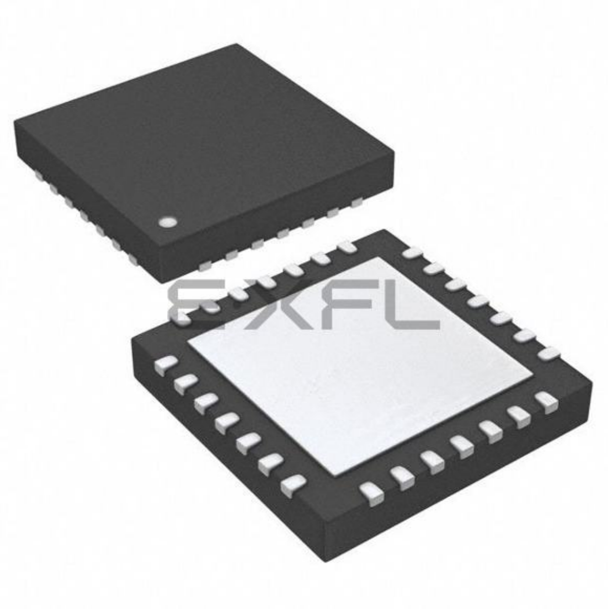
Microchip Technology - PIC18F24K40-I/MV Datasheet



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.

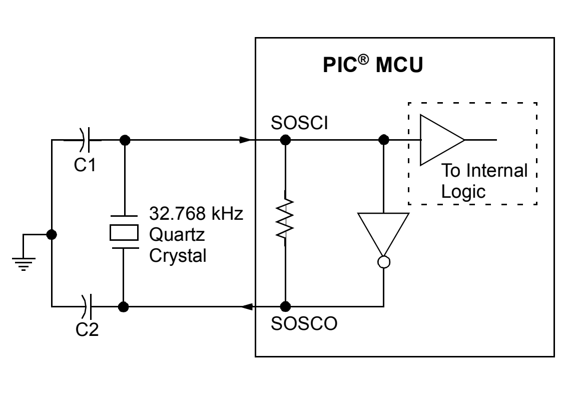
Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	PIC
Core Size	8-Bit
Speed	64MHz
Connectivity	I²C, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, LVD, POR, PWM, WDT
Number of I/O	25
Program Memory Size	16KB (8K x 16)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	2.3V ~ 5.5V
Data Converters	A/D 24x10b; D/A 1x5b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	28-UFQFN Exposed Pad
Supplier Device Package	28-UQFN (4x4)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f24k40-i-mv

4.3.1.5 Secondary Oscillator

The secondary oscillator is a separate oscillator block that can be used as an alternate system clock source. The secondary oscillator is optimized for 32.768 kHz, and can be used with an external crystal oscillator connected to the SOSCI and SOSCO device pins, or an external clock source connected to the SOSCIN pin. The secondary oscillator can be selected during run-time using clock switching. Refer to **Section 4.4 "Clock Switching"** for more information.

FIGURE 4-5: QUARTZ CRYSTAL OPERATION (SECONDARY OSCILLATOR)



Note 1: Quartz crystal characteristics vary according to type, package and manufacturer. The user should consult the manufacturer data sheets for specifications and recommended application.

2: Always verify oscillator performance over the VDD and temperature range that is expected for the application.

3: For oscillator design assistance, reference the following Microchip Application Notes:

- AN826, "*Crystal Oscillator Basics and Crystal Selection for PIC® and PIC® Devices*" (DS00826)
- AN849, "*Basic PIC® Oscillator Design*" (DS00849)
- AN943, "*Practical PIC® Oscillator Analysis and Design*" (DS00943)
- AN949, "*Making Your Oscillator Work*" (DS00949)
- TB097, "*Interfacing a Micro Crystal MS1V-T1K 32.768 kHz Tuning Fork Crystal to a PIC16F690/SS*" (DS91097)
- AN1288, "*Design Practices for Low-Power External Oscillators*" (DS01288)

4.3.2 INTERNAL CLOCK SOURCES

The device may be configured to use the internal oscillator block as the system clock by performing one of the following actions:

- Program the RSTOSC<2:0> bits in Configuration Words to select the INTOSC clock source, which will be used as the default system clock upon a device Reset.
- Write the NOSC<2:0> bits in the OSCCON1 register to switch the system clock source to the internal oscillator during run-time. See **Section 4.4 "Clock Switching"** for more information.

In INTOSC mode, OSC1/CLKIN is available for general purpose I/O. OSC2/CLKOUT is available for general purpose I/O or CLKOUT.

The function of the OSC2/CLKOUT pin is determined by the $\overline{\text{CLKOUTEN}}$ bit in Configuration Words.

The internal oscillator block has two independent oscillators that can produce two internal system clock sources.

1. The **HFINTOSC** (High-Frequency Internal Oscillator) is factory-calibrated and operates from 1 to 64 MHz. The frequency of HFINTOSC can be selected through the OSCFRQ Frequency Selection register, and fine-tuning can be done via the OSCTUNE register.
2. The **LFINTOSC** (Low-Frequency Internal Oscillator) is factory-calibrated and operates at 31 kHz.

5.1 Clock Source

The input to the reference clock output can be selected using the CLKRCLK register.

5.1.1 CLOCK SYNCHRONIZATION

Once the reference clock enable (EN) is set, the module is ensured to be glitch-free at start-up.

When the reference clock output is disabled, the output signal will be disabled immediately.

Clock dividers and clock duty cycles can be changed while the module is enabled, but glitches may occur on the output. To avoid possible glitches, clock dividers and clock duty cycles should be changed only when the CLKREN is clear.

5.2 Programmable Clock Divider

The module takes the clock input and divides it based on the value of the DIV<2:0> bits of the CLKRCON register (Register 5-1).

The following configurations can be made based on the DIV<2:0> bits:

- Base FOSC value
- FOSC divided by 2
- FOSC divided by 4
- FOSC divided by 8
- FOSC divided by 16
- FOSC divided by 32
- FOSC divided by 64
- FOSC divided by 128

The clock divider values can be changed while the module is enabled; however, in order to prevent glitches on the output, the DIV<2:0> bits should only be changed when the module is disabled (EN = 0).

5.3 Selectable Duty Cycle

The DC<1:0> bits of the CLKRCON register can be used to modify the duty cycle of the output clock. A duty cycle of 25%, 50%, or 75% can be selected for all clock rates, with the exception of the undivided base FOSC value.

The duty cycle can be changed while the module is enabled; however, in order to prevent glitches on the output, the DC<1:0> bits should only be changed when the module is disabled (EN = 0).

Note:	The DC1 bit is reset to ‘1’. This makes the default duty cycle 50% and not 0%.

5.4 Operation in Sleep Mode

The reference clock output module clock is based on the system clock. When the device goes to Sleep, the module outputs will remain in their current state. This will have a direct effect on peripherals using the reference clock output as an input signal. No change should occur in the module from entering or exiting from Sleep.

6.1.2 INTERRUPTS DURING DOZE

If an interrupt occurs and the Recover-On-Interrupt bit is clear (ROI = 0) at the time of the interrupt, the Interrupt Service Routine (ISR) continues to execute at the rate selected by DOZE<2:0>. Interrupt latency is extended by the DOZE<2:0> ratio.

If an interrupt occurs and the ROI bit is set (ROI = 1) at the time of the interrupt, the DOZEN bit is cleared and the CPU executes at full speed. The prefetched instruction is executed and then the interrupt vector sequence is executed. In Figure 6-1, the interrupt occurs during the 2^{nd} instruction cycle of the Doze period, and immediately brings the CPU out of Doze. If the Doze-On-Exit (DOE) bit is set (DOE = 1) when the RETFIE operation is executed, DOZEN is set, and the CPU executes at the reduced rate based on the DOZE<2:0> ratio.

EXAMPLE 6-1: DOZE SOFTWARE EXAMPLE

```
//Mainline operation
bool somethingToDo = FALSE:
void main()
{
   initializeSystem();
           // DOZE = 64:1 (for example)
           // ROI = 1;
   GIE = 1; // enable interrupts
   while (1)
   {
      // If ADC completed, process data
      if (somethingToDo)
      {
          doSomething();
          DOZEN = 1; // resume low-power
      }
   }
}

// Data interrupt handler
void interrupt()
{
   // DOZEN = 0 because ROI = 1
   if (ADIF)
   {
      somethingToDo = TRUE;
      DOE = 0; // make main() go fast
      ADIF = 0;
   }
   // else check other interrupts...
   if (TMR0IF)
   {
      timerTick++;
      DOE = 1; // make main() go slow
      TMR0IF = 0;
   }
}
```

6.2 Sleep Mode

Sleep mode is entered by executing the SLEEP instruction, while the Idle Enable (IDLEN) bit of the CPUDOZE register is clear (IDLEN = 0).

Upon entering Sleep mode, the following conditions exist:

1. WDT will be cleared but keeps running if enabled for operation during Sleep
2. The $\overline{\text{PD}}$ bit of the STATUS register is cleared (Register 10-2)
3. The $\overline{\text{TO}}$ bit of the STATUS register is set (Register 10-2)
4. The CPU clock is disabled
5. LFINTOSC, SOSC, HFINTOSC and ADCRC are unaffected and peripherals using them may continue operation in Sleep.
6. I/O ports maintain the status they had before Sleep was executed (driving high, low, or high-impedance)
7. Resets other than WDT are not affected by Sleep mode

Refer to individual chapters for more details on peripheral operation during Sleep.

To minimize current consumption, the following conditions should be considered:

- I/O pins should not be floating
- External circuitry sinking current from I/O pins
- Internal circuitry sourcing current from I/O pins
- Current draw from pins with internal weak pull-ups
- Modules using any oscillator

I/O pins that are high-impedance inputs should be pulled to VDD or VSS externally to avoid switching currents caused by floating inputs.

Examples of internal circuitry that might be sourcing current include modules such as the DAC and FVR modules. See **Section 30.0 “5-Bit Digital-to-Analog Converter (DAC) Module”** and **Section 28.0 “Fixed Voltage Reference (FVR)”** for more information on these modules.

Example 12-3 shows the sequence to do a 16 x 16 unsigned multiplication. Equation 12-1 shows the algorithm that is used. The 32-bit result is stored in four registers (RES<3:0>).

EQUATION 12-1: 16 x 16 UNSIGNED MULTIPLICATION ALGORITHM

$$\begin{aligned}
\text{RES3:RES0} &= \text{ARG1H:ARG1L} \bullet \text{ARG2H:ARG2L} \\
&= (\text{ARG1H} \bullet \text{ARG2H} \bullet 2^{16}) + \\
&\quad (\text{ARG1H} \bullet \text{ARG2L} \bullet 2^{8}) + \\
&\quad (\text{ARG1L} \bullet \text{ARG2H} \bullet 2^{8}) + \\
&\quad (\text{ARG1L} \bullet \text{ARG2L})
\end{aligned}$$

EXAMPLE 12-3: 16 x 16 UNSIGNED MULTIPLY ROUTINE

```
    MOVF     ARG1L, W
    MULWF    ARG2L            ; ARG1L * ARG2L->
                              ; PRODH:PRODL
    MOVFF    PRODH, RES1      ;
    MOVFF    PRODL, RES0      ;
;
    MOVF     ARG1H, W
    MULWF    ARG2H            ; ARG1H * ARG2H->
                              ; PRODH:PRODL
    MOVFF    PRODH, RES3      ;
    MOVFF    PRODL, RES2      ;
;
    MOVF     ARG1L, W
    MULWF    ARG2H            ; ARG1L * ARG2H->
                              ; PRODH:PRODL
    MOVF     PRODL, W         ;
    ADDWF    RES1, F          ; Add cross
    MOVF     PRODH, W         ; products
    ADDWFC   RES2, F          ;
    CLRF     WREG             ;
    ADDWFC   RES3, F          ;
;
    MOVF     ARG1H, W         ;
    MULWF    ARG2L            ; ARG1H * ARG2L->
                              ; PRODH:PRODL
    MOVF     PRODL, W         ;
    ADDWF    RES1, F          ; Add cross
    MOVF     PRODH, W         ; products
    ADDWFC   RES2, F          ;
    CLRF     WREG             ;
    ADDWFC   RES3, F          ;
```

Example 12-4 shows the sequence to do a 16 x 16 signed multiply. Equation 12-2 shows the algorithm used. The 32-bit result is stored in four registers (RES<3:0>). To account for the sign bits of the arguments, the MSb for each argument pair is tested and the appropriate subtractions are done.

EQUATION 12-2: 16 x 16 SIGNED MULTIPLICATION ALGORITHM

$$\begin{aligned}
\text{RES3:RES0} &= \text{ARG1H:ARG1L} \bullet \text{ARG2H:ARG2L} \\
&= (\text{ARG1H} \bullet \text{ARG2H} \bullet 2^{16}) + \\
&\quad (\text{ARG1H} \bullet \text{ARG2L} \bullet 2^{8}) + \\
&\quad (\text{ARG1L} \bullet \text{ARG2H} \bullet 2^{8}) + \\
&\quad (\text{ARG1L} \bullet \text{ARG2L}) + \\
&\quad (-1 \bullet \text{ARG2H<7>} \bullet \text{ARG1H:ARG1L} \bullet 2^{16}) + \\
&\quad (-1 \bullet \text{ARG1H<7>} \bullet \text{ARG2H:ARG2L} \bullet 2^{16})
\end{aligned}$$

EXAMPLE 12-4: 16 x 16 SIGNED MULTIPLY ROUTINE

```
    MOVF     ARG1L, W
    MULWF    ARG2L            ; ARG1L * ARG2L ->
                              ; PRODH:PRODL
    MOVFF    PRODH, RES1      ;
    MOVFF    PRODL, RES0      ;
;
    MOVF     ARG1H, W
    MULWF    ARG2H            ; ARG1H * ARG2H ->
                              ; PRODH:PRODL
    MOVFF    PRODH, RES3      ;
    MOVFF    PRODL, RES2      ;
;
    MOVF     ARG1L, W
    MULWF    ARG2H            ; ARG1L * ARG2H ->
                              ; PRODH:PRODL
    MOVF     PRODL, W         ;
    ADDWF    RES1, F          ; Add cross
    MOVF     PRODH, W         ; products
    ADDWFC   RES2, F          ;
    CLRF     WREG             ;
    ADDWFC   RES3, F          ;
;
    MOVF     ARG1H, W         ;
    MULWF    ARG2L            ; ARG1H * ARG2L ->
                              ; PRODH:PRODL
    MOVF     PRODL, W         ;
    ADDWF    RES1, F          ; Add cross
    MOVF     PRODH, W         ; products
    ADDWFC   RES2, F          ;
    CLRF     WREG             ;
    ADDWFC   RES3, F          ;
;
    BTFSS    ARG2H, 7         ; ARG2H:ARG2L neg?
    BRA      SIGN_ARG1        ; no, check ARG1
    MOVF     ARG1L, W         ;
    SUBWF    RES2             ;
    MOVF     ARG1H, W         ;
    SUBWFB   RES3
;
SIGN_ARG1
    BTFSS    ARG1H, 7         ; ARG1H:ARG1L neg?
    BRA      CONT_CODE        ; no, done
    MOVF     ARG2L, W         ;
    SUBWF    RES2             ;
    MOVF     ARG2H, W         ;
    SUBWFB   RES3
;
CONT_CODE
    :
```

REGISTER 13-12: SCANLADRU: SCAN LOW ADDRESS UPPER BYTE REGISTER

U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	LADR<21:16>[1,2]					
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	

bit 7-6 **Unimplemented:** Read as ‘0’

bit 5-0 **LADR<21:16>:** Scan Start/Current Address bits[1,2]
Upper bits of the current address to be fetched from, value increments on each fetch of memory.

Note 1: Registers SCANLADRU/H/L form a 22-bit value, but are not guarded for atomic or asynchronous access; registers should only be read or written while SCANGO = 0 (SCANCON0 register).

2: While SCANGO = 1 (SCANCON0 register), writing to this register is ignored.

REGISTER 13-13: SCANLADRH: SCAN LOW ADDRESS HIGH BYTE REGISTER

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
LADR<15:8>[1, 2]							
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	

bit 7-0 **LADR<15:8>**: Scan Start/Current Address bits[1, 2]
Most Significant bits of the current address to be fetched from, value increments on each fetch of memory.

Note 1: Registers SCANLADRU/H/L form a 22-bit value, but are not guarded for atomic or asynchronous access; registers should only be read or written while SCANGO = 0 (SCANCON0 register).

2: While SCANGO = 1 (SCANCON0 register), writing to this register is ignored.

TABLE 13-5: SUMMARY OF REGISTERS ASSOCIATED WITH CRC

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
CRCACCH	ACC<15:8>								148
CRCACCL	ACC<7:0>								149
CRCCON0	EN	GO	BUSY	ACCM	—	—	SHIFTM	FULL	147
CRCCON1	DLEN<3:0>				PLEN<3:0>				147
CRCDATH	DATA<15:8>								148
CRCDATL	DATA<7:0>								148
CRCSHIFTH	SHIFT<15:8>								149
CRCSHIFTL	SHIFT<7:0>								149
CRCXORH	X<15:8>								150
CRCXORL	X<7:1>							—	150
PMD0	SYSCMD	FVRMD	HLVDMD	CRCMD	SCANMD	NVMMD	CLKRMD	IOCMD	64
SCANCON0	SCANEN	SCANGO	BUSY	INVALID	INTM	—	MODE<1:0>		151
SCANHADRU	—	—	HADR<21:16>						153
SCANHADRH	HADR<15:8>								154
SCANHADRL	HADR<7:0>								154
SCANLADRU	—	—	LADR<21:16>						152
SCANLADRH	LADR<15:8>								152
SCANLADRL	LADR<7:0>								153
SCANTRIG	—	—	—	—	TSEL<3:0>				155
INTCON	GIE/GIEH	PEIE/GIEL	IPEN	—	—	INT2EDG	INT1EDG	INT0EDG	166
PIR7	SCANIF	CRCIF	NVMIF	—	—	—	—	CWG1IF	174
PIE7	SCANIE	CRCIE	NVMIE	—	—	—	—	CWG1IE	182
IPR7	SCANIP	CRCIP	NVMIP	—	—	—	—	CWG1IP	190

Legend: — = unimplemented location, read as ‘0’. Shaded cells are not used for the CRC module.

14.8 Register Definitions: Interrupt Control

REGISTER 14-1: INTCON: INTERRUPT CONTROL REGISTER

R/W-0/0	R/W-0/0	R/W-0/0	U-0	U-0	R/W-1/1	R/W-1/1	R/W-1/1
GIE/GIEH	PEIE/GIEL	IPEN	—	—	INT2EDG	INT1EDG	INT0EDG
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

If IPEN = 1:
1 = Enables all unmasked interrupts and cleared by hardware for high-priority interrupts only
0 = Disables all interrupts

If IPEN = 0:
1 = Enables all unmasked interrupts and cleared by hardware for all interrupts
0 = Disables all interrupts

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

If IPEN = 1:
1 = Enables all low-priority interrupts and cleared by hardware for low-priority interrupts only
0 = Disables all low-priority interrupts

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

bit 5 **IPEN:** Interrupt Priority Enable bit
1 = Enable priority levels on interrupts
0 = Disable priority levels on interrupts

bit 4-3 **Unimplemented**: Read as ‘0’

bit 2 **INT2EDG:** External Interrupt 2 Edge Select bit
1 = Interrupt on rising edge of INT2 pin
0 = Interrupt on falling edge of INT2 pin

bit 1 **INT1EDG:** External Interrupt 1 Edge Select bit
1 = Interrupt on rising edge of INT1 pin
0 = Interrupt on falling edge of INT1 pin

bit 0 **INT0EDG:** External Interrupt 0 Edge Select bit
1 = Interrupt on rising edge of INT0 pin
0 = Interrupt on falling edge of INT0 pin

Note: Interrupt flag bits are set when an interrupt condition occurs, regardless of the state of its corresponding enable bit or the global 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 15-6: ODCONx: OPEN-DRAIN CONTROL REGISTER

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
ODCx7	ODCx6	ODCx5	ODCx4	ODCx3	ODCx2	ODCx1	ODCx0
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown
-n/n = Value at POR and BOR/Value at all other Resets		

bit 7-0 **ODCx<7:0>**: Open-Drain Configuration on Pins Rx<7:0>
1 = Output drives only low-going signals (sink current only)
0 = Output drives both high-going and low-going signals (source and sink current)

TABLE 15-7: OPEN-DRAIN CONTROL REGISTERS

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ODCONA	ODCA7	ODCA6	ODCA5	ODCA4	ODCA3	ODCA2	ODCA1	ODCA0
ODCONB	ODCB7	ODCB6	ODCB5	ODCB4	ODCB3	ODCB2	ODCB1	ODCB0
ODCONC	ODCC7	ODCC6	ODCC5	ODCC4	ODCC3	ODCC2	ODCC1	ODCC0

16.0 INTERRUPT-ON-CHANGE

PORTA, PORTB, PORTC and pin RE3 of PORTE can be configured to operate as Interrupt-on-Change (IOC) pins on PIC18(L)F2x/4xK40 family devices. An interrupt can be generated by detecting a signal that has either a rising edge or a falling edge. Any individual port pin, or combination of port pins, can be configured to generate an interrupt. The interrupt-on-change module has the following features:

- Interrupt-on-Change enable (Master Switch)
- Individual pin configuration
- Rising and falling edge detection
- Individual pin interrupt flags

Figure 16-1 is a block diagram of the IOC module.

16.1 Enabling the Module

To allow individual port pins to generate an interrupt, the IOCIE bit of the PIE0 register must be set. If the IOCIE bit is disabled, the edge detection on the pin will still occur, but an interrupt will not be generated.

16.2 Individual Pin Configuration

For each port pin, a rising edge detector and a falling edge detector are present. To enable a pin to detect a rising edge, the associated bit of the IOCxP register is set. To enable a pin to detect a falling edge, the associated bit of the IOCxN register is set.

A pin can be configured to detect rising and falling edges simultaneously by setting both associated bits of the IOCxP and IOCxN registers, respectively.

16.3 Interrupt Flags

The IOCAFx, IOCBFx, IOCCFx and IOCEF3 bits located in the IOCAF, IOCBF, IOCCF and IOCEF registers respectively, are status flags that correspond to the interrupt-on-change pins of the associated port. If an expected edge is detected on an appropriately enabled pin, then the status flag for that pin will be set, and an interrupt will be generated if the IOCIE bit is set. The IOCIF bit of the PIR0 register reflects the status of all IOCAFx, IOCBFx, IOCCFx and IOCEF3 bits.

16.4 Clearing Interrupt Flags

The individual status flags, (IOCAFx, IOCBFx, IOCCFx and IOCEF3 bits), can be cleared by resetting them to zero. If another edge is detected during this clearing operation, the associated status flag will be set at the end of the sequence, regardless of the value actually being written.

In order to ensure that no detected edge is lost while clearing flags, only AND operations masking out known changed bits should be performed. The following sequence is an example of what should be performed.

EXAMPLE 16-1: CLEARING INTERRUPT FLAGS (PORTA EXAMPLE)

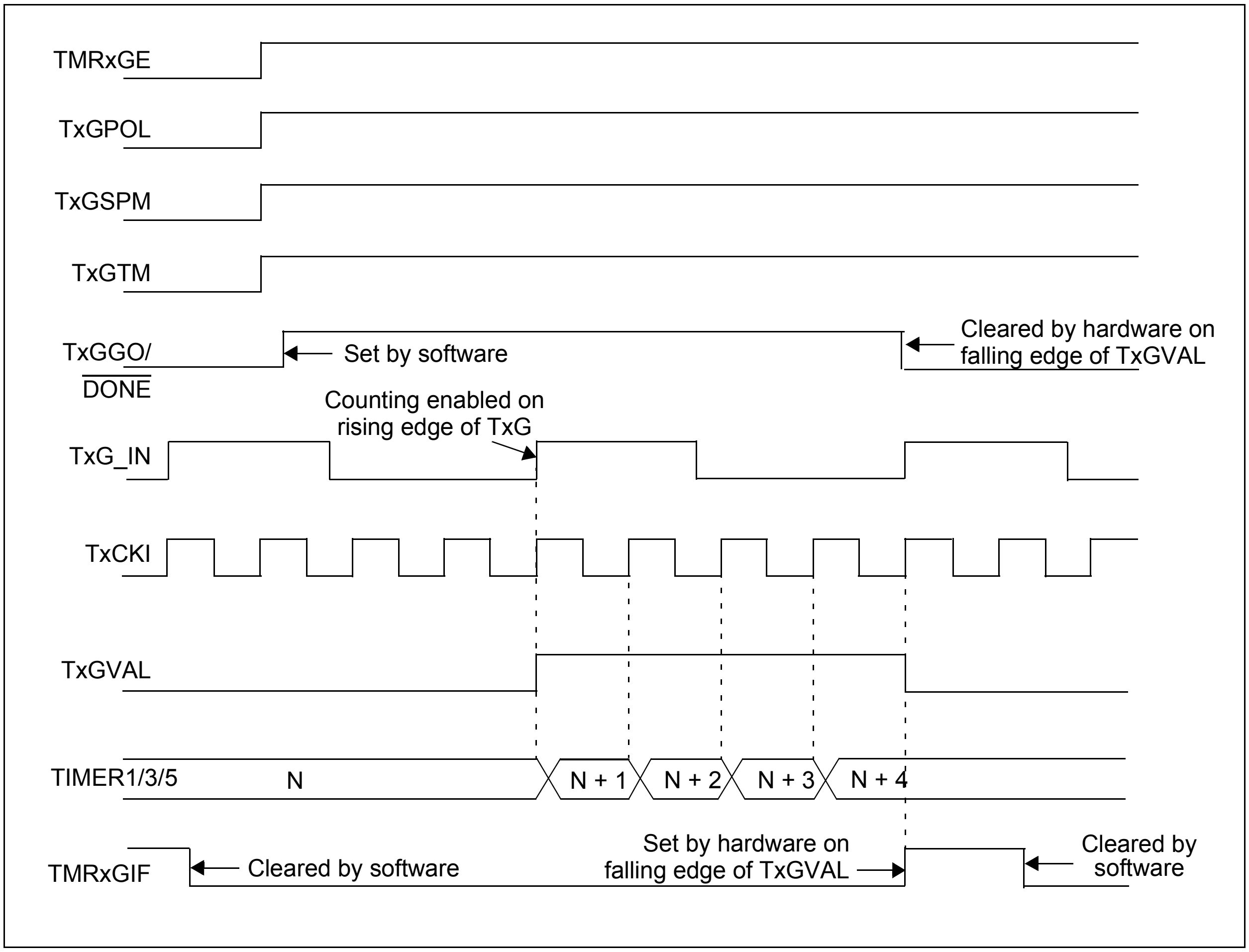
```
MOVLW   0xff
XORWF   IOCAF, W
ANDWF   IOCAF, F
```

16.5 Operation in Sleep

The interrupt-on-change interrupt sequence will wake the device from Sleep mode, if the IOCIE bit is set.

If an edge is detected while in Sleep mode, the IOCxF register will be updated prior to the first instruction executed out of Sleep.

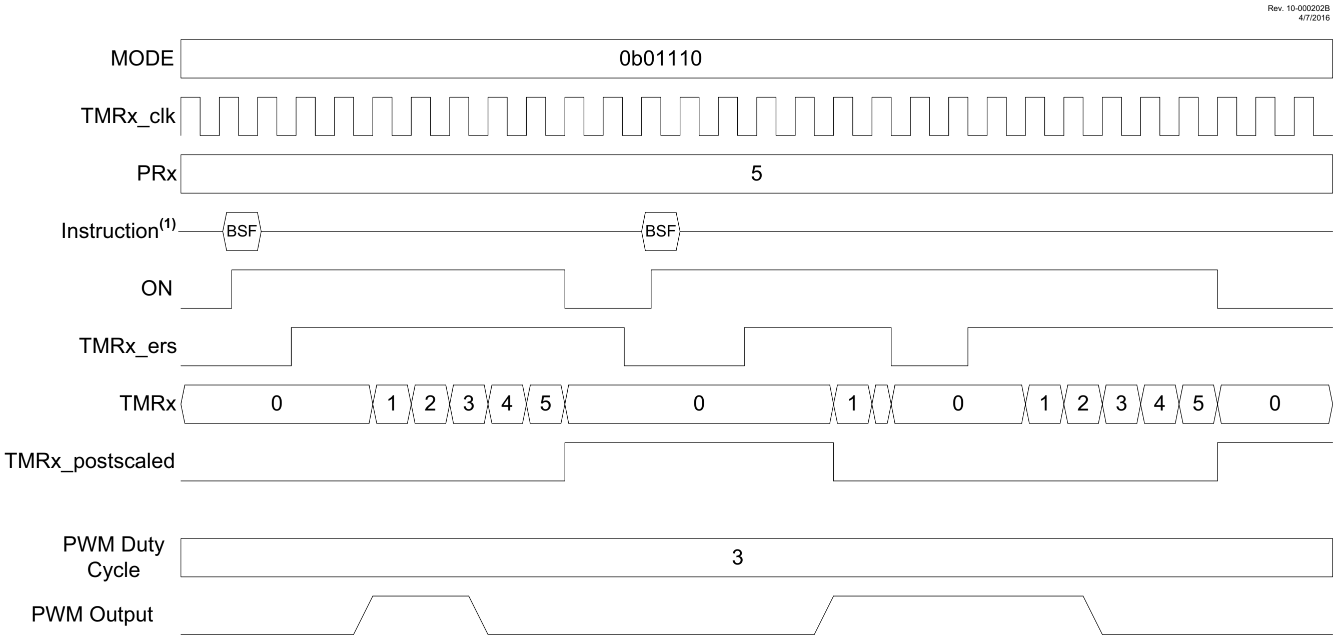
FIGURE 19-7: TIMER1/3/5 GATE SINGLE-PULSE AND TOGGLE COMBINED MODE



19.13 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 PMD1 register. See **Section 7.0 "Peripheral Module Disable (PMD)"** for more information.

FIGURE 20-11: LOW LEVEL RESET, EDGE-TRIGGERED HARDWARE LIMIT ONE-SHOT MODE TIMING DIAGRAM (MODE = `01110`)



Note 1: BSF and BCF represent Bit-Set File and Bit-Clear File instructions executed by the CPU to set or clear the ON bit of TxCON. CPU execution is asynchronous to the timer clock input.

24.9 Dead-Band Jitter

When the rising and falling edges of the input source are asynchronous to the CWG clock, it creates jitter in the dead-band time delay. The maximum jitter is equal to one CWG clock period. Refer to Equation 24-1 for more details.

EQUATION 24-1: DEAD-BAND DELAY TIME CALCULATION

$$T_{DEAD-BAND\_MIN} = \frac{1}{F_{CWG\ CLOCK}} \bullet DBx < 4:0>$$

$$T_{DEAD-BANDMAX} = \frac{1}{F_{CWG\ CLOCK}} \bullet DBx < 4:0>+1$$

$$T_{JITTER} = T_{DEAD-BAND\_MAX} - T_{DEAD-BAND\_MIN}$$

$$T_{JITTER} = \frac{1}{F_{CWG\_CLOCK}}$$

$$T_{DEAD-BAND\_MAX} = T_{DEAD-BAND\_MIN} + T_{JITTER}$$

$EXAMPLE$

$$DBR<4:0>= 0x0A = 10$$

$$F_{CWG\_CLOCK} = 8\ MHz$$

$$T_{JITTER} = \frac{1}{8MHz} = 125\ \text{ns}$$

$$T_{DEAD-BAND\_MIN} = 125\ ns*10 = 125\ \mu s$$

$$T_{DEAD-BAND\_MAX} = 1.25\ \mu s + 0.125\mu s = 1.37\mu s$$

REGISTER 24-7: CWG1AS1: CWG AUTO-SHUTDOWN CONTROL REGISTER 1

U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	AS5E	AS4E	AS3E	AS2E	AS1E	AS0E
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	q = Value depends on condition

bit 7-6 **Unimplemented** Read as ‘0’

bit 5 **AS5E:** CWG Auto-shutdown Source 5 (CMP2 OUT) Enable bit
1 = Auto-shutdown for CMP2 OUT is enabled
0 = Auto-shutdown for CMP2 OUT is disabled

bit 4 **AS4E:** CWG Auto-shutdown Source 4 (CMP1 OUT) Enable bit
1 = Auto-shutdown for CMP1 OUT is enabled
0 = Auto-shutdown for CMP1 OUT is disabled

bit 3 **AS3E:** CWG Auto-shutdown Source 3 (TMR6_Postscaled) Enable bit
1 = Auto-shutdown for TMR6_Postscaled is enabled
0 = Auto-shutdown for TMR6_Postscaled is disabled

bit 2 **AS2E:** CWG Auto-shutdown Source 2 (TMR4_Postscaled) Enable bit
1 = Auto-shutdown for TMR4_Postscaled is enabled
0 = Auto-shutdown for TMR4_Postscaled is disabled

bit 1 **AS1E:** CWG Auto-shutdown Source 1 (TMR2_Postscaled) Enable bit
1 = Auto-shutdown for TMR2_Postscaled is enabled
0 = Auto-shutdown for TMR2_Postscaled is disabled

bit 0 **AS0E:** CWG Auto-shutdown Source 0 (Pin selected by CWG1PPS) Enable bit
1 = Auto-shutdown for CWG1PPS Pin is enabled
0 = Auto-shutdown for CWG1PPS Pin is disabled

REGISTER 24-8: CWG1DBR: CWG RISING DEAD-BAND COUNT REGISTER

U-0	U-0	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
—	—	DBR<5:0>					
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	q = Value depends on condition

bit 7-6 **Unimplemented:** Read as ‘0’

bit 5-0 **DBR<5:0>:** CWG Rising Edge Triggered Dead-Band Count bits

```
11 1111 =  63-64 CWG clock periods
11 1110 =  62-63 CWG clock periods
.
.
.
00 0010 =  2-3 CWG clock periods
00 0001 =  1-2 CWG clock periods
00 0000 =  0 CWG clock periods. Dead-band generation is bypassed
```

REGISTER 24-9: CWG1DBF: CWG FALLING DEAD-BAND COUNT REGISTER

U-0	U-0	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
—	—	DBF<5:0>					
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	q = Value depends on condition

bit 7-6 **Unimplemented:** Read as ‘0’

bit 5-0 **DBF<5:0>:** CWG Falling Edge Triggered Dead-Band Count bits

```
11 1111 =  63-64 CWG clock periods
11 1110 =  62-63 CWG clock periods
.
.
.
00 0010 =  2-3 CWG clock periods
00 0001 =  1-2 CWG clock periods
00 0000 =  0 CWG clock periods. Dead-band generation is bypassed.
```

FIGURE 26-7: SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 0)

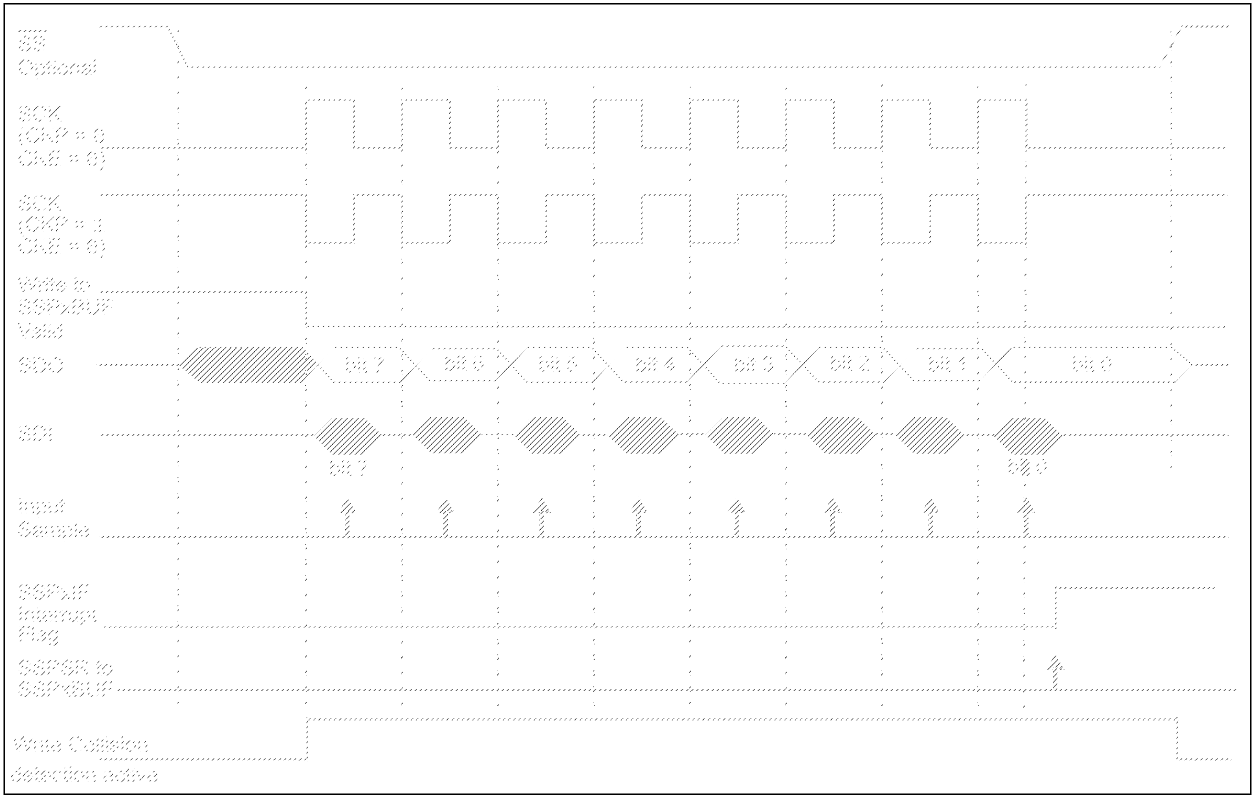


FIGURE 26-8: SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 1)

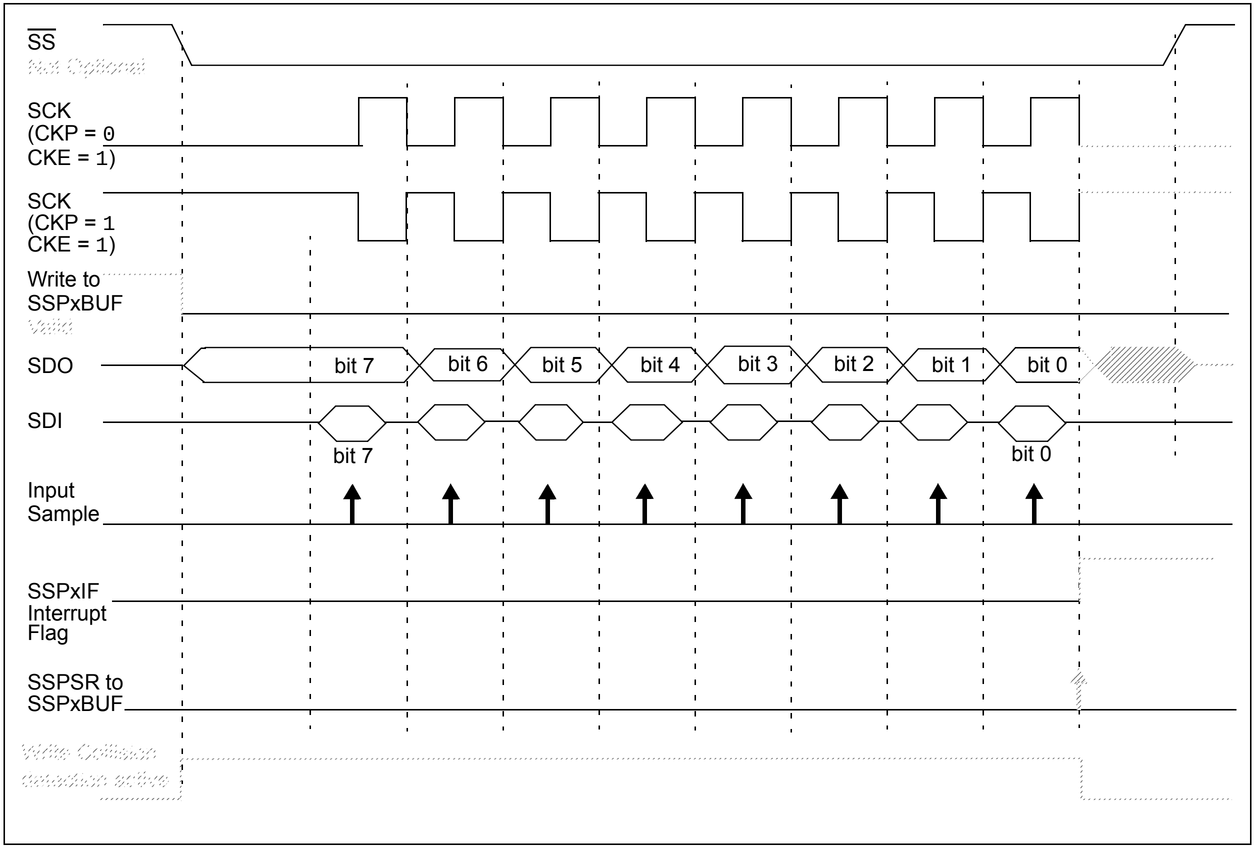
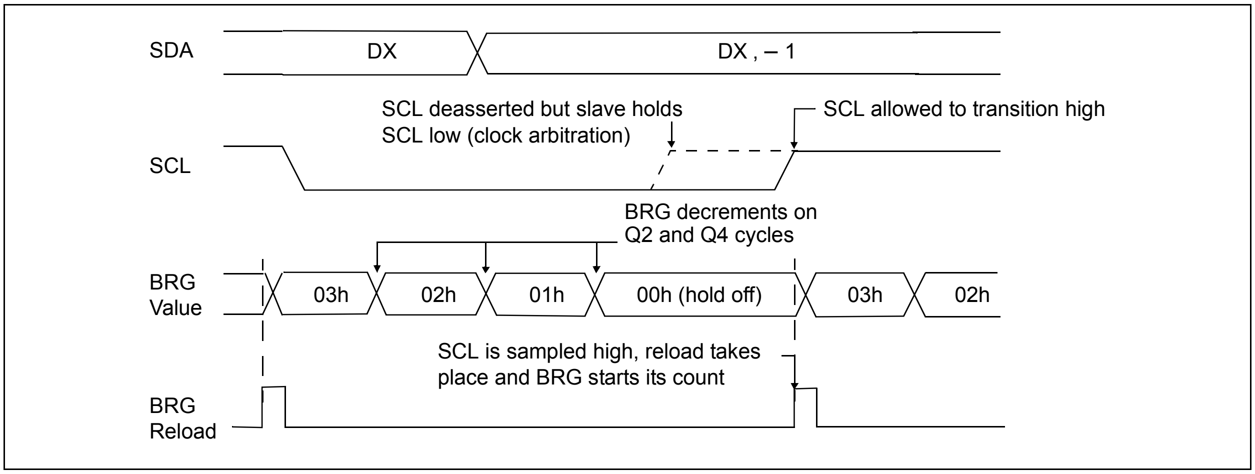


FIGURE 26-25: BAUD RATE GENERATOR TIMING WITH CLOCK ARBITRATION


SDA
DX
DX , – 1
SCL deasserted but slave holds
SCL low (clock arbitration)
SCL allowed to transition high
SCL
BRG decrements on
Q2 and Q4 cycles
BRG
Value
03h
02h
01h
00h (hold off)
03h
02h
SCL is sampled high, reload takes
place and BRG starts its count
BRG
Reload

26.10.3 WCOL STATUS FLAG

If the user writes the SSPxBUF when a Start, Restart, Stop, Receive or Transmit sequence is in progress, the WCOL bit 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 queuing of events is not allowed, writing to the lower five bits of SSPxCON2 is disabled until the Start condition is complete.

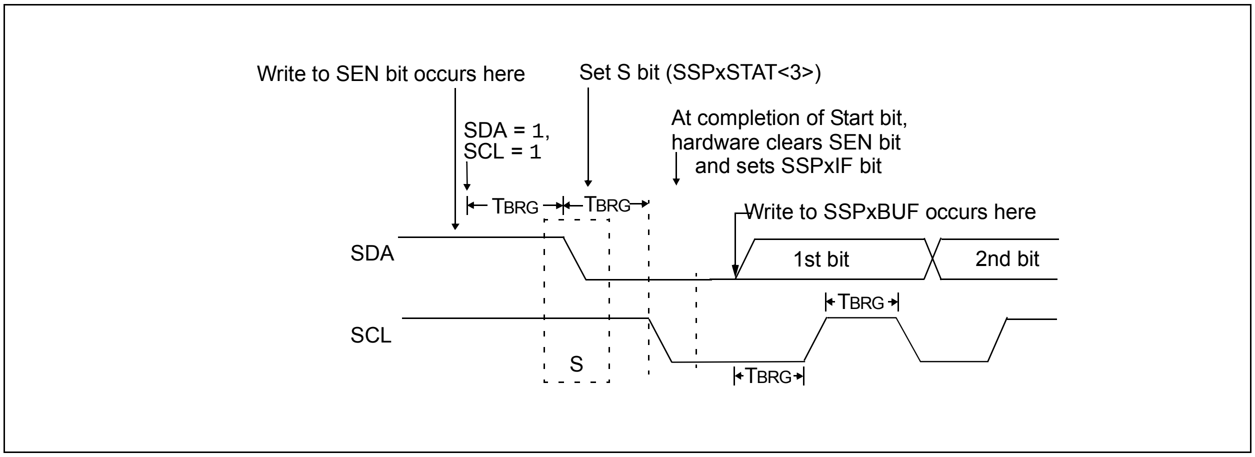
26.10.4 I²C MASTER MODE START CONDITION TIMING

To initiate a Start condition (Figure 26-26), the user sets the Start Enable bit, SEN bit of the SSPxCON2 register. If the SDA and SCL pins are sampled high, the Baud Rate Generator is reloaded with the contents of SSPxADD<7:0> and starts its count. If SCL and SDA are both sampled high when the Baud Rate Generator times out (TBRG), the SDA pin is driven low. The action of the SDA being driven low while SCL is high is the Start condition and causes the S bit of the SSPxSTAT1 register to be set. Following this, the Baud Rate Generator is reloaded with the contents of SSPxADD<7:0> and resumes its count. When the Baud Rate Generator times out (TBRG), the SEN bit of the SSPxCON2 register will be automatically cleared by hardware; the Baud Rate Generator is suspended, leaving the SDA line held low and the Start condition is complete.

Note 1: If at the beginning of the Start condition, the SDA and SCL pins are already sampled low, or if during the Start condition, the SCL line is sampled low before the SDA line is driven low, a bus collision occurs, the Bus Collision Interrupt Flag, BCLxIF, is set, the Start condition is aborted and the I²C module is reset into its Idle state.

2: The Philips I²C specification states that a bus collision cannot occur on a Start.

FIGURE 26-26: FIRST START BIT TIMING



27.5.2 SYNCHRONOUS SLAVE MODE

The following bits are used to configure the EUSART for synchronous slave operation:

- SYNC = 1
- CSRC = 0
- SREN = 0 (for transmit); SREN = 1 (for receive)
- CREN = 0 (for transmit); CREN = 1 (for receive)
- SPEN = 1

Setting the SYNC bit of the TXxSTA register configures the device for synchronous operation. Clearing the CSRC bit of the TXxSTA register configures the device as a slave. Clearing the SREN and CREN bits of the RCxSTA register ensures that the device is in the Transmit mode, otherwise the device will be configured to receive. Setting the SPEN bit of the RCxSTA register enables the EUSART.

27.5.2.1 EUSART Synchronous Slave Transmit

The operation of the Synchronous Master and Slave modes are identical (see **Section 27.5.1.3 "Synchronous Master Transmission")**, except in the case of the Sleep mode.

If two words are written to the TXxREG and then the SLEEP instruction is executed, the following will occur:

1. The first character will immediately transfer to the TSR register and transmit.
2. The second word will remain in the TXxREG register.
3. The TXxIF bit will not be set.
4. After the first character has been shifted out of TSR, the TXxREG register will transfer the second character to the TSR and the TXxIF bit will now be set.
5. If the PEIE and TXxIE bits are set, the interrupt will wake the device from Sleep and execute the next instruction. If the GIE bit is also set, the program will call the Interrupt Service Routine.

27.5.2.2 Synchronous Slave Transmission Setup

1. Set the SYNC and SPEN bits and clear the CSRC bit.
2. Clear the ANSEL bit for the CKx pin (if applicable).
3. Clear the CREN and SREN bits.
4. If interrupts are desired, set the TXxIE bit of the PIE3 register and the GIE and PEIE bits of the INTCON register.
5. If 9-bit transmission is desired, set the TX9 bit.
6. Enable transmission by setting the TXEN bit.
7. If 9-bit transmission is selected, insert the Most Significant bit into the TX9D bit.
8. Start transmission by writing the Least Significant eight bits to the TXxREG register.

REGISTER 31-24: ADSTPTH: ADC THRESHOLD SETPOINT REGISTER HIGH

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
ADSTPT<15:8>							
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	

bit 7-0 **ADSTPT<15:8>**: ADC Threshold Setpoint MSB. Upper byte of ADC threshold setpoint, depending on ADCALC, may be used to determine ADERR, see Register 23-1 for more details.

REGISTER 31-25: ADSTPTL: ADC THRESHOLD SETPOINT REGISTER LOW

R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x
ADSTPT<7:0>							
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	

bit 7-0 **ADSTPT<7:0>**: ADC Threshold Setpoint LSB. Lower byte of ADC threshold setpoint, depending on ADCALC, may be used to determine ADERR, see Register 23-1 for more details.

REGISTER 31-26: ADERRH: ADC SETPOINT ERROR REGISTER HIGH

R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
ADERR<7:0>							
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	

bit 7-0 **ADERR<7:0>**: ADC Setpoint Error MSB. Upper byte of ADC Setpoint Error. Setpoint Error calculation is determined by ADCALC bits of ADCON3, see Register 23-1 for more details.

MULLW Multiply literal with W

Syntax: MULLW k

Operands: $0 \le k \le 255$

Operation: (W) x k → PRODH:PRODL

Status Affected: None

Encoding:

0000	1101	kkkk	kkkk

Description: An unsigned multiplication is carried out between the contents of W and the 8-bit literal ‘k’. The 16-bit result is placed in the PRODH:PRODL register pair. PRODH contains the high byte. W is unchanged.
None of the Status flags are affected.
Note that neither overflow nor carry is possible in this operation. A zero result is possible but not detected.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal ‘k’	Process Data	Write registers PRODH: PRODL

Example: `MULLW 0C4h`

Before Instruction

W	=	E2h
PRODH	=	?
PRODL	=	?

After Instruction

W	=	E2h
PRODH	=	ADh
PRODL	=	08h

MULWF Multiply W with f

Syntax: MULWF f {,a}

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

Operation: (W) x (f) → PRODH:PRODL

Status Affected: None

Encoding:

0000	001a	ffff	ffff

Description: An unsigned multiplication is carried out between the contents of W and the register file location ‘f’. The 16-bit result is stored in the PRODH:PRODL register pair. PRODH contains the high byte. Both W and ‘f’ are unchanged.
None of the Status flags are affected.
Note that neither overflow nor carry is possible in this operation. A zero result is possible but not detected.
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 35.2.3 “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 registers PRODH: PRODL

Example: `MULWF REG, 1`

Before Instruction

W	=	C4h
REG	=	B5h
PRODH	=	?
PRODL	=	?

After Instruction

W	=	C4h
REG	=	B5h
PRODH	=	8Ah
PRODL	=	94h

TABLE 37-2: SUPPLY CURRENT (IDD)$^{(1,2,4)}$

PIC18LF24/25K40			Standard Operating Conditions (unless otherwise stated)					
PIC18F24/25K40								
Param. No.	Symbol	Device Characteristics	Min.	Typ.†	Max.	Units	Conditions	
							VDD	Note
D100	IDD$_{XT4}$	XT = 4 MHz	—	450	650	μA	3.0V	
D100	IDD$_{XT4}$	XT = 4 MHz	—	550	750	μA	3.0V	
D100A	IDD$_{XT4}$	XT = 4 MHz	—	310	—	μA	3.0V	PMD's all 1's
D100A	IDD$_{XT4}$	XT = 4 MHz	—	410	—	μA	3.0V	PMD's all 1's
D101	IDD$_{HFO16}$	HFINTOSC = 16 MHz	—	1.9	2.6	mA	3.0V	
D101	IDD$_{HFO16}$	HFINTOSC = 16 MHz	—	2.0	2.7	mA	3.0V	
D101A	IDD$_{HFO16}$	HFINTOSC = 16 MHz	—	1.4	—	mA	3.0V	PMD's all 1's
D101A	IDD$_{HFO16}$	HFINTOSC = 16 MHz	—	1.5	—	mA	3.0V	PMD's all 1's
D102	IDD$_{HFOPLL}$	HFINTOSC = 64 MHz	—	7.4	9.4	mA	3.0V	
D102	IDD$_{HFOPLL}$	HFINTOSC = 64 MHz	—	7.5	9.5	mA	3.0V	
D102A	IDD$_{HFOPLL}$	HFINTOSC = 64 MHz	—	5.2	—	mA	3.0V	PMD's all 1's
D102A	IDD$_{HFOPLL}$	HFINTOSC = 64 MHz	—	5.3	—	mA	3.0V	PMD's all 1's
D103	IDD$_{HSPLL32}$	HS+PLL = 64 MHz	—	6.9	8.9	mA	3.0V	
D103	IDD$_{HSPLL32}$	HS+PLL = 64 MHz	—	7.0	9.0	mA	3.0V	
D103A	IDD$_{HSPLL32}$	HS+PLL = 64 MHz	—	4.9	—	mA	3.0V	PMD's all 1's
D103A	IDD$_{HSPLL32}$	HS+PLL = 64 MHz	—	5.0	—	mA	3.0V	PMD's all 1's
D104	IDD$_{IDLE}$	IDLE mode, HFINTOSC = 16 MHz	—	1.05	—	mA	3.0V	
D104	IDD$_{IDLE}$	IDLE mode, HFINTOSC = 16 MHz	—	1.15	—	mA	3.0V	
D105	IDD$_{DOZE}^{(3)}$	DOZE mode, HFINTOSC = 16 MHz, Doze Ratio = 16	—	1.1	—	mA	3.0V	
D105	IDD$_{DOZE}^{(3)}$	DOZE mode, HFINTOSC = 16 MHz, Doze Ratio = 16	—	1.2	—	mA	3.0V	

† Data in "Typ." column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: The test conditions for all IDD measurements in active operation mode are: OSC1 = external square wave, from rail-to-rail; all I/O pins are outputs driven low; $\overline{\text{MCLR}}$ = VDD; WDT disabled.

2: The supply current is mainly a function of the operating voltage and frequency. Other factors, such as I/O pin loading and switching rate, oscillator type, internal code execution pattern and temperature, also have an impact on the current consumption.

3: IDD$_{DOZE}$ = [IDD$_{IDLE}$*(N-1)/N] + IDD$_{HFO}$16/N where N = DOZE Ratio (Register 6-2).

4: PMD bits are all in the default state, no modules are disabled.