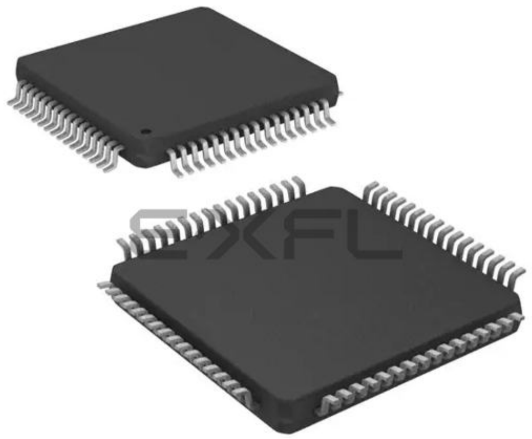


Details	
Product Status	Active
Core Processor	PIC
Core Size	8-Bit
Speed	40MHz
Connectivity	I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, HLVD, POR, PWM, WDT
Number of I/O	54
Program Memory Size	64KB (32K x 16)
Program Memory Type	FLASH
EEPROM Size	1K x 8
RAM Size	3936 x 8
Voltage - Supply (Vcc/Vdd)	2V ~ 5.5V
Data Converters	A/D 12x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	64-TQFP
Supplier Device Package	64-TQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18lf6622-i-pt

4.5 Device Reset Timers

The PIC18F8722 family of devices incorporates three separate on-chip timers that help regulate the Power-on Reset process. Their main function is to ensure that the device clock is stable before code is executed. These timers are:

- Power-up Timer (PWRT)
- Oscillator Start-up Timer (OST)
- PLL Lock Time-out

4.5.1 POWER-UP TIMER (PWRT)

The Power-up Timer (PWRT) of the PIC18F8722 family of devices is an 11-bit counter which uses the INTRC source as the clock input. While the PWRT is counting, the device is held in Reset.

The power-up time delay depends on the INTRC clock and will vary from chip-to-chip due to temperature and process variation. See DC parameter 33 in Table 28-12 for details.

The PWRT is enabled by clearing the $\overline{\text{PWRTEN}}$ Configuration bit.

4.5.2 OSCILLATOR START-UP TIMER (OST)

The Oscillator Start-up Timer (OST) provides a 1024 oscillator cycle (from OSC1 input) delay after the PWRT delay is over (parameter 33, Table 28-12). This ensures that the crystal oscillator or resonator has started and stabilized.

The OST time-out is invoked only for XT, LP, HS and HSPLL modes and only on Power-on Reset, or on exit from most power-managed modes.

4.5.3 PLL LOCK TIME-OUT

With the PLL enabled in its PLL mode, the time-out sequence following a Power-on Reset is slightly different from other oscillator modes. A separate timer is used to provide a fixed time-out that is sufficient for the PLL to lock to the main oscillator frequency. This PLL lock time-out (T_{PLL}) is typically 2 ms and follows the oscillator start-up time-out.

4.5.4 TIME-OUT SEQUENCE

On power-up, the time-out sequence is as follows:

1. After the POR pulse has cleared, PWRT time-out is invoked (if enabled).
2. Then, the OST is activated.

The total time-out will vary based on oscillator configuration and the status of the PWRT. Figure 4-3, Figure 4-4, Figure 4-5, Figure 4-6 and Figure 4-7 all depict time-out sequences on power-up, with the Power-up Timer enabled and the device operating in HS Oscillator mode. Figures 4-3 through 4-6 also apply to devices operating in XT or LP modes. For devices in RC mode and with the PWRT disabled, on the other hand, there will be no time-out at all.

Since the time-outs occur from the POR pulse, if $\overline{\text{MCLR}}$ is kept low long enough, all time-outs will expire. Bringing $\overline{\text{MCLR}}$ high will begin execution immediately (Figure 4-5). This is useful for testing purposes or to synchronize more than one PIC18F8722 family device operating in parallel.

TABLE 4-2: TIME-OUT IN VARIOUS SITUATIONS

Oscillator Configuration	Power-up(2) and Brown-out		Exit from Power-Managed Mode
	$\overline{\text{PWRTEN}}$ = 0	$\overline{\text{PWRTEN}}$ = 1	
HSPLL	T_{PWRT}(1) + 1024 T_{OSC} + T_{PLL}(2)	1024 T_{OSC} + T_{PLL}(2)	1024 T_{OSC} + T_{PLL}(2)
HS, XT, LP	T_{PWRT}(1) + 1024 T_{OSC}	1024 T_{OSC}	1024 T_{OSC}
EC, ECIO	T_{PWRT}(1)	—	—
RC, RCIO	T_{PWRT}(1)	—	—
INTIO1, INTIO2	T_{PWRT}(1)	—	—

Note 1: See parameter 33, Table 28-12.
2: 2 ms is the nominal time required for the PLL to lock.

FIGURE 5-1: PROGRAM MEMORY MAP AND STACK FOR PIC18F8722 FAMILY DEVICES

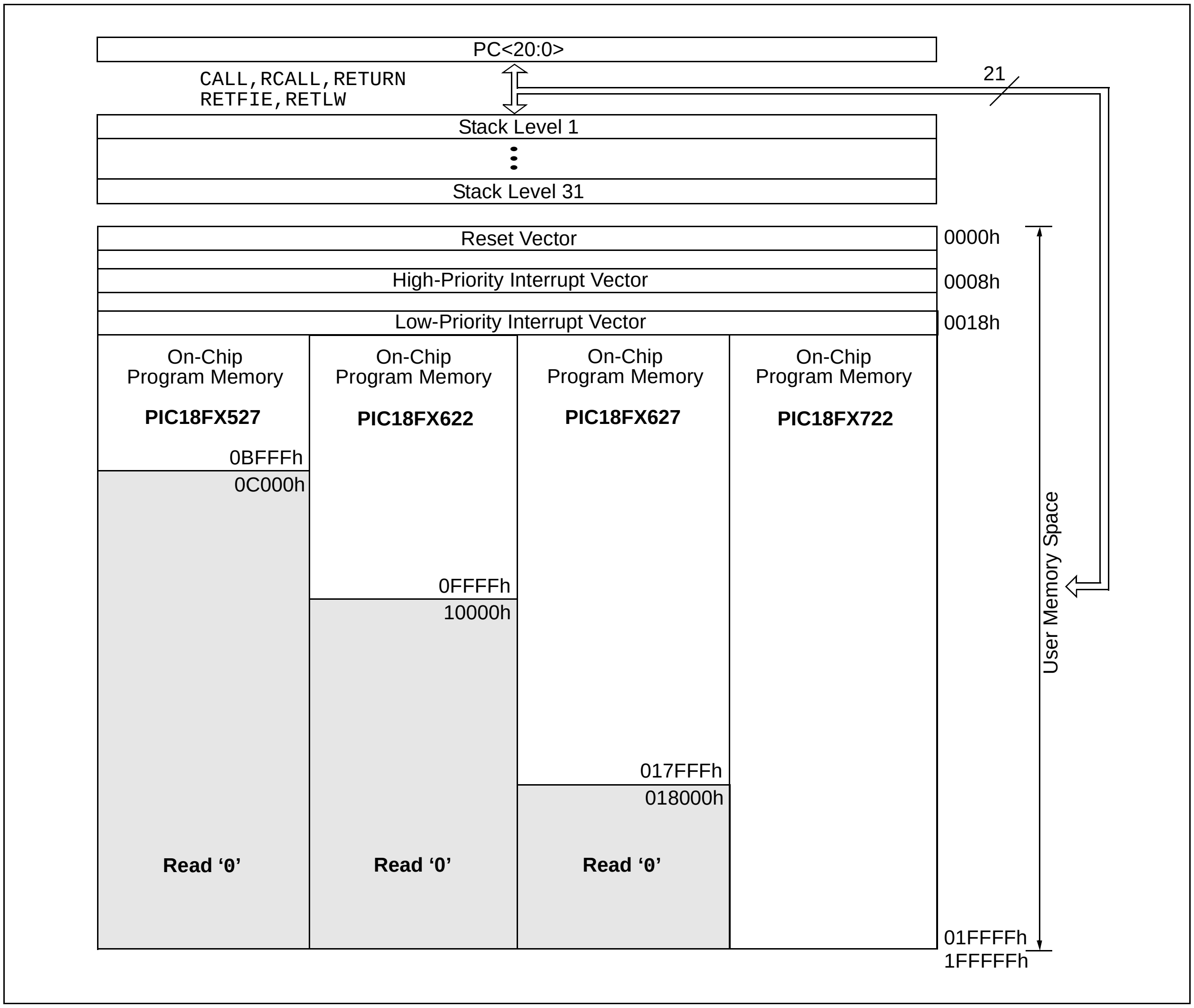


TABLE 5-1: MEMORY ACCESS FOR PIC18F8527/8622/8627/8722 PROGRAM MEMORY MODES

Operating Mode	Internal Program Memory			External Program Memory		
	Execution From	Table Read From	Table Write To	Execution From	Table Read From	Table Write To
Microprocessor	No Access	No Access	No Access	Yes	Yes	Yes
Microprocessor w/ Boot Block	Yes	Yes	Yes	Yes	Yes	Yes
Microcontroller	Yes	Yes	Yes	No Access	No Access	No Access
Extended Microcontroller	Yes	Yes	Yes	Yes	Yes	Yes

5.3.4 SPECIAL FUNCTION REGISTERS

The Special Function Registers (SFRs) are registers used by the CPU and peripheral modules for controlling the desired operation of the device. These registers are implemented as static RAM. SFRs start at the top of data memory (FFFh) and extend downward to occupy the top half of Bank 15 (F60h to FFFh). A list of these registers is given in Table 5-2 and Table 5-3.

The SFRs can be classified into two sets: those associated with the "core" device functionality (ALU, Resets and interrupts) and those related to the peripheral functions. The Reset and interrupt registers are described in their respective chapters, while the ALU's STATUS register is described later in this section. Registers related to the operation of a peripheral feature are described in the chapter for that peripheral.

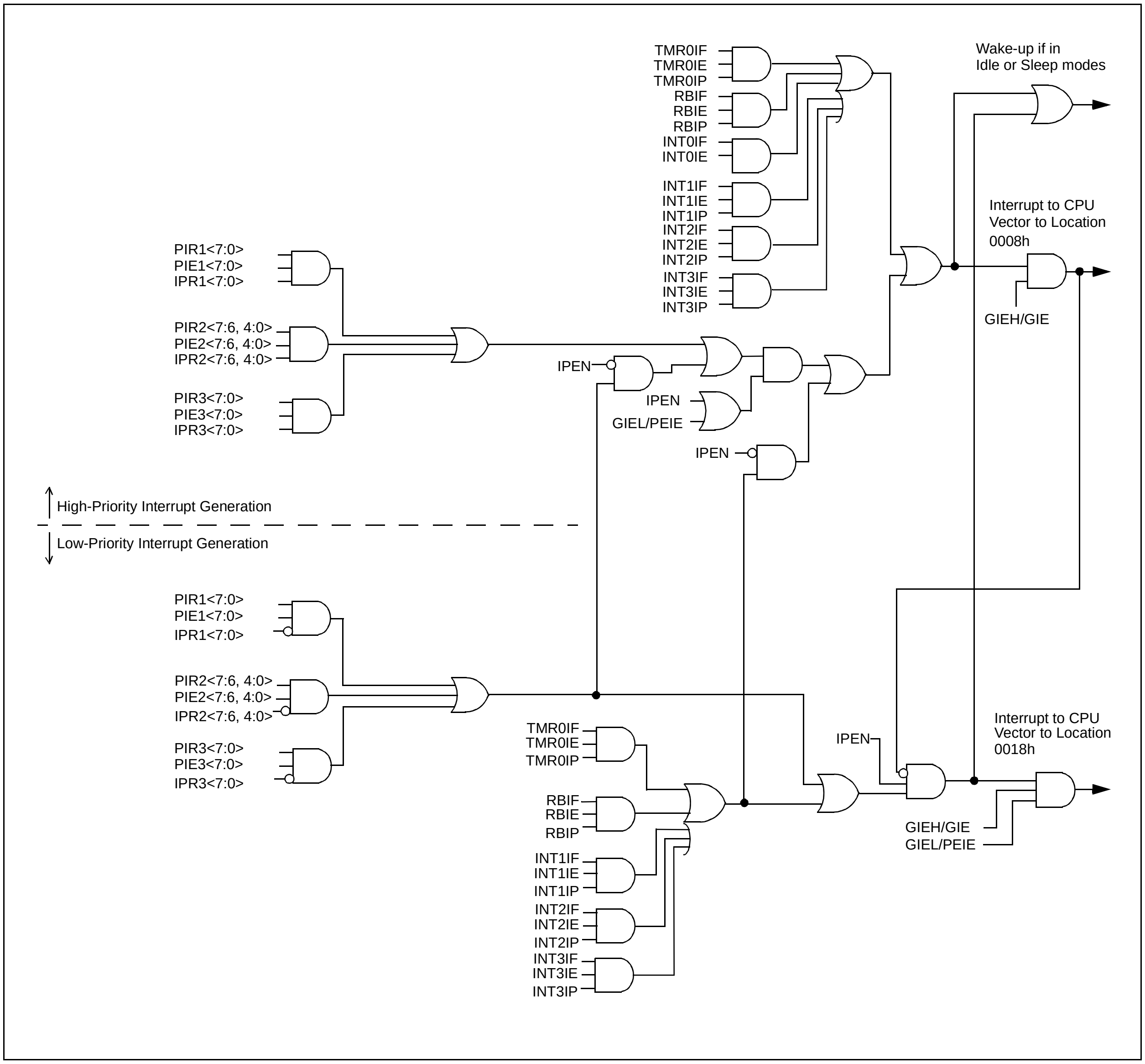
The SFRs are typically distributed among the peripherals whose functions they control. Unused SFR locations are unimplemented and read as '0's.

TABLE 5-2: SPECIAL FUNCTION REGISTER MAP FOR THE PIC18F8722 FAMILY OF DEVICES

Address	Name	Address	Name	Address	Name	Address	Name	Address	Name
FFFh	TOSU	FDFh	INDF2(1)	FBFh	CCPR1H	F9Fh	IPR1	F7Fh	SPBRGH1
FFEh	TOSH	FDEh	POSTINC2(1)	FBEh	CCPR1L	F9Eh	PIR1	F7Eh	BAUDCON1
FFDh	TOSL	FDDh	POSTDEC2(1)	FBDh	CCP1CON	F9Dh	PIE1	F7Dh	SPBRGH2
FFCh	STKPTR	FDCh	PREINC2(1)	FBCh	CCPR2H	F9Ch	MEMCON	F7Ch	BAUDCON2
FFBh	PCLATU	FDBh	PLUSW2(1)	FBBh	CCPR2L	F9Bh	OSCTUNE	F7Bh	—(2)
FFAh	PCLATH	FDAh	FSR2H	FBAh	CCP2CON	F9Ah	TRISJ(3)	F7Ah	—(2)
FF9h	PCL	FD9h	FSR2L	FB9h	CCPR3H	F99h	TRISH(3)	F79h	ECCP1DEL
FF8h	TBLPTRU	FD8h	STATUS	FB8h	CCPR3L	F98h	TRISG	F78h	TMR4
FF7h	TBLPTRH	FD7h	TMR0H	FB7h	CCP3CON	F97h	TRISF	F77h	PR4
FF6h	TBLPTRL	FD6h	TMR0L	FB6h	ECCP1AS	F96h	TRISE	F76h	T4CON
FF5h	TABLAT	FD5h	T0CON	FB5h	CVRCON	F95h	TRISD	F75h	CCPR4H
FF4h	PRODH	FD4h	—(2)	FB4h	CMCON	F94h	TRISC	F74h	CCPR4L
FF3h	PRODL	FD3h	OSCCON	FB3h	TMR3H	F93h	TRISB	F73h	CCP4CON
FF2h	INTCON	FD2h	HLVDCON	FB2h	TMR3L	F92h	TRISA	F72h	CCPR5H
FF1h	INTCON2	FD1h	WDTCON	FB1h	T3CON	F91h	LATJ(3)	F71h	CCPR5L
FF0h	INTCON3	FD0h	RCON	FB0h	PSPCON	F90h	LATH(3)	F70h	CCP5CON
FEFh	INDF0(1)	FCFh	TMR1H	FAFh	SPBRG1	F8Fh	LATG	F6Fh	SPBRG2
FEEh	POSTINC0(1)	FCEh	TMR1L	FAEh	RCREG1	F8Eh	LATF	F6Eh	RCREG2
FEDh	POSTDEC0(1)	FCDh	T1CON	FADh	TXREG1	F8Dh	LATE	F6Dh	TXREG2
FECh	PREINC0(1)	FCCh	TMR2	FACh	TXSTA1	F8Ch	LATD	F6Ch	TXSTA2
FEBh	PLUSW0(1)	FCBh	PR2	FABh	RCSTA1	F8Bh	LATC	F6Bh	RCSTA2
FEAh	FSR0H	FCAh	T2CON	FAAh	EEADRH	F8Ah	LATB	F6Ah	ECCP3AS
FE9h	FSR0L	FC9h	SSP1BUF	FA9h	EEADR	F89h	LATA	F69h	ECCP3DEL
FE8h	WREG	FC8h	SSP1ADD	FA8h	EEDATA	F88h	PORTJ(3)	F68h	ECCP2AS
FE7h	INDF1(1)	FC7h	SSP1STAT	FA7h	EECON2(1)	F87h	PORTH(3)	F67h	ECCP2DEL
FE6h	POSTINC1(1)	FC6h	SSP1CON1	FA6h	EECON1	F86h	PORTG	F66h	SSP2BUF
FE5h	POSTDEC1(1)	FC5h	SSP1CON2	FA5h	IPR3	F85h	PORTF	F65h	SSP2ADD
FE4h	PREINC1(1)	FC4h	ADRESH	FA4h	PIR3	F84h	PORTE	F64h	SSP2STAT
FE3h	PLUSW1(1)	FC3h	ADRESL	FA3h	PIE3	F83h	PORTD	F63h	SSP2CON1
FE2h	FSR1H	FC2h	ADCON0	FA2h	IPR2	F82h	PORTC	F62h	SSP2CON2
FE1h	FSR1L	FC1h	ADCON1	FA1h	PIR2	F81h	PORTB	F61h	—(2)
FE0h	BSR	FC0h	ADCON2	FA0h	PIE2	F80h	PORTA	F60h	—(2)

Note 1: This is not a physical register.
2: Unimplemented registers are read as '0'.
3: This register is not available on 64-pin devices.

FIGURE 10-1: PIC18F8722 FAMILY INTERRUPT LOGIC



10.1 INTCON Registers

The INTCON registers are readable and writable registers which contain various enable, priority and flag bits.

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

REGISTER 10-1: INTCON: INTERRUPT 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-x
GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF(1)
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 **GIE/GIEH:** Global Interrupt Enable bit

When IPEN = 0:
1 = Enables all unmasked interrupts
0 = Disables all interrupts

When IPEN = 1:
1 = Enables all high-priority interrupts
0 = Disables all interrupts

bit 6 **PEIE/GIEL:** Peripheral Interrupt Enable bit

When IPEN = 0:
1 = Enables all unmasked peripheral interrupts
0 = Disables all peripheral interrupts

When IPEN = 1:
1 = Enables all low-priority peripheral interrupts
0 = Disables all low-priority peripheral interrupts

bit 5 **TMR0IE:** TMR0 Overflow Interrupt Enable bit
1 = Enables the TMR0 overflow interrupt
0 = Disables the TMR0 overflow interrupt

bit 4 **INT0IE:** INT0 External Interrupt Enable bit
1 = Enables the INT0 external interrupt
0 = Disables the INT0 external interrupt

bit 3 **RBIE:** RB Port Change Interrupt Enable bit
1 = Enables the RB port change interrupt
0 = Disables the RB port change interrupt

bit 2 **TMR0IF:** TMR0 Overflow Interrupt Flag bit
1 = TMR0 register has overflowed (must be cleared in software)
0 = TMR0 register did not overflow

bit 1 **INT0IF:** INT0 External Interrupt Flag bit
1 = The INT0 external interrupt occurred (must be cleared in software)
0 = The INT0 external interrupt did not occur

bit 0 **RBIF:** RB Port Change Interrupt Flag bit(1)
1 = At least one of the RB7:RB4 pins changed state (must be cleared in software)
0 = None of the RB7:RB4 pins have changed state

Note 1: A mismatch condition will continue to set this bit. Reading PORTB will end the mismatch condition and allow the bit to be cleared.

TABLE 11-7: PORTD FUNCTIONS (CONTINUED)

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RD5/AD5/PSP5/SDI2/SDA2	RD5	0	O	DIG	LATD<5> data output.
		1	I	ST	PORTD<5> data input.
	AD5(1)	x	O	DIG	External memory interface, address/data bit 5 output. Takes priority over PSP, MSSP and port data.
		x	I	TTL	External memory interface, data bit 5 input.
	PSP5	x	O	DIG	PSP read data output (LATD<5>). Takes priority over port data.
		x	I	TTL	PSP write data input.
	SDI2	1	I	ST	SPI data input (MSSP2 module).
	SDA2	1	O	DIG	I^2C™ data output (MSSP2 module). Takes priority over PSP and port data.
		1	I	I^2C/SMB	I^2C data input (MSSP2 module); input type depends on module setting.
RD6/AD6/PSP6/SCK2/SCL2	RD6	0	O	DIG	LATD<6> data output.
		1	I	ST	PORTD<6> data input.
	AD6(1)	x	O	DIG-3	External memory interface, address/data bit 6 output. Takes priority over PSP, MSSP and port data.
		x	I	TTL	External memory interface, data bit 6 input.
	PSP6	x	O	DIG	PSP read data output (LATD<6>). Takes priority over port data.
		x	I	TTL	PSP write data input.
	SCK2	0	O	DIG	SPI clock output (MSSP2 module). Takes priority over PSP and port data.
		1	I	ST	SPI clock input (MSSP2 module).
	SCL2	0	O	DIG	I^2C clock output (MSSP2 module). Takes priority over PSP and port data.
		1	I	I^2C/SMB	I^2C clock input (MSSP2 module); input type depends on module setting.
RD7/AD7/PSP7/$\overline{SS2}$	RD7	0	O	DIG	LATD<7> data output.
		1	I	ST	PORTD<7> data input.
	AD7(1)	x	O	DIG	External memory interface, address/data bit 7 output. Takes priority over PSP and port data.
		x	I	TTL	External memory interface, data bit 7 input.
	PSP7	x	O	DIG	PSP read data output (LATD<7>). Takes priority over port data.
		x	I	TTL	PSP write data input.
	$\overline{SS2}$	1	I	TTL	Slave select input for SSP (MSSP2 module).

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 11-8: SUMMARY OF REGISTERS ASSOCIATED WITH PORTD

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on page
PORTD	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0	60
LATD	LATD7	LATD6	LATD5	LATD4	LATD3	LATD2	LATD1	LATD0	60
TRISD	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0	60

11.6 PORTF, LATF and TRISF Registers

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

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

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

PORTF is multiplexed with several analog peripheral functions, including the A/D converter and comparator inputs, as well as the comparator outputs. Pins RF1 through RF2 may be used as comparator inputs or outputs by setting the appropriate bits in the CMCON register. To use RF<6:0:> as digital inputs, it is necessary to turn off the A/D inputs.

Note 1: On a Power-on Reset, the RF<6:0> pins are configured as analog inputs and read as '0'.

2: To configure PORTF as digital I/O, set the ADCON1 register.

EXAMPLE 11-6: INITIALIZING PORTF

```
CLRF    PORTF   ; Initialize PORTF by
                ; clearing output
                ; data latches
CLRF    LATF    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   0x0F    ;
MOVWF   ADCON1  ; Set PORTF as digital I/O
MOVLW   0xCF    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISF   ; Set RF3:RF0 as inputs
                ; RF5:RF4 as outputs
                ; RF7:RF6 as inputs
```

12.0 TIMER0 MODULE

The Timer0 module incorporates the following features:

- Software selectable operation as a timer or counter in both 8-bit or 16-bit modes
- Readable and writable registers
- Dedicated 8-bit, software programmable prescaler
- Selectable clock source (internal or external)
- Edge select for external clock
- Interrupt-on-overflow

The T0CON register (Register 12-1) controls all aspects of the module’s operation, including the prescale selection. It is both readable and writable.

A simplified block diagram of the Timer0 module in 8-bit mode is shown in Figure 12-1. Figure 12-2 shows a simplified block diagram of the Timer0 module in 16-bit mode.

REGISTER 12-1: T0CON: TIMER0 CONTROL REGISTER

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0
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 **TMR0ON:** Timer0 On/Off Control bit
1 = Enables Timer0
0 = Stops Timer0

bit 6 **T08BIT**: Timer0 8-bit/16-bit Control bit
1 = Timer0 is configured as an 8-bit timer/counter
0 = Timer0 is configured as a 16-bit timer/counter

bit 5 **T0CS**: Timer0 Clock Source Select bit
1 = Transition on T0CKI pin
0 = Internal instruction cycle clock (CLKO)

bit 4 **T0SE**: Timer0 Source Edge Select bit
1 = Increment on high-to-low transition on T0CKI pin
0 = Increment on low-to-high transition on T0CKI pin

bit 3 **PSA**: Timer0 Prescaler Assignment bit
1 = TImer0 prescaler is NOT assigned. Timer0 clock input bypasses prescaler.
0 = Timer0 prescaler is assigned. Timer0 clock input comes from prescaler output.

bit 2-0 **T0PS<2:0>**: Timer0 Prescaler Select bits
111 = 1:256 Prescale value
110 = 1:128 Prescale value
101 = 1:64 Prescale value
100 = 1:32 Prescale value
011 = 1:16 Prescale value
010 = 1:8 Prescale value
001 = 1:4 Prescale value
000 = 1:2 Prescale value

13.1 Timer1 Operation

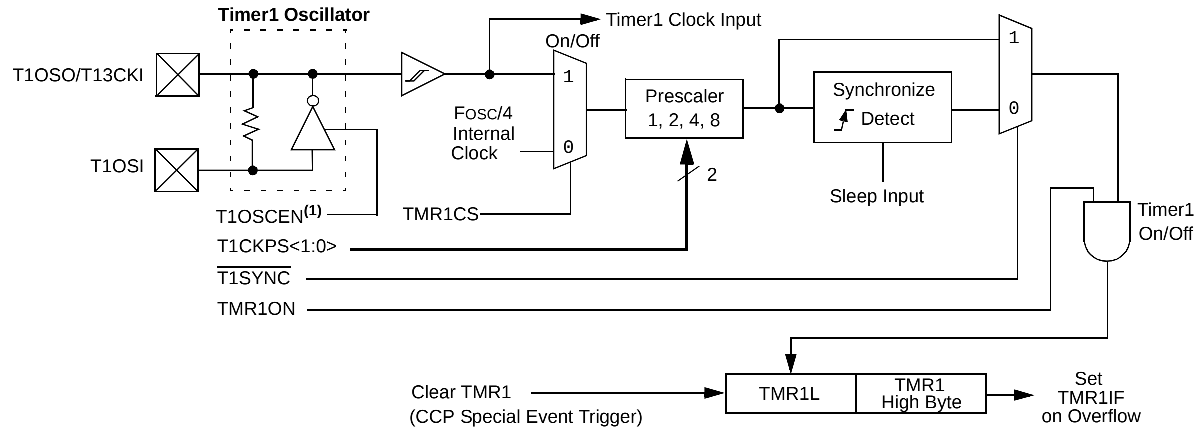
Timer1 can operate in one of these modes:

- Timer
- Synchronous Counter
- Asynchronous Counter

The operating mode is determined by the clock select bit, TMR1CS (T1CON<1>). When TMR1CS is cleared (= 0), Timer1 increments on every internal instruction cycle (FOSC/4). When the bit is set, Timer1 increments on every rising edge of the Timer1 external clock input or the Timer1 oscillator, if enabled.

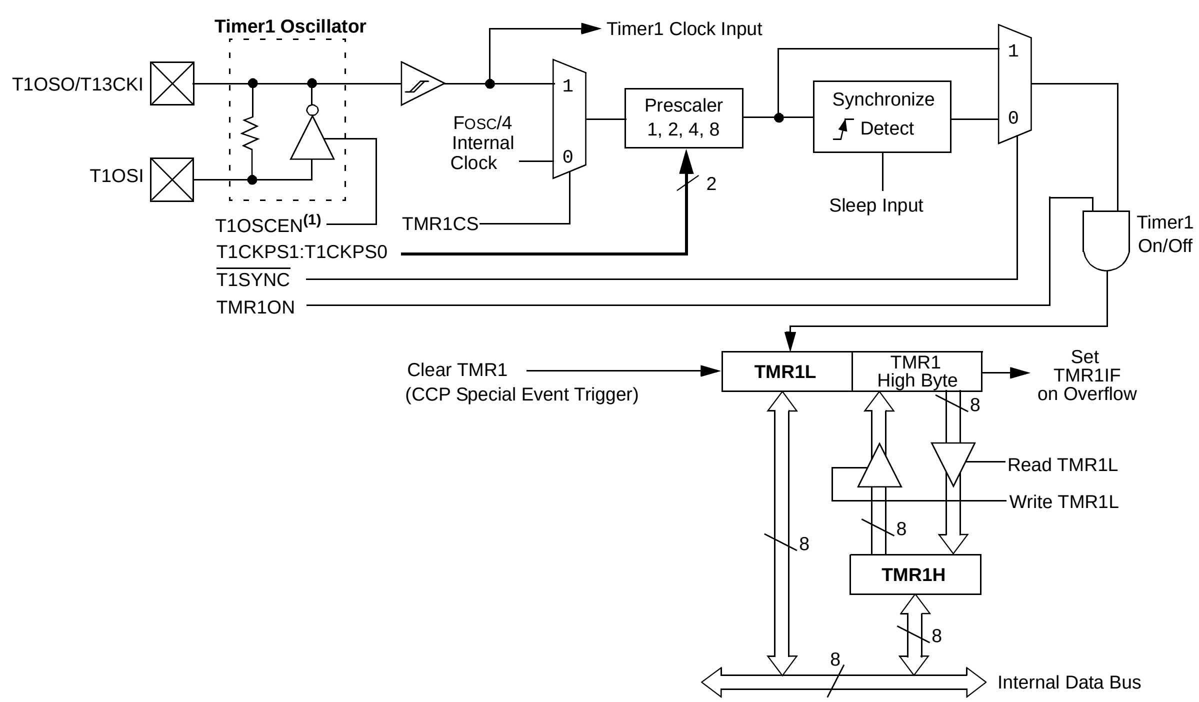
When Timer1 is enabled, the RC1/T1OSI and RC0/T1OSO/T13CKI pins become inputs. This means the values of TRISC<1:0> are ignored and the pins are read as '0'.

FIGURE 13-1: TIMER1 BLOCK DIAGRAM



Note 1: When enable bit, T1OSCEN, is cleared, the inverter and feedback resistor are turned off to eliminate power drain.

FIGURE 13-2: TIMER1 BLOCK DIAGRAM (16-BIT READ/WRITE MODE)



Note 1: When enable bit, T1OSCEN, is cleared, the inverter and feedback resistor are turned off to eliminate power drain.

13.3.3 TIMER1 OSCILLATOR LAYOUT CONSIDERATIONS

The Timer1 oscillator circuit draws very little power during operation. Due to the low-power nature of the oscillator, it may also be sensitive to rapidly changing signals in close proximity.

The oscillator circuit, shown in Figure 13-3, should be located as close as possible to the microcontroller. There should be no circuits passing within the oscillator circuit boundaries other than VSS or VDD.

If a high-speed circuit must be located near the Timer1 oscillator, a grounded guard ring around the oscillator circuit may be helpful when used on a single-sided PCB or in addition to a ground plane.

13.4 Timer1 Interrupt

The TMR1 register pair (TMR1H:TMR1L) increments from 0000h to FFFFh and rolls over to 0000h. The Timer1 interrupt, if enabled, is generated on overflow, which is latched in interrupt flag bit, TMR1IF (PIR1<0>). This interrupt can be enabled or disabled by setting or clearing the Timer1 Interrupt Enable bit, TMR1IE (PIE1<0>).

13.5 Resetting Timer1 Using the CCP Special Event Trigger

If any of the CCP modules are configured to use Timer1 and generate a Special Event Trigger in Compare mode (CCPxM<3:0>, this signal will reset Timer1. The trigger from the ECCP2 module will also start an A/D conversion if the A/D module is enabled (see **Section 17.3.4 "Special Event Trigger"** for more information).

The module must be configured as either a timer or a synchronous counter to take advantage of this feature. When used this way, the CCPRH:CCPRL register pair effectively becomes a period register for Timer1.

If Timer1 is running in Asynchronous Counter mode, this Reset operation may not work.

In the event that a write to Timer1 coincides with a Special Event Trigger, the write operation will take precedence.

Note:	The Special Event Triggers from the CCPx module will not set the TMR1IF interrupt flag bit (PIR1<0>).

13.6 Using Timer1 as a Real-Time Clock

Adding an external LP oscillator to Timer1 (such as the one described in **Section 13.3 "Timer1 Oscillator"** above) gives users the option to include RTC functionality to their applications. This is accomplished with an inexpensive watch crystal to provide an accurate time base and several lines of application code to calculate the time. When operating in Sleep mode and using a battery or supercapacitor as a power source, it can completely eliminate the need for a separate RTC device and battery backup.

The application code routine, `RTCisr`, shown in Example 13-1, demonstrates a simple method to increment a counter at one-second intervals using an Interrupt Service Routine. Incrementing the TMR1 register pair to overflow triggers the interrupt and calls the routine, which increments the seconds counter by one; additional counters for minutes and hours are incremented as the previous counter overflow.

Since the register pair is 16 bits wide, counting up to overflow the register directly from a 32.768 kHz clock would take 2 seconds. To force the overflow at the required one-second intervals, it is necessary to preload it; the simplest method is to set the MSb of TMR1H with a BSF instruction. Note that the TMR1L register is never preloaded or altered; doing so may introduce cumulative error over many cycles.

For this method to be accurate, Timer1 must operate in Asynchronous mode and the Timer1 overflow interrupt must be enabled (PIE1<0> = `1`), as shown in the routine, `RTCinit`. The Timer1 oscillator must also be enabled and running at all times.

EXAMPLE 13-1: IMPLEMENTING A REAL-TIME CLOCK USING A TIMER1 INTERRUPT SERVICE

```
RTCinit
           MOVLW   80h            ; Preload TMR1 register pair
           MOVWF   TMR1H          ; for 1 second overflow
           CLRF    TMR1L
           MOVLW   b'00001111'    ; Configure for external clock,
           MOVWF   T1CON          ; Asynchronous operation, external oscillator
           CLRF    secs           ; Initialize timekeeping registers
           CLRF    mins           ;
           MOVLW   .12
           MOVWF   hours
           BSF     PIE1, TMR1IE   ; Enable Timer1 interrupt
           RETURN
RTCisr
           BSF     TMR1H, 7       ; Preload for 1 sec overflow
           BCF     PIR1, TMR1IF   ; Clear interrupt flag
           INCF    secs, F        ; Increment seconds
           MOVLW   .59            ; 60 seconds elapsed?
           CPFSGT  secs
           RETURN                 ; No, done
           CLRF    secs           ; Clear seconds
           INCF    mins, F        ; Increment minutes
           MOVLW   .59            ; 60 minutes elapsed?
           CPFSGT  mins
           RETURN                 ; No, done
           CLRF    mins           ; clear minutes
           INCF    hours, F       ; Increment hours
           MOVLW   .23            ; 24 hours elapsed?
           CPFSGT  hours
           RETURN                 ; No, done
           CLRF    hours          ; Reset hours
           RETURN                 ; Done
```

TABLE 13-2: 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	Reset Values on page
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	57
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	60
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	60
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	60
TMR1L	Timer1 Register Low Byte								58
TMR1H	Timer1 Register High Byte								58
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON	58

Legend: Shaded cells are not used by the Timer1 module.

15.1 Timer3 Operation

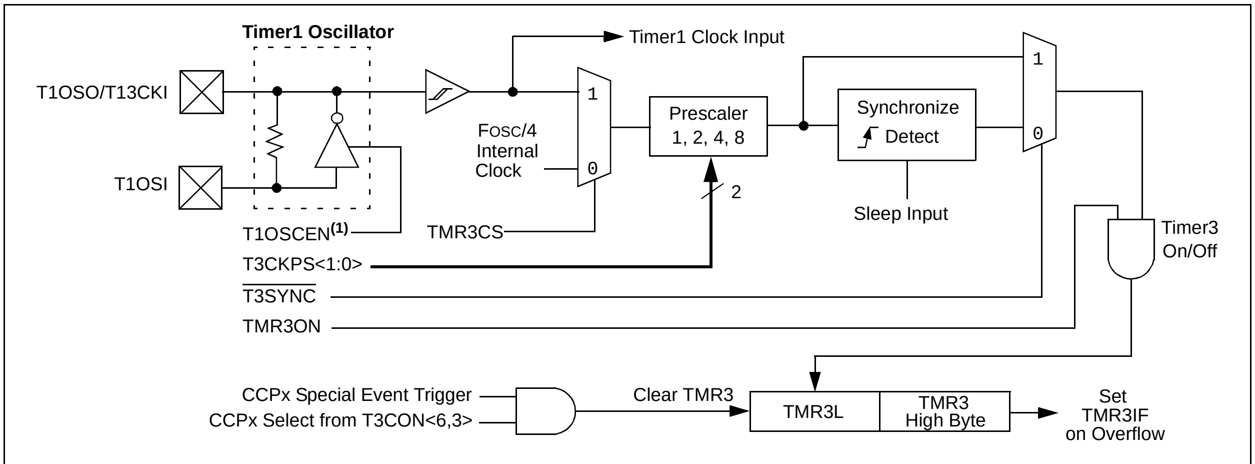
Timer3 can operate in one of three modes:

- Timer
- Synchronous Counter
- Asynchronous Counter

The operating mode is determined by the clock select bit, TMR3CS (T3CON<1>). When TMR3CS is cleared (= `0`), Timer3 increments on every internal instruction cycle (FOSC/4). When the bit is set, Timer3 increments on every rising edge of the Timer1 external clock input or the Timer1 oscillator, if enabled.

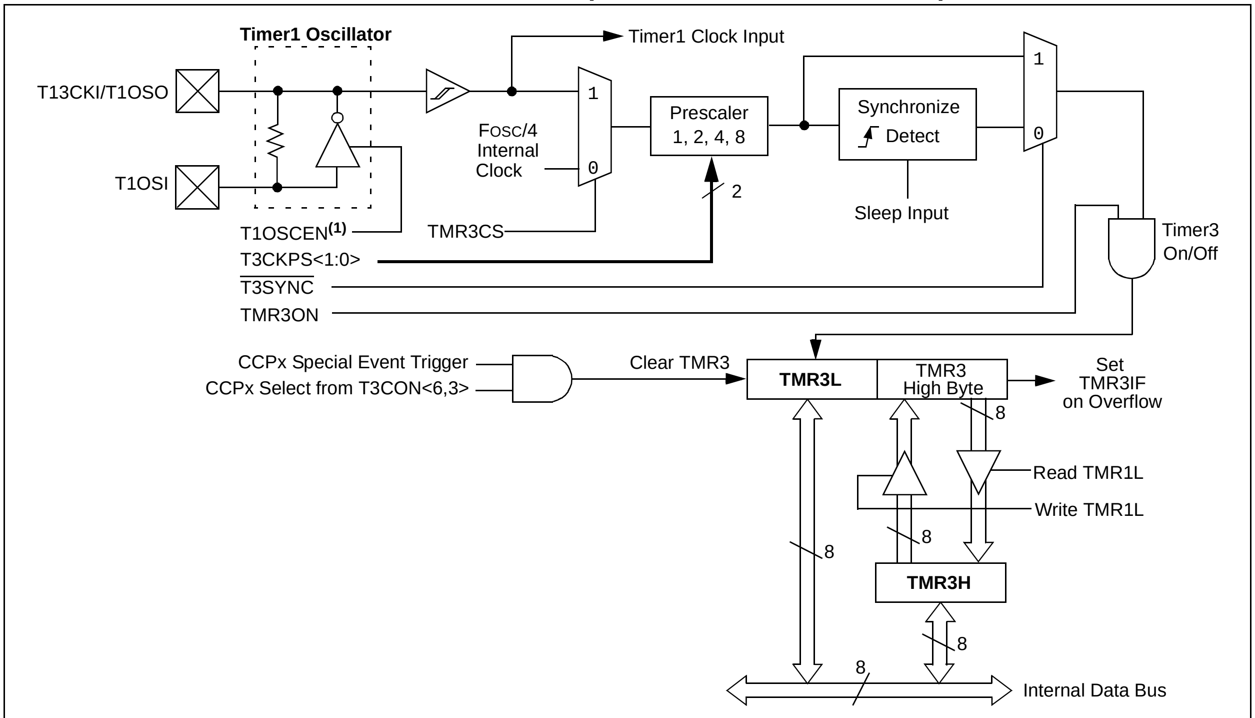
As with Timer1, the RC1/T1OSI and RC0/T1OSO/T13CKI pins become inputs when the Timer1 oscillator is enabled. This means the values of TRISC<1:0> are ignored and the pins are read as '`0`'.

FIGURE 15-1: TIMER3 BLOCK DIAGRAM



Note 1: When enable bit, T1OSCEN, is cleared, the inverter and feedback resistor are turned off to eliminate power drain.

FIGURE 15-2: TIMER3 BLOCK DIAGRAM (16-BIT READ/WRITE MODE)



Note 1: When enable bit, T1OSCEN, is cleared, the inverter and feedback resistor are turned off to eliminate power drain.

FIGURE 20-3: EUSART TRANSMIT BLOCK DIAGRAM

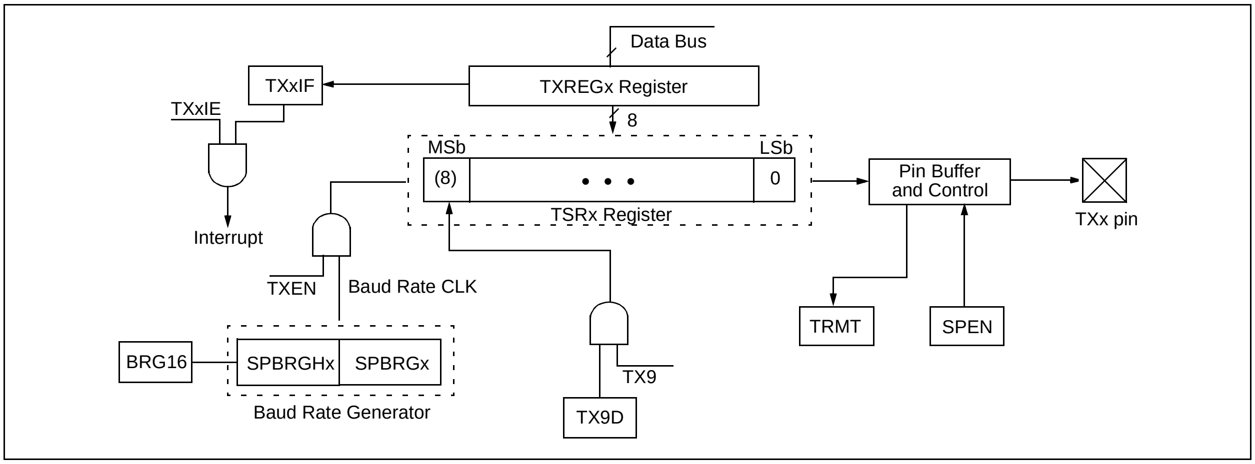


FIGURE 20-4: ASYNCHRONOUS TRANSMISSION

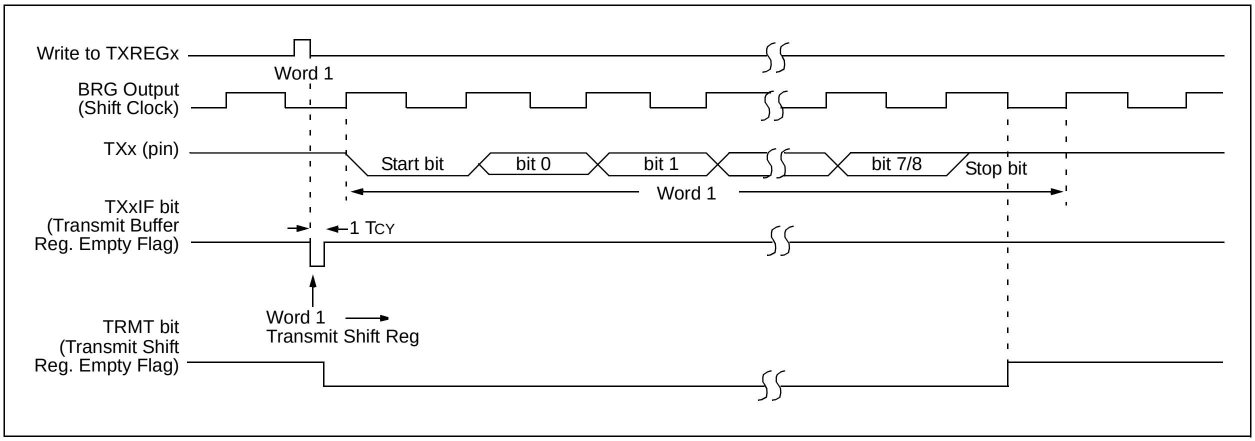
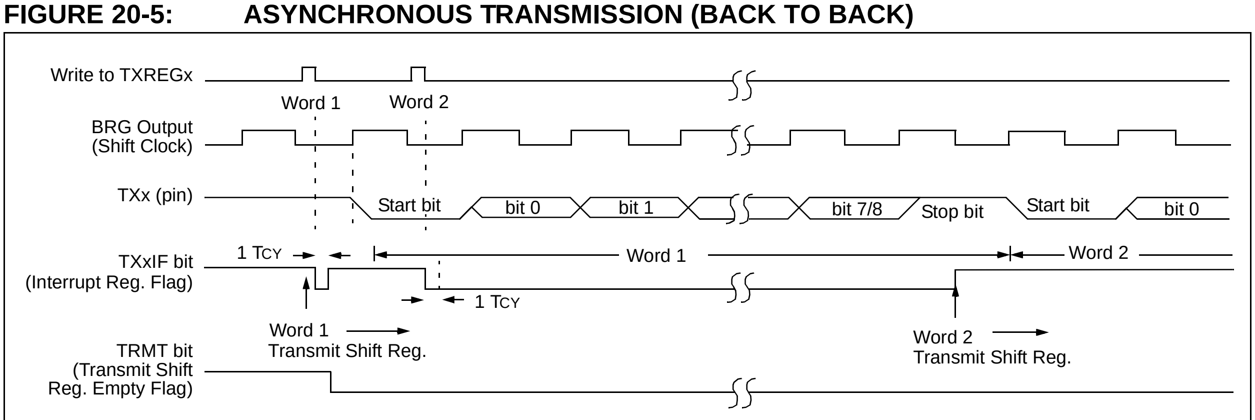


FIGURE 20-5: ASYNCHRONOUS TRANSMISSION (BACK TO BACK)



Note: This timing diagram shows two consecutive transmissions.

NOTES:

REGISTER 25-11: CONFIG7L: CONFIGURATION REGISTER 7 LOW (BYTE ADDRESS 30000Ch)

R/C-1	R/C-1	R/C-1	R/C-1	R/C-1	R/C-1	R/C-1	R/C-1
EBTR7(1)	EBTR6(1)	EBTR5(2)	EBTR4(2)	EBTR3(3)	EBTR2	EBTR1	EBTR0
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 **EBTR7:** Table Read Protection bit(1)
1 = Block 7 (01C000-01FFFFh) not protected from table reads executed in other blocks
0 = Block 7 (01C000-01FFFFh) protected from table reads executed in other blocks

bit 6 **EBTR6:** Table Read Protection bit(1)
1 = Block 6 (018000-01BFFFh) not protected from table reads executed in other blocks
0 = Block 6 (018000-01BFFFh) protected from table reads executed in other blocks

bit 5 **EBTR5:** Table Read Protection bit(2)
1 = Block 5 (014000-017FFFh) not protected from table reads executed in other blocks
0 = Block 5 (014000-017FFFh) protected from table reads executed in other blocks

bit 4 **EBTR4:** Table Read Protection bit(2)
1 = Block 4 (010000-013FFFh) not protected from table reads executed in other blocks
0 = Block 4 (010000-013FFFh) protected from table reads executed in other blocks

bit 3 **EBTR3:** Table Read Protection bit(3)
1 = Block 3 (00C000-00FFFFh) not protected from table reads executed in other blocks
0 = Block 3 (00C000-00FFFFh) protected from table reads executed in other blocks

bit 2 **EBTR2**: Table Read Protection bit
1 = Block 2 (008000-00BFFFh) not protected from table reads executed in other blocks
0 = Block 2 (008000-00BFFFh) protected from table reads executed in other blocks

bit 1 **EBTR1:** Table Read Protection bit
1 = Block 1 (004000-007FFFh) not protected from table reads executed in other blocks
0 = Block 1 (004000-007FFFh) protected from table reads executed in other blocks

bit 0 **EBTR0:** Table Read Protection bit
1 = Block 0 (000800, 001000 or 002000(4)-003FFFh) not protected from table reads executed in other blocks
0 = Block 0 (000800, 001000 or 002000(4)-003FFFh) protected from table reads executed in other blocks

Note 1: Unimplemented in PIC18F6527/6622/6627/8527/8622/8627 devices; maintain this bit set.
2: Unimplemented in PIC18F6527/6622/8527/8622 devices; maintain this bit set.
3: Unimplemented in PIC18F6527/8527 devices; maintain this bit set.
4: Unimplemented in PIC18F6527/8527 devices; maintain this bit set.

ADDWFC	ADD W and Carry bit to f
Syntax:	ADDWFC f {,d {,a}}
Operands:	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
Operation:	(W) + (f) + (C) → dest
Status Affected:	N,OV, C, DC, Z
Encoding:	`0010` `00da` `ffff` `ffff`
Description:	Add W, the Carry flag and data memory location ‘f’. If ‘d’ is ‘0’, the result is placed in W. If ‘d’ is ‘1’, the result is placed in data memory location ‘f’. 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

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register ‘f’	Process Data	Write to destination

Example: `ADDWFC REG, 0, 1`

Before Instruction
- Carry bit = 1
- REG = 02h
- W = 4Dh

After Instruction
- Carry bit = 0
- REG = 02h
- W = 50h

ANDLW	AND Literal with W
Syntax:	ANDLW k
Operands:	0 ≤ k ≤ 255
Operation:	(W) .AND. k → W
Status Affected:	N, Z
Encoding:	`0000` `1011` `kkkk` `kkkk`
Description:	The contents of W are ANDed 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: `ANDLW 05Fh`

Before Instruction
- W = A3h

After Instruction
- W = 03h

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 1110 00ff k11kkk` `1111 0000 k7kkk 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

SUBLW **Subtract W from literal**

Syntax:	SUBLW k
Operands:	$0 \le k \le 255$
Operation:	$k - (W) \rightarrow W$
Status Affected:	N, OV, C, DC, Z
Encoding:	`0000` `1000` `kkkk` `kkkk`
Description:	W is subtracted from the eight-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 1: `SUBLW 02h`

Before Instruction
W = 01h
C = ?
After Instruction
W = 01h
C = 1 ; result is positive
Z = 0
N = 0

Example 2: `SUBLW 02h`

Before Instruction
W = 02h
C = ?
After Instruction
W = 00h
C = 1 ; result is zero
Z = 1
N = 0

Example 3: `SUBLW 02h`

Before Instruction
W = 03h
C = ?
After Instruction
W = FFh ; (2's complement)
C = 0 ; result is negative
Z = 0
N = 1

SUBWF **Subtract W from f**

Syntax:	SUBWF f {,d {,a}}
Operands:	$0 \le f \le 255$ $d \in [0,1]$ $a \in [0,1]$
Operation:	$(f) - (W) \rightarrow dest$
Status Affected:	N, OV, C, DC, Z
Encoding:	`0101` `11da` `ffff` `ffff`
Description:	Subtract W from register 'f' (2's complement method). If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register 'f' (default). 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 to destination

Example 1: `SUBWF REG, 1, 0`

Before Instruction
REG = 3
W = 2
C = ?
After Instruction
REG = 1
W = 2
C = 1 ; result is positive
Z = 0
N = 0

Example 2: `SUBWF REG, 0, 0`

Before Instruction
REG = 2
W = 2
C = ?
After Instruction
REG = 2
W = 0
C = 1 ; result is zero
Z = 1
N = 0

Example 3: `SUBWF REG, 1, 0`

Before Instruction
REG = 1
W = 2
C = ?
After Instruction
REG = FFh ;(2's complement)
W = 2
C = 0 ; result is negative
Z = 0
N = 1

THE MICROCHIP WEB SITE

Microchip provides online support via our WWW site at www.microchip.com. This web site is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the web site contains the following information:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQ), technical support requests, online discussion groups, Microchip consultant program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

CUSTOMER CHANGE NOTIFICATION SERVICE

Microchip's customer notification service helps keep customers current on Microchip products. Subscribers will receive e-mail notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, access the Microchip web site at www.microchip.com, click on Customer Change Notification and follow the registration instructions.

CUSTOMER SUPPORT

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support
- Development Systems Information Line

Customers should contact their distributor, representative or field application engineer (FAE) for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in the back of this document.

Technical support is available through the web site at: http://support.microchip.com