timer0 Register Low Byte							
TMR0H	Timer0 Register High Byte							
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
T0CON	TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0

Legend: Shaded cells are not used by Timer0.

FIGURE 13-7: TIMER1 GATE SINGLE PULSE AND TOGGLE COMBINED MODE

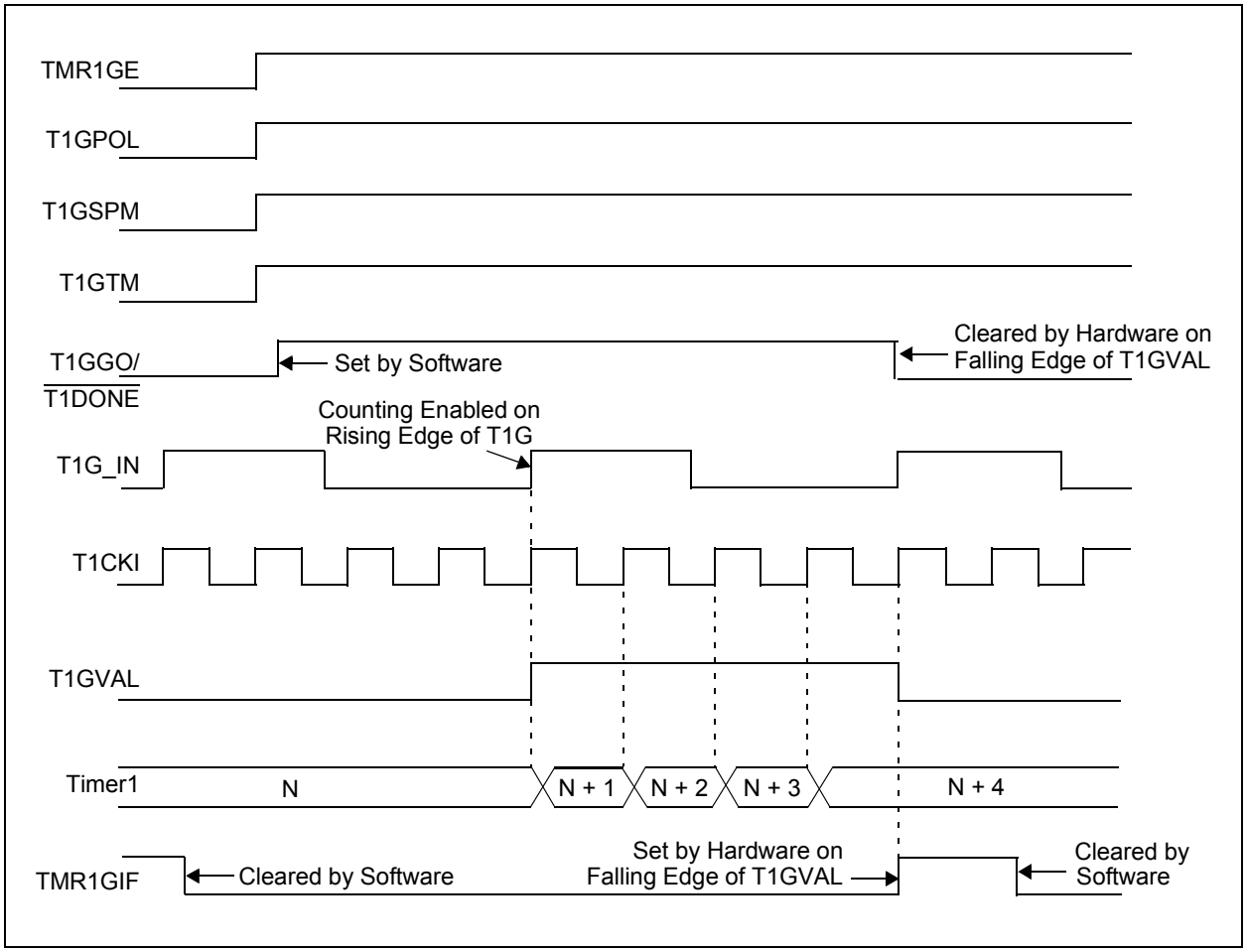


TABLE 13-5: REGISTERS ASSOCIATED WITH TIMER1 AS A TIMER/COUNTER

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
PIR1	PMPIF(1)	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF
PIE1	PMPIE(1)	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE
IPR1	PMPIP(1)	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP
TMR1L	Timer1 Register Low Byte							
TMR1H	Timer1 Register High Byte							
T1CON	TMR1CS1	TMR1CS0	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	RD16	TMR1ON
T1GCON	TMR1GE	T1GPOL	T1GTM	T1GSPM	$\overline{\text{T1GGO}}$/$\overline{\text{T1DONE}}$	T1GVAL	T1GSS1	T1GSS0
OSCCON2	—	SOSCRUN	—	SOSCDRV	SOSCGO	PRISD	—	—

Legend: — = unimplemented, read as '0'. Shaded cells are not used by the Timer1 module.

Note 1: These bits are only available in 44-pin devices.

REGISTER 18-3: CCPTMRS2: CCP4-10 TIMER SELECT 2 REGISTER (BANKED F50h)

U-0	U-0	U-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0
—	—	—	C10TSEL0	—	C9TSEL0	C8TSEL1	C8TSEL0
bit 7							bit 0

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

bit 7-5 **Unimplemented:** Read as '0'

bit 4 **C10TSEL0:** CCP10 Timer Selection bit
0 = CCP10 is based off of TMR1/TMR2
1 = Reserved; do not use

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

bit 2 **C9TSEL0:** CCP9 Timer Selection bit
0 = CCP9 is based off of TMR1/TMR2
1 = CCP9 is based off of TMR1/TMR4

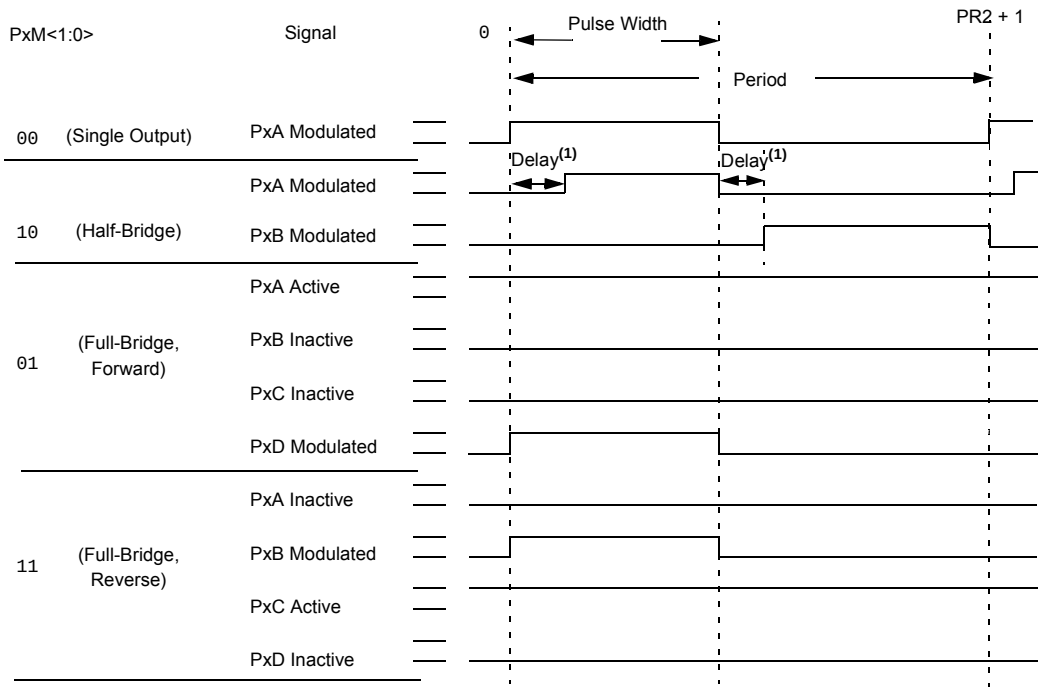
bit 1-0 **C8TSEL<1:0>:** CCP8 Timer Selection bits
00 = CCP8 is based off of TMR1/TMR2
01 = CCP8 is based off of TMR1/TMR4
10 = CCP8 is based off of TMR1/TMR6
11 = Reserved; do not use

TABLE 19-3: EXAMPLE PIN ASSIGNMENTS FOR VARIOUS PWM ENHANCED MODES
TABLE 19-3:

ECCP Mode	PxM<1:0>	PxA	PxB	PxC	PxD
Single	00	Yes(1)	Yes(1)	Yes(1)	Yes(1)
Half-Bridge	10	Yes	Yes	No	No
Full-Bridge, Forward	01	Yes	Yes	Yes	Yes
Full-Bridge, Reverse	11	Yes	Yes	Yes	Yes

Note 1: Outputs are enabled by pulse steering in Single mode (see Register 19-6).

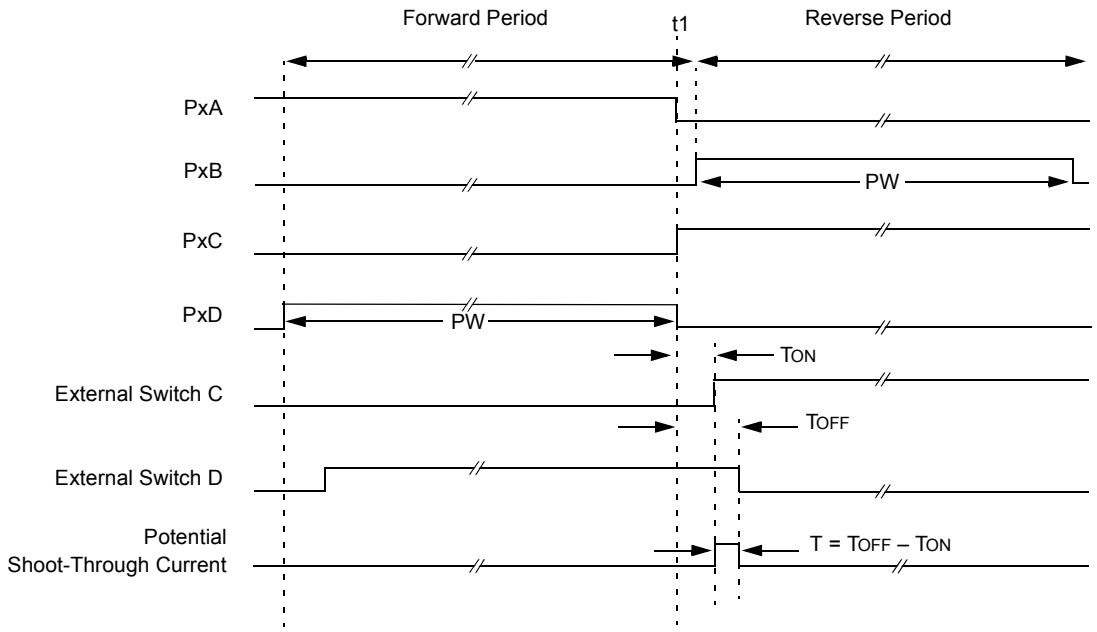
FIGURE 19-4: PWM (ENHANCED MODE) OUTPUT RELATIONSHIPS (ACTIVE-HIGH STATE) EXAMPLE



Relationships:
- Period = 4 * TOSC * (PR2 + 1) * (TMR2 Prescale Value)
- Pulse Width = TOSC * (CCPRxL<7:0>:CCPxCON<5:4>) * (TMR2 Prescale Value)
- Delay = 4 * TOSC * (ECCPxDEL<6:0>)

Note 1: Dead-band delay is programmed using the ECCPxDEL register (see **Section 19.4.6 "Programmable Dead-Band Delay Mode"**).

FIGURE 19-11: EXAMPLE OF PWM DIRECTION CHANGE AT NEAR 100% DUTY CYCLE



Note 1: All signals are shown as active-high.

2: TON is the turn-on delay of power switch, QC, and its driver.

3: TOFF is the turn-off delay of power switch, QD, and its driver.

19.4.3 START-UP CONSIDERATIONS

When any PWM mode is used, the application hardware must use the proper external pull-up and/or pull-down resistors on the PWM output pins.

Note:	When the microcontroller is released from Reset, all of the I/O pins are in the high-impedance state. The external circuits must keep the power switch devices in the OFF state until the microcontroller drives the I/O pins with the proper signal levels or activates the PWM output(s).

The CCPxM<1:0> bits of the CCPxCON register allow the user to choose whether the PWM output signals are active-high or active-low for each pair of PWM output pins (PxA/PxC and PxB/PxD). The PWM output polarities must be selected before the PWM pin output drivers are enabled. Changing the polarity configuration while the PWM pin output drivers are enabled is not recommended, since it may result in damage to the application circuits.

The PxA, PxB, PxC and PxD output latches may not be in the proper states when the PWM module is initialized. Enabling the PWM pin output drivers at the same time as the Enhanced PWM modes may cause damage to the application circuit. The Enhanced PWM modes must be enabled in the proper Output mode and complete a full PWM cycle before enabling the PWM pin output drivers. The completion of a full PWM cycle is indicated by the TMR2IF or TMR4IF bit of the PIR1 or PIR3 register being set as the second PWM period begins.

19.4.4 ENHANCED PWM AUTO-SHUTDOWN MODE

The PWM mode supports an Auto-Shutdown mode that will disable the PWM outputs when an external shutdown event occurs. Auto-Shutdown mode places the PWM output pins into a predetermined state. This mode is used to help prevent the PWM from damaging the application.

The auto-shutdown sources are selected using the ECCPxAS<2:0> bits (ECCPxAS<6:4>). A shutdown event may be generated by:

- A logic '0' on the pin that is assigned to the FLT0 input function
- Comparator C1
- Comparator C2
- Setting the ECCPxASE bit in firmware

A shutdown condition is indicated by the ECCPxASE (Auto-Shutdown Event Status) bit (ECCPxAS<7>). If the bit is a '0', the PWM pins are operating normally. If the bit is a '1', the PWM outputs are in the shutdown state.

REGISTER 19-6: PSTRxCON: PULSE STEERING CONTROL (1, ACCESS FBFh; 2, FB9h; 3, BANKED F1Ah)(1)

R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-1
CMPL1	CMPL0	—	STRSYNC	STRD	STRC	STRB	STRA
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 **CMPL<1:0>:** Complementary Mode Output Assignment Steering Sync bits
1 = Modulated output pin toggles between PxA and PxB for each period
0 = Complementary output assignment disabled; the STRD:STRA bits are used to determine Steering mode

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

bit 4 **STRSYNC:** Steering Sync bit
1 = Output steering update occurs on next PWM period
0 = Output steering update occurs at the beginning of the instruction cycle boundary

bit 3 **STRD:** Steering Enable D bit
1 = PxD pin has the PWM waveform with polarity control from CCPxM<1:0>
0 = PxD pin is assigned to port pin

bit 2 **STRC:** Steering Enable C bit
1 = PxC pin has the PWM waveform with polarity control from CCPxM<1:0>
0 = PxC pin is assigned to port pin

bit 1 **STRB:** Steering Enable B bit
1 = PxB pin has the PWM waveform with polarity control from CCPxM<1:0>
0 = PxB pin is assigned to port pin

bit 0 **STRA:** Steering Enable A bit
1 = PxA pin has the PWM waveform with polarity control from CCPxM<1:0>
0 = PxA pin is assigned to port pin

Note 1: The PWM Steering mode is available only when the CCPxCON register bits, CCPxM<3:2> = 11 and PxM<1:0> = 00.

20.4.6 USING THE SPI DMA MODULE

The following steps would typically be taken to enable and use the SPI DMA module:

1. Configure the I/O pins, which will be used by MSSP2.
   a) Assign SCK2, SDO2, SDI2 and $\overline{SS2}$ to the RPn pins as appropriate for the SPI mode which will be used. Only functions which will be used need to be assigned to a pin.
   b) Initialize the associated LATx registers for the desired Idle SPI bus state.
   c) If Open-Drain Output mode on SDO2 and SCK2 (Master mode) is desired, set ODCON3<1>.
   d) Configure corresponding TRISx bits for each I/O pin used.
2. Configure and enable MSSP2 for the desired SPI operating mode.
   a) Select the desired operating mode (Master or Slave, SPI Mode 0, 1, 2 and 3) and configure the module by writing to the SSP2STAT and SSP2CON1 registers.
   b) Enable MSSP2 by setting SSP2CON1<5> = 1.
3. Configure the SPI DMA engine.
   a) Select the desired operating mode by writing the appropriate values to DMACON2 and DMACON1.
   b) Initialize the TXADDRH/TXADDRL Pointer (Full-Duplex or Half-Duplex Transmit Only mode).
   c) Initialize the RXADDRH/RXADDRL Pointer (Full-Duplex or Half-Duplex Receive Only mode).
   d) Initialize the DMABCH/DMABCL Byte Count register with the number of bytes to be transferred in the next SPI DMA operation.
   e) Set the DMAEN bit (DMACON1<0>).

   In SPI Master modes, this will initiate a DMA transaction. In SPI Slave modes, this will complete the initialization process, and the module will now be ready to begin receiving and/or transmitting data to the master device once the master starts the transaction.
4. Detect the SSP2IF interrupt condition (PIR3<7).
   a) If the interrupt was configured to occur at the completion of the SPI DMA transaction, the DMAEN bit (DMACON1<0>) will be clear. User firmware may prepare the module for another transaction by repeating steps 3.b through 3.e.
   b) If the interrupt was configured to occur prior to the completion of the SPI DMA transaction, the DMAEN bit may still be set, indicating the transaction is still in progress. User firmware would typically use this interrupt condition to begin preparing new data for the next DMA transaction. Firmware should not repeat steps 3.b. through 3.e. until the DMAEN bit is cleared by the hardware, indicating the transaction is complete.

Example 20-2 provides example code demonstrating the initialization process and the steps needed to use the SPI DMA module to perform a 512-byte Full-Duplex Master mode transfer.

20.5.3.4 7-Bit Address Masking Mode

Unlike 5-Bit Address Masking mode, 7-Bit Address Masking mode uses a mask of up to eight bits (in 10-bit addressing) to define a range of addresses than can be Acknowledged, using the lowest bits of the incoming address. This allows the module to Acknowledge up to 127 different addresses with 7-bit addressing, or 255 with 10-bit addressing (see Example 20-4). This mode is the default configuration of the module and is selected when MSSPMSK is unprogrammed ('1').

The address mask for 7-Bit Address Masking mode is stored in the SSPxMSK register, instead of the SSPxCON2 register. SSPxMSK is a separate, hardware register within the module, but it is not directly addressable. Instead, it shares an address in the SFR space with the SSPxADD register. To access the SSPxMSK register, it is necessary to select MSSP mode, '1001' (SSPCON1<3:0> = 1001), and then read or write to the location of SSPxADD.

To use 7-Bit Address Masking mode, it is necessary to initialize SSPxMSK with a value before selecting the I^2C Slave Addressing mode. Thus, the required sequence of events is:

1. Select SSPxMSK Access mode (SSPxCON2<3:0> = 1001).
2. Write the mask value to the appropriate SSPxADD register address (FC8h for MSSP1, F6Eh for MSSP2).
3. Set the appropriate I^2C Slave mode (SSPxCON2<3:0> = 0111 for 10-bit addressing, 0110 for 7-bit addressing).

Setting or clearing mask bits in SSPxMSK behaves in the opposite manner of the ADMSK bits in 5-Bit Address Masking mode. That is, clearing a bit in SSPxMSK causes the corresponding address bit to be masked; setting the bit requires a match in that position. SSPxMSK resets to all '1's upon any Reset condition, and therefore, has no effect on the standard MSSP operation until written with a mask value.

With 7-Bit Address Masking mode, the SSPxMSK<7:1> bits mask the corresponding address bits in the SSPxADD register. For any SSPxMSK bits that are active (SSPxMSK<n> = 0), the corresponding SSPxADD address bit is ignored (SSPxADD<n> = x). For the module to issue an address Acknowledge, it is sufficient to match only on addresses that do not have an active address mask.

With 10-Bit Address Masking mode, SSPxMSK<7:0> bits mask the corresponding address bits in the SSPxADD register. For any SSPxMSK bits that are active (= 0), the corresponding SSPxADD address bit is ignored (SSPxADD<n> = x).

Note:	The two MSbs of the address are not affected by address masking.

EXAMPLE 20-4: ADDRESS MASKING EXAMPLES IN 7-BIT MASKING MODE

7-Bit Addressing:

SSPxADD<7:1>= 1010 000

SSPxMSK<7:1>= 1111 001

Addresses Acknowledged = A8h, A6h, A4h, A0h

10-Bit Addressing:

SSPxADD<7:0> = 1010 0000 (the two MSbs are ignored in this example since they are not affected)

SSPxMSK<5:1> = 1111 0

Addresses Acknowledged = A8h, A6h, A4h, A0h

FIGURE 20-10: I²C SLAVE MODE TIMING (TRANSMISSION, 7-BIT ADDRESS)

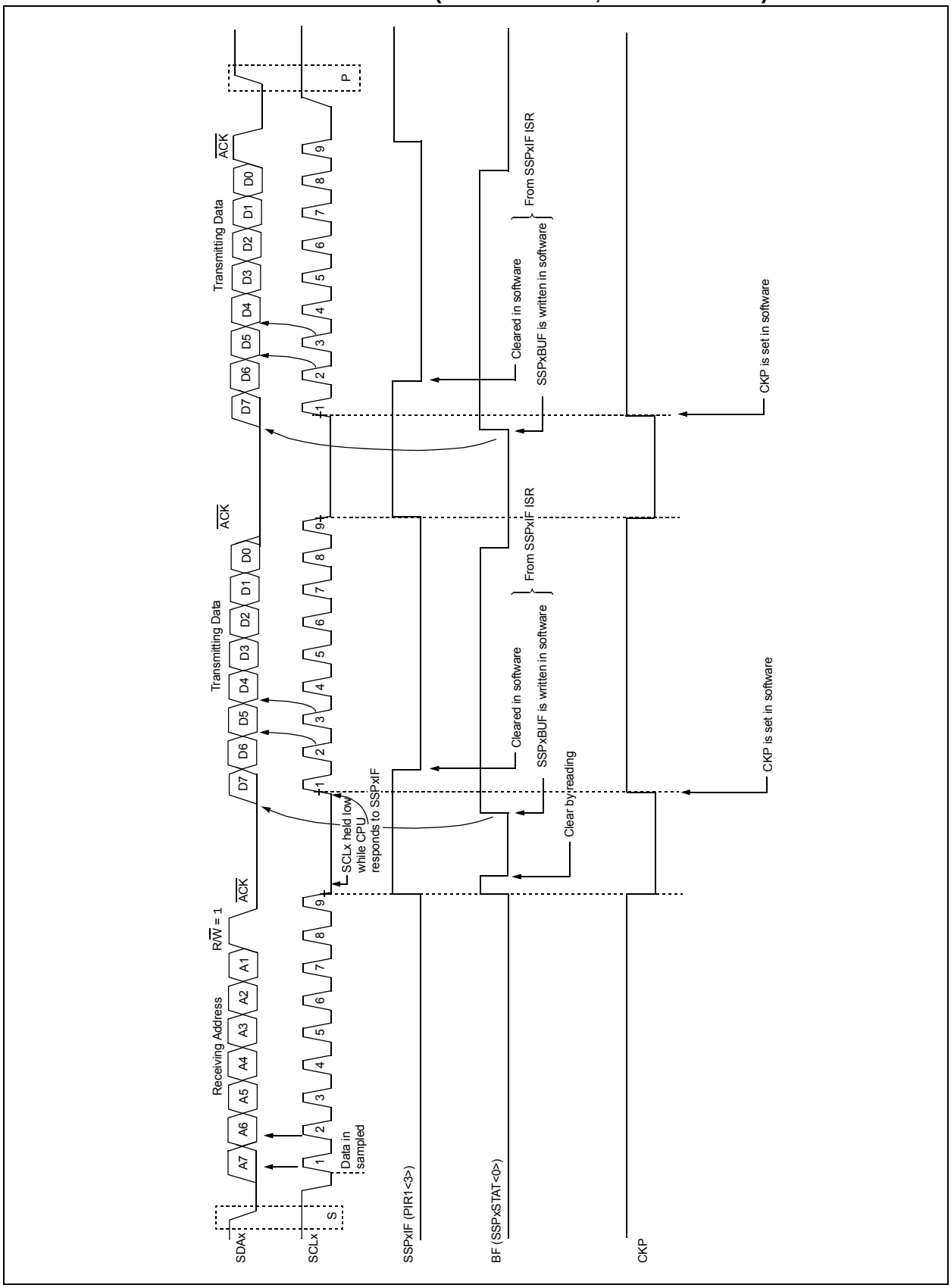


TABLE 20-4: REGISTERS ASSOCIATED WITH I²C OPERATION

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
PIR1	PMPIF(3)	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF
PIE1	PMPIE(3)	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE
IPR1	PMPIP(3)	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP
PIR2	OSCFIF	CM2IF	CM1IF	USBIF	BCL1IF	HLVDIF	TMR3IF	CCP2IF
PIE2	OSCFIE	CM2IE	CM1IE	USBIE	BCL1IE	HLVDIE	TMR3IE	CCP2IE
IPR2	OSCFIP	CM2IP	CM1IP	USBIP	BCL1IP	HLVDIP	TMR3IP	CCP2IP
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CTMUIF	TMR3GIF	RTCCIF
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CTMUIE	TMR3GIE	RTCCIE
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CTMUIP	TMR3GIP	RTCCIP
TRISC	TRISC7	TRISC6	—	—	—	TRISC2	TRISC1	TRISC0
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0
SSP1BUF	MSSP1 Receive Buffer/Transmit Register							
SSP1ADD	MSSP1 Address Register (I²C Slave mode), MSSP1 Baud Rate Reload Register (I²C Master mode)							
SSPxMSK(1)	MSK7	MSK6	MSK5	MSK4	MSK3	MSK2	MSK1	MSK0
SSPxCON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0
SSPxCON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN
	GCEN	ACKSTAT	ADMSK5(2)	ADMSK4(2)	ADMSK3(2)	ADMSK2(2)	ADMSK1(2)	SEN
SSPxSTAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF
SSP2BUF	MSSP2 Receive Buffer/Transmit Register							
SSP2ADD	MSSP2 Address Register (I²C Slave mode), MSSP2 Baud Rate Reload Register (I²C Master mode)							

Legend: — = unimplemented, read as '0'. Shaded cells are not used by the MSSPx module in I²C mode.

Note 1: SSPxMSK shares the same address in SFR space as SSPxADD, but is only accessible in certain I²C Slave mode operations in 7-Bit Masking mode. See **Section 20.5.3.4 "7-Bit Address Masking Mode"** for more details.

2: Alternate bit definitions for use in I²C Slave mode operations only.

3: These bits are only available on 44-pin devices.

FIGURE 22-3: A/D CONVERSION TAD CYCLES (ACQT<2:0> = 000, TACQ = 0)

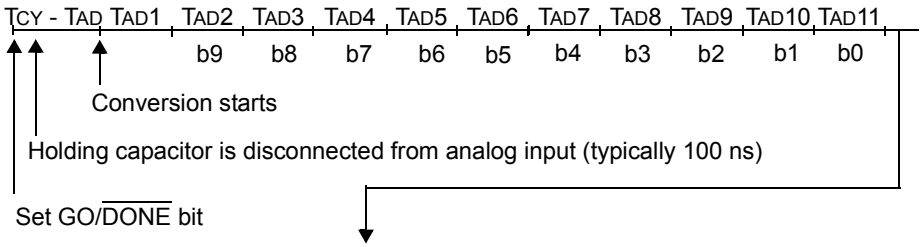
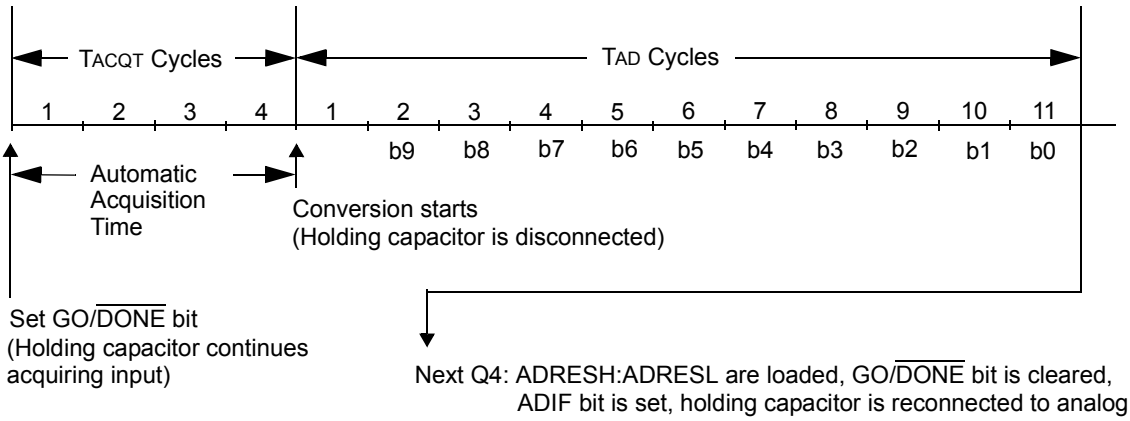


FIGURE 22-4: A/D CONVERSION TAD CYCLES (ACQT<2:0> = 010, TACQ = 4 TAD)



22.7 A/D Converter Calibration

The A/D Converter in the PIC18F47J53 family of devices includes a self-calibration feature, which compensates for any offset generated within the module. The calibration process is automated and is initiated by setting the ADCAL bit (ADCON1<6>). The next time the GO/DONE bit is set, the module will perform an offset calibration and store the result internally. Thus, subsequent offsets will be compensated.

Example 22-1 provides an example of a calibration routine.

The calibration process assumes that the device is in a relatively steady-state operating condition. If A/D calibration is used, it should be performed after each device Reset or if there are other major changes in operating conditions.

23.5.3 USB ERROR INTERRUPT STATUS REGISTER (UEIR)

The USB Error Interrupt Status register (Register 23-9) contains the flag bits for each of the error sources within the USB peripheral. Each of these sources is controlled by a corresponding interrupt enable bit in the UEIE register. All of the USB error flags are ORed together to generate the USB Error Interrupt Flag (UERRIF) at the top level of the interrupt logic.

Each error bit is set as soon as the error condition is detected. Thus, the interrupt will typically not correspond with the end of a token being processed.

Once an interrupt bit has been set by the SIE, it must be cleared in software by writing a '0'.

REGISTER 23-9: UEIR: USB ERROR INTERRUPT STATUS REGISTER (ACCESS F63h)

R/C-0	U-0	U-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0
BTSEF	—	—	BTOEF	DFN8EF	CRC16EF	CRC5EF	PIDEF
bit 7							bit 0

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

bit 7 **BTSEF:** Bit Stuff Error Flag bit
1 = A bit stuff error has been detected
0 = No bit stuff error has been detected

bit 6-5 **Unimplemented:** Read as '0'

bit 4 **BTOEF:** Bus Turnaround Time-out Error Flag bit
1 = Bus turnaround time-out has occurred (more than 16 bit times of Idle from previous EOP elapsed)
0 = No bus turnaround time-out has occurred

bit 3 **DFN8EF:** Data Field Size Error Flag bit
1 = The data field was not an integral number of bytes
0 = The data field was an integral number of bytes

bit 2 **CRC16EF:** CRC16 Failure Flag bit
1 = The CRC16 failed
0 = The CRC16 passed

bit 1 **CRC5EF:** CRC5 Host Error Flag bit
1 = The token packet was rejected due to a CRC5 error
0 = The token packet was accepted

bit 0 **PIDEF:** PID Check Failure Flag bit
1 = PID check failed
0 = PID check passed

25.2 Voltage Reference Accuracy/Error

The full range of voltage reference cannot be realized due to the construction of the module. The transistors on the top and bottom of the resistor ladder network (see Figure 25-1) keep CVREF from approaching the reference source rails. The voltage reference is derived from the reference source; therefore, the CVREF output changes with fluctuations in that source. The tested absolute accuracy of the voltage reference can be found in **Section 31.0 "Electrical Characteristics"**.

25.3 Connection Considerations

The voltage reference module operates independently of the comparator module. The output of the reference generator may be connected to the RA2 pin if the CVROE bit is set. Enabling the voltage reference output onto RA2 when it is configured as a digital input will increase current consumption. Connecting RA2 as a digital output with CVRSS enabled will also increase current consumption.

The RA2 pin can be used as a simple D/A output with limited drive capability. Due to the limited current drive capability, a buffer must be used on the voltage reference output for external connections to VREF. See Figure 25-2 for an example buffering technique.

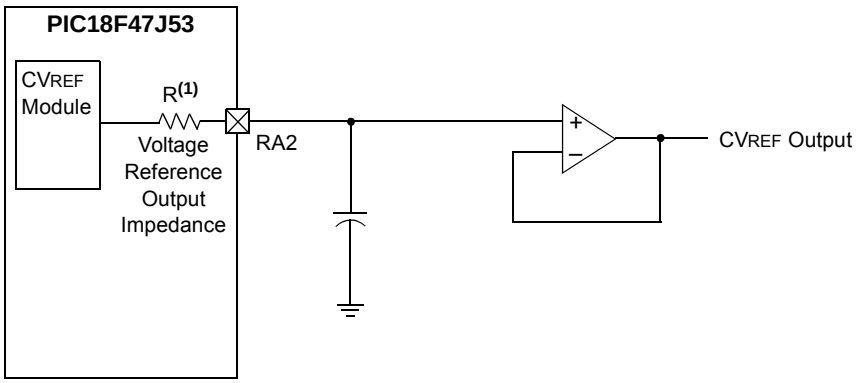
25.4 Operation During Sleep

When the device wakes up from Sleep through an interrupt or a Watchdog Timer time-out, the contents of the CVRCON register are not affected. To minimize current consumption in Sleep mode, the voltage reference should be disabled.

25.5 Effects of a Reset

A device Reset disables the voltage reference by clearing bit, CVREN (CVRCON<7>). This Reset also disconnects the reference from the RA2 pin by clearing bit, CVROE (CVRCON<6>) and selects the high-voltage range by clearing bit, CVRR (CVRCON<5>). The CVR value select bits are also cleared.

FIGURE 25-2: COMPARATOR VOLTAGE REFERENCE OUTPUT BUFFER EXAMPLE



Note 1: R is dependent upon the Comparator Voltage Reference Configuration bits, CVRCON<5> and CVRCON<3:0>.

TABLE 25-1: REGISTERS ASSOCIATED WITH COMPARATOR VOLTAGE REFERENCE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0
CM1CON	CON	COE	CPOL	EVPOL1	EVPOL0	CREF	CCH1	CCH0
CM2CON	CON	COE	CPOL	EVPOL1	EVPOL0	CREF	CCH1	CCH0
CM3CON	CON	COE	CPOL	EVPOL1	EVPOL0	CREF	CCH1	CCH0
TRISA	TRISA7	TRISA6	TRISA5	—	TRISA3	TRISA2	TRISA1	TRISA0
ANCON0	PCFG7(1)	PCFG6(1)	PCFG5(1)	PCFG4	PCFG3	PCFG2	PCFG1	PCFG0

Legend: — = unimplemented, read as '0', r = reserved. Shaded cells are not used with the comparator voltage reference.

Note 1: These bits are only available on 44-pin devices.

29.0 INSTRUCTION SET SUMMARY

The PIC18F47J53 family of devices incorporate the standard set of 75 PIC18 core instructions, as well as an extended set of 8 new instructions for the optimization of code that is recursive or that utilizes a software stack. The extended set is discussed later in this section.

29.1 Standard Instruction Set

The standard PIC18 MCU instruction set adds many enhancements to the previous PIC® MCU instruction sets, while maintaining an easy migration from these PIC MCU instruction sets. Most instructions are a single program memory word (16 bits), but there are four instructions that require two program memory locations.

Each single-word instruction is a 16-bit word divided into an opcode, which specifies the instruction type and one or more operands, which further specify the operation of the instruction.

The instruction set is highly orthogonal and is grouped into four basic categories:

- **Byte-oriented** operations
- **Bit-oriented** operations
- **Literal** operations
- **Control** operations

The PIC18 instruction set summary in Table 29-2 lists **byte-oriented**, **bit-oriented**, **literal** and **control** operations. Table 29-1 shows the opcode field descriptions.

Most **byte-oriented** instructions have three operands:

1. The file register (specified by ‘f’)
2. The destination of the result (specified by ‘d’)
3. The accessed memory (specified by ‘a’)

The file register designator, ‘f’, specifies which file register is to be used by the instruction. The destination designator, ‘d’, specifies where the result of the operation is to be placed. If ‘d’ is zero, the result is placed in the WREG register. If ‘d’ is one, the result is placed in the file register specified in the instruction.

All **bit-oriented** instructions have three operands:

1. The file register (specified by ‘f’)
2. The bit in the file register (specified by ‘b’)
3. The accessed memory (specified by ‘a’)

The bit field designator, ‘b’, selects the number of the bit affected by the operation, while the file register designator, ‘f’, represents the number of the file in which the bit is located.

The **literal** instructions may use some of the following operands:

- A literal value to be loaded into a file register (specified by ‘k’)
- The desired FSR register to load the literal value into (specified by ‘f’)
- No operand required (specified by ‘—’)

The **control** instructions may use some of the following operands:

- A program memory address (specified by ‘n’)
- The mode of the CALL or RETURN instructions (specified by ‘s’)
- The mode of the table read and table write instructions (specified by ‘m’)
- No operand required (specified by ‘—’)

All instructions are a single word, except for four double-word instructions. These instructions were made double-word to contain the required information in 32 bits. In the second word, the 4 MSbs are ‘1’s. If this second word is executed as an instruction (by itself), it will execute as a NOP.

All single-word instructions are executed in a single instruction cycle, unless a conditional test is true or the program counter is changed as a result of the instruction. In these cases, the execution takes two instruction cycles with the additional instruction cycle(s) executed as a NOP.

The double-word instructions execute in two instruction cycles.

One instruction cycle consists of four oscillator periods. Thus, for an oscillator frequency of 4 MHz, the normal instruction execution time is 1 μs. If a conditional test is true, or the program counter is changed as a result of an instruction, the instruction execution time is 2 μs. Two-word branch instructions (if true) would take 3 μs.

Figure 29-1 shows the general formats that the instructions can have. All examples use the convention ‘nnh’ to represent a hexadecimal number.

The Instruction Set Summary, shown in Table 29-2, lists the standard instructions recognized by the Microchip MPASM™ Assembler.

Section 29.1.1 “Standard Instruction Set” provides a description of each instruction.

BTFSC **Bit Test File, Skip if Clear**

Syntax: BTFSC f, b {,a}

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

Operation: skip if (f<b>) = 0

Status Affected: None

Encoding:

1011	bbba	ffff	ffff

Description: If bit 'b' in register 'f' is '0', then the next instruction is skipped. If bit 'b' is '0', then the next instruction fetched during the current instruction execution is discarded and a NOP is executed instead, making this a 2-cycle instruction.

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 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    BTFSC   FLAG, 1, 0
FALSE   :
TRUE    :
```

Before Instruction
PC = address (HERE)
After Instruction
If FLAG<1> = 0;
PC = address (TRUE)
If FLAG<1> = 1;
PC = address (FALSE)

BTFSS **Bit Test File, Skip if Set**

Syntax: BTFSS f, b {,a}

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

Operation: skip if (f<b>) = 1

Status Affected: None

Encoding:

1010	bbba	ffff	ffff

Description: If bit 'b' in register 'f' is '1', then the next instruction is skipped. If bit 'b' is '1', then the next instruction fetched during the current instruction execution is discarded and a NOP is executed instead, making this a 2-cycle instruction.

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 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    BTFSS   FLAG, 1, 0
FALSE   :
TRUE    :
```

Before Instruction
PC = address (HERE)
After Instruction
If FLAG<1> = 0;
PC = address (FALSE)
If FLAG<1> = 1;
PC = address (TRUE)

TBLRD **Table Read**

Syntax: TBLRD (*; *+; *-; +*)

Operands: None

Operation: if TBLRD *,
(Prog Mem (TBLPTR)) → TABLAT;
TBLPTR – No Change
if TBLRD *+,
(Prog Mem (TBLPTR)) → TABLAT;
(TBLPTR) + 1 → TBLPTR
if TBLRD *-,
(Prog Mem (TBLPTR)) → TABLAT;
(TBLPTR) – 1 → TBLPTR
if TBLRD +*,
(TBLPTR) + 1 → TBLPTR;
(Prog Mem (TBLPTR)) → TABLAT

Status Affected: None

Encoding:

0000	0000	0000	10nn nn=0 * =1 *+ =2 *- =3 +*

Description: This instruction is used to read the contents of Program Memory (P.M.). To address the program memory, a pointer called Table Pointer (TBLPTR) is used.

The TBLPTR (a 21-bit pointer) points to each byte in the program memory. TBLPTR has a 2-Mbyte address range.

TBLPTR[0] = 0: Least Significant Byte of Program Memory Word

TBLPTR[0] = 1: Most Significant Byte of Program Memory Word

The TBLRD instruction can modify the value of TBLPTR as follows:

- no change
- post-increment
- post-decrement
- pre-increment

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	No operation	No operation
No operation	No operation (Read Program Memory)	No operation	No operation (Write TABLAT)

TBLRD **Table Read (Continued)**

Example 1:
```
TBLRD   *+ ;
```
Before Instruction
TABLAT = 55h
TBLPTR = 00A356h
MEMORY(00A356h) = 34h
After Instruction
TABLAT = 34h
TBLPTR = 00A357h

Example 2:
```
TBLRD   +* ;
```
Before Instruction
TABLAT = AAh
TBLPTR = 01A357h
MEMORY(01A357h) = 12h
MEMORY(01A358h) = 34h
After Instruction
TABLAT = 34h
TBLPTR = 01A358h