Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	PIC
Core Size	8-Bit
Speed	40MHz
Connectivity	EBI/EMI, I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, HLVD, POR, PWM, WDT
Number of I/O	70
Program Memory Size	128KB (64K x 16)
Program Memory Type	FLASH
EEPROM Size	1K x 8
RAM Size	3.8K x 8
Voltage - Supply (Vcc/Vdd)	4.2V ~ 5.5V
Data Converters	A/D 16x10b
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/pic18f8722t-i-pt

TABLE 4-4: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices				Power-on Reset, Brown-out Reset	$\overline{\text{MCLR}}$ Resets, WDT Reset, RESET Instruction, Stack Resets	Wake-up via WDT or Interrupt
FSR1H	6X27	6X22	8X27	8X22	---- 0000	---- 0000	---- uuuu
FSR1L	6X27	6X22	8X27	8X22	xxxx xxxx	uuuu uuuu	uuuu uuuu
BSR	6X27	6X22	8X27	8X22	---- 0000	---- 0000	---- uuuu
INDF2	6X27	6X22	8X27	8X22	N/A	N/A	N/A
POSTINC2	6X27	6X22	8X27	8X22	N/A	N/A	N/A
POSTDEC2	6X27	6X22	8X27	8X22	N/A	N/A	N/A
PREINC2	6X27	6X22	8X27	8X22	N/A	N/A	N/A
PLUSW2	6X27	6X22	8X27	8X22	N/A	N/A	N/A
FSR2H	6X27	6X22	8X27	8X22	---- 0000	---- 0000	---- uuuu
FSR2L	6X27	6X22	8X27	8X22	xxxx xxxx	uuuu uuuu	uuuu uuuu
STATUS	6X27	6X22	8X27	8X22	---x xxxx	---u uuuu	---u uuuu
TMR0H	6X27	6X22	8X27	8X22	0000 0000	0000 0000	uuuu uuuu
TMR0L	6X27	6X22	8X27	8X22	xxxx xxxx	uuuu uuuu	uuuu uuuu
T0CON	6X27	6X22	8X27	8X22	1111 1111	1111 1111	uuuu uuuu
OSCCON	6X27	6X22	8X27	8X22	0100 q000	0100 q000	uuuu uuqu
HLVDCON	6X27	6X22	8X27	8X22	0-00 0101	0-00 0101	u-uu uuuu
WDTCON	6X27	6X22	8X27	8X22	---- ---0	---- ---0	---- ---u
RCON(4)	6X27	6X22	8X27	8X22	0q-1 11q0	0q-q qquu	uq-u qquu
TMR1H	6X27	6X22	8X27	8X22	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR1L	6X27	6X22	8X27	8X22	xxxx xxxx	uuuu uuuu	uuuu uuuu
T1CON	6X27	6X22	8X27	8X22	0000 0000	u0uu uuuu	uuuu uuuu
TMR2	6X27	6X22	8X27	8X22	0000 0000	0000 0000	uuuu uuuu
PR2	6X27	6X22	8X27	8X22	1111 1111	uuuu uuuu	uuuu uuuu
T2CON	6X27	6X22	8X27	8X22	-000 0000	-000 0000	-uuu uuuu
SSP1BUF	6X27	6X22	8X27	8X22	xxxx xxxx	uuuu uuuu	uuuu uuuu
SSP1ADD	6X27	6X22	8X27	8X22	0000 0000	0000 0000	uuuu uuuu
SSP1STAT	6X27	6X22	8X27	8X22	0000 0000	0000 0000	uuuu uuuu
SSP1CON1	6X27	6X22	8X27	8X22	0000 0000	0000 0000	uuuu uuuu
SSP1CON2	6X27	6X22	8X27	8X22	0000 0000	0000 0000	uuuu uuuu

Legend: u = unchanged, x = unknown, - = unimplemented bit, read as '0', q = value depends on condition.
Shaded cells indicate conditions do not apply for the designated device.

Note 1: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

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: 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.

4: See Table 4-3 for Reset value for specific condition.

5: Bits 6 and 7 of PORTA, LATA and TRISA are enabled, depending on the oscillator mode selected. When not enabled as PORTA pins, they are disabled and read '0'.

7.1 External Memory Bus Control

The operation of the interface is controlled by the MEMCON register (Register 7-1). This register is available in all program memory operating modes except Microcontroller mode. In this mode, the register is disabled and cannot be written to.

The EBDIS bit (MEMCON<7>) controls the operation of the bus and related port functions. Clearing EBDIS enables the interface and disables the I/O functions of the ports, as well as any other functions multiplexed to those pins. Setting the bit enables the I/O ports and other functions but allows the interface to override everything else on the pins when an external memory operation is required. By default, the external bus is always enabled and disables all other I/O.

The operation of the EBDIS bit is also influenced by the program memory mode being used. This is discussed in more detail in **Section 7.4 "Program Memory Modes and the External Memory Bus"**.

The WAIT bits allow for the addition of wait states to external memory operations. The use of these bits is discussed in **Section 7.3 "Wait States"**.

The WM bits select the particular operating mode used when the bus is operating in 16-bit Data Width mode. These are discussed in more detail in **Section 7.5 "16-Bit Data Width Modes"**. These bits have no effect when an 8-bit Data Width mode is selected.

WM<1:0>: TBLWT Operation with 16-Bit Data Bus Width Select bits

1x = Word Write mode: TABLAT0 and TABLAT1 word output, $\overline{WRH}$ active when TABLAT1 written

01 = Byte Select mode: TABLAT data copied on both MSB and LSB; $\overline{WRH}$ and ($\overline{UB}$ or $\overline{LB}$) will activate

REGISTER 7-1: MEMCON: EXTERNAL MEMORY BUS CONTROL REGISTER

R/W-0	U-0	R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0
EBDIS	—	WAIT1	WAIT0	—	—	WM1	WM0
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 **EBDIS**: External Bus Disable bit
- 1 = External bus enabled when microcontroller accesses external memory; otherwise, all external bus drivers are mapped as I/O ports
- 0 = External bus always enabled, I/O ports are disabled

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

bit 5-4 **WAIT<1:0>:** Table Reads and Writes Bus Cycle Wait Count bits
- 11 = Table reads and writes will wait 0 TCY
- 10 = Table reads and writes will wait 1 TCY
- 01 = Table reads and writes will wait 2 TCY
- 00 = Table reads and writes will wait 3 TCY

bit 3-2 **Unimplemented**: Read as '0'

bit 1-0 **WM<1:0>:** TBLWT Operation with 16-Bit Data Bus Width Select bits
- 1 = Result was negative
- 0 = Result was positive

7.7 Operation in Power-Managed Modes

In alternate power-managed Run modes, the external bus continues to operate normally. If a clock source with a lower speed is selected, bus operations will run at that speed. In these cases, excessive access times for the external memory may result if wait states have been enabled and added to external memory operations. If operations in a lower power Run mode are anticipated, users should provide in their applications for adjusting memory access times at the lower clock speeds.

In Sleep and Idle modes, the microcontroller core does not need to access data; bus operations are suspended. The state of the external bus is frozen with the address/data pins and most of the control pins holding at the same state they were in when the mode was invoked. The only potential changes are the $\overline{CE}$, $\overline{LB}$ and $\overline{UB}$ pins which are held at logic high.

TABLE 7-3: SUMMARY OF REGISTERS ASSOCIATED WITH POWER-MANAGED MODES

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on page
MEMCON(1)	EBDIS	—	WAIT1	WAIT0	—	—	WM1	WM0	60
CONFIG3L(2)	WAIT	BW	ABW1	ABW0	—	—	PM1	PM0	302
CONFIG3H	MCLRE	—	—	—	—	LPT1OSC	ECCPMX(2)	CCP2MX	303

Legend: — = unimplemented, read as '0'. Shaded cells are not used by the External Memory Bus.

Note 1: This register is not implemented on 64-pin devices.

2: Unimplemented in PIC18F6527/6622/6627/6722 devices.

TABLE 11-4: SUMMARY OF REGISTERS ASSOCIATED WITH PORTB

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on page
PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	60
LATB	LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0	60
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	60
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	57
INTCON2	$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	RBIP	57
INTCON3	INT2IP	INT1IP	INT3IE	INT2IE	INT1IE	INT3IF	INT2IF	INT1IF	57

Legend: Shaded cells are not used by PORTB.

11.4 PORTD, TRISD and LATD Registers

PORTD is an 8-bit wide, bidirectional port. The corresponding Data Direction register is TRISD. Setting a TRISD bit (= 1) will make the corresponding PORTD pin an input (i.e., put the corresponding output driver in a high-impedance mode). Clearing a TRISD bit (= 0) will make the corresponding PORTD pin an output (i.e., put the contents of the output latch on the selected pin).

The Data Latch register (LATD) is also memory mapped. Read-modify-write operations on the LATD register read and write the latched output value for PORTD.

All pins on PORTD are implemented with Schmitt Trigger input buffers. Each pin is individually configurable as an input or output.

Note:	On a Power-on Reset, these pins are configured as digital inputs.

In 80-pin devices, PORTD is multiplexed with the system bus as part of the external memory interface. I/O port and other functions are only available when the interface is disabled by setting the EBDIS bit (MEMCON<7>). When the interface is enabled, PORTD is the low-order byte of the multiplexed address/data bus (AD<7:0>). The TRISD bits are also overridden.

PORTD can also be configured to function as an 8-bit wide parallel microprocessor port by setting the PSPMODE control bit (PSPCON<4>). In this mode, parallel port data takes priority over other digital I/O (but not the external memory interface). When the parallel port is active, the input buffers are TTL. For more information, refer to **Section 11.10 "Parallel Slave Port"**.

EXAMPLE 11-4: INITIALIZING PORTD

```
CLRF    PORTD   ; Initialize PORTD by
                ; clearing output
                ; data latches
CLRF    LATD    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   0CFh    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISD   ; Set RD<3:0> as inputs
                ; RD<5:4> as outputs
                ; RD<7:6> as inputs
```

TABLE 11-7: PORTD FUNCTIONS

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RD0/AD0/PSP0	RD0	0	O	DIG	LATD<0> data output.
		1	I	ST	PORTD<0> data input.
	AD0(1)	x	O	DIG	External memory interface, address/data bit 0 output. Takes priority over PSP and port data.
		x	I	TTL	External memory interface, data bit 0 input.
	PSP0	x	O	DIG	PSP read data output (LATD<0>). Takes priority over port data.
		x	I	TTL	PSP write data input.
RD1/AD1/PSP1	RD1	0	O	DIG	LATD<1> data output.
		1	I	ST	PORTD<1> data input.
	AD1(1)	x	O	DIG	External memory interface, address/data bit 1 output. Takes priority over PSP and port data.
		x	I	TTL	External memory interface, data bit 1 input.
	PSP1	x	O	DIG	PSP read data output (LATD<1>). Takes priority over port data.
		x	I	TTL	PSP write data input.
RD2/AD2/PSP2	RD2	0	O	DIG	LATD<2> data output.
		1	I	ST	PORTD<2> data input.
	AD2(1)	x	O	DIG	External memory interface, address/data bit 2 output. Takes priority over PSP and port data.
		x	I	TTL	External memory interface, data bit 2 input.
	PSP2	x	O	DIG	PSP read data output (LATD<2>). Takes priority over port data.
		x	I	TTL	PSP write data input.
RD3/AD3/PSP3	RD3	0	O	DIG	LATD<3> data output.
		1	I	ST	PORTD<3> data input.
	AD3(1)	x	O	DIG	External memory interface, address/data bit 3 output. Takes priority over PSP and port data.
		x	I	TTL	External memory interface, data bit 3 input.
	PSP3	x	O	DIG	PSP read data output (LATD<3>). Takes priority over port data.
		x	I	TTL	PSP write data input.
RD4/AD4/ PSP4/SDO2	RD4	0	O	DIG	LATD<4> data output.
		1	I	ST	PORTD<4> data input.
	AD4(1)	x	O	DIG	External memory interface, address/data bit 4 output. Takes priority over PSP, MSSP and port data.
		x	I	TTL	External memory interface, data bit 4 input.
	PSP4	x	O	DIG	PSP read data output (LATD<4>). Takes priority over port and PSP data.
		x	I	TTL	PSP write data input.
	SDO2	0	O	DIG	SPI data output (MSSP2 module). Takes priority over PSP and port data.

Legend: PWR = Power Supply, O = Output, I = Input, ANA = Analog Signal, DIG = Digital Output, ST = Schmitt Buffer Input, TTL = TTL Buffer Input, x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

Note 1: Implemented on 80-pin devices only.

TABLE 11-17: PORTJ FUNCTIONS

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RJ0/ALE	RJ0	0	O	DIG	LATJ<0> data output.
		1	I	ST	PORTJ<0> data input.
	ALE	x	O	DIG	External memory interface address latch enable control output. Takes priority over digital I/O.
RJ1/$\overline{OE}$	RJ1	0	O	DIG	LATJ<1> data output.
		1	I	ST	PORTJ<1> data input.
	$\overline{OE}$	x	O	DIG	External memory interface output enable control output. Takes priority over digital I/O.
RJ2/$\overline{WRL}$	RJ2	0	O	DIG	LATJ<2> data output.
		1	I	ST	PORTJ<2> data input.
	$\overline{WRL}$	x	O	DIG	External Memory Bus write low byte control. Takes priority over digital I/O.
RJ3/$\overline{WRH}$	RJ3	0	O	DIG	LATJ<3> data output.
		1	I	ST	PORTJ<3> data input.
	$\overline{WRH}$	x	O	DIG	External memory interface write high byte control output. Takes priority over digital I/O.
RJ4/BA0	RJ4	0	O	DIG	LATJ<4> data output.
		1	I	ST	PORTJ<4> data input.
	BA0	x	O	DIG	External memory interface byte address 0 control output. Takes priority over digital I/O.
RJ5/$\overline{CE}$	RJ5	0	O	DIG	LATJ<5> data output.
		1	I	ST	PORTJ<5> data input.
	$\overline{CE}$	x	O	DIG	External memory interface chip enable control output. Takes priority over digital I/O.
RJ6/$\overline{LB}$	RJ6	0	O	DIG	LATJ<6> data output.
		1	I	ST	PORTJ<6> data input.
	$\overline{LB}$	x	O	DIG	External memory interface lower byte enable control output. Takes priority over digital I/O.
RJ7/$\overline{UB}$	RJ7	0	O	DIG	LATJ<7> data output.
		1	I	ST	PORTJ<7> data input.
	$\overline{UB}$	x	O	DIG	External memory interface upper byte enable control output. Takes priority over digital I/O.

Legend: PWR = Power Supply, O = Output, I = Input, ANA = Analog Signal, DIG = Digital Output, ST = Schmitt Buffer Input, TTL = TTL Buffer Input, x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

TABLE 11-18: SUMMARY OF REGISTERS ASSOCIATED WITH PORTJ

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on page
PORTJ	RJ7	RJ6	RJ5	RJ4	RJ3	RJ2	RJ1	RJ0	60
LATJ	LATJ7	LATJ6	LATJ5	LATJ4	LATJ3	LATJ2	LATJ1	LATJ0	60
TRISJ	TRISJ7	TRISJ6	TRISJ5	TRISJ4	TRISJ3	TRISJ2	TRISJ1	TRISJ0	60

13.0 TIMER1 MODULE

The Timer1 timer/counter module incorporates these features:

- Software selectable operation as a 16-bit timer or counter
- Readable and writable 8-bit registers (TMR1H and TMR1L)
- Selectable clock source (internal or external) with device clock or Timer1 oscillator internal options
- Interrupt-on-overflow
- Reset on CCP Special Event Trigger
- Device clock status flag (T1RUN)

A simplified block diagram of the Timer1 module is shown in Figure 13-1. A block diagram of the module's operation in Read/Write mode is shown in Figure 13-2.

The module incorporates its own low-power oscillator to provide an additional clocking option. The Timer1 oscillator can also be used as a low-power clock source for the microcontroller in power-managed operation.

Timer1 can also be used to provide Real-Time Clock (RTC) functionality to applications with only a minimal addition of external components and code overhead.

Timer1 is controlled through the T1CON Control register (Register 13-1). It also contains the Timer1 Oscillator Enable bit (T1OSCEN). Timer1 can be enabled or disabled by setting or clearing control bit, TMR1ON (T1CON<0>).

REGISTER 13-1: T1CON: TIMER1 CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON
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 **RD16:** 16-Bit Read/Write Mode Enable bit
1 = Enables register read/write of Timer1 in one 16-bit operation
0 = Enables register read/write of Timer1 in two 8-bit operations

bit 6 **T1RUN:** Timer1 System Clock Status bit
1 = Device clock is derived from Timer1 oscillator
0 = Device clock is derived from another source

bit 5-4 **T1CKPS<1:0>:** Timer1 Input Clock Prescale Select bits
11 = 1:8 Prescale value
10 = 1:4 Prescale value
01 = 1:2 Prescale value
00 = 1:1 Prescale value

bit 3 **T1OSCEN:** Timer1 Oscillator Enable bit
1 = Timer1 oscillator is enabled
0 = Timer1 oscillator is shut off
The oscillator inverter and feedback resistor are turned off to eliminate power drain.

bit 2 **$\overline{\text{T1SYNC}}$:** Timer1 External Clock Input Synchronization Select bit
When TMR1CS = 1:
1 = Do not synchronize external clock input
0 = Synchronize external clock input
When TMR1CS = 0:
This bit is ignored. Timer1 uses the internal clock when TMR1CS = 0.

bit 1 **TMR1CS:** Timer1 Clock Source Select bit
1 = External clock from pin RC0/T1OSO/T13CKI (on the rising edge)
0 = Internal clock (FOSC/4)

bit 0 **TMR1ON:** Timer1 On bit
1 = Enables Timer1
0 = Stops Timer1

13.2 Timer1 16-bit Read/Write Mode

Timer1 can be configured for 16-bit reads and writes (see Figure 13-2). When the RD16 control bit (T1CON<7>) is set, the address for TMR1H is mapped to a buffer register for the high byte of Timer1. A read from TMR1L will load the contents of the high byte of Timer1 into the Timer1 high byte buffer. This provides the user with the ability to accurately read all 16 bits of Timer1 without having to determine whether a read of the high byte, followed by a read of the low byte, has become invalid due to a rollover between reads.

A write to the high byte of Timer1 must also take place through the TMR1H Buffer register. The Timer1 high byte is updated with the contents of TMR1H when a write occurs to TMR1L. This allows a user to write all 16 bits to both the high and low bytes of Timer1 at once.

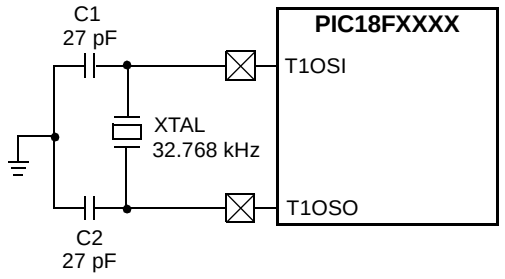
The high byte of Timer1 is not directly readable or writable in this mode. All reads and writes must take place through the Timer1 High Byte Buffer register. Writes to TMR1H do not clear the Timer1 prescaler. The prescaler is only cleared on writes to TMR1L.

13.3 Timer1 Oscillator

An on-chip crystal oscillator circuit is incorporated between pins T1OSI (input) and T1OSO (amplifier output). It is enabled by setting the Timer1 Oscillator Enable bit, T1OSCEN (T1CON<3>). The oscillator is a low-power circuit rated for 32 kHz crystals. It will continue to run during all power-managed modes. The circuit for a typical LP oscillator is shown in Figure 13-3. Table 13-1 shows the capacitor selection for the Timer1 oscillator.

The user must provide a software time delay to ensure proper start-up of the Timer1 oscillator.

FIGURE 13-3: EXTERNAL COMPONENTS FOR THE TIMER1 LP OSCILLATOR



Note: See the Notes with Table 13-1 for additional information about capacitor selection.

TABLE 13-1: CAPACITOR SELECTION FOR THE TIMER OSCILLATOR(2,3,4)

Osc Type	Freq	C1	C2
LP	32 kHz	27 pF(1)	27 pF(1)

Note 1: Microchip suggests these values as a starting point in validating the oscillator circuit.

2: Higher capacitance increases the stability of the oscillator but also increases the start-up time.

3: Since each resonator/crystal has its own characteristics, the user should consult the resonator/crystal manufacturer for appropriate values of external components.

4: Capacitor values are for design guidance only.

13.3.1 USING TIMER1 AS A CLOCK SOURCE

The Timer1 oscillator is also available as a clock source in power-managed modes. By setting the clock select bits, SCS<1:0> (OSCCON<1:0>), to '01', the device switches to SEC_RUN mode; both the CPU and peripherals are clocked from the Timer1 oscillator. If the IDLEN bit (OSCCON<7>) is cleared and a SLEEP instruction is executed, the device enters SEC_IDLE mode. Additional details are available in **Section 3.0 "Power-Managed Modes"**.

Whenever the Timer1 oscillator is providing the clock source, the Timer1 system clock status flag, T1RUN (T1CON<6>), is set. This can be used to determine the controller's current clocking mode. It can also indicate the clock source being currently used by the Fail-Safe Clock Monitor. If the Clock Monitor is enabled and the Timer1 oscillator fails while providing the clock, polling the T1RUN bit will indicate whether the clock is being provided by the Timer1 oscillator or another source.

13.3.2 LOW-POWER TIMER1 OPTION

The Timer1 oscillator can operate at two distinct levels of power consumption based on device configuration. When the LPT1OSC Configuration bit is set, the Timer1 oscillator operates in a low-power mode. When LPT1OSC is not set, Timer1 operates at a higher power level. Power consumption for a particular mode is relatively constant, regardless of the device's operating mode. The default Timer1 configuration is the higher power mode.

As the low-power Timer1 mode tends to be more sensitive to interference, high noise environments may cause some oscillator instability. The low-power option is, therefore, best suited for low noise applications where power conservation is an important design consideration.

16.0 TIMER4 MODULE

The Timer4 timer module has the following features:

- 8-bit Timer register (TMR4)
- 8-bit Period register (PR4)
- Readable and writable (both registers)
- Software programmable prescaler (1:1, 1:4, 1:16)
- Software programmable postscaler (1:1 to 1:16)
- Interrupt on TMR4 match of PR4

Timer4 has a control register shown in Register 16-1. Timer4 can be shut off by clearing control bit, TMR4ON (T4CON<2>), to minimize power consumption. The prescaler and postscaler selection of Timer4 are also controlled by this register. Figure 16-1 is a simplified block diagram of the Timer4 module.

16.1 Timer4 Operation

Timer4 can be used as the PWM time base for the PWM mode of the CCP modules. The TMR4 register is readable and writable and is cleared on any device Reset. The input clock (FOSC/4) has a prescale option of 1:1, 1:4 or 1:16, selected by control bits T4CKPS<1:0> (T4CON<1:0>). The match output of TMR4 goes through a 4-bit postscaler (which gives a 1:1 to 1:16 scaling inclusive) to generate a TMR4 interrupt, latched in flag bit, TMR4IF (PIR3<3>).

The prescaler and postscaler counters are cleared when any of the following occurs:

- a write to the TMR4 register
- a write to the T4CON register
- any device Reset (Power-on Reset, $\overline{\text{MCLR}}$ Reset, Watchdog Timer Reset or Brown-out Reset)

TMR4 is not cleared when T4CON is written.

REGISTER 16-1: T4CON: TIMER4 CONTROL REGISTER

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	T4OUTPS3	T4OUTPS2	T4OUTPS1	T4OUTPS0	TMR4ON	T4CKPS1	T4CKPS0
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 **T4OUTPS<3:0>:** Timer4 Output Postscale Select bits
0000 = 1:1 Postscale
0001 = 1:2 Postscale
•
•
•
1111 = 1:16 Postscale

bit 2 **TMR4ON**: Timer4 On bit
1 = Timer4 is on
0 = Timer4 is off

bit 1-0 **T4CKPS<1:0>**: Timer4 Clock Prescale Select bits
00 = Prescaler is 1
01 = Prescaler is 4
1x = Prescaler is 16

17.4 PWM Mode

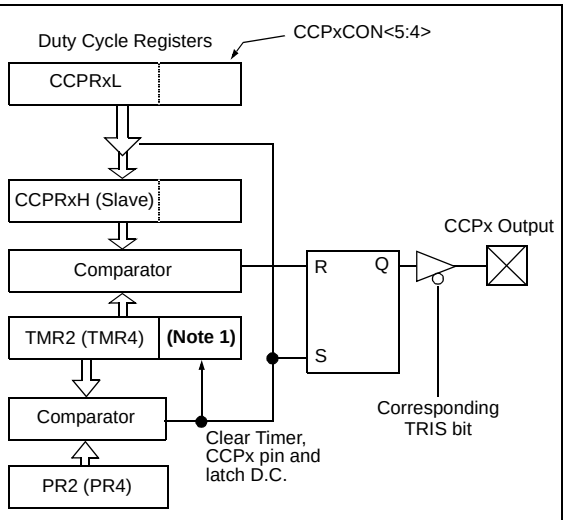
In Pulse-Width Modulation (PWM) mode, the CCPx pin produces up to a 10-bit resolution PWM output. Since the CCP4 and CCP5 pins are multiplexed with a PORTG data latch, the appropriate TRISG bit must be cleared to make the CCP4 or CCP5 pin an output.

Note:	Clearing the CCP4CON or CCP5CON register will force the RG3 or RG4 output latch (depending on device configuration) to the default low level. This is not the PORTG I/O data latch.

Figure 17-4 shows a simplified block diagram of the CCP module in PWM mode.

For a step-by-step procedure on how to set up a CCP module for PWM operation, see **Section 17.4.3 "Setup for PWM Operation"**.

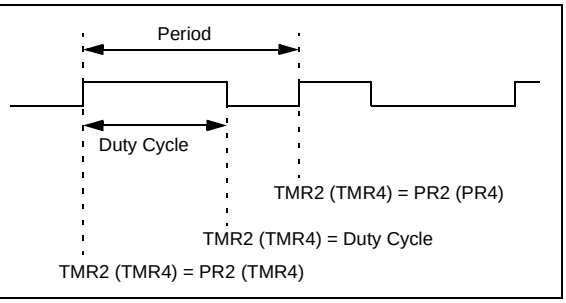
FIGURE 17-4: SIMPLIFIED PWM BLOCK DIAGRAM



Note 1: The 8-bit TMR2 or TMR4 value is concatenated with the 2-bit internal Q clock, or 2 bits of the prescaler, to create the 10-bit time base.

A PWM output (Figure 17-5) has a time base (period) and a time that the output stays high (duty cycle). The frequency of the PWM is the inverse of the period (1/period).

FIGURE 17-5: PWM OUTPUT



17.4.1 PWM PERIOD

The PWM period is specified by writing to the PR2 (PR4) register. The PWM period can be calculated using the following formula:

EQUATION 17-1:

$$\text{PWM Period} = [(PR2) + 1] \cdot 4 \cdot T_{OSC} \cdot (\text{TMR2 Prescale Value})$$

PWM frequency is defined as 1/[PWM period].

When TMR2 (TMR4) is equal to PR2 (PR4), the following three events occur on the next increment cycle:

- TMR2 (TMR4) is cleared
- The CCPx pin is set (exception: if PWM duty cycle = 0%, the CCPx pin will not be set)
- The PWM duty cycle is latched from CCPRxL into CCPRxH

Note:	The Timer2 and Timer 4 postscalers (see **Section 14.0 "Timer2 Module"** and **Section 16.0 "Timer4 Module"**) are not used in the determination of the PWM frequency. The postscaler could be used to have a servo update rate at a different frequency than the PWM output.

17.4.2 PWM DUTY CYCLE

The PWM duty cycle is specified by writing to the CCPRxL register and to the CCPxCON<5:4> bits. Up to 10-bit resolution is available. The CCPRxL contains the eight MSbs and the CCPxCON<5:4> contains the two LSbs. This 10-bit value is represented by CCPRxL:CCPxCON<5:4>. The following equation is used to calculate the PWM duty cycle in time:

EQUATION 17-2:

$$\text{PWM Duty Cycle} = (\text{CCPRxL:CCPxCON<5:4>}) \cdot T_{OSC} \cdot (\text{TMR2 Prescale Value})$$

CCPRxL and CCPxCON<5:4> can be written to at any time, but the duty cycle value is not latched into CCPRxH until after a match between PR2 (PR4) and TMR2 (TMR4) occurs (i.e., the period is complete). In PWM mode, CCPRxH is a read-only register.

19.0 MASTER SYNCHRONOUS SERIAL PORT (MSSP) MODULE

19.1 Master SSP (MSSP) Module Overview

The Master Synchronous Serial Port (MSSP) module is a serial interface, useful for communicating with other peripheral or microcontroller devices. These peripheral devices may be serial EEPROMs, shift registers, display drivers, A/D converters, etc. The MSSP module can operate in one of two modes:

- Serial Peripheral Interface (SPI)
- Inter-Integrated Circuit (I^2C™)
  - Full Master mode
  - Slave mode (with general address call)

The I^2C interface supports the following modes in hardware:

- Master mode
- Multi-Master mode
- Slave mode

All members of the PIC18F8722 family have two MSSP modules, designated as MSSP1 and MSSP2. Each module operates independently of the other.

Note:	Throughout this section, generic references to an MSSP module in any of its operating modes may be interpreted as being equally applicable to MSSP1 or MSSP2. Register names and module I/O signals use the generic designator 'x' to indicate the use of a numeral to distinguish a particular module when required. Control bit names are not individuated.

19.2 Control Registers

Each MSSP module has three associated control registers. These include a status register (SSPxSTAT) and two control registers (SSPxCON1 and SSPxCON2). The use of these registers and their individual Configuration bits differ significantly depending on whether the MSSP module is operated in SPI or I^2C mode.

Additional details are provided under the individual sections.

Note:	In devices with more than one MSSP module, it is very important to pay close attention to SSPCON register names. SSP1CON1 and SSP1CON2 control different operational aspects of the same module, while SSP1CON1 and SSP2CON1 control the same features for two different modules.

19.3 SPI Mode

The SPI mode allows 8 bits of data to be synchronously transmitted and received simultaneously. All four modes of SPI are supported. To accomplish communication, typically three pins are used:

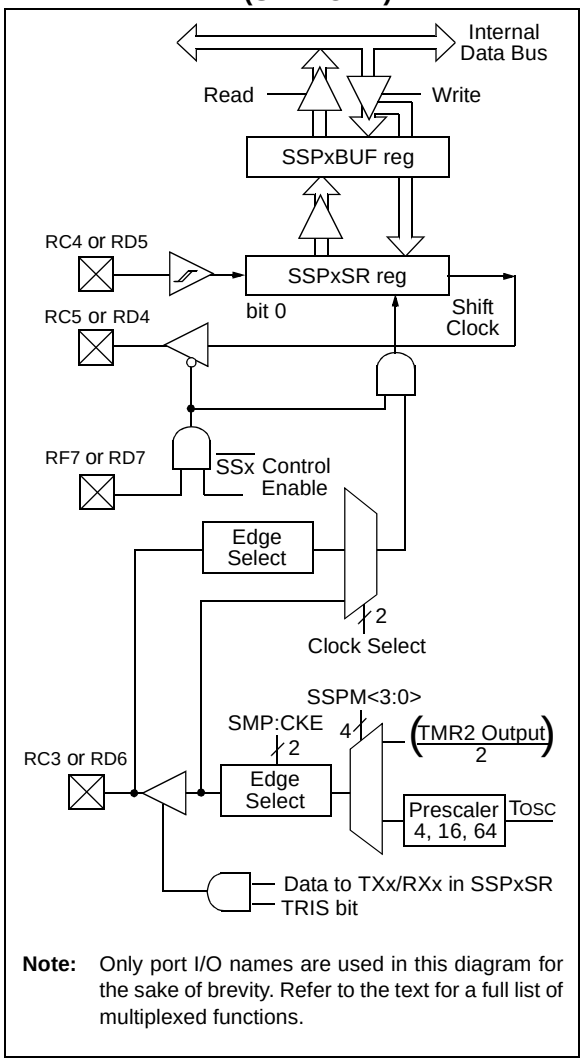
- Serial Data Out (SDOx) – RC5/SDO1 or RD4/SDO2
- Serial Data In (SDIx) – RC4/SDI1/SDA1 or RD5/SDI2/SDA2
- Serial Clock (SCKx) – RC3/SCK1/SCL1 or RD6/SCK2/SCL2

Additionally, a fourth pin may be used when in a Slave mode of operation:

- Slave Select ($\overline{SSx}$) – RF7/$\overline{SS1}$ or RD7/$\overline{SS2}$

Figure 19-1 shows the block diagram of the MSSP module when operating in SPI mode.

FIGURE 19-1: MSSP BLOCK DIAGRAM (SPI MODE)



Note: Only port I/O names are used in this diagram for the sake of brevity. Refer to the text for a full list of multiplexed functions.

21.0 10-BIT ANALOG-TO-DIGITAL CONVERTER (A/D) MODULE

The Analog-to-Digital (A/D) converter module has 12 inputs for the 64-pin devices and 16 for the 80-pin devices. This module allows conversion of an analog input signal to a corresponding 10-bit digital number.

The module has five registers:

- A/D Result High Register (ADRESH)
- A/D Result Low Register (ADRESL)
- A/D Control Register 0 (ADCON0)
- A/D Control Register 1 (ADCON1)
- A/D Control Register 2 (ADCON2)

The ADCON0 register, shown in Register 21-1, controls the operation of the A/D module. The ADCON1 register, shown in Register 21-2, configures the functions of the port pins. The ADCON2 register, shown in Register 21-3, configures the A/D clock source, programmed acquisition time and justification.

REGISTER 21-1: ADCON0: A/D CONTROL REGISTER

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	CHS3(1)	CHS2(1)	CHS1(1)	CHS0(1)	GO/$\overline{\text{DONE}}$	ADON
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 **Unimplemented:** Read as ‘0’

bit 5-2 **CHS<3:0>** Analog Channel Select bits(1)
0000 = Channel 0 (AN0)
0001 = Channel 1 (AN1)
0010 = Channel 2 (AN2)
0011 = Channel 3 (AN3)
0100 = Channel 4 (AN4)
0101 = Channel 5 (AN5)
0110 = Channel 6 (AN6)
0111 = Channel 7 (AN7)
1000 = Channel 8 (AN8)
1001 = Channel 9 (AN9)
1010 = Channel 10 (AN10)
1011 = Channel 11 (AN11)
1100 = Channel 12 (AN12)(1)
1101 = Channel 13 (AN13)(1)
1110 = Channel 14 (AN14)(1)
1111 = Channel 15 (AN15)(1)

bit 1 **GO/$\overline{\text{DONE}}$:** A/D Conversion Status bit
When ADON = 1:
1 = A/D conversion in progress
0 = A/D Idle

bit 0 **ADON:** A/D On bit
1 = A/D converter module is enabled
0 = A/D converter module is disabled

Note 1: These channels are not implemented on 64-pin devices.

21.4 Operation in Power-Managed Modes

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

If the A/D is expected to operate while the device is in a power-managed mode, the ACQT<2:0> and ADCS<2:0> bits in ADCON2 should be updated in accordance with the clock source to be used in that mode. After entering the mode, an A/D acquisition or conversion may be started. Once started, the device should continue to be clocked by the same clock source until the conversion has been completed.

If desired, the device may be placed into the corresponding Idle mode during the conversion. If the device clock frequency is less than 1 MHz, the A/D RC clock source should be selected.

Operation in the Sleep mode requires the A/D FRC clock to be selected. If bits ACQT<2:0> are set to '000' and a conversion is started, the conversion will be delayed one instruction cycle to allow execution of the SLEEP instruction and entry to Sleep mode. The IDLEN bit (OSCCON<7>) must have already been cleared prior to starting the conversion.

21.5 Configuring Analog Port Pins

The ADCON1, TRISA, TRISF and TRISH registers all configure the A/D port pins. The port pins needed as analog inputs must have their corresponding TRIS bits set (input). If the TRIS bit is cleared (output), the digital output level (V_{OH} or V_{OL}) will be converted.

The A/D operation is independent of the state of the CHS<3:0> bits and the TRIS bits.

Note 1:	When reading the Port register, all pins configured as analog input channels will read as cleared (a low level). Pins configured as digital inputs will convert as analog inputs. Analog levels on a digitally configured input will be accurately converted.
2:	Analog levels on any pin defined as a digital input may cause the digital input buffer to consume current out of the device's specification limits.

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 two-cycle instruction.

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

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 26.2.3 "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 two-cycle instruction.

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

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 26.2.3 "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)

LFSR **Load FSR**

Syntax: LFSR f, k

Operands: $0 \le f \le 2$
$0 \le k \le 4095$

Operation: $k \rightarrow FSRf$

Status Affected: None

Encoding:

1110 1111	1110 0000	00ff k7kkk	k11kkk kkkk

Description: The 12-bit literal 'k' is loaded into the file select register pointed to by 'f'.

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k' MSB	Process Data	Write literal 'k' MSB to FSRfH
Decode	Read literal 'k' LSB	Process Data	Write literal 'k' to FSRfL

Example: LFSR 2, 3ABh

After Instruction
FSR2H = 03h
FSR2L = ABh

MOVF **Move f**

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

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

Operation: $f \rightarrow dest$

Status Affected: N, Z

Encoding:

0101	00da	ffff	ffff

Description: The contents of register 'f' are moved to a destination dependent upon the status of 'd'. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f' (default). Location 'f' can be anywhere in the 256-byte bank.

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 26.2.3 "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 W

Example: MOVF REG, 0, 0

Before Instruction
REG = 22h
W = FFh
After Instruction
REG = 22h
W = 22h

MOVLW **Move Literal to W**

Syntax: MOVLW k

Operands: $0 \le k \le 255$

Operation: $k \rightarrow W$

Status Affected: None

Encoding:

0000	1110	kkkk	kkkk

Description: The eight-bit literal ‘k’ is loaded into W.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal ‘k’	Process Data	Write to W

Example: MOVLW 5Ah

After Instruction
W = 5Ah

MOVWF **Move W to f**

Syntax: MOVWF f {,a}

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

Operation: $(W) \rightarrow f$

Status Affected: None

Encoding:

0110	111a	ffff	ffff

Description: Move data from W to register ‘f’. Location ‘f’ can be anywhere in the 256-byte bank.

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 26.2.3 “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: MOVWF REG, 0

Before Instruction
W = 4Fh
REG = FFh
After Instruction
W = 4Fh
REG = 4Fh

TSTFSZ **Test f, Skip if 0**

Syntax: TSTFSZ f {,a}

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

Operation: skip if f = 0

Status Affected: None

Encoding:

0110	011a	ffff	ffff

Description: If 'f' = 0, the next instruction fetched during the current instruction execution is discarded and a NOP is executed, making this a two-cycle instruction.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank (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 26.2.3 "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    TSTFSZ  CNT, 1
NZERO   :
ZERO    :
```
Before Instruction
PC = Address (HERE)
After Instruction
If CNT = 00h,
PC = Address (ZERO)
If CNT ≠ 00h,
PC = Address (NZERO)

XORLW **Exclusive OR Literal with W**

Syntax: XORLW k

Operands: $0 \le k \le 255$

Operation: (W) .XOR. $k \rightarrow W$

Status Affected: N, Z

Encoding:

0000	1010	kkkk	kkkk

Description: The contents of W are XORed with the 8-bit literal 'k'. The result is placed in W.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to W

Example:
```
XORLW   0AFh
```
Before Instruction
W = B5h
After Instruction
W = 1Ah

MOVSS **Move Indexed to Indexed**

Syntax: MOVSS [z_s], [z_d]

Operands: $0 \le z_s \le 127$
$0 \le z_d \le 127$

Operation: ((FSR2) + z_s) → ((FSR2) + z_d)

Status Affected: None

Encoding:

1st word (source)	1110	1011	1zzz	zzzzs
2nd word (dest.)	1111	xxxx	xzzz	zzzzd

Description: The contents of the source register are moved to the destination register. The addresses of the source and destination registers are determined by adding the 7-bit literal offsets 'z_s' or 'z_d', respectively, to the value of FSR2. Both registers can be located anywhere in the 4096-byte data memory space (000h to FFFh).

The MOVSS instruction cannot use the PCL, TOSU, TOSH or TOSL as the destination register.

If the resultant source address points to an Indirect Addressing register, the value returned will be 00h. If the resultant destination address points to an Indirect Addressing register, the instruction will execute as a NOP.

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Determine source addr	Determine source addr	Read source reg
Decode	Determine dest addr	Determine dest addr	Write to dest reg

Example:
```
MOVSS  [05h], [06h]
```
Before Instruction
FSR2 = 80h
Contents of 85h = 33h
Contents of 86h = 11h
After Instruction
FSR2 = 80h
Contents of 85h = 33h
Contents of 86h = 33h

PUSHL **Store Literal at FSR2, Decrement FSR2**

Syntax: PUSHL k

Operands: $0 \le k \le 255$

Operation: k → (FSR2),
FSR2 – 1→ FSR2

Status Affected: None

Encoding:

1111	1010	kkkk	kkkk

Description: The 8-bit literal 'k' is written to the data memory address specified by FSR2. FSR2 is decremented by 1 after the operation.

This instruction allows users to push values onto a software stack.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read 'k'	Process data	Write to destination

Example:
```
PUSHL  08h
```
Before Instruction
FSR2H:FSR2L = 01ECh
Memory (01ECh) = 00h

After Instruction
FSR2H:FSR2L = 01EBh
Memory (01ECh) = 08h

APPENDIX C: CONVERSION CONSIDERATIONS

This appendix discusses the considerations for converting from previous versions of a device to the ones listed in this data sheet. Typically, these changes are due to the differences in the process technology used. An example of this type of conversion is from a PIC16C74A to a PIC16C74B.

Not Applicable

APPENDIX D: MIGRATION FROM BASELINE TO ENHANCED DEVICES

This section discusses how to migrate from a Baseline device (i.e., PIC16C5X) to an Enhanced MCU device (i.e., PIC18FXXX).

The following are the list of modifications over the PIC16C5X microcontroller family:

Not Currently Available