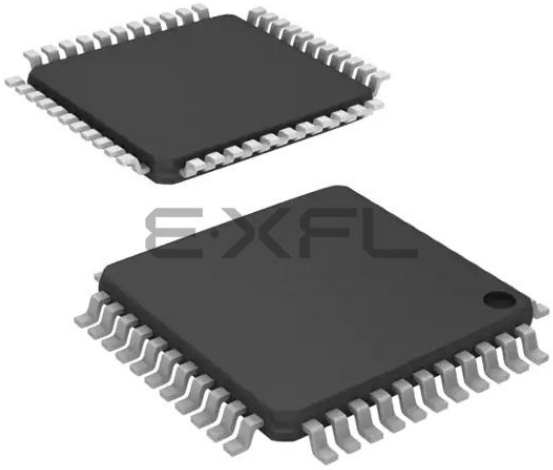


Welcome to E-XFL.COM

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, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	34
Program Memory Size	32KB (16K x 16)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	3.8K x 8
Voltage - Supply (Vcc/Vdd)	2.15V ~ 3.6V
Data Converters	A/D 13x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	44-TQFP
Supplier Device Package	44-TQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f45j11-i-pt

TO OUR VALUED CUSTOMERS

It is our intention to provide our valued customers with the best documentation possible to ensure successful use of your Microchip products. To this end, we will continue to improve our publications to better suit your needs. Our publications will be refined and enhanced as new volumes and updates are introduced.

If you have any questions or comments regarding this publication, please contact the Marketing Communications Department via E-mail at **docerrors@microchip.com** or fax the **Reader Response Form** in the back of this data sheet to (480) 792-4150. We welcome your feedback.

Most Current Data Sheet

To obtain the most up-to-date version of this data sheet, please register at our Worldwide Web site at:

http://www.microchip.com

You can determine the version of a data sheet by examining its literature number found on the bottom outside corner of any page. The last character of the literature number is the version number, (e.g., DS30000A is version A of document DS30000).

Errata

An errata sheet, describing minor operational differences from the data sheet and recommended workarounds, may exist for current devices. As device/documentation issues become known to us, we will publish an errata sheet. The errata will specify the revision of silicon and revision of document to which it applies.

To determine if an errata sheet exists for a particular device, please check with one of the following:

- Microchip's Worldwide Web site; http://www.microchip.com
- Your local Microchip sales office (see last page)

When contacting a sales office, please specify which device, revision of silicon and data sheet (include literature number) you are using.

Customer Notification System

Register on our web site at **www.microchip.com** to receive the most current information on all of our products.

TABLE 1-4: PIC18F4XJ11 PINOUT I/O DESCRIPTIONS

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	44-QFN	44-TQFP			
$\overline{MCLR}$	18	18	I	ST	Master Clear (Reset) input; this is an active-low Reset to the device.
OSC1/CLKI/RA7	32	30			Oscillator crystal or external clock input.
OSC1			I	ST	Oscillator crystal input or external clock source input. ST buffer when configured in RC mode; otherwise CMOS. Main oscillator input connection.
CLKI			I	CMOS	External clock source input; always associated with pin function OSC1 (see related OSC1/CLKI pins).
RA7(1)			I/O	TTL	Digital I/O.
OSC2/CLKO/RA6	33	31			Oscillator crystal or clock output
OSC2			O	—	Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode.
CLKO			O	—	Main oscillator feedback output connection in RC mode, OSC2 pin outputs CLKO, which has 1/4 the frequency of OSC1 and denotes the instruction cycle rate.
RA6(1)			I/O	TTL	Digital I/O.

Legend: TTL = TTL compatible input; CMOS = CMOS compatible input or output; ST = Schmitt Trigger input with CMOS levels; Analog = Analog input; I = Input; O = Output; P = Power; OD = Open-Drain (no P diode to V_{DD}); DIG = Digital output

Note 1: RA7 and RA6 will be disabled if OSC1 and OSC2 are used for the clock function.

TABLE 5-2: INITIALIZATION CONDITIONS FOR ALL REGISTERS

Register	Applicable Devices		Power-on Reset, Brown-out Reset, Wake From Deep Sleep	$\overline{\text{MCLR}}$ Resets WDT Reset RESET Instruction Stack Resets CM Resets	Wake-up via WDT or Interrupt
TOSU	PIC18F2XJ11	PIC18F4XJ11	---0 0000	---0 0000	---u uuuu(1)
TOSH	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu(1)
TOSL	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu(1)
STKPTR	PIC18F2XJ11	PIC18F4XJ11	00-0 0000	uu-0 0000	uu-u uuuu(1)
PCLATU	PIC18F2XJ11	PIC18F4XJ11	---0 0000	---0 0000	---u uuuu
PCLATH	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
PCL	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	PC + 2(2)
TBLPTRU	PIC18F2XJ11	PIC18F4XJ11	--00 0000	--00 0000	--uu uuuu
TBLPTRH	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
TBLPTRL	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
TABLAT	PIC18F2XJ11	PIC18F4XJ11	0000 0000	0000 0000	uuuu uuuu
PRODH	PIC18F2XJ11	PIC18F4XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
PRODL	PIC18F2XJ11	PIC18F4XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
INTCON	PIC18F2XJ11	PIC18F4XJ11	0000 000x	0000 000u	uuuu uuuu(3)
INTCON2	PIC18F2XJ11	PIC18F4XJ11	1111 1111	1111 1111	uuuu uuuu(3)
INTCON3	PIC18F2XJ11	PIC18F4XJ11	1100 0000	1100 0000	uuuu uuuu(3)
INDF0	PIC18F2XJ11	PIC18F4XJ11	N/A	N/A	N/A
POSTINC0	PIC18F2XJ11	PIC18F4XJ11	N/A	N/A	N/A
POSTDEC0	PIC18F2XJ11	PIC18F4XJ11	N/A	N/A	N/A
PREINC0	PIC18F2XJ11	PIC18F4XJ11	N/A	N/A	N/A
PLUSW0	PIC18F2XJ11	PIC18F4XJ11	N/A	N/A	N/A
FSR0H	PIC18F2XJ11	PIC18F4XJ11	---- 0000	---- 0000	---- uuuu
FSR0L	PIC18F2XJ11	PIC18F4XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
WREG	PIC18F2XJ11	PIC18F4XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
INDF1	PIC18F2XJ11	PIC18F4XJ11	N/A	N/A	N/A
POSTINC1	PIC18F2XJ11	PIC18F4XJ11	N/A	N/A	N/A
POSTDEC1	PIC18F2XJ11	PIC18F4XJ11	N/A	N/A	N/A
PREINC1	PIC18F2XJ11	PIC18F4XJ11	N/A	N/A	N/A
PLUSW1	PIC18F2XJ11	PIC18F4XJ11	N/A	N/A	N/A
FSR1H	PIC18F2XJ11	PIC18F4XJ11	---- 0000	---- 0000	---- uuuu
FSR1L	PIC18F2XJ11	PIC18F4XJ11	xxxx xxxx	uuuu uuuu	uuuu uuuu
BSR	PIC18F2XJ11	PIC18F4XJ11	---- 0000	---- 0000	---- uuuu

Legend: u = unchanged, x = unknown, - = unimplemented bit, read as '0', q = value depends on condition.

Note 1: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

2: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

3: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

4: See Table 5-1 for Reset value for specific condition.

5: Not implemented for PIC18F2XJ11 devices.

6: Not implemented on "LF" devices.

6.2.3 INSTRUCTIONS IN PROGRAM MEMORY

The program memory is addressed in bytes. Instructions are stored as 2 bytes or 4 bytes in program memory. The Least Significant Byte (LSB) of an instruction word is always stored in a program memory location with an even address (LSB = 0). To maintain alignment with instruction boundaries, the PC increments in steps of 2 and the LSB will always read '0' (see **Section 6.1.3 "Program Counter"**).

Figure 6-5 provides an example of how instruction words are stored in the program memory.

The CALL and GOTO instructions have the absolute program memory address embedded into the instruction. Since instructions are always stored on word boundaries, the data contained in the instruction is a word address. The word address is written to PC<20:1>, which accesses the desired byte address in program memory. Instruction #2 in Figure 6-5 displays how the instruction, GOTO 0006h, is encoded in the program memory. Program branch instructions, which encode a relative address offset, operate in the same manner. The offset value stored in a branch instruction represents the number of single-word instructions that the PC will be offset by. **Section 27.0 "Instruction Set Summary"** provides further details of the instruction set.

FIGURE 6-5: INSTRUCTIONS IN PROGRAM MEMORY

Program Memory Byte Locations →

			LSB = 1	LSB = 0	Word Address ↓
					000000h
					000002h
					000004h
					000006h
Instruction 1:	MOVLW	055h	0Fh	55h	000008h
Instruction 2:	GOTO	0006h	EFh	03h	00000Ah
			F0h	00h	00000Ch
Instruction 3:	MOVFF	123h, 456h	C1h	23h	00000Eh
			F4h	56h	000010h
					000012h
					000014h

6.2.4 TWO-WORD INSTRUCTIONS

The standard PIC18 instruction set has four two-word instructions: CALL, MOVFF, GOTO and LSFR. In all cases, the second word of the instructions always has '1111' as its four Most Significant bits (MSbs); the other 12 bits are literal data, usually a data memory address.

The use of '1111' in the 4 MSbs of an instruction specifies a special form of NOP. If the instruction is executed in proper sequence immediately after the first word, the data in the second word is accessed and used by the instruction sequence. If the first word is skipped for some reason, and the second word is executed by itself, a NOP is executed instead. This is necessary for cases when the two-word instruction is preceded by a conditional instruction that changes the PC. Example 6-4 illustrates how this works.

Note:	See **Section 6.5 "Program Memory and the Extended Instruction Set"** for information on two-word instructions in the extended instruction set.

EXAMPLE 6-4: TWO-WORD INSTRUCTIONS

CASE 1:

Object Code	Source Code		
0110 0110 0000 0000	TSTFSZ	REG1	; is RAM location 0?
1100 0001 0010 0011	MOVFF	REG1, REG2	; No, skip this word
1111 0100 0101 0110			; Execute this word as a NOP
0010 0100 0000 0000	ADDWF	REG3	; continue code

CASE 2:

Object Code	Source Code		
0110 0110 0000 0000	TSTFSZ	REG1	; is RAM location 0?
1100 0001 0010 0011	MOVFF	REG1, REG2	; Yes, execute this word
1111 0100 0101 0110			; 2nd word of instruction
0010 0100 0000 0000	ADDWF	REG3	; continue code

6.3.4.1 Context Defined SFRs

There are several registers that share the same address in the SFR space. The register's definition and usage depends on the operating mode of its associated peripheral. These registers are:

- SSPxADD and SSPxMSK: These are two separate hardware registers, accessed through a single SFR address. The operating mode of the MSSP modules determines which register is being accessed. See **Section 19.5.3.4 "7-Bit Address Masking Mode"** for additional details.
- PMADDRH/L and PMDOUT2H/L: In this case, these named buffer pairs are actually the same physical registers. The Parallel Master Port (PMP) module's operating mode determines what function the registers take on. See **Section 11.1.2 "Data Registers"** for additional details.

7.5.3 WRITE VERIFY

Depending on the application, good programming practice may dictate that the value written to the memory should be verified against the original value. This should be used in applications where excessive writes can stress bits near the specification limit.

7.5.4 UNEXPECTED TERMINATION OF WRITE OPERATION

If a write is terminated by an unplanned event, such as loss of power or an unexpected Reset, the memory location just programmed should be verified and reprogrammed if needed. If the write operation is interrupted by a MCLR Reset or a WDT time-out Reset during normal operation, the user can check the WRERR bit and rewrite the location(s) as needed.

7.6 Flash Program Operation During Code Protection

See **Section 26.6 "Program Verification and Code Protection"** for details on code protection of Flash program memory.

TABLE 7-2: REGISTERS ASSOCIATED WITH PROGRAM FLASH MEMORY

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
TBLPTRU	—	—	bit 21	Program Memory Table Pointer Upper Byte (TBLPTR<20:16>)					69
TBPLTRH	Program Memory Table Pointer High Byte (TBLPTR<15:8>)								69
TBLPTRL	Program Memory Table Pointer Low Byte (TBLPTR<7:0>)								69
TABLAT	Program Memory Table Latch								69
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	69
EECON2	Program Memory Control Register 2 (not a physical register)								71
EECON1	—	—	WPROG	FREE	WRERR	WREN	WR	—	71

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

REGISTER 11-7: PMSTATH: PARALLEL PORT STATUS REGISTER HIGH BYTE (BANKED F55h)(1)

R-0	R/W-0	U-0	U-0	R-0	R-0	R-0	R-0
IBF	IBOV	—	—	IB3F	IB2F	IB1F	IB0F
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 = All writable input buffer registers are full
0 = Some or all of the writable input buffer registers are empty

bit 6 **IBOV:** Input Buffer Overflow Status bit
1 = A write attempt to a full input byte register occurred (must be cleared in software)
0 = No overflow occurred

bit 5-4 **Unimplemented:** Read as '0'

bit 3-0 **IB3F:IB0F:** Input Buffer x Status Full bits
1 = Input buffer contains data that has not been read (reading buffer will clear this bit)
0 = Input buffer does not contain any unread data

Note 1: This register is only available in 44-pin devices.

REGISTER 11-8: PMSTATL: PARALLEL PORT STATUS REGISTER LOW BYTE (BANKED F54h)(1)

R-1	R/W-0	U-0	U-0	R-1	R-1	R-1	R-1
OBE	OBUF	—	—	OB3E	OB2E	OB1E	OB0E
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 **OBE:** Output Buffer Empty Status bit
1 = All readable output buffer registers are empty
0 = Some or all of the readable output buffer registers are full

bit 6 **OBUF:** Output Buffer Underflow Status bit
1 = A read occurred from an empty output byte register (must be cleared in software)
0 = No underflow occurred

bit 5-4 **Unimplemented:** Read as '0'

bit 3-0 **OB3E:OB0E:** Output Buffer x Status Empty bits
1 = Output buffer is empty (writing data to the buffer will clear this bit)
0 = Output buffer contains data that has not been transmitted

Note 1: This register is only available in 44-pin devices.

11.1.2 DATA REGISTERS

The PMP module uses eight registers for transferring data into and out of the microcontroller. They are arranged as four pairs to allow the option of 16-bit data operations:

- PMDIN1H and PMDIN1L
- PMDIN2H and PMDIN2L
- PMADDRH/PMDOUT1H and PMADDRL/PMDOUT1L
- PMDOUT2H and PMDOUT2L

The PMDIN1 register is used for incoming data in Slave modes and both input and output data in Master modes. The PMDIN2 register is used for buffering input data in select Slave modes.

The PMADDR/PMDOUT1 registers are actually a single register pair; the name and function are dictated by the module's operating mode. In Master modes, the registers function as the PMADDRH and PMADDRL registers and contain the address of any incoming or outgoing data. In Slave modes, the registers function as PMDOUT1H and PMDOUT1L and are used for outgoing data.

PMADDRH differs from PMADDRL in that it can also have limited PMP control functions. When the module is operating in select Master mode configurations, the upper two bits of the register can be used to determine the operation of chip select signals. If these are not used, PMADDR simply functions to hold the upper 8 bits of the address. Register 11-9 provides the function of the individual bits in PMADDRH.

The PMDOUT2H and PMDOUT2L registers are only used in Buffered Slave modes and serve as a buffer for outgoing data.

11.1.3 PAD CONFIGURATION CONTROL REGISTER

In addition to the module level configuration options, the PMP module can also be configured at the I/O pin for electrical operation. This option allows users to select either the normal Schmitt Trigger input buffer on digital I/O pins shared with the PMP, or use TTL level compatible buffers instead. Buffer configuration is controlled by the PMPTTL bit in the PADCFG1 register.

FIGURE 11-18: WRITE TIMING, 8-BIT DATA, PARTIALLY MULTIPLEXED ADDRESS, ENABLE STROBE

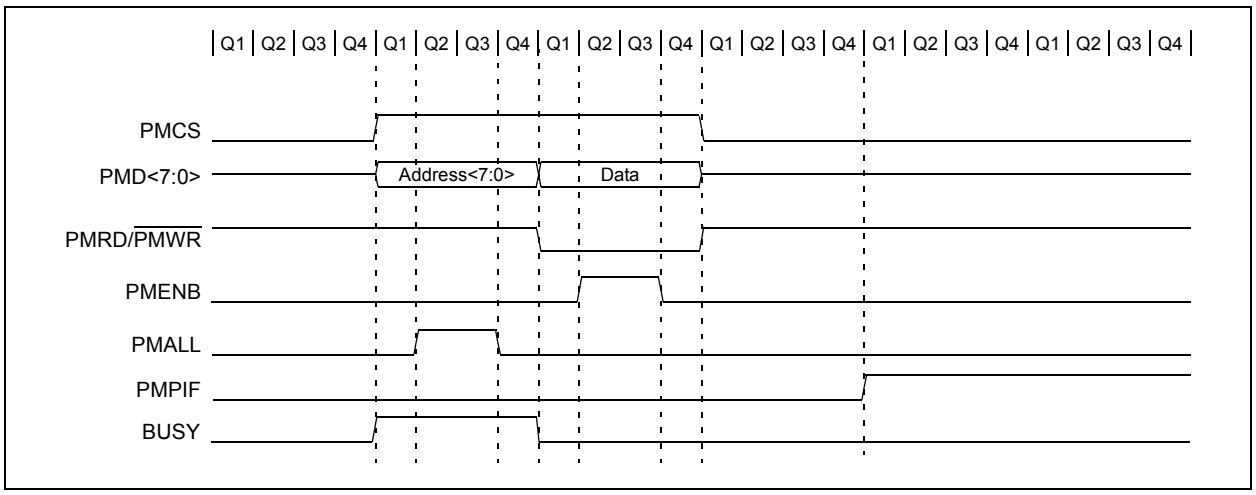


FIGURE 11-19: READ TIMING, 8-BIT DATA, FULLY MULTIPLEXED 16-BIT ADDRESS

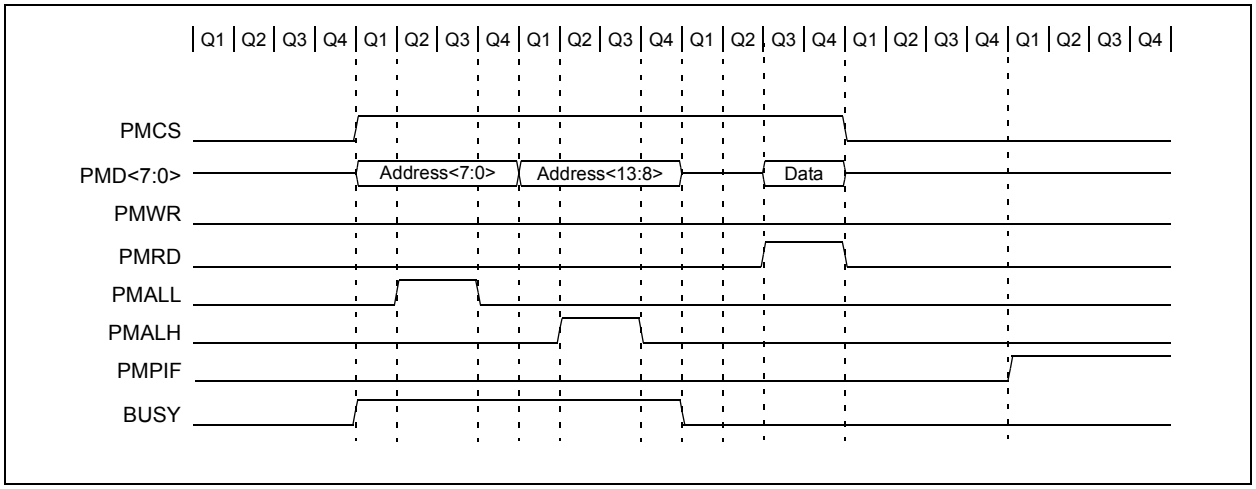
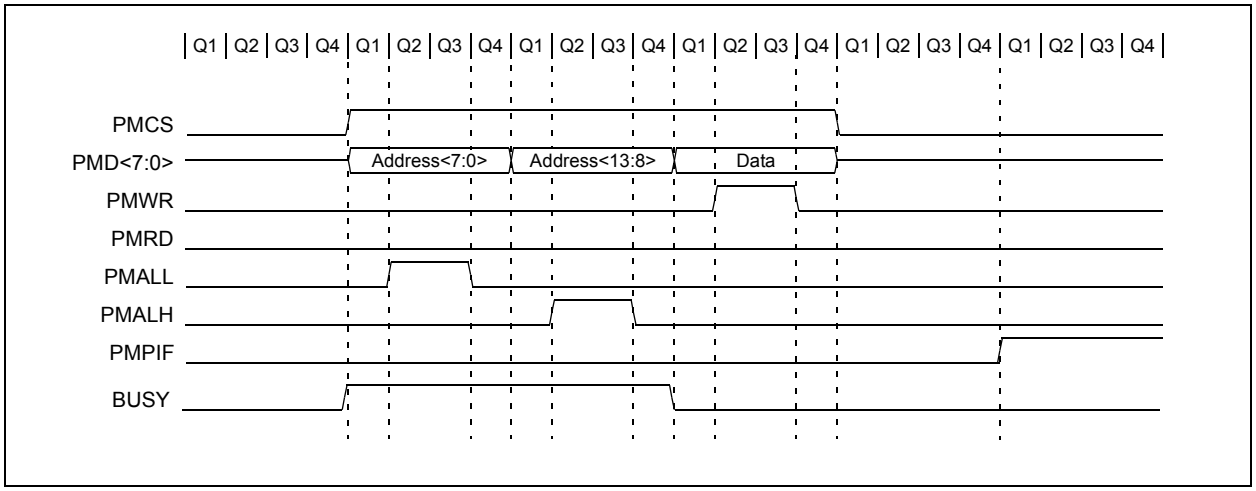


FIGURE 11-20: WRITE TIMING, 8-BIT DATA, FULLY MULTIPLEXED 16-BIT ADDRESS



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).

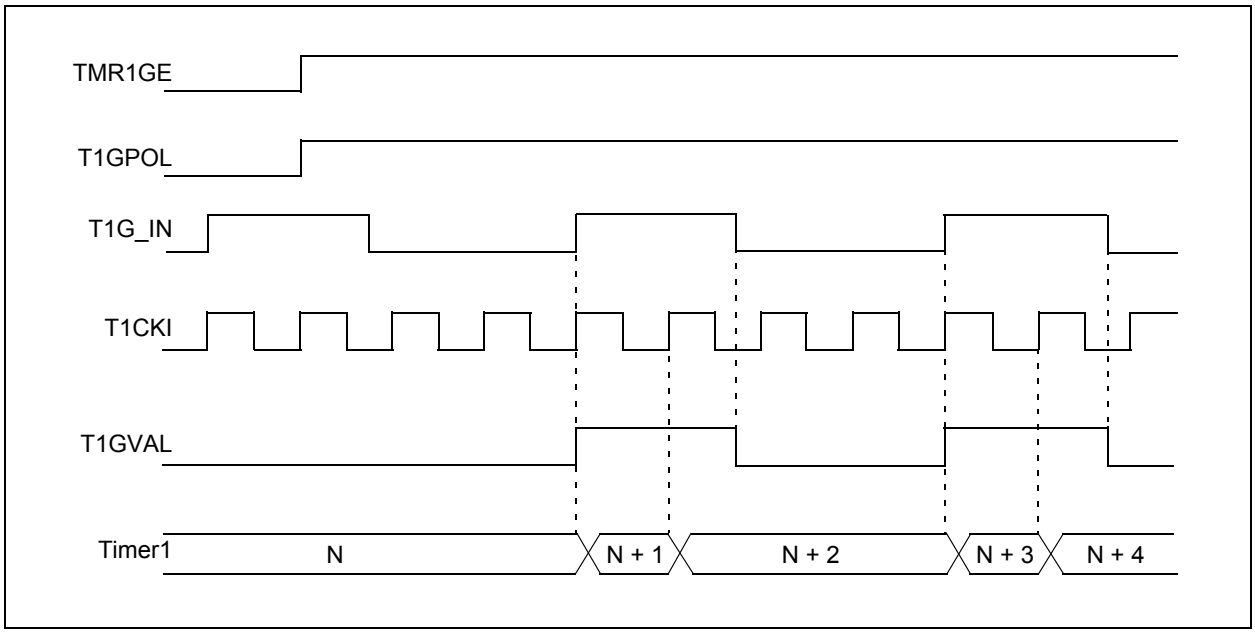
Since Timer0 is shut down in Sleep mode, the TMR0 interrupt cannot awaken the processor from Sleep.

TABLE 12-1: REGISTERS ASSOCIATED WITH TIMER0

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
TMR0L	Timer0 Register Low Byte								91
TMR0H	Timer0 Register High Byte								91
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	90
T0CON	TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0	91

Legend: — = unimplemented, read as ‘0’. Shaded cells are not used by Timer0.

FIGURE 13-4: TIMER1 GATE COUNT ENABLE MODE



13.8.2 TIMER1 GATE SOURCE SELECTION

The Timer1 gate source can be selected from one of four different sources. Source selection is controlled by the T1GSSx bits of the T1GCON register. The polarity for each available source is also selectable. Polarity selection is controlled by the T1GPOL bit of the T1GCON register.

TABLE 13-4: TIMER1 GATE SOURCES

T1GSS<1:0>	Timer1 Gate Source
00	Timer1 Gate Pin
01	Overflow of Timer0 (TMR0 increments from FFh to 00h)
10	TMR2 to Match PR2 (TMR2 increments to match PR2)

13.8.2.1 T1G Pin Gate Operation

The T1G pin is one source for Timer1 gate control. It can be used to supply an external source to the Timer1 gate circuitry.

13.8.2.2 Timer0 Overflow Gate Operation

When Timer0 increments from FFh to 00h, a low-to-high pulse will automatically be generated and internally supplied to the Timer1 gate circuitry.

13.8.2.3 Timer2 Match Gate Operation

The TMR2 register will increment until it matches the value in the PR2 register. On the very next increment cycle, TMR2 will be reset to 00h. When this Reset occurs, a low-to-high pulse will automatically be generated and internally supplied to the Timer1 gate circuitry.

The pulse remains high for one instruction cycle and returns to low until the next match.

When T1GPOL = 1, Timer1 increments for a single instruction cycle following TMR2 matching PR2.

With T1GPOL = 0, Timer1 increments except during the cycle following the match.

17.1.1 RTCC CONTROL REGISTERS

REGISTER 17-1: RTCCFG: RTCC CONFIGURATION REGISTER (BANKED F3Fh)(1)

R/W-0	U-0	R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0
RTCEN(2)	—	RTCWREN	RTCSYNC	HALFSEC(3)	RTCOE	RTCPTR1	RTCPTR0
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 **RTCEN:** RTCC Enable bit(2)
1 = RTCC module is enabled
0 = RTCC module is disabled

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

bit 5 **RTCWREN:** RTCC Value Registers Write Enable bit
1 = RTCVALH and RTCVALL registers can be written to by the user
0 = RTCVALH and RTCVALL registers are locked out from being written to by the user

bit 4 **RTCSYNC:** RTCC Value Registers Read Synchronization bit
1 = RTCVALH, RTCVALL and ALRMRPT registers can change while reading due to a rollover ripple resulting in an invalid data read
If the register is read twice and results in the same data, the data can be assumed to be valid.
0 = RTCVALH, RTCVALL or ALCFGRPT registers can be read without concern over a rollover ripple

bit 3 **HALFSEC:** Half-Second Status bit(3)
1 = Second half period of a second
0 = First half period of a second

bit 2 **RTCOE:** RTCC Output Enable bit
1 = RTCC clock output enabled
0 = RTCC clock output disabled

bit 1-0 **RTCPTR<1:0>:** RTCC Value Register Window Pointer bits
Points to the corresponding RTCC Value registers when reading the RTCVALH<7:0> and RTCVALL<7:0> registers; the RTCPTR<1:0> value decrements on every read or write of RTCVALH<7:0> until it reaches '00'.
RTCVALH<7:0>:
00 = Minutes
01 = Weekday
10 = Month
11 = Reserved
RTCVALL<7:0>:
00 = Seconds
01 = Hours
10 = Day
11 = Year

Note 1: The RTCCFG register is only affected by a POR.
2: A write to the RTCEN bit is only allowed when RTCWREN = 1.
3: This bit is read-only. It is cleared to '0' on a write to the lower half of the MINSEC register.

17.1.4 RTCEN BIT WRITE

An attempt to write to the RTCEN bit while RTCWREN = 0 will be ignored. RTCWREN must be set before a write to RTCEN can take place.

Like the RTCEN bit, the RTCVALH<15:8> and RTCVALL<7:0> registers can only be written to when RTCWREN = 1. A write to these registers, while RTCWREN = 0, will be ignored.

17.2 Operation

17.2.1 REGISTER INTERFACE

The register interface for the RTCC and alarm values is implemented using the Binary Coded Decimal (BCD) format. This simplifies the firmware, when using the module, as each of the digits is contained within its own 4-bit value (see Figure 17-2 and Figure 17-3).

FIGURE 17-2: TIMER DIGIT FORMAT

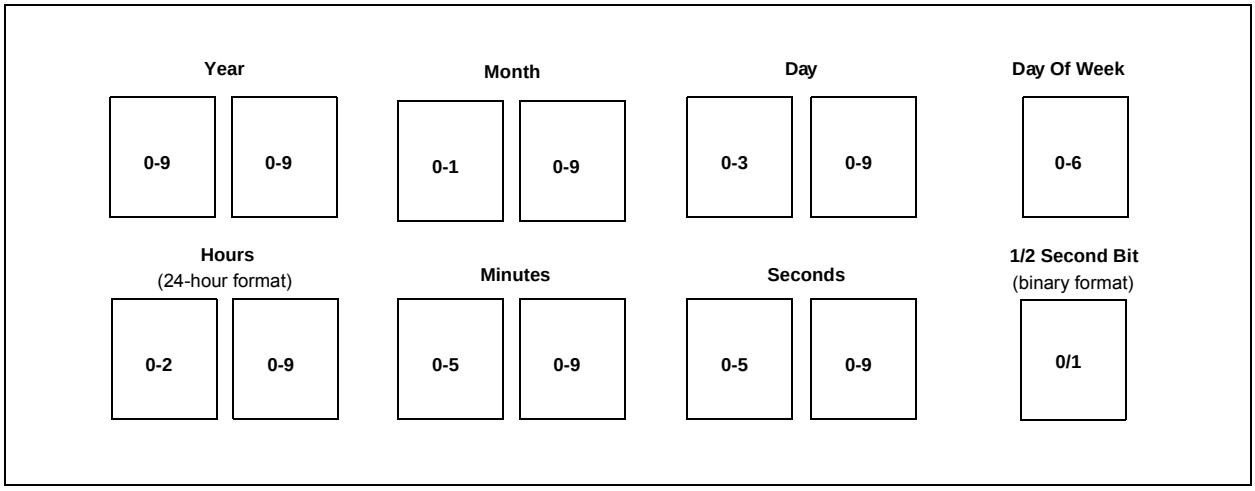


FIGURE 17-3: ALARM DIGIT FORMAT

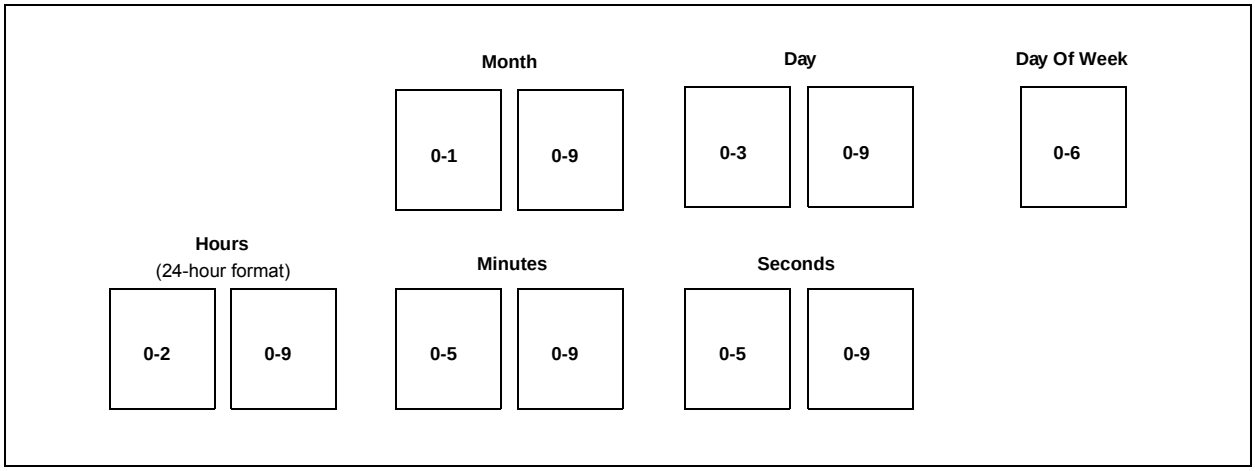
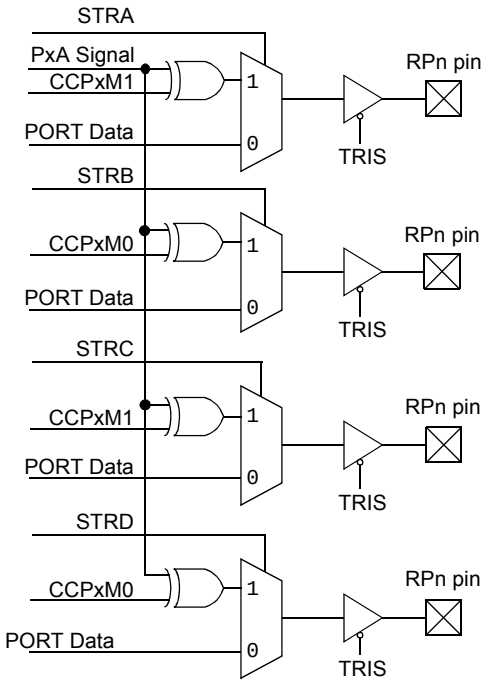


FIGURE 18-18: SIMPLIFIED STEERING BLOCK DIAGRAM



Note 1: Port outputs are configured as displayed when the CCPxCON register bits, PxM<1:0> = 00 and CCP1M<3:2> = 11.

2: Single PWM output requires setting at least one of the STRx bits.

18.5.7.1 Steering Synchronization

The STRSYNC bit of the PSTRxCON register gives the user two selections of when the steering event will happen. When the STRSYNC bit is '0', the steering event will happen at the end of the instruction that writes to the PSTRxCON register. In this case, the output signal at the Px<D:A> pins may be an incomplete PWM waveform. This operation is useful when the user firmware needs to immediately remove a PWM signal from the pin.

When the STRSYNC bit is '1', the effective steering update will happen at the beginning of the next PWM period. In this case, steering on/off the PWM output will always produce a complete PWM waveform.

Figures 18-19 and 18-20 illustrate the timing diagrams of the PWM steering depending on the STRSYNC setting.

FIGURE 18-19: EXAMPLE OF STEERING EVENT AT END OF INSTRUCTION (STRSYNC = 0)

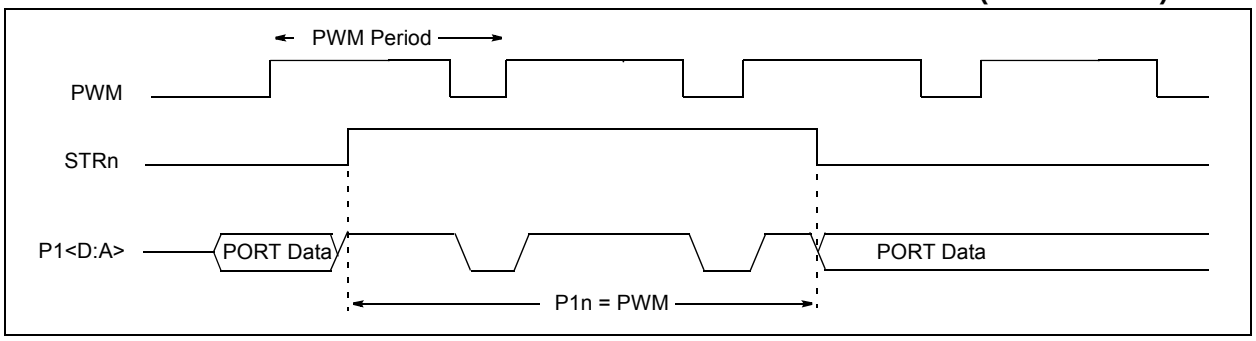
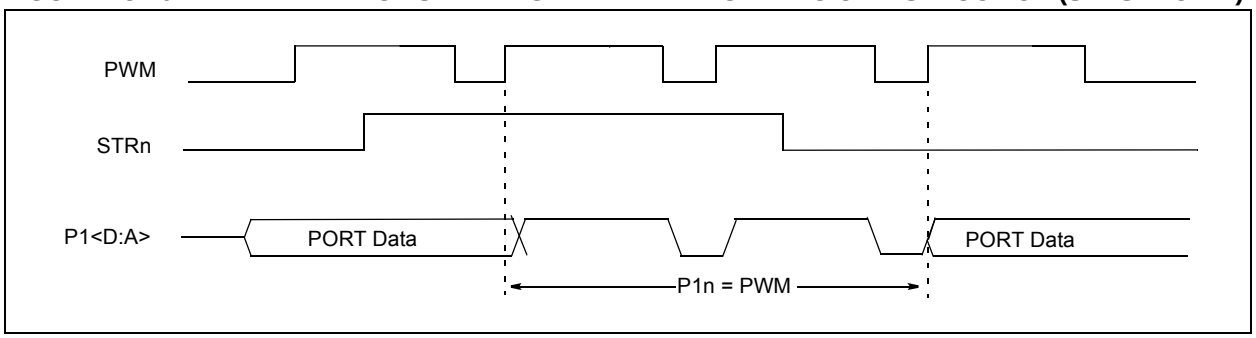


FIGURE 18-20: EXAMPLE OF STEERING EVENT AT BEGINNING OF INSTRUCTION (STRSYNC = 1)



19.3.6 MASTER MODE

The master can initiate the data transfer at any time because it controls the SCKx. The master determines when the slave (Processor 2, Figure 19-2) is to broadcast data by the software protocol.

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

Note:	To avoid lost data in Master mode, a read of the SSPxBUF must be performed to clear the Buffer Full (BF) detect bit (SSPxSTAT<0>) between each transmission.

The CKP is selected by appropriately programming the CKP bit (SSPxCON1<4>). This then, would give waveforms for SPI communication as illustrated in Figure 19-3, Figure 19-5 and Figure 19-6, where the Most Significant Byte (MSB) is transmitted first. In Master mode, the SPI clock rate (bit rate) is user-programmable to be one of the following:

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

When using the Timer2 output/2 option, the Period Register 2 (PR2) can be used to determine the SPI bit rate. However, only PR2 values of 0x01 to 0xFF are valid in this mode.

Figure 19-3 illustrates the waveforms for Master mode. When the CKE bit is set, the SDOx data is valid before there is a clock edge on SCKx. The change of the input sample is shown based on the state of the SMP bit. The time when the SSPxBUF is loaded with the received data is shown.

FIGURE 19-3: SPI MODE WAVEFORM (MASTER MODE)

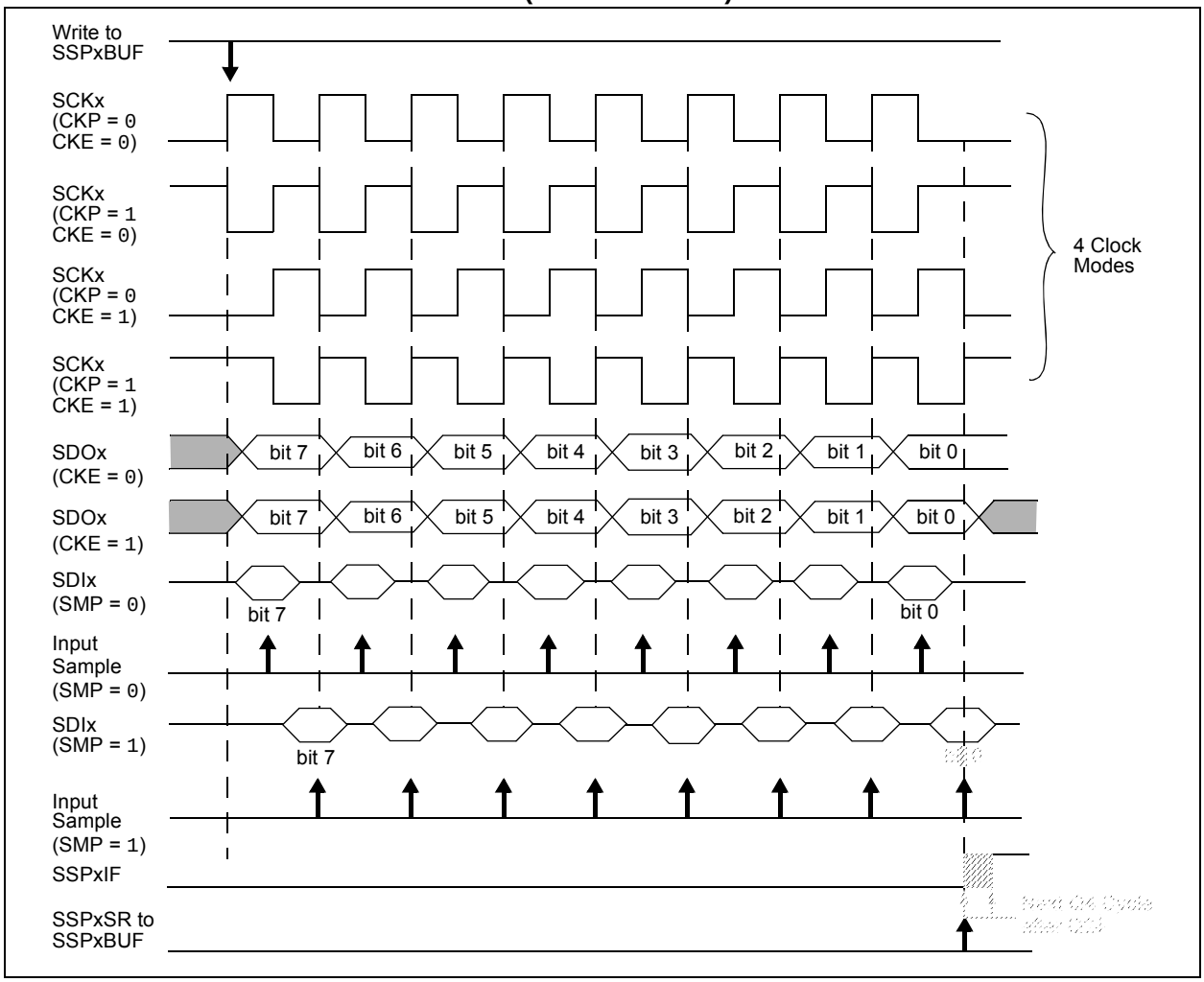


TABLE 19-2: REGISTERS ASSOCIATED WITH SPI OPERATION

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	69
PIR1	PMPIF(2)	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	72
PIE1	PMPIE(2)	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	72
IPR1	PMPIP(2)	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	72
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CTMUIF	TMR3GIF	RTCCIF	72
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CTMUIE	TMR3GIE	RTCCIE	72
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CTMUIP	TMR3GIP	RTCCIP	72
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	72
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	72
SSP1BUF	MSSP1 Receive Buffer/Transmit Register								70
SSPxCON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	70
SSPxSTAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	70
SSP2BUF	MSSP2 Receive Buffer/Transmit Register								73
ODCON3(1)	—	—	—	—	—	—	SPI2OD	SPI1OD	74

Legend: Shaded cells are not used by the MSSP module in SPI mode.

Note 1: Configuration SFR overlaps with default SFR at this address; available only when WDTCON<4> = 1.

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

19.4 SPI DMA Module

The SPI DMA module contains control logic to allow the MSSP2 module to perform SPI direct memory access transfers. This enables the module to quickly transmit or receive large amounts of data with relatively little CPU intervention. When the SPI DMA module is used, MSSP2 can directly read and write to general purpose SRAM. When the SPI DMA module is not enabled, MSSP2 functions normally, but without DMA capability.

The SPI DMA module is composed of control logic, a Destination Receive Address Pointer, a Transmit Source Address Pointer, an interrupt manager and a Byte Count register for setting the size of each DMA transfer. The DMA module may be used with all SPI Master and Slave modes, and supports both half-duplex and full-duplex transfers.

19.4.1 I/O PIN CONSIDERATIONS

When enabled, the SPI DMA module uses the MSSP2 module. All SPI related input and output signals related to MSSP2 are routed through the Peripheral Pin Select module. The appropriate initialization procedure as described in **Section 19.4.6 "Using the SPI DMA Module"** will need to be followed prior to using the SPI DMA module. The output pins assigned to the SDO2 and SCK2 functions can optionally be configured as open-drain outputs, such as for level shifting operations mentioned in the same section.

19.4.2 RAM TO RAM COPY OPERATIONS

Although the SPI DMA module is primarily intended to be used for SPI communication purposes, the module can also be used to perform RAM to RAM copy operations. To do this, configure the module for Full-Duplex Master mode operation, but assign the SDO2 output and SDI2 input functions onto the same RPn pin in the PPS module. This will allow the module to operate in Loopback mode, providing RAM copy capability.

19.4.3 IDLE AND SLEEP CONSIDERATIONS

The SPI DMA module remains fully functional when the microcontroller is in Idle mode.

During normal sleep, the SPI DMA module is not functional and should not be used. To avoid corrupting a transfer, user firmware should be careful to make certain that pending DMA operations are complete by polling the DMAEN bit in the DMACON1 register prior to putting the microcontroller into Sleep.

In SPI Slave modes, the MSSP2 module is capable of transmitting and/or receiving one byte of data while in Sleep mode. This allows the SSP2IF flag in the PIR3 register to be used as a wake-up source. When the DMAEN bit is cleared, the SPI DMA module is effectively disabled, and the MSSP2 module functions normally, but without DMA capabilities. If the DMAEN bit is clear prior to entering Sleep, it is still possible to use the SSP2IF as a wake-up source without any data loss.

Neither MSSP2 nor the SPI DMA module will provide any functionality in Deep Sleep. Upon exiting from Deep Sleep, all of the I/O pins, MSSP2 and SPI DMA related registers will need to be fully reinitialized before the SPI DMA module can be used again.

19.4.4 REGISTERS

The SPI DMA engine is enabled and controlled by the following Special Function Registers:

- DMACON1
- DMACON2
- TXADDRH
- TXADDRL
- RXADDRH
- RXADDRL
- DMABCH
- DMABCL

19.4.4.1 DMACON1

The DMACON1 register is used to select the main operating mode of the SPI DMA module. The SSCON1 and SSCON0 bits are used to control the slave select pin.

When MSSP2 is used in SPI Master mode with the SPI DMA module, $\overline{\text{SSDMA}}$ can be controlled by the DMA module as an output pin. If MSSP2 will be used to communicate with an SPI slave device that needs the $\overline{\text{SS}}$ pin to be toggled periodically, the SPI DMA hardware can automatically be used to deassert $\overline{\text{SS}}$ between each byte, every two bytes or every four bytes.

Alternatively, user firmware can manually generate slave select signals with normal general purpose I/O pins, if required by the slave device(s).

When the TXINC bit is set, the TXADDR register will automatically increment after each transmitted byte. Automatic transmit address increment can be disabled by clearing the TXINC bit. If the automatic transmit address increment is disabled, each byte which is output on SDO2, will be the same (the contents of the SRAM pointed to by the TXADDR register) for the entire DMA transaction.

When the RXINC bit is set, the RXADDR register will automatically increment after each received byte. Automatic receive address increment can be disabled by clearing the RXINC bit. If RXINC is disabled in Full-Duplex or Half-Duplex Receive modes, all incoming data bytes on SDI2 will overwrite the same memory location pointed to by the RXADDR register. After the SPI DMA transaction has completed, the last received byte will reside in the memory location pointed to by the RXADDR register.

The SPI DMA module can be used for either half-duplex receive only communication, half-duplex transmit only communication or full-duplex simultaneous transmit and receive operations. All modes are available for both SPI master and SPI slave configurations. The DUPLEX0 and DUPLEX1 bits can be used to select the desired operating mode.

The behavior of the DLYINTEN bit varies greatly depending on the SPI operating mode. For example behavior for each of the modes, see Figure 19-3 through Figure 19-6.

SPI Slave mode, DLYINTEN = 1: In this mode, an SSP2IF interrupt will be generated during a transfer if the time between successful byte transmission events is longer than the value set by the DLYCYC<3:0> bits in the DMACON2 register. This interrupt allows slave firmware to know that the master device is taking an unusually large amount of time between byte transmissions. For example, this information may be useful for implementing application-defined communication protocols involving time-outs if the bus remains Idle for too long. When DLYINTEN = 1, the DLYLVL<3:0> interrupts occur normally according to the selected setting.

SPI Slave mode, DLYINTEN = 0: In this mode, the time-out based interrupt is disabled. No additional SSP2IF interrupt events will be generated by the SPI DMA module, other than those indicated by the INTLVL<3:0> bits in the DMACON2 register. In this mode, always set DLYCYC<3:0> = 0000.

SPI Master mode, DLYINTEN = 0: The DLYCYC<3:0> bits in the DMACON2 register determine the amount of additional inter-byte delay, which is added by the SPI DMA module during a transfer. The Master mode $\overline{SS2}$ output feature may be used.

SPI Master mode, DLYINTEN = 1: The amount of hardware overhead is slightly reduced in this mode, and the minimum inter-byte delay is 8 TCY for FOSC/4, 9 TCY for FOSC/16 and 15 TCY for FOSC/64. This mode can potentially be used to obtain slightly higher effective SPI bandwidth. In this mode, the $\overline{SS2}$ control feature cannot be used, and should always be disabled (DMACON1<7:6> = 00). Additionally, the interrupt generating hardware (used in Slave mode) remains active. To avoid extraneous SSP2IF interrupt events, set the DMACON2 delay bits, DLYCYC<3:0> = 1111, and ensure that the SPI serial clock rate is no slower than FOSC/64.

In SPI Master modes, the DMAEN bit is used to enable the SPI DMA module and to initiate an SPI DMA transaction. After user firmware sets the DMAEN bit, the DMA hardware will begin transmitting and/or receiving data bytes according to the configuration used. In SPI Slave modes, setting the DMAEN bit will finish the initialization steps needed to prepare the SPI DMA module for communication (which must still be initiated by the master device).

To avoid possible data corruption, once the DMAEN bit is set, user firmware should not attempt to modify any of the MSSP2 or SPI DMA related registers, with the exception of the INTLVL bits in the DMACON2 register.

If user firmware wants to halt an ongoing DMA transaction, the DMAEN bit can be manually cleared by the firmware. Clearing the DMAEN bit while a byte is currently being transmitted will not immediately halt the byte in progress. Instead, any byte currently in progress will be completed before the MSSP2 and SPI DMA modules go back to their Idle conditions. If user firmware clears the DMAEN bit, the TXADDR, RXADDR and DMABC registers will no longer update, and the DMA module will no longer make any additional read or writes to SRAM; therefore, state information can be lost.

21.1 A/D Acquisition Requirements

For the A/D Converter to meet its specified accuracy, the charge holding capacitor (CHOLD) must be allowed to fully charge to the input channel voltage level. The analog input model is illustrated in Figure 21-2. The source impedance (RS) and the internal sampling switch (RSS) impedance directly affect the time required to charge the capacitor CHOLD. The sampling switch (RSS) impedance varies over the device voltage (VDD). The source impedance affects the offset voltage at the analog input (due to pin leakage current). **The maximum recommended impedance for analog sources is 2.5 kΩ.** After the analog input channel is selected (changed), the channel must be sampled for at least the minimum acquisition time before starting a conversion.

Note:	When the conversion is started, the holding capacitor is disconnected from the input pin.

To calculate the minimum acquisition time, Equation 21-1 may be used. This equation assumes that 1/2 LSb error is used (1024 steps for the A/D). The 1/2 LSb error is the maximum error allowed for the A/D to meet its specified resolution.

Equation 21-3 provides the calculation of the minimum required acquisition time, TACQ. This calculation is based on the following application system assumptions:

CHOLD	=	25 pF
Rs	=	2.5 kΩ
Conversion Error	≤	1/2 LSb
VDD	=	3V → Rss = 2 kΩ
Temperature	=	85°C (system max.)

EQUATION 21-1: ACQUISITION TIME

$$T_{ACQ} = \text{Amplifier Settling Time} + \text{Holding Capacitor Charging Time} + \text{Temperature Coefficient}$$
$$= T_{AMP} + T_C + T_{COFF}$$

EQUATION 21-2: A/D MINIMUM CHARGING TIME

$$V_{HOLD} = (V_{REF} - (V_{REF}/2048)) \bullet (1 - e^{(-T_C/C_{HOLD}(R_{IC} + R_{SS} + R_S))})$$
or
$$T_C = -(C_{HOLD})(R_{IC} + R_{SS} + R_S) \ln(1/2048)$$

EQUATION 21-3: CALCULATING THE MINIMUM REQUIRED ACQUISITION TIME

$$T_{ACQ} = T_{AMP} + T_C + T_{COFF}$$
$$T_{AMP} = 0.2\ \mu s$$
$$T_{COFF} = (\text{Temp} - 25°C)(0.02\ \mu s/°C)$$
$$(85°C - 25°C)(0.02\ \mu s/°C)$$
$$1.2\ \mu s$$

Temperature coefficient is only required for temperatures > 25°C. Below 25°C, $T_{COFF} = 0\ \mu s$.

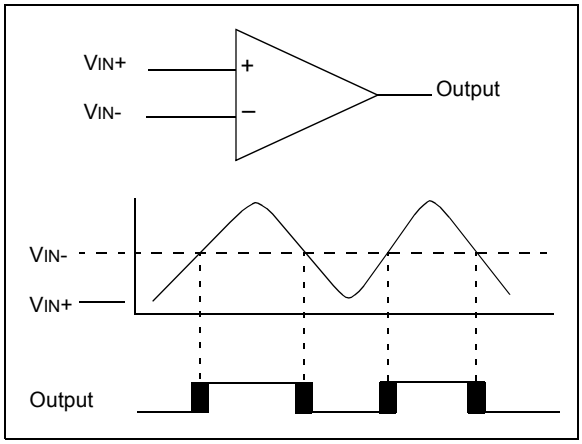
$$T_C = -(C_{HOLD})(R_{IC} + R_{SS} + R_S) \ln(1/2048)\ \mu s$$
$$-(25\ pF)\ (1\ k\Omega + 2\ k\Omega + 2.5\ k\Omega) \ln(0.0004883)\ \mu s$$
$$1.05\ \mu s$$

$$T_{ACQ} = 0.2\ \mu s + 1.05\ \mu s + 1.2\ \mu s$$
$$2.45\ \mu s$$

22.2 Comparator Operation

A single comparator is shown in Figure 22-2, along with the relationship between the analog input levels and the digital output. When the analog input at VIN+ is less than the analog input, VIN-, the output of the comparator is a digital low level. When the analog input at VIN+ is greater than the analog input, VIN-, the output of the comparator is a digital high level. The shaded areas of the output of the comparator in Figure 22-2 represent the uncertainty due to input offsets and response time.

FIGURE 22-2: SINGLE COMPARATOR



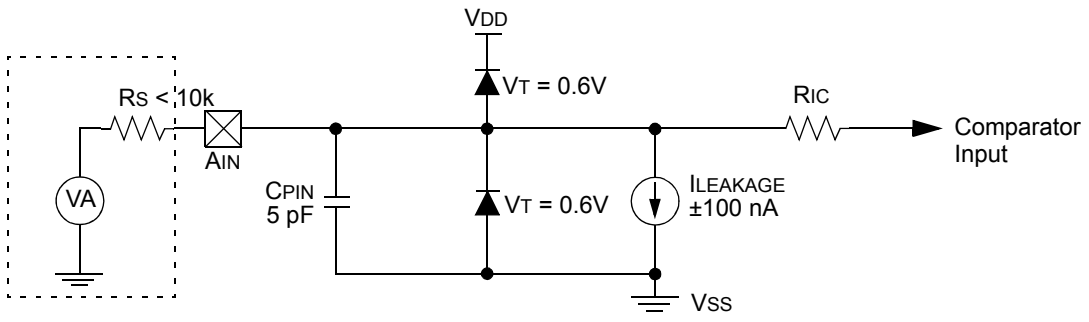
22.3 Comparator Response Time

Response time is the minimum time, after selecting a new reference voltage or input source, before the comparator output has a valid level. The response time of the comparator differs from the settling time of the voltage reference. Therefore, both of these times must be considered when determining the total response to a comparator input change. Otherwise, the maximum delay of the comparators should be used (see **Section 29.0 "Electrical Characteristics"**).

22.4 Analog Input Connection Considerations

Figure 22-3 provides a simplified circuit for an analog input. Since the analog pins are connected to a digital output, they have reverse biased diodes to VDD and VSS. The analog input, therefore, must be between VSS and VDD. If the input voltage deviates from this range by more than 0.6V in either direction, one of the diodes is forward biased and a latch-up condition may occur. A maximum source impedance of 10 kΩ is recommended for the analog sources. Any external component connected to an analog input pin, such as a capacitor or a Zener diode, should have very little leakage current.

FIGURE 22-3: COMPARATOR ANALOG INPUT MODEL



Legend:

CPIN	=	Input Capacitance
VT	=	Threshold Voltage
ILEAKAGE	=	Leakage Current at the pin due to various junctions
RIC	=	Interconnect Resistance
RS	=	Source Impedance
VA	=	Analog Voltage