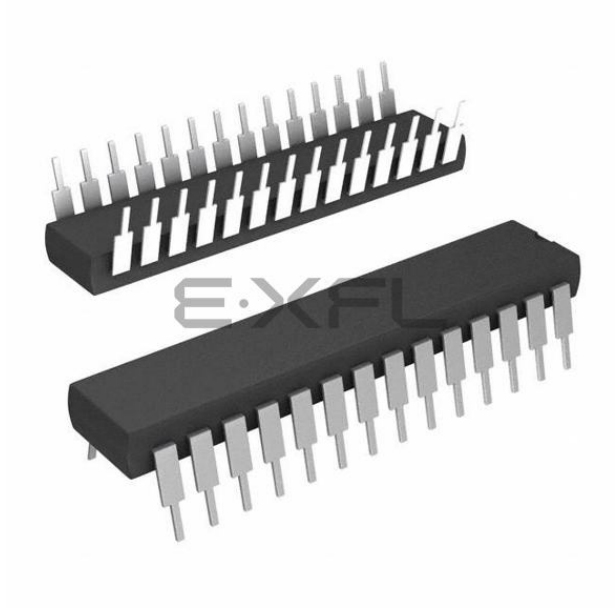


Welcome to E-XFL.COM

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"

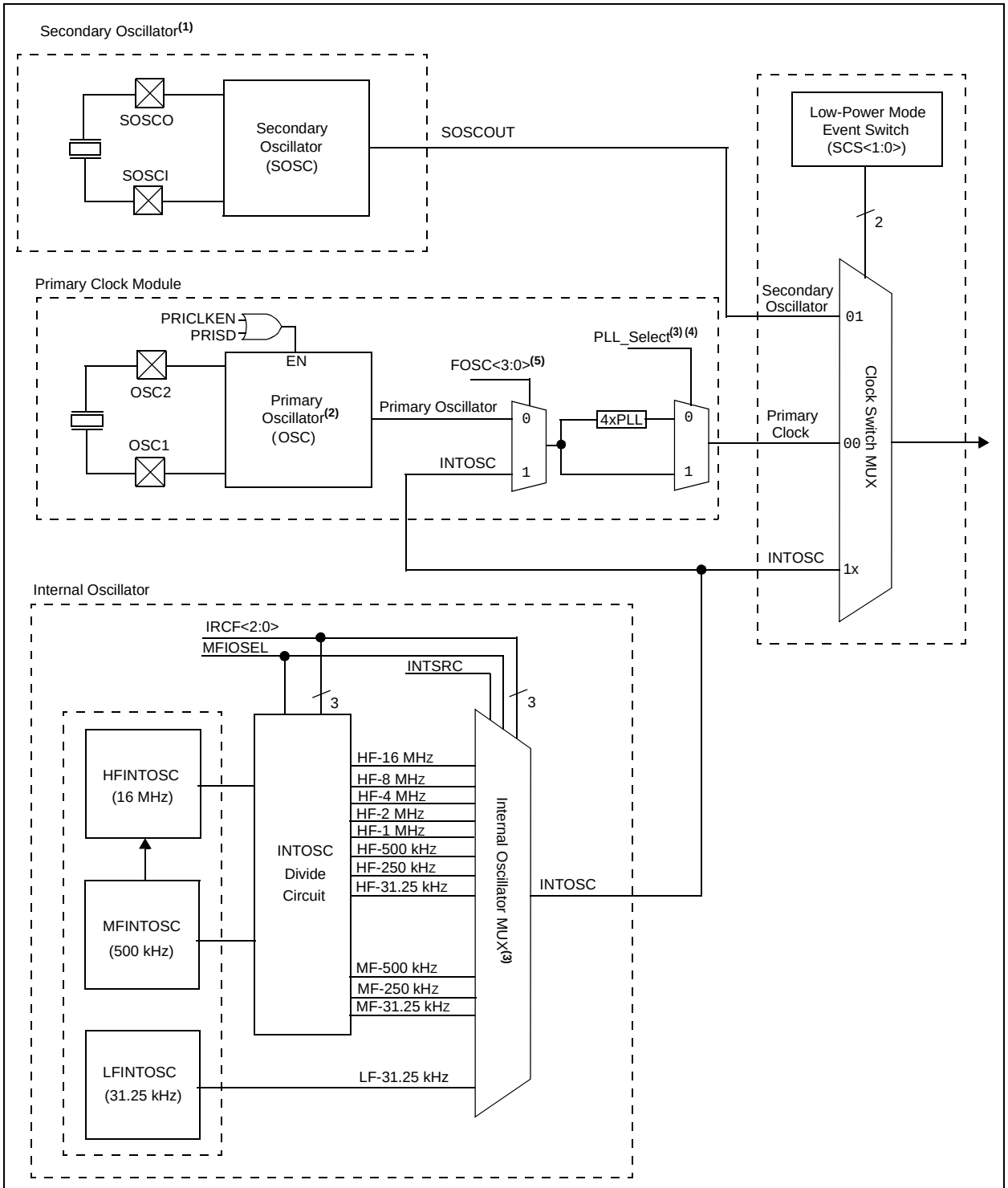
Details	
Product Status	Active
Core Processor	PIC
Core Size	8-Bit
Speed	48MHz
Connectivity	I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, HLVD, POR, PWM, WDT
Number of I/O	24
Program Memory Size	16KB (8K x 16)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	768 x 8
Voltage - Supply (Vcc/Vdd)	2.3V ~ 5.5V
Data Converters	A/D 19x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Through Hole
Package / Case	28-DIP (0.300", 7.62mm)
Supplier Device Package	28-SPDIP
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f24k22-e-sp

TABLE 3: PIC18(L)F4XK22 PIN SUMMARY

40-PDIP	40-UQFN	44-TQFP	44-QFN	I/O	Analog	Comparator	CTMU	SR Latch	Reference	(E)CCP	EUSART	MSSP	Timers	Interrupts	Pull-up	Basic
2	17	19	19	RA0	AN0	C12IN0-										
3	18	20	20	RA1	AN1	C12IN1-										
4	19	21	21	RA2	AN2	C2IN+			VREF- DACOUT							
5	20	22	22	RA3	AN3	C1IN+			VREF+							
6	21	23	23	RA4		C1OUT		SRQ					T0CKI			
7	22	24	24	RA5	AN4	C2OUT		SRNQ	HLVDIN			$\overline{SS}$1				
14	29	31	33	RA6												OSC2 CLKO
13	28	30	32	RA7												OSC1 CLKI
33	8	8	9	RB0	AN12			SRI		FLT0				INT0	Y	
34	9	9	10	RB1	AN10	C12IN3-								INT1	Y	
35	10	10	11	RB2	AN8		CTED1							INT2	Y	
36	11	11	12	RB3	AN9	C12IN2-	CTED2			CCP2 P2A(1)					Y	
37	12	14	14	RB4	AN11								T5G	IOC	Y	
38	13	15	15	RB5	AN13					CCP3 P3A(3)			T1G T3CKI(2)	IOC	Y	
39	14	16	16	RB6										IOC	Y	PGC
40	15	17	17	RB7										IOC	Y	PGD
15	30	32	34	RC0						P2B(4)			SOSCO T1CKI T3CKI(2) T3G			
16	31	35	35	RC1						CCP2(1) P2A			SOSCI			
17	32	36	36	RC2	AN14		CTPLS			CCP1 P1A			T5CKI			
18	33	37	37	RC3	AN15							SCK1 SCL1				
23	38	42	42	RC4	AN16							SDI1 SDA1				
24	39	43	43	RC5	AN17							SDO1				
25	40	44	44	RC6	AN18						TX1 CK1					
26	1	1	1	RC7	AN19						RX1 DT1					
19	34	38	38	RD0	AN20							SCK2 SCL2				
20	35	39	39	RD1	AN21					CCP4		SDI2 SDA2				
21	36	40	40	RD2	AN22					P2B(4)						
22	37	41	41	RD3	AN23					P2C		$\overline{SS}$2				
27	2	2	2	RD4	AN24					P2D		SD02				
28	3	3	3	RD5	AN25					P1B						
29	4	4	4	RD6	AN26					P1C	TX2 CK2					
30	5	5	5	RD7	AN27					P1D	RX2 DT2					
8	23	25	25	RE0	AN5					CCP3 P3A(3)						

Note 1: CCP2 multiplexed in fuses.
2: T3CKI multiplexed in fuses.
3: CCP3/P3A multiplexed in fuses.
4: P2B multiplexed in fuses.

FIGURE 2-1: SIMPLIFIED OSCILLATOR SYSTEM BLOCK DIAGRAM



Note 1: Details in Figure 2-4.

2: Details in Figure 2-2.

3: Details in Figure 2-3.

4: Details in Table 2-1.

5: The Primary Oscillator MUX uses the INTOSC branch when FOSC<3:0> = 100x.

2.8 PLL Frequency Multiplier

A Phase-Locked Loop (PLL) circuit is provided as an option for users who wish to use a lower frequency oscillator circuit or to clock the device up to its highest rated frequency from the crystal oscillator. This may be useful for customers who are concerned with EMI due to high-frequency crystals or users who require higher clock speeds from an internal oscillator.

2.8.1 PLL IN EXTERNAL OSCILLATOR MODES

The PLL can be enabled for any of the external oscillator modes using the OSC1/OSC2 pins by either setting the PLLCFG bit (CONFIG1H<4>), or setting the PLLEN bit (OSCTUNE<6>). The PLL is designed for input frequencies of 4 MHz up to 16 MHz. The PLL then multiplies the oscillator output frequency by four to produce an internal clock frequency up to 64 MHz. Oscillator frequencies below 4 MHz should not be used with the PLL.

2.8.2 PLL IN HFINTOSC MODES

The 4x frequency multiplier can be used with the internal oscillator block to produce faster device clock speeds than are normally possible with the internal oscillator. When enabled, the PLL multiplies the HFINTOSC by four to produce clock rates up to 64 MHz.

Unlike external clock modes, when internal clock modes are enabled, the PLL can only be controlled through software. The PLLEN control bit of the OSCTUNE register is used to enable or disable the PLL operation when the HFINTOSC is used.

The PLL is designed for input frequencies of 4 MHz up to 16 MHz.

4.0 RESET

The PIC18(L)F2X/4XK22 devices differentiate between various kinds of Reset:

a) Power-on Reset (POR)
b) $\overline{\text{MCLR}}$ Reset during normal operation
c) $\overline{\text{MCLR}}$ Reset during power-managed modes
d) Watchdog Timer (WDT) Reset (during execution)
e) Programmable Brown-out Reset (BOR)
f) RESET Instruction
g) Stack Full Reset
h) Stack Underflow Reset

This section discusses Resets generated by $\overline{\text{MCLR}}$, POR and BOR and covers the operation of the various start-up timers. Stack Reset events are covered in **Section 5.2.0.1 "Stack Full and Underflow Resets"**. WDT Resets are covered in **Section 24.3 "Watchdog Timer (WDT)"**.

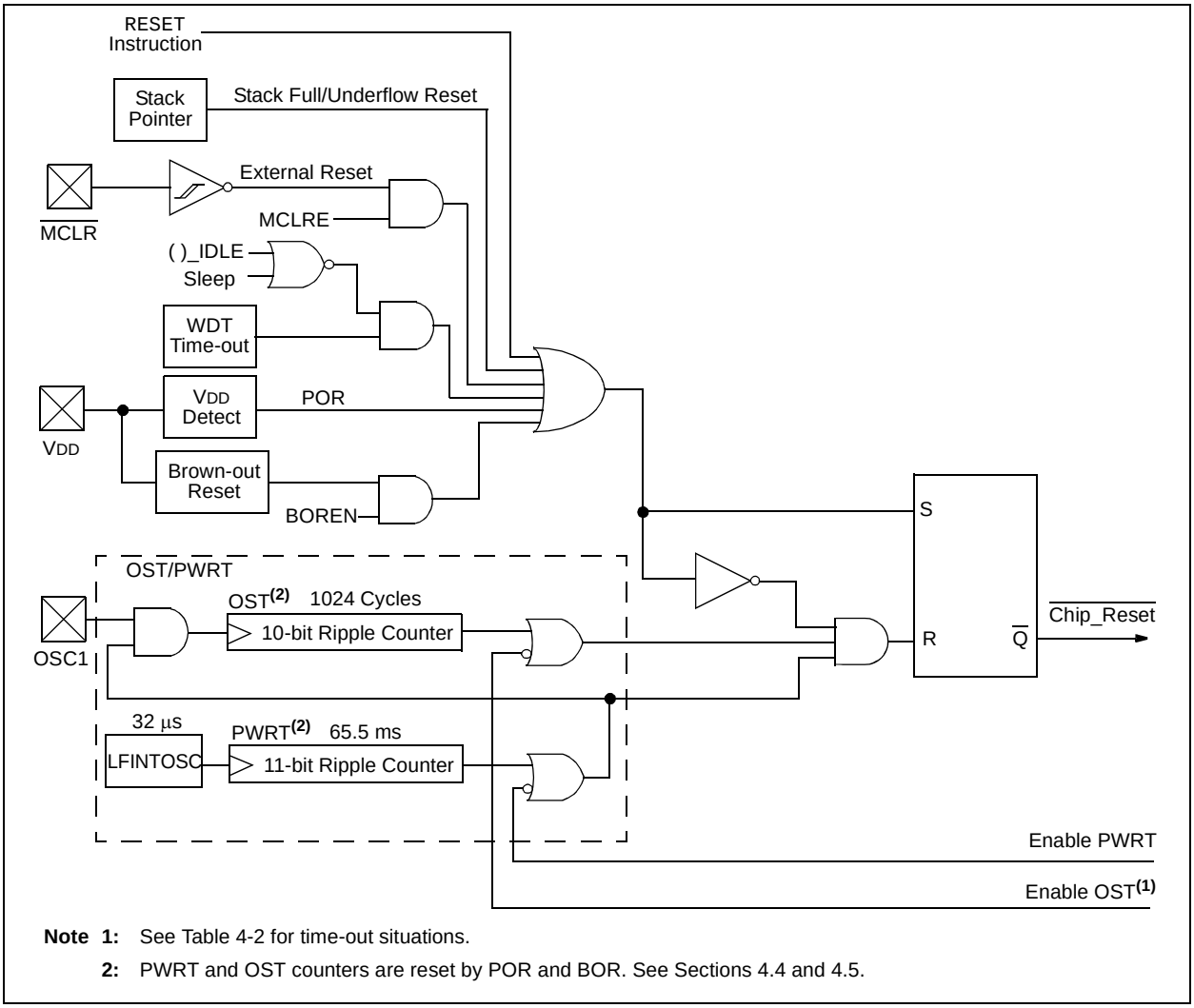
A simplified block diagram of the On-Chip Reset Circuit is shown in Figure 4-1.

4.1 RCON Register

Device Reset events are tracked through the RCON register (Register 4-1). The lower five bits of the register indicate that a specific Reset event has occurred. In most cases, these bits can only be cleared by the event and must be set by the application after the event. The state of these flag bits, taken together, can be read to indicate the type of Reset that just occurred. This is described in more detail in **Section 4.7 "Reset State of Registers"**.

The RCON register also has control bits for setting interrupt priority (IPEN) and software control of the BOR (SBOREN). Interrupt priority is discussed in **Section 9.0 "Interrupts"**. BOR is covered in **Section 4.5 "Brown-out Reset (BOR)"**.

FIGURE 4-1: SIMPLIFIED BLOCK DIAGRAM OF ON-CHIP RESET CIRCUIT



Note 1: See Table 4-2 for time-out situations.

2: PWRT and OST counters are reset by POR and BOR. See Sections 4.4 and 4.5.

9.0 INTERRUPTS

The PIC18(L)F2X/4XK22 devices have multiple interrupt sources and an interrupt priority feature that allows most interrupt sources to be assigned a high or low priority level (INT0 does not have a priority bit, it is always a high priority). The high priority interrupt vector is at 0008h and the low priority interrupt vector is at 0018h. A high priority interrupt event will interrupt a low priority interrupt that may be in progress.

There are 19 registers used to control interrupt operation.

These registers are:

- INTCON, INTCON2, INTCON3
- PIR1, PIR2, PIR3, PIR4, PIR5
- PIE1, PIE2, PIE3, PIE4, PIE5
- IPR1, IPR2, IPR3, IPR4, IPR5
- RCON

It is recommended that the Microchip header files supplied with MPLAB® IDE be used for the symbolic bit names in these registers. This allows the assembler/ compiler to automatically take care of the placement of these bits within the specified register.

In general, interrupt sources have three bits to control their operation. They are:

- **Flag bit** to indicate that an interrupt event occurred
- **Enable bit** that allows program execution to branch to the interrupt vector address when the flag bit is set
- **Priority bit** to select high priority or low priority

9.1 Mid-Range Compatibility

When the IPEN bit is cleared (default state), the interrupt priority feature is disabled and interrupts are compatible with PIC® microcontroller mid-range devices. In Compatibility mode, the interrupt priority bits of the IPRx registers have no effect. The PEIE/GIEL bit of the INTCON register is the global interrupt enable for the peripherals. The PEIE/GIEL bit disables only the peripheral interrupt sources and enables the peripheral interrupt sources when the GIE/GIEH bit is also set. The GIE/GIEH bit of the INTCON register is the global interrupt enable which enables all non-peripheral interrupt sources and disables all interrupt sources, including the peripherals. All interrupts branch to address 0008h in Compatibility mode.

9.2 Interrupt Priority

The interrupt priority feature is enabled by setting the IPEN bit of the RCON register. When interrupt priority is enabled the GIE/GIEH and PEIE/GIEL global interrupt enable bits of Compatibility mode are replaced by the GIEH high priority, and GIEL low priority, global interrupt enables. When set, the GIEH bit of the INTCON register enables all interrupts that have their associated IPRx register or INTCONx register priority bit set (high priority). When clear, the GIEH bit disables all interrupt sources including those selected as low priority. When clear, the GIEL bit of the INTCON register disables only the interrupts that have their associated priority bit cleared (low priority). When set, the GIEL bit enables the low priority sources when the GIEH bit is also set.

When the interrupt flag, enable bit and appropriate Global Interrupt Enable (GIE) bit are all set, the interrupt will vector immediately to address 0008h for high priority, or 0018h for low priority, depending on level of the interrupting source's priority bit. Individual interrupts can be disabled through their corresponding interrupt enable bits.

9.3 Interrupt Response

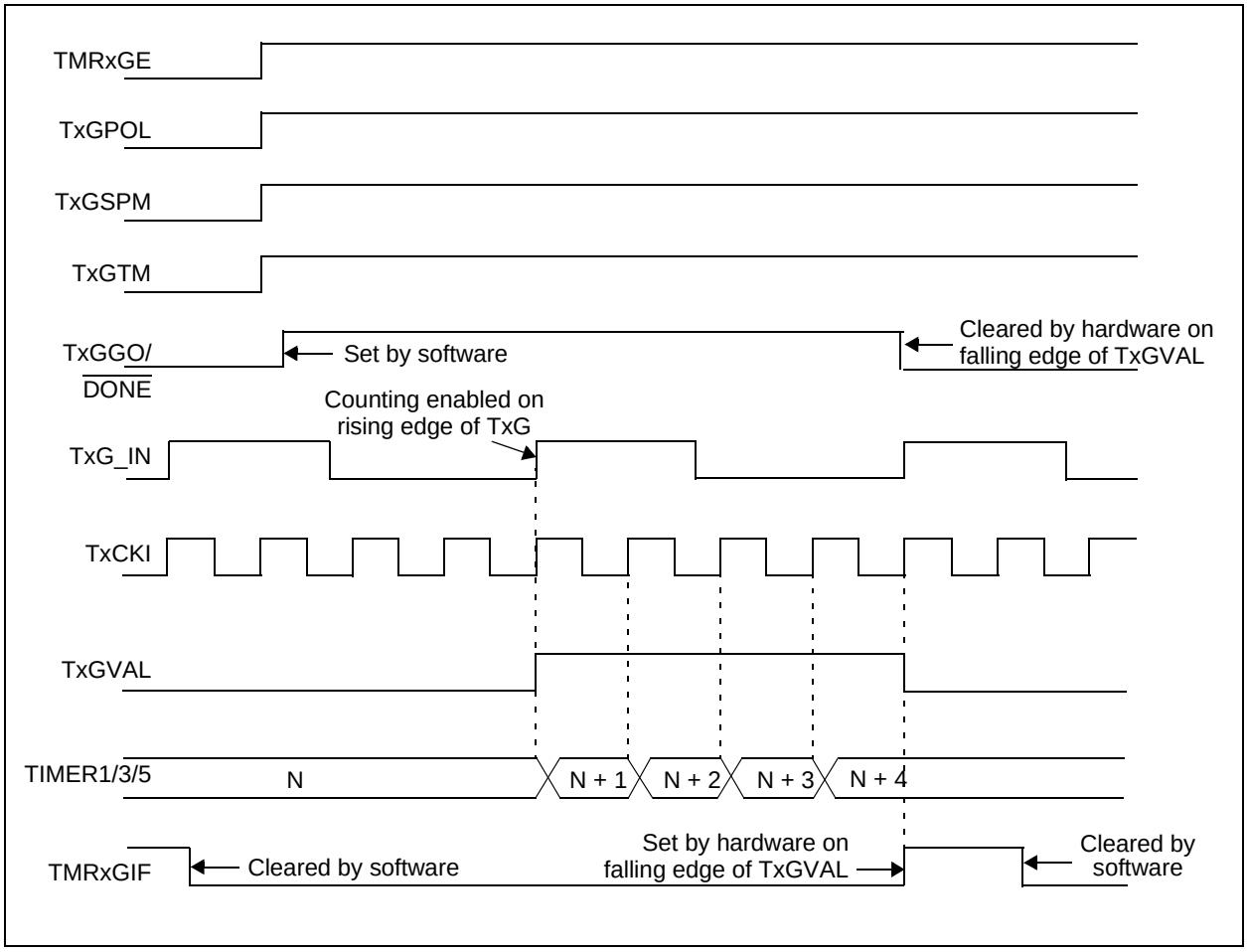
When an interrupt is responded to, the Global Interrupt Enable bit is cleared to disable further interrupts. The GIE/GIEH bit is the Global Interrupt Enable when the IPEN bit is cleared. When the IPEN bit is set, enabling interrupt priority levels, the GIEH bit is the high priority global interrupt enable and the GIEL bit is the low priority Global Interrupt Enable. High priority interrupt sources can interrupt a low priority interrupt. Low priority interrupts are not processed while high priority interrupts are in progress.

The return address is pushed onto the stack and the PC is loaded with the interrupt vector address (0008h or 0018h). Once in the Interrupt Service Routine, the source(s) of the interrupt can be determined by polling the interrupt flag bits in the INTCONx and PIRx registers. The interrupt flag bits must be cleared by software before re-enabling interrupts to avoid repeating the same interrupt.

The "return from interrupt" instruction, `RETFIE`, exits the interrupt routine and sets the GIE/GIEH bit (GIEH or GIEL if priority levels are used), which re-enables interrupts.

For external interrupt events, such as the INT pins or the PORTB interrupt-on-change, the interrupt latency will be three to four instruction cycles. The exact latency is the same for one-cycle or two-cycle instructions. Individual interrupt flag bits are set, regardless of the status of their corresponding enable bits or the Global Interrupt Enable bit.

FIGURE 12-7: TIMER1/3/5 GATE SINGLE-PULSE AND TOGGLE COMBINED MODE



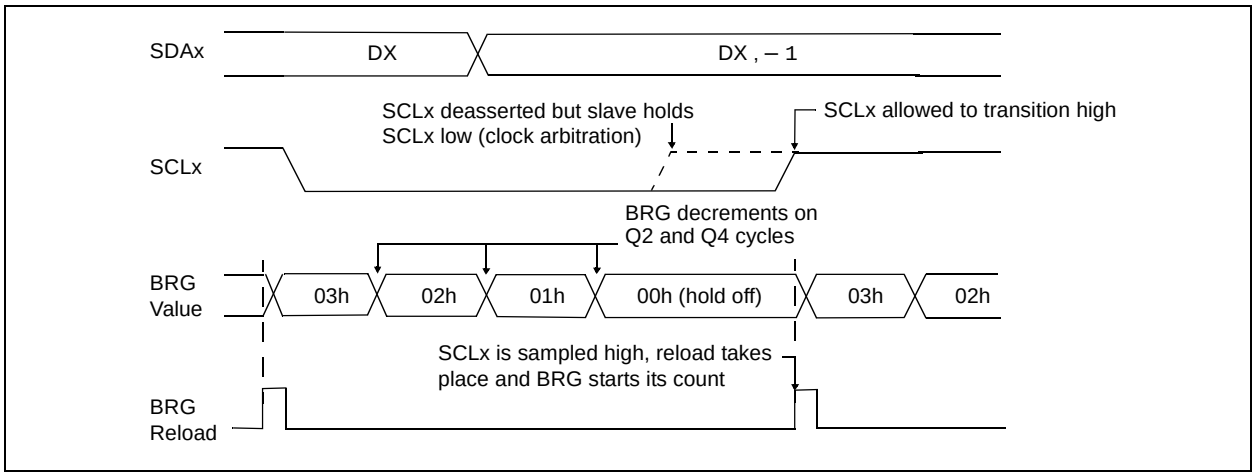
12.12 Peripheral Module Disable

When a peripheral module is not used or inactive, the module can be disabled by setting the Module Disable bit in the PMD registers. This will reduce power consumption to an absolute minimum. Setting the PMD bits holds the module in Reset and disconnects the module's clock source. The Module Disable bits for Timer1 (TMR1MD), Timer3 (TMR3MD) and Timer5 (TMR5MD) are in the PMD0 Register. See **Section 3.0 "Power-Managed Modes"** for more information.

15.6.2 CLOCK ARBITRATION

Clock arbitration occurs when the master, during any receive, transmit or Repeated Start/Stop condition, releases the SCLx pin (SCLx allowed to float high). When the SCLx pin is allowed to float high, the Baud Rate Generator (BRG) is suspended from counting until the SCLx pin is actually sampled high. When the SCLx pin is sampled high, the Baud Rate Generator is reloaded with the contents of SSPxADD<7:0> and begins counting. This ensures that the SCLx high time will always be at least one BRG rollover count in the event that the clock is held low by an external device (Figure 15-25).

FIGURE 15-25: BAUD RATE GENERATOR TIMING WITH CLOCK ARBITRATION



15.6.3 WCOL STATUS FLAG

If the user writes the SSPxBUF when a Start, Restart, Stop, Receive or Transmit sequence is in progress, the WCOL is set and the contents of the buffer are unchanged (the write does not occur). Any time the WCOL bit is set it indicates that an action on SSPxBUF was attempted while the module was not Idle.

Note:	Because queueing of events is not allowed, writing to the lower 5 bits of SSPxCON2 is disabled until the Start condition is complete.

15.6.13 MULTI -MASTER COMMUNICATION, BUS COLLISION AND BUS ARBITRATION

Multi-Master mode support is achieved by bus arbitration. When the master outputs address/data bits onto the SDAx pin, arbitration takes place when the master outputs a '1' on SDAx, by letting SDAx float high and another master asserts a '0'. When the SCLx pin floats high, data should be stable. If the expected data on SDAx is a '1' and the data sampled on the SDAx pin is '0', then a bus collision has taken place. The master will set the Bus Collision Interrupt Flag, BCLxIF, and reset the I^2C port to its Idle state (Figure 15-32).

If a transmit was in progress when the bus collision occurred, the transmission is halted, the BF flag is cleared, the SDAx and SCLx lines are deasserted and the SSPxBUF can be written to. When the user services the bus collision Interrupt Service Routine and if the I^2C bus is free, the user can resume communication by asserting a Start condition.

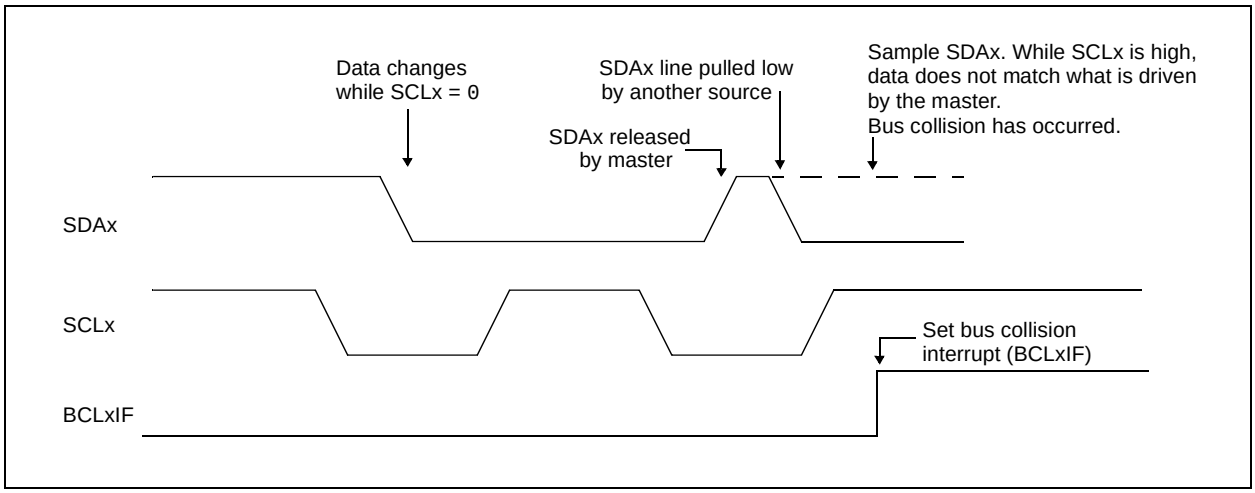
If a Start, Repeated Start, Stop or Acknowledge condition was in progress when the bus collision occurred, the condition is aborted, the SDAx and SCLx lines are deasserted and the respective control bits in the SSPxCON2 register are cleared. When the user services the bus collision Interrupt Service Routine and if the I^2C bus is free, the user can resume communication by asserting a Start condition.

The master will continue to monitor the SDAx and SCLx pins. If a Stop condition occurs, the SSPxIF bit will be set.

A write to the SSPxBUF will start the transmission of data at the first data bit, regardless of where the transmitter left off when the bus collision occurred.

In Multi-Master mode, the interrupt generation on the detection of Start and Stop conditions allows the determination of when the bus is free. Control of the I^2C bus can be taken when the P bit is set in the SSPxSTAT register, or the bus is Idle and the S and P bits are cleared.

FIGURE 15-32: BUS COLLISION TIMING FOR TRANSMIT AND ACKNOWLEDGE



15.6.13.3 Bus Collision During a Stop Condition

Bus collision occurs during a Stop condition if:

a) After the SDAx pin has been deasserted and allowed to float high, SDAx is sampled low after the BRG has timed out (Case 1).

b) After the SCLx pin is deasserted, SCLx is sampled low before SDAx goes high (Case 2).

The Stop condition begins with SDAx asserted low. When SDAx is sampled low, the SCLx pin is allowed to float. When the pin is sampled high (clock arbitration), the Baud Rate Generator is loaded with SSPxADD and counts down to zero. After the BRG times out, SDAx is sampled. If SDAx is sampled low, a bus collision has occurred. This is due to another master attempting to drive a data '0' (Figure 15-38). If the SCLx pin is sampled low before SDAx is allowed to float high, a bus collision occurs. This is another case of another master attempting to drive a data '0' (Figure 15-39).

FIGURE 15-38: BUS COLLISION DURING A STOP CONDITION (CASE 1)

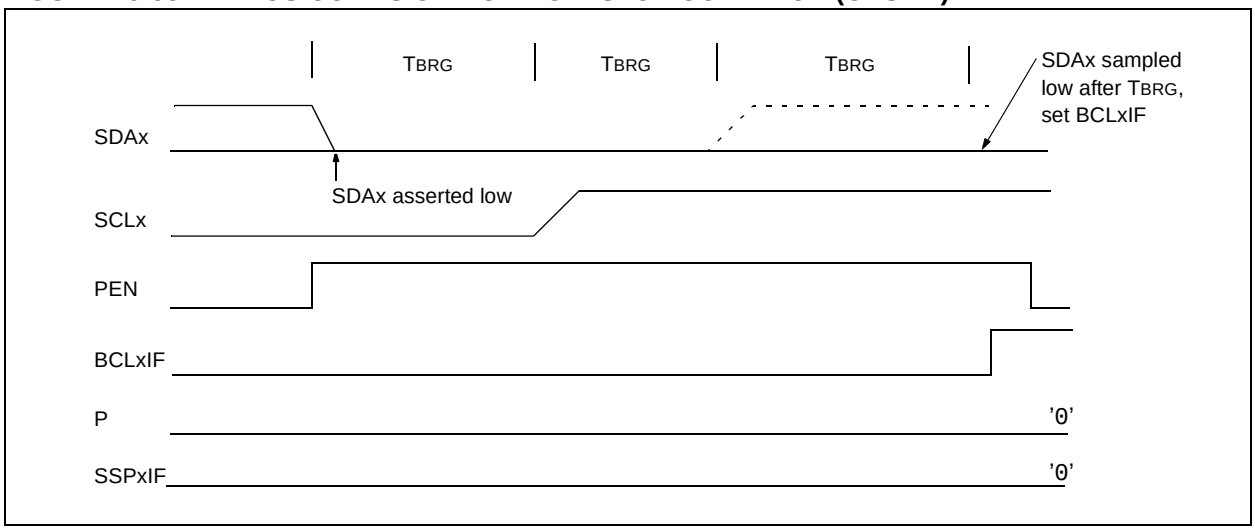
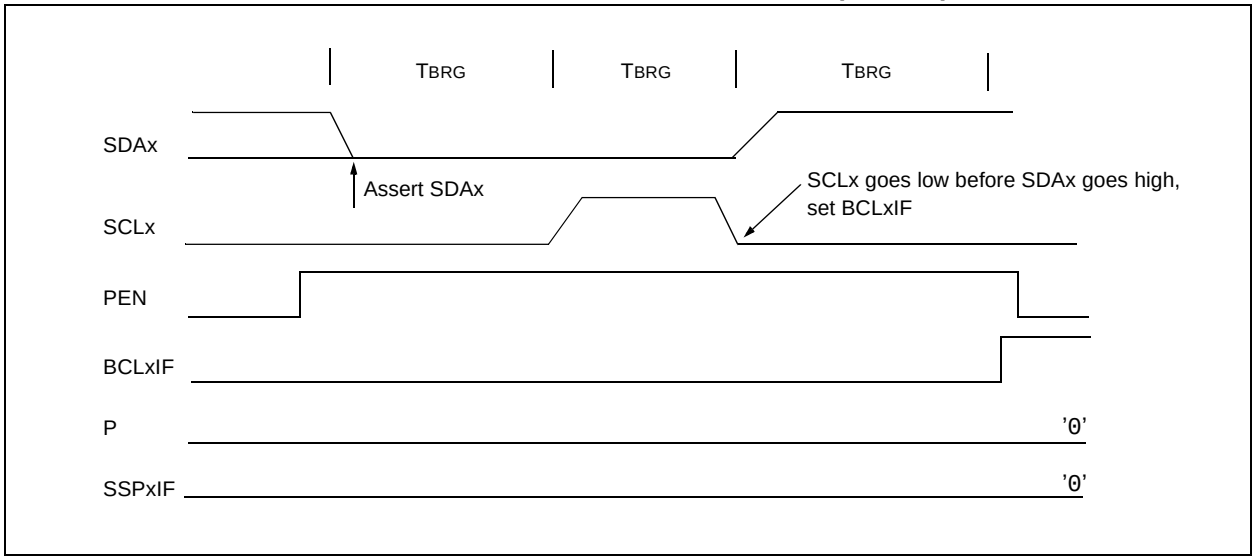


FIGURE 15-39: BUS COLLISION DURING A STOP CONDITION (CASE 2)



REGISTER 15-3: SSPxCON1: SSPx CONTROL REGISTER 1

R/C/HS-0	R/C/HS-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPxOV	SSPxEN	CKP	SSPxM<3:0>			
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets	
'1' = Bit is set	'0' = Bit is cleared	HS = Bit is set by hardware	C = User cleared

bit 7 **WCOL:** Write Collision Detect bit
Master mode:
1 = A write to the SSPxBUF register was attempted while the I^2C conditions were not valid for a transmission to be started
0 = No collision
Slave mode:
1 = The SSPxBUF register is written while it is still transmitting the previous word (must be cleared in software)
0 = No collision

bit 6 **SSPxOV:** Receive Overflow Indicator bit**(1)**
In SPI mode:
1 = A new byte is received while the SSPxBUF register is still holding the previous data. In case of overflow, the data in SSPxSR is lost. Overflow can only occur in Slave mode. In Slave mode, the user must read the SSPxBUF, even if only transmitting data, to avoid setting overflow. In Master mode, the overflow bit is not set since each new reception (and transmission) is initiated by writing to the SSPxBUF register (must be cleared in software).
0 = No overflow
In I^2C mode:
1 = A byte is received while the SSPxBUF register is still holding the previous byte. SSPxOV is a "don't care" in Transmit mode (must be cleared in software).
0 = No overflow

bit 5 **SSPxEN:** Synchronous Serial Port Enable bit
In both modes, when enabled, these pins must be properly configured as input or output
In SPI mode:
1 = Enables serial port and configures SCKx, SDOx, SDIx and $\overline{SSx}$ as the source of the serial port pins**(2)**
0 = Disables serial port and configures these pins as I/O port pins
In I^2C mode:
1 = Enables the serial port and configures the SDAx and SCLx pins as the source of the serial port pins**(3)**
0 = Disables serial port and configures these pins as I/O port pins

bit 4 **CKP:** Clock Polarity Select bit
In SPI mode:
1 = Idle state for clock is a high level
0 = Idle state for clock is a low level
In I^2C Slave mode:
SCLx release control
1 = Enable clock
0 = Holds clock low (clock stretch). (Used to ensure data setup time.)
In I^2C Master mode:
Unused in this mode

16.4.4 BREAK CHARACTER SEQUENCE

The EUSART module has the capability of sending the special Break character sequences that are required by the LIN bus standard. A Break character consists of a Start bit, followed by 12 '0' bits and a Stop bit.

To send a Break character, set the SENDB and TXEN bits of the TXSTAx register. The Break character transmission is then initiated by a write to the TXREGx. The value of data written to TXREGx will be ignored and all '0's will be transmitted.

The SENDB bit is automatically reset by hardware after the corresponding Stop bit is sent. This allows the user to preload the transmit FIFO with the next transmit byte following the Break character (typically, the Sync character in the LIN specification).

The TRMT bit of the TXSTAx register indicates when the transmit operation is active or Idle, just as it does during normal transmission. See Figure 16-9 for the timing of the Break character sequence.

16.4.4.1 Break and Sync Transmit Sequence

The following sequence will start a message frame header made up of a Break, followed by an auto-baud Sync byte. This sequence is typical of a LIN bus master.

1. Configure the EUSART for the desired mode.
2. Set the TXEN and SENDB bits to enable the Break sequence.
3. Load the TXREGx with a dummy character to initiate transmission (the value is ignored).
4. Write '55h' to TXREGx to load the Sync character into the transmit FIFO buffer.
5. After the Break has been sent, the SENDB bit is reset by hardware and the Sync character is then transmitted.

When the TXREGx becomes empty, as indicated by the TXxIF, the next data byte can be written to TXREGx.

16.4.5 RECEIVING A BREAK CHARACTER

The Enhanced EUSART module can receive a Break character in two ways.

The first method to detect a Break character uses the FERR bit of the RCSTAx register and the Received data as indicated by RCREGx. The Baud Rate Generator is assumed to have been initialized to the expected baud rate.

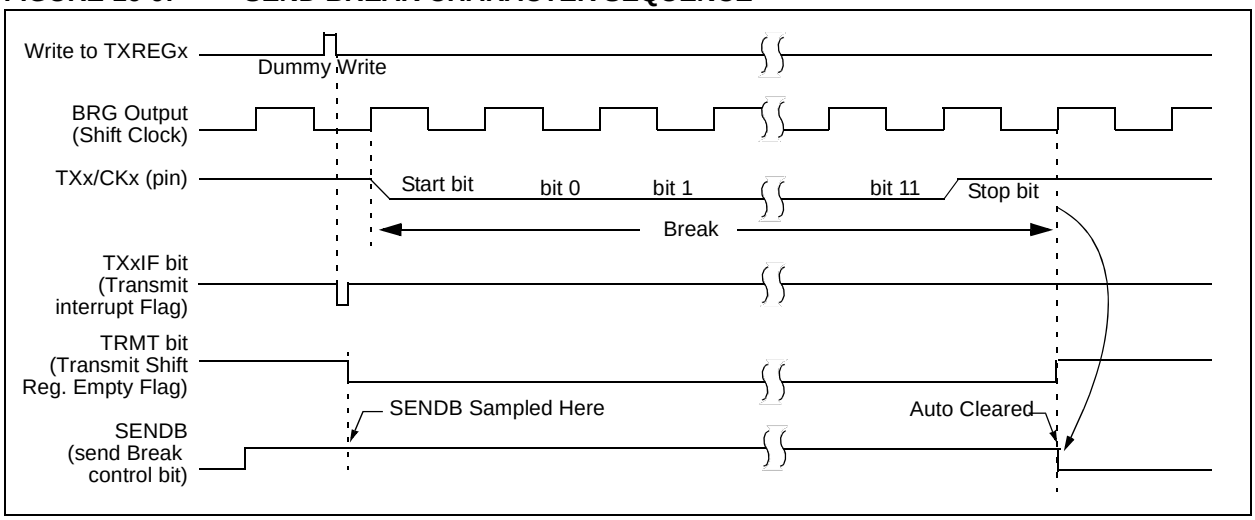
A Break character has been received when;

- RCxIF bit is set
- FERR bit is set
- RCREGx = 00h

The second method uses the Auto-Wake-up feature described in **Section 16.4.3 "Auto-Wake-up on Break"**. By enabling this feature, the EUSART will sample the next two transitions on RXx/DTx, cause an RCxIF interrupt, and receive the next data byte followed by another interrupt.

Note that following a Break character, the user will typically want to enable the Auto-Baud Detect feature. For both methods, the user can set the ABDEN bit of the BAUDCONx register before placing the EUSART in Sleep mode.

FIGURE 16-9: SEND BREAK CHARACTER SEQUENCE



16.5.1.6 Synchronous Master Reception

Data is received at the RXx/DTx pin. The RXx/DTx pin output driver must be disabled by setting the corresponding TRIS bits when the EUSART is configured for synchronous master receive operation.

In Synchronous mode, reception is enabled by setting either the Single Receive Enable bit (SREN of the RCSTAx register) or the Continuous Receive Enable bit (CREN of the RCSTAx register).

When SREN is set and CREN is clear, only as many clock cycles are generated as there are data bits in a single character. The SREN bit is automatically cleared at the completion of one character. When CREN is set, clocks are continuously generated until CREN is cleared. If CREN is cleared in the middle of a character the CK clock stops immediately and the partial character is discarded. If SREN and CREN are both set, then SREN is cleared at the completion of the first character and CREN takes precedence.

To initiate reception, set either SREN or CREN. Data is sampled at the RXx/DTx pin on the trailing edge of the TXx/CKx clock pin and is shifted into the Receive Shift Register (RSR). When a complete character is received into the RSR, the RCxIF bit is set and the character is automatically transferred to the two character receive FIFO. The Least Significant eight bits of the top character in the receive FIFO are available in RCREGx. The RCxIF bit remains set as long as there are un-read characters in the receive FIFO.

16.5.1.7 Slave Clock

Synchronous data transfers use a separate clock line, which is synchronous with the data. A device configured as a slave receives the clock on the TXx/CKx line. The TXx/CKx pin output driver must be disabled by setting the associated TRIS bit when the device is configured for synchronous slave transmit or receive operation. Serial data bits change on the leading edge to ensure they are valid at the trailing edge of each clock. One data bit is transferred for each clock cycle. Only as many clock cycles should be received as there are data bits.

16.5.1.8 Receive Overrun Error

The receive FIFO buffer can hold two characters. An overrun error will be generated if a third character, in its entirety, is received before RCREGx is read to access the FIFO. When this happens the OERR bit of the RCSTAx register is set. Previous data in the FIFO will not be overwritten. The two characters in the FIFO buffer can be read, however, no additional characters will be received until the error is cleared. The OERR bit can only be cleared by clearing the overrun condition. If the overrun error occurred when the SREN bit is set and CREN is clear then the error is cleared by reading RCREGx.

If the overrun occurred when the CREN bit is set then the error condition is cleared by either clearing the CREN bit of the RCSTAx register or by clearing the SPEN bit which resets the EUSART.

16.5.1.9 Receiving 9-bit Characters

The EUSART supports 9-bit character reception. When the RX9 bit of the RCSTAx register is set the EUSART will shift 9-bits into the RSR for each character received. The RX9D bit of the RCSTAx register is the ninth, and Most Significant, data bit of the top unread character in the receive FIFO. When reading 9-bit data from the receive FIFO buffer, the RX9D data bit must be read before reading the eight Least Significant bits from the RCREGx.

16.5.1.10 Synchronous Master Reception Setup:

1. Initialize the SPBRGHx, SPBRGx register pair for the appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate.
2. Set the RXx/DTx and TXx/CKx TRIS controls to ‘1’.
3. Enable the synchronous master serial port by setting bits SYNC, SPEN and CSRC. Disable RXx/DTx and TXx/CKx output drivers by setting the corresponding TRIS bits.
4. Ensure bits CREN and SREN are clear.
5. If using interrupts, set the GIE/GIEH and PEIE/GIEL bits of the INTCON register and set RCxIE.
6. If 9-bit reception is desired, set bit RX9.
7. Start reception by setting the SREN bit or for continuous reception, set the CREN bit.
8. Interrupt flag bit RCxIF will be set when reception of a character is complete. An interrupt will be generated if the enable bit RCxIE was set.
9. Read the RCSTAx register to get the ninth bit (if enabled) and determine if any error occurred during reception.
10. Read the 8-bit received data by reading the RCREGx register.
11. If an overrun error occurs, clear the error by either clearing the CREN bit of the RCSTAx register or by clearing the SPEN bit which resets the EUSART.

17.1.7 RESULT FORMATTING

The 10-bit A/D conversion result can be supplied in two formats, left justified or right justified. The ADFM bit of the ADCON2 register controls the output format.

Figure 17-2 shows the two output formats.

FIGURE 17-2: 10-BIT A/D CONVERSION RESULT FORMAT

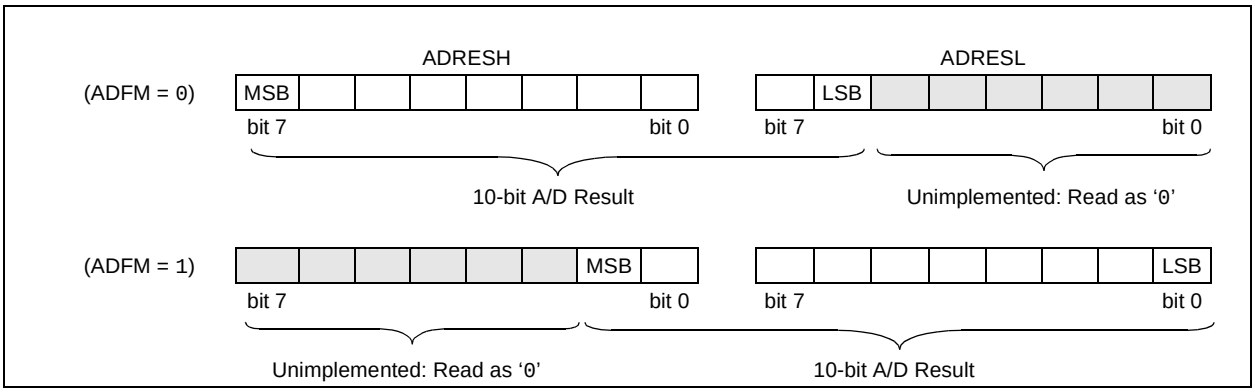


FIGURE 20-1: DIVSRCLK BLOCK DIAGRAM

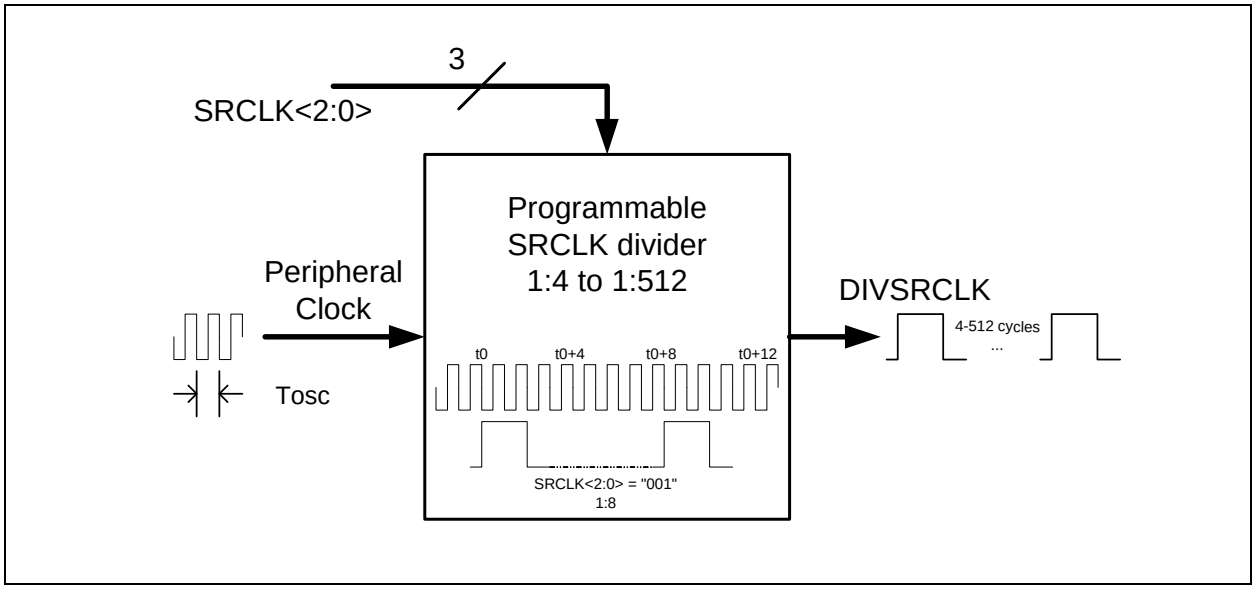
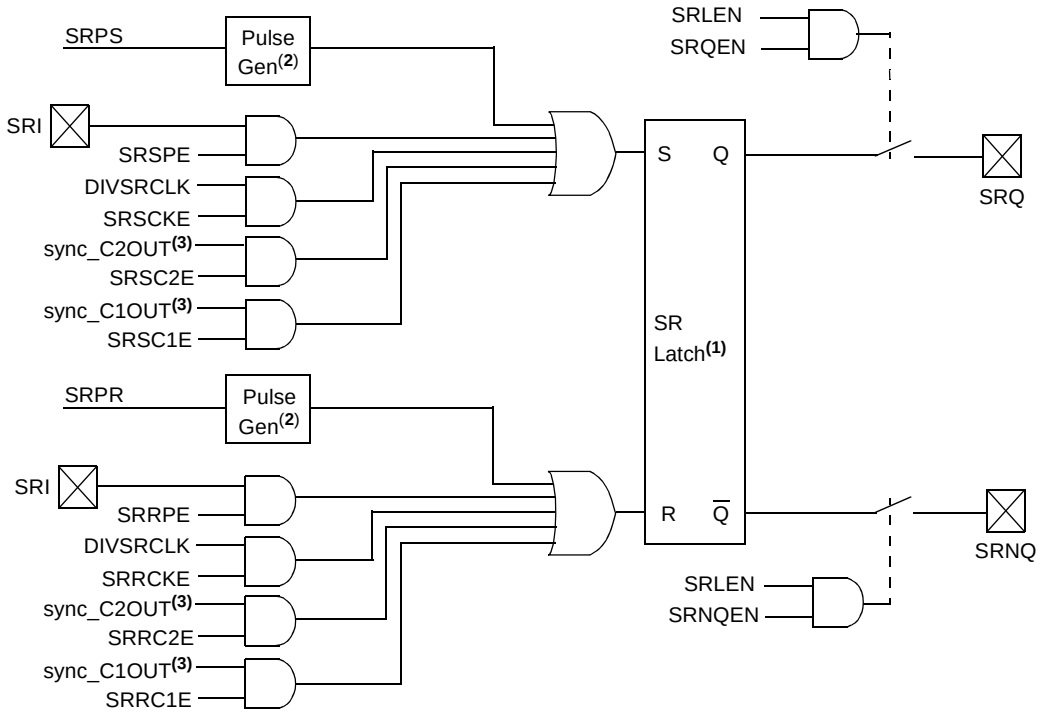


FIGURE 20-2: SR LATCH SIMPLIFIED BLOCK DIAGRAM



Note 1: If R = 1 and S = 1 simultaneously, Q = 0, $\overline{Q}$ = 1
2: Pulse generator causes a pulse width of 2 TOSC clock cycles.
3: Name denotes the connection point at the comparator output.

20.5 Register Definitions: SR Latch Control

REGISTER 20-1: SRCON0: SR LATCH 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
SRLEN	SRCLK<2:0>			SRQEN	SRNQEN	SRPS	SRPR
bit 7							bit 0

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

bit 7 **SRLEN:** SR Latch Enable bit(1)
1 = SR latch is enabled
0 = SR latch is disabled

bit 6-4 **SRCLK<2:0>:** SR Latch Clock Divider Bits
000 = Generates a 2 TOSC wide pulse on DIVSRCLK every 4 peripheral clock cycles
001 = Generates a 2 TOSC wide pulse on DIVSRCLK every 8 peripheral clock cycles
010 = Generates a 2 TOSC wide pulse on DIVSRCLK every 16 peripheral clock cycles
011 = Generates a 2 TOSC wide pulse on DIVSRCLK every 32 peripheral clock cycles
100 = Generates a 2 TOSC wide pulse on DIVSRCLK every 64 peripheral clock cycles
101 = Generates a 2 TOSC wide pulse on DIVSRCLK every 128 peripheral clock cycles
110 = Generates a 2 TOSC wide pulse on DIVSRCLK every 256 peripheral clock cycles
111 = Generates a 2 TOSC wide pulse on DIVSRCLK every 512 peripheral clock cycles

bit 3 **SRQEN:** SR Latch Q Output Enable bit
1 = Q is present on the SRQ pin
0 = Q is internal only

bit 2 **SRNQEN:** SR Latch $\overline{Q}$ Output Enable bit
1 = $\overline{Q}$ is present on the SRNQ pin
0 = $\overline{Q}$ is internal only

bit 1 **SRPS:** Pulse Set Input of the SR Latch bit(2)
1 = Pulse set input for two TOSC clock cycles
0 = No effect on set input

bit 0 **SRPR:** Pulse Reset Input of the SR Latch bit(2)
1 = Pulse reset input for two TOSC clock cycles
0 = No effect on Reset input

Note 1: Changing the SRCLK bits while the SR latch is enabled may cause false triggers to the set and Reset inputs of the latch.
2: Set only, always reads back '0'.

REGISTER 24-9: CONFIG6H: CONFIGURATION REGISTER 6 HIGH

R/C-1	R/C-1	R-1	U-0	U-0	U-0	U-0	U-0
WRTD	WRTB	WRTC(1)	—	—	—	—	—
bit 7							bit 0

Legend:	
R = Readable bit	U = Unimplemented bit, read as '0'
-n = Value when device is unprogrammed	C = Clearable only bit

bit 7 **WRTD:** Data EEPROM Write Protection bit
1 = Data EEPROM not write-protected
0 = Data EEPROM write-protected

bit 6 **WRTB:** Boot Block Write Protection bit
1 = Boot Block not write-protected
0 = Boot Block write-protected

bit 5 **WRTC:** Configuration Register Write Protection bit(1)
1 = Configuration registers not write-protected
0 = Configuration registers write-protected

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

Note 1: This bit is read-only in normal execution mode; it can be written only in Program mode.

REGISTER 24-10: CONFIG7L: CONFIGURATION REGISTER 7 LOW

U-0	U-0	U-0	U-0	R/C-1	R/C-1	R/C-1	R/C-1
—	—	—	—	EBTR3(1)	EBTR2(1)	EBTR1	EBTR0
bit 7							bit 0

Legend:	
R = Readable bit	U = Unimplemented bit, read as '0'
-n = Value when device is unprogrammed	C = Clearable only bit

bit 7-4 **Unimplemented:** Read as '0'

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

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

bit 1 **EBTR1:** Table Read Protection bit
1 = Block 1 not protected from table reads executed in other blocks
0 = Block 1 protected from table reads executed in other blocks

bit 0 **EBTR0:** Table Read Protection bit
1 = Block 0 not protected from table reads executed in other blocks
0 = Block 0 protected from table reads executed in other blocks

Note 1: Available on PIC18(L)FX5K22 and PIC18(L)FX6K22s devices.

25.1.1 STANDARD INSTRUCTION SET

ADDLW **ADD literal to W**

Syntax: ADDLW k

Operands: $0 \le k \le 255$

Operation: $(W) + k \rightarrow W$

Status Affected: N, OV, C, DC, Z

Encoding:

0000	1111	kkkk	kkkk

Description: The contents of W are added to the 8-bit literal 'k' and 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: ADDLW 15h

Before Instruction
W = 10h
After Instruction
W = 25h

ADDWF **ADD W to f**

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

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

Operation: $(W) + (f) \rightarrow$ dest

Status Affected: N, OV, C, DC, Z

Encoding:

0010	01da	ffff	ffff

Description: Add W to register 'f'. 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.
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 25.2.3 "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: ADDWF REG, 0, 0

Before Instruction
W = 17h
REG = 0C2h
After Instruction
W = 0D9h
REG = 0C2h

Note: All PIC18 instructions may take an optional label argument preceding the instruction mnemonic for use in symbolic addressing. If a label is used, the instruction format then becomes: {label} instruction argument(s).

CLRF **Clear f**

Syntax: CLRF f {,a}

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

Operation: 000h → f
1 → Z

Status Affected: Z

Encoding:

0110	101a	ffff	ffff

Description: Clears the contents of the specified register.
If 'a' is '0', the Access Bank is selected.
If 'a' is '1', the BSR is used to select the GPR bank.
If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 25.2.3 "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: CLRF FLAG_REG, 1

Before Instruction
FLAG_REG = 5Ah
After Instruction
FLAG_REG = 00h

CLRWDT **Clear Watchdog Timer**

Syntax: CLRWDT

Operands: None

Operation: 000h → WDT,
000h → WDT postscaler,
1 → $\overline{TO}$,
1 → $\overline{PD}$

Status Affected: $\overline{TO}$, $\overline{PD}$

Encoding:

0000	0000	0000	0100

Description: CLRWDT instruction resets the Watchdog Timer. It also resets the postscaler of the WDT. Status bits, $\overline{TO}$ and $\overline{PD}$, are set.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	Process Data	No operation

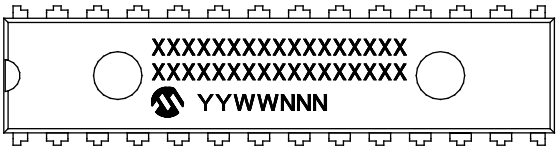
Example: CLRWDT

Before Instruction
WDT Counter = ?
After Instruction
WDT Counter = 00h
WDT Postscaler = 0
$\overline{TO}$ = 1
$\overline{PD}$ = 1

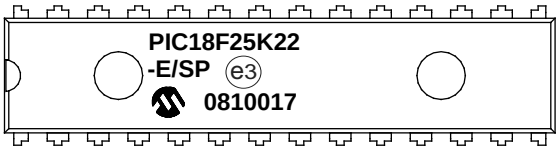
29.0 PACKAGING INFORMATION

29.1 Package Marking Information

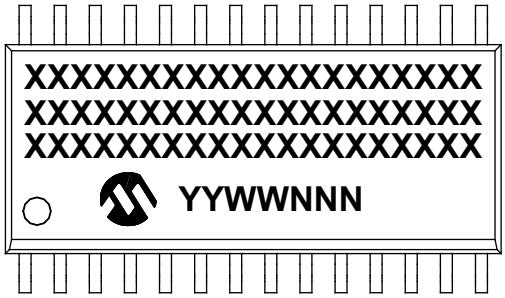
28-Lead SPDIP (.300”)



Example

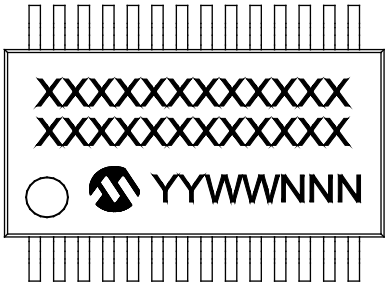


28-Lead SOIC (7.50 mm)

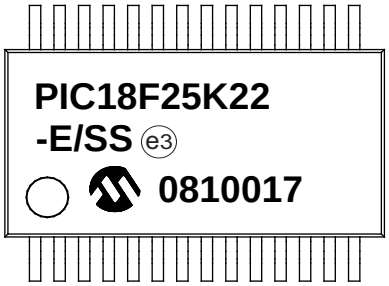


Example

28-Lead SSOP (5.30 mm)



Example

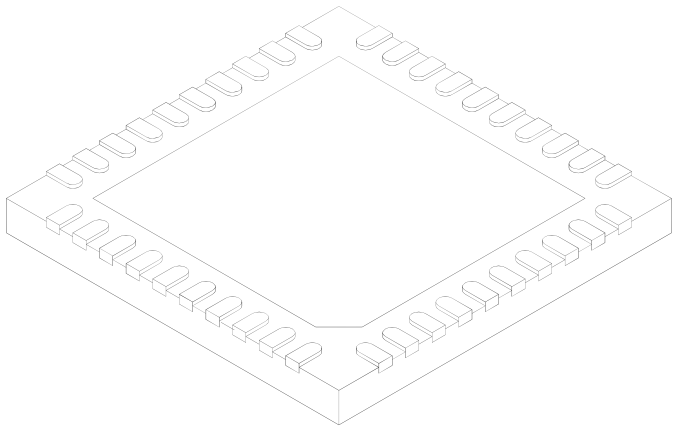


Legend:	XX...X	Customer-specific information or Microchip part number
	Y	Year code (last digit of calendar year)
	YY	Year code (last 2 digits of calendar year)
	WW	Week code (week of January 1 is week ‘01’)
	NNN	Alphanumeric traceability code
	(e3)	Pb-free JEDEC® designator for Matte Tin (Sn)
	*	This package is Pb-free. The Pb-free JEDEC designator ((e3)) can be found on the outer packaging for this package.

Note: In the event the full Microchip part number cannot be marked on one line, it will be carried over to the next line, thus limiting the number of available characters for customer-specific information.

40-Lead Ultra Thin Plastic Quad Flat, No Lead Package (MV) – 5x5x0.5 mm Body [UQFN]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N		40	
Pitch	e		0.40 BSC	
Overall Height	A	0.45	0.50	0.55
Standoff	A1	0.00	0.02	0.05
Contact Thickness	A3		0.127 REF	
Overall Width	E		5.00 BSC	
Exposed Pad Width	E2	3.60	3.70	3.80
Overall Length	D		5.00 BSC	
Exposed Pad Length	D2	3.60	3.70	3.80
Contact Width	b	0.15	0.20	0.25
Contact Length	L	0.30	0.40	0.50
Contact-to-Exposed Pad	K	0.20	-	-

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Package is saw singulated.
3. Dimensioning and tolerancing per ASME Y14.5M.

   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-156A Sheet 2 of 2