Section 17. Master Synchronous Serial Port (MSSP)

HIGHLIGHTS

This section of the manual contains the following major topics:

17.1 Introduction ............................................................................................................... 17-2
17.2 Control Register ........................................................................................................... 17-4
17.3 SPI Mode ................................................................................................................... 17-9
17.4 SSP I²C™ Operation ................................................................................................. 17-18
17.5 Connection Considerations for I²C Bus ................................................................. 17-56
17.6 Initialization ............................................................................................................. 17-57
17.7 Design Tips .............................................................................................................. 17-58
17.8 Related Application Notes ....................................................................................... 17-59
17.9 Revision History ....................................................................................................... 17-60

Note: At present NO Mid-Range MCU devices are available with this module. Devices are planned, but there is no schedule for availability. Please refer to Microchip's Web site or BBS for release of Product Briefs. You will be able to find out the details and the features for new devices.

This module is available on Microchip's High End family (PIC17CXXX). Please refer to Microchip's Web site, BBS, Regional Sales Office, or Factory Representatives.

I²C is a trademark of Philips Corporation.
17.1 Introduction

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

- Serial Peripheral Interface (SPI™)
- Inter-Integrated Circuit (I²C™)
  - Full Master Mode
  - Slave mode (with general address call)

Figure 17-1 shows a block diagram for the SPI mode, while Figure 17-2, and Figure 17-3 show the block diagrams for the two different I²C modes of operation.

Figure 17-1: SPI Mode Block Diagram
Figure 17-2: I²C Slave Mode Block Diagram

Figure 17-3: I²C Master Mode Block Diagram
## Control Register

### Register 17-1: SSPSTAT: SSP Status Register

<table>
<thead>
<tr>
<th>R/W-0</th>
<th>R/W-0</th>
<th>R-0</th>
<th>R-0</th>
<th>R-0</th>
<th>R-0</th>
<th>R-0</th>
<th>R-0</th>
<th>R-0</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMP</td>
<td>CKE</td>
<td>D/A</td>
<td>P</td>
<td>S</td>
<td>R/W</td>
<td>UA</td>
<td>BF</td>
<td></td>
</tr>
</tbody>
</table>

**bit 7**  
**SMP**: Sample bit  
SPI Master Mode  
1 = Input data sampled at end of data output time  
0 = Input data sampled at middle of data output time  

SPI Slave Mode  
SMP must be cleared when SPI is used in slave mode  

In **I²C master or slave mode**:

1 = Slew rate control disabled for standard speed mode (100 kHz and 1 MHz)  
0 = Slew rate control enabled for high speed mode (400 kHz)

**bit 6**  
**CKE**: SPI Clock Edge Select  

<table>
<thead>
<tr>
<th>CKP</th>
<th></th>
</tr>
</thead>
</table>
| 0    | 1 = Data transmitted on rising edge of SCK  
0 = Data transmitted on falling edge of SCK  
| 1    | 1 = Data transmitted on falling edge of SCK  
0 = Data transmitted on rising edge of SCK  

**bit 5**  
**D/A**: Data/Address bit (**I²C** mode only)  
1 = Indicates that the last byte received or transmitted was data  
0 = Indicates that the last byte received or transmitted was address  

**bit 4**  
**P**: Stop bit  
(**I²C** mode only. This bit is cleared when the SSP module is disabled, SSPEN is cleared)  
1 = Indicates that a stop bit has been detected last (this bit is '0' on RESET)  
0 = Stop bit was not detected last  

**bit 3**  
**S**: Start bit  
(**I²C** mode only. This bit is cleared when the SSP module is disabled, SSPEN is cleared)  
1 = Indicates that a start bit has been detected last (this bit is '0' on RESET)  
0 = Start bit was not detected last  

**bit 2**  
**R/W**: Read/Write bit information (**I²C** mode only)  
This bit holds the R/W bit information following the last address match. This bit is only valid from the address match to the next start bit, stop bit, or not ACK bit.  

In **I²C slave mode**:

1 = Read  
0 = Write  

In **I²C master mode**:

1 = Transmit is in progress  
0 = Transmit is not in progress.  

Or'ing this bit with SEN, RSEN, PEN, RCEN, or ACKEN will indicate if the SSP is in IDLE mode.

**bit 1**  
**UA**: Update Address (10-bit **I²C** mode only)  
1 = Indicates that the user needs to update the address in the SSPADD register  
0 = Address does not need to be updated
### Register 17-1: SSPSTAT: SSP Status Register (Cont'd)

<table>
<thead>
<tr>
<th>Bit 0</th>
<th>BF: Buffer Full Status bit</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Receive (SPI and I²C modes)</td>
</tr>
<tr>
<td></td>
<td>1 = Receive complete, SSPBUF is full</td>
</tr>
<tr>
<td></td>
<td>0 = Receive not complete, SSPBUF is empty</td>
</tr>
<tr>
<td></td>
<td>Transmit (I²C mode only)</td>
</tr>
<tr>
<td></td>
<td>1 = Data Transmit in progress (does not include the ACK and stop bits), SSPBUF is full</td>
</tr>
<tr>
<td></td>
<td>0 = Data Transmit complete (does not include the ACK and stop bits), SSPBUF is empty</td>
</tr>
</tbody>
</table>

**Legend**

- **R** = Readable bit
- **W** = Writable bit
- **U** = Unimplemented bit, read as '0'
- **n** = Value at POR reset
Register 17-2: SSPCON1: SSP Control Register1

<table>
<thead>
<tr>
<th>R/W-0</th>
<th>R/W-0</th>
<th>R/W-0</th>
<th>R/W-0</th>
<th>R/W-0</th>
<th>R/W-0</th>
<th>R/W-0</th>
<th>R/W-0</th>
</tr>
</thead>
<tbody>
<tr>
<td>WCOL</td>
<td>SSPOV</td>
<td>SSPEN</td>
<td>CKP</td>
<td>SSPM3</td>
<td>SSPM2</td>
<td>SSPM1</td>
<td>SSPM0</td>
</tr>
</tbody>
</table>

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

bit 6  
**SSPOV**: Receive Overflow Indicator bit  
In **SPI mode**:  
1 = A new byte is received while the SSPBUF register is still holding the previous data. In case of overflow, the data in SSPSR is lost. Overflow can only occur in slave mode. In slave mode, the user must read the SSPBUF, 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 SSPBUF register.  
0 = No overflow  
In **I^2^C mode**:  
1 = A byte is received while the SSPBUF register is still holding the previous byte. SSPOV is a “don’t care” in transmit mode. SSPOV must be cleared in software in either mode. (must be cleared in software)  
0 = No overflow

bit 5  
**SSPEN**: 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 SCK, SDO, SDI, and SS as the source of the serial port pins  
0 = Disables serial port and configures these pins as I/O port pins  
In **I^2^C mode**:  
1 = Enables the serial port and configures the SDA and SCL pins as the source of the serial port pins  
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^2^C slave mode**:  
SCK release control  
1 = Enable clock  
0 = Holds clock low (clock stretch) (Used to ensure data setup time)  
In **I^2^C master mode**:  
Unused in this mode
### Section 17. MSSP

Register 17-2: **SSPCON1: SSP Control Register1 (Cont’d)**

<table>
<thead>
<tr>
<th>bit 3 - 0</th>
<th><strong>SSPM3:SSPM0</strong>: Synchronous Serial Port Mode Select bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>SPI master mode, clock = FOSC/4</td>
</tr>
<tr>
<td>0001</td>
<td>SPI master mode, clock = FOSC/16</td>
</tr>
<tr>
<td>0010</td>
<td>SPI master mode, clock = FOSC/64</td>
</tr>
<tr>
<td>0011</td>
<td>SPI master mode, clock = TMR2 output/2</td>
</tr>
<tr>
<td>0100</td>
<td>SPI slave mode, clock = SCK pin. SS pin control enabled.</td>
</tr>
<tr>
<td>0101</td>
<td>SPI slave mode, clock = SCK pin. SS pin control disabled.</td>
</tr>
<tr>
<td>0110</td>
<td>I2C slave mode, 7-bit address</td>
</tr>
<tr>
<td>0111</td>
<td>I2C slave mode, 10-bit address</td>
</tr>
<tr>
<td>1000</td>
<td>I2C master mode, clock = FOSC / (4 * (SSPADD+1))</td>
</tr>
<tr>
<td>1xx1</td>
<td>Reserved</td>
</tr>
<tr>
<td>1x1x</td>
<td>Reserved</td>
</tr>
</tbody>
</table>

**Legend**

- **R** = Readable bit
- **W** = Writable bit
- **U** = Unimplemented bit, read as ‘0’
- **- n** = Value at POR reset
Register 17-3: SSPCON2: SSP Control Register

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>GCEN</td>
<td>General Call Enable bit (In I²C slave mode only)</td>
</tr>
<tr>
<td></td>
<td>ACKSTAT</td>
<td>Acknowledge Status bit (In I²C master mode only)</td>
</tr>
<tr>
<td></td>
<td>ACKDT</td>
<td>Acknowledge Data bit (In I²C master mode only)</td>
</tr>
<tr>
<td></td>
<td>ACKEN</td>
<td>Acknowledge Sequence Enable bit (In I²C master mode only)</td>
</tr>
<tr>
<td></td>
<td>RCEN</td>
<td>Receive Enable bit (In I²C master mode only)</td>
</tr>
<tr>
<td></td>
<td>PEN</td>
<td>Stop Condition Enable bit (In I²C master mode only)</td>
</tr>
<tr>
<td></td>
<td>RSEN</td>
<td>Repeated Start Condition Enabled bit (In I²C master mode only)</td>
</tr>
<tr>
<td></td>
<td>SEN</td>
<td>Start Condition Enabled bit (In I²C master mode only)</td>
</tr>
</tbody>
</table>

**Legend**

- **R** = Readable bit
- **W** = Writable bit
- **U** = Unimplemented bit, read as ‘0’
- **- n** = Value at POR reset
17.3 SPI Mode

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

- Serial Data Out (SDO)
- Serial Data In (SDI)
- Serial Clock (SCK)

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

- Slave Select (SS)

17.3.1 Operation

When initializing the SPI, several options need to be specified. This is done by programming the appropriate control bits in the SSPCON1 register (SSPCON1<5:0>) and SSPSTAT<7:6>. These control bits allow the following to be specified:

- Master Mode (SCK is the clock output)
- Slave Mode (SCK is the clock input)
- Clock Polarity (Idle state of SCK)
- Data input sample phase (middle or end of data output time)
- Clock edge (output data on rising/falling edge of SCK)
- Clock Rate (Master mode only)
- Slave Select Mode (Slave mode only)

Figure 17-4 shows the block diagram of the SSP module, when in SPI mode.

Figure 17-4: SSP Block Diagram (SPI Mode)
The SSP consists of a transmit/receive Shift Register (SSPSR) and a buffer register (SSPBUF). The SSPSR shifts the data in and out of the device, MSb first. The SSPBUF holds the data that was written to the SSPSR, until the received data is ready. Once the 8-bits of data have been received, that byte is moved to the SSPBUF register. Then the buffer full detect bit, BF (SSPSTAT<0>), and the interrupt flag bit, SSPIF, are set. This double buffering of the received data (SSPBUF) allows the next byte to start reception before reading the data that was just received. Any write to the SSPBUF register during transmission/reception of data will be ignored, and the write collision detect bit, WCOL (SSPSTAT<7>), will be set. User software must clear the WCOL bit so that it can be determined if the following write(s) to the SSPBUF register completed successfully.

When the application software is expecting to receive valid data, the SSPBUF should be read before the next byte of data to transfer is written to the SSPBUF. Buffer full bit, BF (SSPSTAT<0>), indicates when SSPBUF has been loaded with the received data (transmission is complete). When the SSPBUF is read, the BF bit is cleared. This data may be irrelevant if the SPI is only a transmitter. Generally the SSP Interrupt is used to determine when the transmission/reception has completed. The SSPBUF must be read and/or written. If the interrupt method is not going to be used, then software polling can be done to ensure that a write collision does not occur. Example 17-1 shows the loading of the SSPBUF (SSPSR) for data transmission.

Example 17-1: Loading the SSPBUF (SSPSR) Register

```
BCF STATUS, RP1    ;Specify Bank1
BSF STATUS, RP0    ;
LOOP BTFSS SSPSTAT, BF    ;Has data been received (transmit complete)?
  GOTO LOOP           ;No
BCF STATUS, RP0    ;Specify Bank0
MOVF SSPBUF, W      ;W reg = contents of SSPBUF
MOVWF RXDATA         ;Save in user RAM, if data is meaningful
MOVF TXDATA, W      ;W reg = contents of TXDATA
MOVWF SSPBUF         ;New data to xmit
```

The SSPSR is not directly readable or writable, and can only be accessed by addressing the SSPBUF register. Additionally, the SSP status register (SSPSTAT) indicates the various status conditions.

17.3.2 Enabling SPI I/O

To enable the serial port, SSP Enable bit, SSPEN (SSPCON1<5>), must be set. To reset or reconfigure SPI mode, clear the SSPEN bit, re-initialize the SSPCON registers, and then set the SSPEN bit. This configures the SDI, SDO, SCK, and SS pins as serial port pins. For the pins to behave as the serial port function, some must have their data direction bits (in the TRIS register) appropriately programmed. That is:

- SDI is automatically controlled by the SPI module
- SDO must have TRIS bit cleared
- SCK (Master mode) must have TRIS bit cleared
- SCK (Slave mode) must have TRIS bit set
- SS must have TRIS bit set

Any serial port function that is not desired may be overridden by programming the corresponding data direction (TRIS) register to the opposite value.
17.3.3 Typical Connection

Figure 17-5 shows a typical connection between two microcontrollers. The master controller (Processor 1) initiates the data transfer by sending the SCK signal. Data is shifted out of both shift registers on their programmed clock edge, and latched on the opposite edge of the clock. Both processors should be programmed to the same Clock Polarity (CKP), then both controllers would send and receive data at the same time. Whether the data is meaningful (or dummy data) depends on the application software. This leads to three scenarios for data transmission:

- Master sends data — Slave sends dummy data
- Master sends data — Slave sends data
- Master sends dummy data — Slave sends data

**Figure 17-5: SPI Master/Slave Connection**
17.3.4 Master Mode

The master can initiate the data transfer at any time because it controls the SCK. The master determines when the slave (Processor 2, Figure 17-5) is to broadcast data by the software protocol.

In master mode the data is transmitted/received as soon as the SSPBUF register is written to. If the SPI is only going to receive, the SDO output could be disabled (programmed as an input). The SSPSR register will continue to shift in the signal present on the SDI pin at the programmed clock rate. As each byte is received, it will be loaded into the SSPBUF register as if a normal received byte (interrupts and status bits appropriately set). This could be useful in receiver applications as a "line activity monitor" mode.

The clock polarity is selected by appropriately programming the CKP bit (SSPCON1<4>). This then would give waveforms for SPI communication as shown in Figure 17-6, Figure 17-8, and Figure 17-9 where the MSb is transmitted first. In master mode, the SPI clock rate (bit rate) is user programmable to be one of the following:

- \( \frac{F_{OSC}}{4} \) (or \( T_{CY} \))
- \( \frac{F_{OSC}}{16} \) (or \( 4 \cdot T_{CY} \))
- \( \frac{F_{OSC}}{64} \) (or \( 16 \cdot T_{CY} \))
- Timer2 output/2

This allows a maximum data rate (at 20 MHz) of 8.25 Mbps.

Figure 17-6 Shows the waveforms for master mode. When the CKE bit is set, the SDO data is valid before there is a clock edge on SCK. The change of the input sample is shown based on the state of the SMP bit. The time when the SSPBUF is loaded with the received data is shown.
Figure 17-6: SPI Mode Waveform (Master Mode)

- **Write to SSPBUF**
- **SCK** (CKP = 0, CKE = 0)
- **SCK** (CKP = 1, CKE = 0)
- **SCK** (CKP = 0, CKE = 1)
- **SCK** (CKP = 1, CKE = 1)
- **SDO** (CKE = 0)
- **SDO** (CKE = 1)
- **SDI** (SMP = 0)
- **SDI** (SMP = 1)
- **Input Sample (SMP = 0)**
- **Input Sample (SMP = 1)**
- **SSPIF**
- **SSPSR to SSPBUF**

4 clock modes

**Input Sample**

- **bit7**
- **bit6**
- **bit5**
- **bit4**
- **bit3**
- **bit2**
- **bit1**
- **bit0**

**Next Q4 cycle after Q2↓**

**Write to SSPBUF**
17.3.5 Slave Mode

In slave mode, the data is transmitted and received as the external clock pulses appear on SCK. When the last bit is latched, the SSPIF interrupt flag bit is set.

While in slave mode the external clock is supplied by the external clock source on the SCK pin. This external clock must meet the minimum high and low times as specified in the electrical specifications.

While in sleep mode, the slave can transmit/receive data. When a byte is receive the device will wake-up from sleep.

17.3.6 Slave Select Synchronization

The SS pin allows a synchronous slave mode. The SPI must be in slave mode with SS pin control enabled (SSPCON1<3:0> = 04h). The pin must not be driven low for the SS pin to function as an input. The Data Latch must be high. When the SS pin is low, transmission and reception are enabled and the SDO pin is driven. When the SS pin goes high, the SDO pin is no longer driven, even if in the middle of a transmitted byte, and becomes a floating output. External pull-up/ pull-down resistors may be desirable, depending on the application.

| Note 1: | When the SPI is in Slave Mode with SS pin control enabled, (SSPCON<3:0> = 0100) the SPI module will reset if the SS pin is set to VDD. |
| Note 2: | If the SPI is used in Slave Mode with CKE is set, then the SS pin control must be enabled. |

When the SPI module resets, the bit counter is forced to 0. This can be done by either by forcing the SS pin to a high level or clearing the SSPEN bit.

To emulate two-wire communication, the SDO pin can be connected to the SDI pin. When the SPI needs to operate as a receiver the SDO pin can be configured as an input. This disables transmissions from the SDO. The SDI can always be left as an input (SDI function) since it cannot create a bus conflict.
### Section 17. MSSP

#### Figure 17-7: Slave Synchronization Waveform

- **SS**
- **SCK** (CKP = 0, CKE = 0)
- **SCK** (CKP = 1, CKE = 0)
- **Write to SSPBUF**
- **SDO**: bit7, bit6, bit7, bit0
- **SDI** (SMP = 0): bit7, bit7, bit0
- **SSPIF Interrupt Flag**
- **SSPSR to SSPBUF**

#### Figure 17-8: SPI Mode Waveform (Slave Mode with CKE = 0)

- **SS** optional
- **SCK** (CKP = 0, CKE = 0)
- **SCK** (CKP = 1, CKE = 0)
- **Write to SSPBUF**
- **SDO**: bit7, bit6, bit5, bit4, bit3, bit2, bit1, bit0
- **SDI** (SMP = 0): bit7, bit7, bit0
- **SSPIF Interrupt Flag**
- **SSPSR to SSPBUF**

**Input Sample (SMP = 0)**

**SSPSR to SSPBUF**

Next Q4 cycle after Q2↓
Figure 17-9: SPI Mode Waveform (Slave Mode with CKE = 1)

- SS: not optional
- SCK (CKP = 0, CKE = 1)
- SCK (CKP = 1, CKE = 1)
- Write to SSPBUF
- SDO:
  - bit7
  - bit6
  - bit5
  - bit4
  - bit3
  - bit2
  - bit1
  - bit0
- SDI (SMP = 0)
- Input Sample (SMP = 0)
- SSPIF: Interrupt Flag
- SSPSR to SSPBUF

Next Q4 cycle after Q2↓
17.3.7 Sleep Operation

In master mode all module clocks are halted, and the transmission/reception will remain in that state until the device wakes from sleep. After the device returns to normal mode, the module will continue to transmit/receive data.

In slave mode, the SPI transmit/receive shift register operates asynchronously to the device. This allows the device to be placed in sleep mode, and data to be shifted into the SPI transmit/receive shift register. When all 8-bits have been received, the MSSP interrupt flag bit will be set and if enabled will wake the device from sleep.

17.3.8 Effects of a Reset

A reset disables the MSSP module and terminates the current transfer.

Table 17-1: Registers Associated with SPI Operation

<table>
<thead>
<tr>
<th>Name</th>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
<th>Value on POR, BOR</th>
<th>Value on all other resets</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTCON</td>
<td>GIE</td>
<td>PEIE</td>
<td>T0IE</td>
<td>INTE</td>
<td>RBIE(2)</td>
<td>T0IF</td>
<td>INTF</td>
<td>RBF(2)</td>
<td>0000 0000</td>
<td>0000 0000</td>
</tr>
<tr>
<td>PIR</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>SSPIF(1)</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>PIE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>SSPIE(1)</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>SSPBUF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>xxxx xxxx</td>
<td>xxxx xxxx</td>
</tr>
<tr>
<td>SSPCON1</td>
<td>WCOL</td>
<td>SSPOV</td>
<td>SSPEN</td>
<td>CKP</td>
<td>SSPM3</td>
<td>SSPM2</td>
<td>SSPM1</td>
<td>SSPM0</td>
<td>0000 0000</td>
<td>0000 0000</td>
</tr>
<tr>
<td>SSPSTAT</td>
<td>SMP</td>
<td>CKE</td>
<td>D/A</td>
<td>P</td>
<td>S</td>
<td>R/W</td>
<td>UA</td>
<td>BF</td>
<td>0000 0000</td>
<td>0000 0000</td>
</tr>
</tbody>
</table>

Legend:  x = unknown,  u = unchanged,  – = unimplemented read as ‘0’.
Shaded cells are not used by the SSP in SPI mode.

Note 1: The position of this bit is device dependent.
2: These bits may also be named GPIE and GPIF.
17.4  **SSP I²C™ Operation**

The MSSP module in I²C mode fully implements all master and slave functions (including general call support) and provides interrupts on start and stop bits in hardware to determine a free bus (multi-master function). The SSP module implements the standard mode specifications as well as 7-bit and 10-bit addressing. **Appendix A** gives an overview of the I²C bus specification.

A “glitch” filter is on the SCL and SDA pins when the pin is an input. This filter operates in both the 100 KHz and 400 KHz modes. In the 100 KHz mode, when these pins are an output, there is a slew rate control of the pin that is independent of device frequency.

**Figure 17-10: I²C Slave Mode Block Diagram**

**Figure 17-11: I²C Master Mode Block Diagram**
Section 17. MSSP

Two pins are used for data transfer. These are the SCL pin, which is the clock, and the SDA pin, which is the data. Pins that are on the port are automatically configured when the I²C mode is enabled. The SSP module functions are enabled by setting SSP Enable bit, SSPEN (SSPCON1<5>),

The SSP module has six registers for I²C operation. They are the:

- SSP Control Register1 (SSPCON1)
- SSP Control Register2 (SSPCON2)
- SSP Status Register (SSPSTAT)
- Serial Receive/Transmit Buffer (SSPBUF)
- SSP Shift Register (SSPSR) - Not directly accessible
- SSP Address Register (SSPADD)

The SSPCON1 register allows control of the I²C operation. Four mode selection bits (SSPCON1<3:0>) allow one of the following I²C modes to be selected:

- I²C Slave mode (7-bit address)
- I²C Slave mode (10-bit address)
- I²C Master mode, clock = OSC/4 (SSPADD +1)

Before selecting any I²C mode, the SCL and SDA pins must be programmed to inputs by setting the appropriate TRIS bits. Selecting an I²C mode, by setting the SSPEN bit, enables the SCL and SDA pins to be used as the clock and data lines in I²C mode.

The SSPSTAT register gives the status of the data transfer. This information includes detection of a START or STOP bit, specifies if the received byte was data or address, if the next byte is the completion of 10-bit address, and if this will be a read or write data transfer.

The SSPBUF is the register to which transfer data is written to or read from. The SSPSR register shifts the data in or out of the device. In receive operations, the SSPBUF and SSPSR create a double buffered receiver. This allows reception of the next byte to begin before reading the last byte of received data. When the complete byte is received, it is transferred to the SSPBUF register and flag bit SSPIF is set. If another complete byte is received before the SSPBUF register is read, a receiver overflow has occurred and the SSPOV bit (SSPCON1<6>) is set and the byte in the SSPSR is lost.

The SSPADD register holds the slave address. In 10-bit mode, the user needs to write the high byte of the address (1111 0 A9 A8 0). Following the high byte address match, the low byte of the address needs to be loaded (A7:A0).
17.4.1 Slave Mode

In slave mode, the SCL and SDA pins must be configured as inputs. The SSP module will over-ride the input state with the output data when required (slave-transmitter).

When an address is matched or the data transfer after an address match is received, the hardware automatically will generate the acknowledge (ACK) pulse, and then load the SSPBUF register with the received value currently in the SSPSR register.

There are certain conditions that will cause the SSP module not to give this ACK pulse. These are if either (or both):

a) The buffer full bit, BF (SSPSTAT<0>), was set before the transfer was received.
b) The overflow bit, SSPOV (SSPCON1<6>), was set before the transfer was received.

If the BF bit is set, the SSPSR register value is not loaded into the SSPBUF, but the SSPIF and SSPOV bits are set. Table 17-2 shows what happens when a data transfer byte is received, given the status of the BF and SSPOV bits. The shaded cells show the condition where user software did not properly clear the overflow condition. The BF flag bit is cleared by reading the SSPBUF register while bit SSPOV is cleared through software.

The SCL clock input must have a minimum high and low time for proper operation. The high and low times of the I^2C specification as well as the requirement of the SSP module is shown in timing parameters 100 and 101 of the “Electrical Specifications” section.
17.4.1.1 Addressing

Once the SSP module has been enabled, it waits for a START condition to occur. Following the START condition, the 8-bits are shifted into the SSPSR register. All incoming bits are sampled with the rising edge of the clock (SCL) line. The value of register SSPSR<7:1> is compared to the value of the SSPADD register. The address is compared on the falling edge of the eighth clock (SCL) pulse. If the addresses match, and the BF and SSPOV bits are clear, the following events occur:

a) The SSPSR register value is loaded into the SSPBUF register on the falling edge of the eighth SCL pulse.
b) The buffer full bit, BF, is set on the falling edge of the eighth SCL pulse.
c) An ACK pulse is generated.
d) SSP interrupt flag bit, SSPIF, is set (interrupt is generated if enabled) - on the falling edge of the ninth SCL pulse.

In 10-bit address mode, two address bytes need to be received by the slave. The five Most Significant bits (MSbs) of the first address byte specify if this is a 10-bit address. The R/W bit (SSPSTAT<2>) must specify a write so the slave device will receive the second address byte. For a 10-bit address the first byte would equal ‘1111 0 A9 A8 0’, where A9 and A8 are the two MSbs of the address. The sequence of events for a 10-bit address is as follows, with steps 7-9 for slave-transmitter:

1. Receive first (high) byte of Address (the SSPIF, BF, and UA (SSPSTAT<1>) bits are set).
2. Update the SSPADD register with second (low) byte of Address (clears the UA bit and releases the SCL line).
3. Read the SSPBUF register (clears the BF bit) and clear flag bit SSPIF.
4. Receive second (low) byte of Address (the SSPIF, BF, and UA bits are set).
5. Update the SSPADD register with the first (high) byte of Address. This will clear the UA bit and release the SCL line.
6. Read the SSPBUF register (clears the BF bit) and clear the SSPIF flag bit.
7. Receive repeated START condition.
8. Receive first (high) byte of Address (the SSPIF and BF bits are set).
9. Read the SSPBUF register (clears the BF bit) and clear the SSPIF flag bit.

**Table 17-2: Data Transfer Received Byte Actions**

<table>
<thead>
<tr>
<th>Status Bits as Data Transfer is Received</th>
<th>SSPSR → SSPBUF</th>
<th>Generate ACK Pulse</th>
<th>Set bit SSPIF (SSP Interrupt occurs if enabled)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BF</td>
<td>SSPOV</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

Note: Following the Repeated Start condition (step 7) in 10-bit mode, the user only needs to match the first 7-bit address. The user does not update the SSPADD for the second half of the address.

**Table 17-2: Data Transfer Received Byte Actions**

<table>
<thead>
<tr>
<th>Status Bits as Data Transfer is Received</th>
<th>SSPSR → SSPBUF</th>
<th>Generate ACK Pulse</th>
<th>Set bit SSPIF (SSP Interrupt occurs if enabled)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BF</td>
<td>SSPOV</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

Note: Shaded cells show the conditions where the user software did not properly clear the overflow condition.
17.4.1.2 Slave Reception

When the R/W bit of the address byte is clear and an address match occurs, the R/W bit of the SSPSTAT register is cleared. The received address is loaded into the SSPBUF register.

When the address byte overflow condition exists, then no acknowledge (ACK) pulse is given. An overflow condition is defined as either the BF bit (SSPSTAT<0>) is set or the SSPOV bit (SSPCON1<6>) is set.

An SSP interrupt is generated for each data transfer byte. The SSPIF flag bit must be cleared in software. The SSPSTAT register is used to determine the status of the received byte.

**Note:** The SSPBUF will be loaded if the SSPOV bit is set and the BF flag bit is cleared. If a read of the SSPBUF was performed, but the user did not clear the state of the SSPOV bit before the next receive occurred. The ACK is not sent and the SSPBUF is updated.
### Section 17. MSSP

#### 17.4.1.3 Slave Transmission

When the R/W bit of the incoming address byte is set and an address match occurs, the R/W bit of the SSPSTAT register is set. The received address is loaded into the SSPBUF register. The \textit{ACK} pulse will be sent on the ninth bit, and the SCL pin is held low. The transmit data must be loaded into the SSPBUF register, which also loads the SSPSR register. Then the SCL pin should be enabled by setting the CKP bit (SSPCON1<4>). The master should monitor the SCL pin prior to asserting another clock pulse. The slave devices may be holding off the master by stretching the clock. The eight data bits are shifted out on the falling edge of the SCL input. This ensures that the SDA signal is valid during the SCL high time (Figure 17-13).

An SSP interrupt is generated for each data transfer byte. The SSPIF flag bit must be cleared in software, and the SSPSTAT register is used to determine the status of the byte transfer. The SSPIF flag bit is set on the falling edge of the ninth clock pulse.

As a slave-transmitter, the \textit{ACK} pulse from the master-receiver is latched on the rising edge of the ninth SCL input pulse. If the SDA line was high (not \textit{ACK}), then the data transfer is complete. When the not \textit{ACK} is latched by the slave, the slave logic is reset and the slave then monitors for another occurrence of the START bit. If the SDA line was low (\textit{ACK}), the transmit data must be loaded into the SSPBUF register, which also loads the SSPSR register. Then the SCL pin should be enabled by setting the CKP bit.

#### Figure 17-12: I²C Slave Mode Waveforms for Reception (7-bit Address)

#### Figure 17-13: I²C Slave Mode Waveforms for Transmission (7-bit Address)
Figure 17-14: I^2C Slave Mode Waveform (Transmission 10-bit Address)
Figure 17-15: I^2C Slave Mode Waveform (Reception 10-bit Address)
17.4.2 General Call Address Support

The addressing procedure for the I²C bus is such that the first byte after the START condition usually determines which device will be the slave addressed by the master. The exception is the general call address which can address all devices. When this address is used, all devices should, in theory, respond with an acknowledge.

The general call address is one of eight addresses reserved for specific purposes by the I²C protocol. It consists of all 0’s with \( R/W = 0 \).

The general call address is recognized when the General Call Enable bit (GCEN) is enabled (SSPCON2<7> set). Following a start-bit detect, 8-bits are shifted into SSPSR and the address is compared against SSPADD, and is also compared to the general call address, fixed in hardware.

If the general call address matches, the SSPSR is transferred to the SSPBUF, the BF flag bit is set (eight bit), and on the falling edge of the ninth bit (\( \text{ACK} \) bit) the SSPIF interrupt flag bit is set. When the interrupt is serviced. The source for the interrupt can be checked by reading the contents of the SSPBUF to determine if the address was device specific or a general call address.

In 10-bit mode, the SSPADD is required to be updated for the second half of the address to match, and the UA bit is set (SSPSTAT<1>). If the general call address is sampled when the GCEN bit is set while the slave is configured in 10-bit address mode, then the second half of the address is not necessary, the UA bit will not be set, and the slave will begin receiving data after the acknowledge (Figure 17-16).

Figure 17-16: Slave Mode General Call Address Sequence (7 or 10-bit Address Mode)
17.4.3 Sleep Operation

While in sleep mode, the \( \text{I}^2\text{C} \) module can receive addresses or data, and when an address match or complete byte transfer occurs wake the processor from sleep (if the MSSP interrupt is enabled).

17.4.4 Effect of a Reset

A reset disables the MSSP module and terminates the current transfer.

### Table 17-3: Registers Associated with \( \text{I}^2\text{C} \) Operation

<table>
<thead>
<tr>
<th>Name</th>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
<th>Value on POR, BOR</th>
<th>Value on all other resets</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTCN</td>
<td>GIE</td>
<td>PEIE</td>
<td>T0IE</td>
<td>INTE</td>
<td>RBIE(2)</td>
<td>T0IF</td>
<td>INTF</td>
<td>RBIF(2)</td>
<td>0000 0000</td>
<td>0000 0000</td>
</tr>
<tr>
<td>PIR</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>SSPIF, BCLIF(1)</td>
<td></td>
<td></td>
<td></td>
<td>0, 0</td>
<td>0, 0</td>
</tr>
<tr>
<td>PIE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>SSPIE, BCLIF(1)</td>
<td></td>
<td></td>
<td></td>
<td>0, 0</td>
<td>0, 0</td>
</tr>
<tr>
<td>SSPADD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Synchronous Serial Port (( \text{I}^2\text{C} ) mode) Address Register (slave mode)/Baud Rate Generator (master mode)</td>
<td></td>
<td></td>
<td></td>
<td>0000 0000</td>
<td>0000 0000</td>
</tr>
<tr>
<td>SSPBUF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Synchronous Serial Port Receive Buffer/Transmit Register</td>
<td>xxxxx</td>
<td>xxxxx</td>
<td>uuuu</td>
<td>uuuu</td>
<td></td>
</tr>
<tr>
<td>SSPCON1</td>
<td>WCOL</td>
<td>SSPOV</td>
<td>SSPEN</td>
<td>CKP</td>
<td>SSPM3</td>
<td>SSPM2</td>
<td>SSPM1</td>
<td>SSPM0</td>
<td>0000 0000</td>
<td>0000 0000</td>
</tr>
<tr>
<td>SSPCON2</td>
<td>GCE</td>
<td>ACKSTAT</td>
<td>ACKDT</td>
<td>ACKEN</td>
<td>RCEN</td>
<td>PEN</td>
<td>RSEN</td>
<td>SEN</td>
<td>0000 0000</td>
<td>0000 0000</td>
</tr>
<tr>
<td>SSPSTAT</td>
<td>SMP</td>
<td>CKE</td>
<td>D/A</td>
<td>P</td>
<td>S</td>
<td>R/W</td>
<td>UA</td>
<td>BF</td>
<td>0000 0000</td>
<td>0000 0000</td>
</tr>
</tbody>
</table>

Legend: \( x \) = unknown, \( u \) = unchanged, \( - \) = unimplemented read as ‘0’.

Shaded cells are not used by the SSP in \( \text{I}^2\text{C} \) mode.

**Note 1:** The position of these bits is device dependent.

2: These bits may also be named GPIE and GPIF.
17.4.5 Master Mode

Master mode of operation is supported by interrupt generation on the detection of the START and STOP conditions. The STOP (P) and START (S) bits are cleared from a reset or when the SSP module is disabled. Control of the I2C bus may be taken when the P bit is set, or the bus is idle with both the S and P bits clear.

In master mode the SCL and SDA lines are manipulated by the SSP hardware.

The following events will cause SSP Interrupt Flag bit, SSPIF, to be set (SSP Interrupt if enabled):

- START condition
- STOP condition
- Data transfer byte transmitted/received
- Acknowledge Transmit
- Repeated Start

Figure 17-17: SSP Block Diagram (I²C Master Mode)
17.4.6 Multi-Master Mode

In multi-master mode, the interrupt generation on the detection of the START and STOP conditions allows the determination of when the bus is free. The STOP (P) and START (S) bits are cleared from a reset or when the SSP module is disabled. Control of the I2C bus may be taken when the P bit (SSPSTAT<4>) is set, or the bus is idle with both the S and P bits clear. When the bus is busy, enabling the SSP Interrupt will generate the interrupt when the STOP condition occurs.

In multi-master operation, the SDA line must be monitored, for arbitration, to see if the signal level is the expected output level. This check is performed in hardware, with the result placed in the BCLIF bit.

The states where arbitration can be lost are:

- Address Transfer
- Data Transfer
- A Start Condition
- A Repeated Start Condition
- An Acknowledge Condition

17.4.7 I2C Master Mode Support

Master Mode is enabled by setting and clearing the appropriate SSPM bits in SSPCON1 and by setting the SSPEN bit. Once master mode is enabled, the user has six options.

1. Assert a start condition on SDA and SCL.
2. Assert a Repeated Start condition on SDA and SCL.
3. Write to the SSPBUF register initiating transmission of data/address.
4. Generate a stop Condition on SDA and SCL.
5. Configure the I2C port to receive data.
6. Generate an acknowledge condition at the end of a received byte of data.

Note: The SSP Module when configured in I2C Master Mode does not allow queueing of events. For instance: The user is not allowed to initiate a start condition, and immediately write the SSPBUF register to imitate transmission before the START condition is complete. In this case the SSPBUF will not be written to, and the WCOL bit will be set, indicating that a write to the SSPBUF did not occur.
17.4.7.1 I\(^2\)C Master Mode Operation

The master device generates all of the serial clock pulses and the START and STOP conditions. A transfer is ended with a STOP condition or with a repeated START condition. Since the repeated START condition is also the beginning of the next serial transfer, the I\(^2\)C bus will not be released.

In Master transmitter mode serial data is output through SDA, while SCL outputs the serial clock. The first byte transmitted contains the slave address of the receiving device, (7 bits) and the Read/Write (R/W) bit. In this case the R/W bit will be logic ‘0’. Serial data is transmitted 8 bits at a time. After each byte is transmitted, an acknowledge bit is received. START and STOP conditions are output to indicate the beginning and the end of a serial transfer.

In Master receive mode the first byte transmitted contains the slave address of the transmitting device (7 bits) and the R/W bit. In this case the R/W bit will be logic ‘1’. Thus the first byte transmitted is a 7-bit slave address followed by a ‘1’ to indicate receive bit. Serial data is received via SDA while SCL outputs the serial clock. Serial data is received 8 bits at a time. After each byte is received, an acknowledge bit is transmitted. START and STOP conditions indicate the beginning and end of transmission.

The baud rate generator used for SPI mode operation is used to set the SCL clock frequency for either 100 kHz, 400 kHz, or 1 MHz I\(^2\)C operation. The baud rate generator reload value is contained in the lower 7 bits of the SSPADD register. The baud rate generator will automatically begin counting on a write to the SSPBUF. Once the given operation is complete (i.e., transmission of the last data bit is followed by ACK) the internal clock will automatically stop counting and the SCL pin will remain in its last state.

A typical transmit sequence would go as follows:

a) The user generates a Start Condition by setting the START enable bit, SEN (SSPCON2<0>).
b) SSPIF is set. The SSP module will wait the required start time before any other operation takes place.
c) The user loads the SSPBUF with the address to transmit.
d) Address is shifted out the SDA pin until all 8 bits are transmitted.
e) The SSP Module shifts in the ACK bit from the slave device, and writes its value into the SSPCON2 register (SSPCON2<6>).
f) The SSP module generates an interrupt at the end of the ninth clock cycle by setting the SSPIF bit.
g) The user loads the SSPBUF with eight bits of data.
h) DATA is shifted out the SDA pin until all 8 bits are transmitted.
i) The SSP Module shifts in the ACK bit from the slave device, and writes its value into the SSPCON2 register (SSPCON2<6>).
j) The SSP module generates an interrupt at the end of the ninth clock cycle by setting the SSPIF bit.
k) The user generates a STOP condition by setting the STOP enable bit, PEN (SSPCON2<2>).
l) Interrupt is generated once the stop condition is complete.
17.4.8 Baud Rate Generator

In I²C master mode, the reload value for the BRG is located in the lower 7 bits of the SSPADD register (Figure 17-18). When the BRG is loaded with this value, the BRG counts down to 0 and stops until another reload has taken place. In I²C master mode, the BRG is reloaded automatically. If Clock Arbitration is taking place for instance, the BRG will be reloaded when the SCL pin is sampled high (Figure 17-19).

**Figure 17-18: Baud Rate Generator Block Diagram**

**Figure 17-19: Baud Rate Generator Timing With Clock Arbitration**
I2C Master Mode Start Condition Timing

To initiate a START condition the user sets the start condition enable bit, SEN (SSPCON2<0>). If the SDA and SCL pins are sampled high, the baud rate generator is re-loaded with the contents of SSPADD<6:0>, 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 (SSPSTAT<3>) to be set. Following this, the baud rate generator is reloaded with the contents of SSPADD<6:0> and resumes its count. When the baud rate generator times out (TBRG) the SEN bit (SSPCON2<0>) 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:** If at the beginning of 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, BCLIF, is set, the START condition is aborted, and the I^2^C module is reset into its IDLE state.

17.4.9.1 WCOL Status Flag

If the user writes the SSPBUF when an START sequence is in progress, then WCOL is set and the contents of the buffer are unchanged (the write doesn’t occur).

**Note:** Because queueing of events is not allowed, writing to the lower 5 bits of SSPCON2 is disabled until the START condition is complete.

Figure 17-20: First Start Bit Timing

- **SDA**
  - **S**
  - **T_{brg}**
- **SCL**
  - **T_{brg}**

Write to SEN bit occurs here.

Set S bit (SSPSTAT<3>)

At completion of start bit, Hardware clears SEN bit and sets SSPIF bit

Write to SSPBUF occurs here

SDA = 1, SCL = 1

1st Bit

2nd Bit
Figure 17-21: Start Condition Flowchart

1. **Idle Mode**
2. **SEN (SSPCON2<0> = 1)**
   - **Bus collision detected, Set BCLIF, Release SCL, Clear SEN**
   - **SDA = 1?**
     - **Yes**
       - **Load BRG with Ye s**
       - **BRG Rollover?**
         - **No**
           - **Clear SEN**
           - **Start Condition Done, No**
             - **Reset BRG**
     - **No**
       - **Force SDA = 0, Load BRG with SSPADD<6:0>, Set S bit**
   - **No**
     - **Force SDA = 0, Load BRG with SSPADD<6:0>, Set S bit**
     - **BRG rollover?**
       - **No**
         - **SCL = 0?**
           - **Yes**
             - **Reset BRG**
           - **No**
             - **Force SCL = 0, Start Condition Done, Clear SEN, Set SSPIF**
     - **Yes**
       - **SCL = 0?**
         - **Yes**
           - **Reset BRG**
         - **No**
           - **Force SCL = 0, Start Condition Done, Clear SEN, Set SSPIF**
   - **SCL = 1? SCL = 1?**
     - **Yes**
       - **Load BRG with SSPADD<6:0>**
     - **No**
       - **Force SCL = 0, Release SCL, SSPEN = 1, SSPCON1<3:0> = 1000**

3. **SSPEN = 1, SSPCON1<3:0> = 1000**

---

Section 17. MSSP
### 17.4.10 I²C Master Mode Repeated Start Condition Timing

A Repeated Start condition occurs when the RSEN bit (SSPCON2<1>) is programmed high and the I²C logic module is in the idle state. When the RSEN bit is set, the SCL pin is asserted low. When the SCL pin is sampled low, the baud rate generator is loaded with the contents of SSPADD<5:0>, and begins counting. The SDA pin is released (brought high) for one baud rate generator count (T_{BRG}). When the baud rate generator times out, if SDA is sampled high, the SCL pin will be de-asserted (brought high). When SCL is sampled high the baud rate generator is re-loaded with the contents of SSPADD<6:0> and begins counting. SDA and SCL must be sampled high for one T_{BRG}. This action is then followed by assertion of the SDA pin (SDA = 0) for one T_{BRG} while SCL is high. Following this, the RSEN bit (SSPCON2<1>) will be automatically cleared, and the baud rate generator is not reloaded, leaving the SDA pin held low. As soon as a start condition is detected on the SDA and SCL pins, the S bit (SSPSTAT<3>) will be set. The SSPIF bit will not be set until the baud rate generator has timed-out.

**Note 1:** If RSEN is programmed while any other event is in progress, it will not take effect.

**Note 2:** A bus collision during the Repeated Start condition occurs if:
- SDA is sampled low when SCL goes from low to high.
- SCL goes low before SDA is asserted low. This may indicates that another master is attempting to transmit a data ‘1’.

Immediately following the SSPIF bit getting set, the user may write the SSPBUF with the 7-bit address in 7-bit mode, or the default first address in 10-bit mode. After the first eight bits are transmitted and an ACK is received, the user may then transmit an additional eight bits of address (10-bit mode) or eight bits of data (7-bit mode).
17.4.10.1 WCOL Status Flag

If the user writes the SSPBUF when a Repeated Start sequence is in progress, then WCOL is set and the contents of the buffer are unchanged (the write doesn't occur).

**Note:** Because queueing of events is not allowed, writing of the lower 5 bits of SSPCON2 is disabled until the Repeated Start condition is complete.

**Figure 17-22: Repeat Start Condition Waveform**

- Write to SSPCON2 occurs here. SDA = 1, SCL (no change)
- At completion of start bit, hardware clear RSEN bit and set SSPIF
- Set S (SSPSTAT<3>)
- Write to SSPBUF occurs here.
- Falling edge of ninth clock
- End of Xmit
- Sr = Repeated Start
Figure 17-23: Repeated Start Condition Flowchart (part 1 of 2)
Figure 17-24: Repeated Start Condition Flowchart (part 2 of 2)

- If SCL = 1, go to B.
- If SDA = 0, go to C.
- If BRG rollover, Force SDA = 0, Load BRG with SSPADD<6:0>.
- If SCL = '0', go to A.
- If SDA = 0, Yes, Set S.
- If BRG rollover, Force SDA = 0, Load BRG with SSPADD<6:0>.
- If SCL = '0', reset BRG.
- If BRG rollover, Reset BRG.
- Force SCL = 0, Repeated Start condition done, Clear RSEN, Set SSPIF.
17.4.11 I\(^2\)C Master Mode Transmission

Transmission of a data byte, a 7-bit address, or the either half of a 10-bit address is accomplished by simply writing a value to SSPBUF register. This action will set the buffer full flag bit, BF, and allow the baud rate generator to begin counting and start the next transmission. Each bit of address/data will be shifted out onto the SDA pin after the falling edge of SCL is asserted (see data hold time specification parameters 106). SCL is held low for one baud rate generator roll over count (\(T_{BRG}\)). Data should be valid before SCL is released high (see Data setup time specification parameters 107). When the SCL pin is released high, it is held that way for \(T_{BRG}\), the data on the SDA pin must remain stable for that duration and some hold time after the next falling edge of SCL. After the eighth bit is shifted out (the falling edge of the eighth clock), the BF flag is cleared and the master releases SDA allowing the slave device being addressed to respond with an \(A\overline{C}K\) bit during the ninth bit time, if an address match occurs or if data was received properly. The status of \(A\overline{C}K\) is written into the ACKDT bit on the falling edge of the ninth clock. If the master receives an acknowledge, the acknowledge status bit, ACKSTAT, is cleared. If not, the bit is set. After the ninth clock the SSPIF bit is set, and the master clock (baud rate generator) is suspended until the next data byte is loaded into the SSPBUF leaving SCL low and SDA unchanged (Figure 17-26).

After the write to the SSPBUF, each bit of address will be shifted out on the falling edge of SCL until all seven address bits and the R/W bit are completed. On the falling edge of the eighth clock the master will de-assert the SDA pin allowing the slave to respond with an acknowledge. On the falling edge of the ninth clock the master will sample the SDA pin to see if the address was recognized by a slave. The status of the \(A\overline{C}K\) bit is loaded into the ACKSTAT status bit (SSPCON2<6>). Following the falling edge of the ninth clock transmission of the address, the SSPIF is set, the BF flag is cleared, and the baud rate generator is turned off until another write to the SSPBUF takes place, holding SCL low and allowing SDA to float.

17.4.11.1 BF Status Flag

In transmit mode, the BF bit (SSPSTAT<0>) is set when the CPU writes to SSPBUF and is cleared when all 8 bits are shifted out.

17.4.11.2 WCOL Status Flag

If the user writes the SSPBUF when a transmit is already in progress (i.e. SSPSR is still shifting out a data byte), then WCOL is set and the contents of the buffer are unchanged (the write doesn’t occur).

WCOL must be cleared in software.

17.4.11.3 ACKSTAT Status Flag

In transmit mode, the ACKSTAT bit (SSPCON2<6>) is cleared when the slave has sent an acknowledge (\(A\overline{C}K = 0\)), and is set when the slave does not acknowledge (\(A\overline{C}K = 1\)). A slave sends an acknowledge when it has recognized its address (including a general call), or when the slave has properly received its data.
Figure 17-25: Master Transmit Flowchart

Idle Mode
- Write SSPBUF
- Num_Clocks = 0, BF = 1
- Force SCL = 0
- Yes: Num_Clocks = 8?
  - No: Load BRG with SSPADD<6:0>, start BRG count, SDA = Current Data bit
  - Yes: BRG rollover?
    - No: Stop BRG, Force SCL = 1, (Clock Arbitration)
    - Yes: SCL = 1?
      - No: SDA = Data bit?
        - Yes: Load BRG with SSPADD<6:0>, count SCL high time
        - No: BRG rollover?
          - Yes: No SDA = Data bit?
            - Yes: Reset BRG, Num_Clocks = Num_Clocks + 1
            - No: SCL = 0?
              - Yes: BCLIF, hold prescale off, Clear XMIT enable
              - No: SDA = Data bit?
                - Yes: Load BRG with SSPADD<6:0>, count SCL high time
                - No: Rollover?
                  - No: SCL = 0?
                    - Yes: Set SSPIF
                    - No: SDA = Data bit?
                      - Yes: Force BF = 0
                      - No: Rollover?
                        - Yes: Load BRG with SSPADD<6:0>, count high time
                        - No: Yes

- No: SDA = Data bit?
  - Yes: Load BRG with SSPADD<6:0>, start BRG count, SDA = Current Data bit
  - No: Yes
- Release SDA so slave can drive ACK, Force BF = 0
- Load BRG with SSPADD<6:0>, start BRG count
- BRG rollover?
  - Yes: Force SCL = 1, Stop BRG, (Clock Arbitration)
  - No: SCL = 1?
    - Yes: Load BRG with SSPADD<6:0>, count high time
    - No: Bus collision detected: Set BCLIF, hold prescale off, Clear XMIT enable
- Rollover?
  - Yes: Load BRG with SSPADD<6:0>, count high time
  - No: Yes
- Force SCL = 0, Set SSPIF
Figure 17-26: \( \text{I}^2\text{C} \) Master Mode Waveform (Transmission, 7 or 10-bit Address)
17.4.12 I2C Master Mode Reception

Master mode reception is enabled by programming the receive enable bit, RCEN (SSPCON2<3>).

Note: The SSP Module must be in an IDLE STATE before the RCEN bit is set, or the RCEN bit will be disregarded.

The baud rate generator begins counting, and on each rollover, the state of the SCL pin changes (high to low/low to high), and data is shifted into the SSPSR. After the falling edge of the eighth clock, the receive enable flag is automatically cleared, the contents of the SSPSR are loaded into the SSPBUF, the BF flag bit is set, the SSPIF flag bit is set, and the baud rate generator is suspended from counting, holding SCL low. The SSP is now in IDLE state, awaiting the next command. When the buffer is read by the CPU, the BF flag bit is automatically cleared. The user can then send an acknowledge bit at the end of reception, by setting the acknowledge sequence enable bit, ACKEN (SSPCON2<4>).

17.4.12.1 BF Status Flag

In receive operation, the BF bit is set when an address or data byte is loaded into SSPBUF from SSPSR. It is cleared when the SSPBUF register is read.

17.4.12.2 SSPOV Status Flag

In receive operation, the SSPOV bit is set when 8 bits are received into the SSPSR, and the BF flag bit is already set from a previous reception.

17.4.12.3 WCOL Status Flag

If the user writes the SSPBUF when a receive is already in progress (i.e. SSPSR is still shifting in a data byte), then the WCOL bit is set and the contents of the buffer are unchanged (the write doesn’t occur).
Figure 17-27: Master Receiver Flowchart

Idle mode

RCEN = 1

Num_Clocks = 0,
Release SDA

Force SCL=0,
Load BRG w/ SSPADD<6:0>,
start count

BRG rollover?

Yes

Release SCL

SCL = 1?

No

Yes

Sample SDA,
Shift data into SSPSR

Load BRG with SSPADD<6:0>,
start count,

BRG rollover?

SCL = 0?

No

Yes

Num_Clocks = Num_Clocks + 1

Num_Clocks = 8?

No

Yes

Force SCL = 0,
Set SSPIF,
Set BF,
Move contents of SSPSR into SSPBUF,
Clear RCEN.
Figure 17-28: \( I^2C \) Master Mode Waveform (Reception 7-Bit Address)
17.4.13 Acknowledge Sequence Timing

An acknowledge sequence is enabled by setting the acknowledge sequence enable bit, ACKEN (SSPCON2<4>). When this bit is set, the SCL pin is pulled low and the contents of the acknowledge data bit is presented on the SDA pin. If the user wishes to generate an acknowledge, then the ACKDT bit should be cleared. If not, the user should set the ACKDT bit before starting an acknowledge sequence. The baud rate generator then counts for one rollover period ($T_{BRG}$), and the SCL pin is de-asserted (pulled high). When the SCL pin is sampled high (clock arbitration), the baud rate generator counts for $T_{BRG}$. The SCL pin is then pulled low. Following this, the ACKEN bit is automatically cleared, the baud rate generator is turned off, and the SSP module then goes into IDLE mode (Figure 17-29).

17.4.13.1 WCOL Status Flag

If the user writes the SSPBUF when an acknowledge sequence is in progress, then WCOL is set and the contents of the buffer are unchanged (the write doesn't occur).

Figure 17-29: Acknowledge Sequence Waveform

- Acknowledge sequence starts here, Write to SSPCON2 ACKEN = 1, ACKDT = 0
- $T_{BRG}$
- SDA
- SCL
- SSPIF
- Set SSPIF at the end of receive
- Cleared in software
- Cleared in software
- ACKEN automatically cleared
- Note: $T_{BRG}$ = one baud rate generator period.
Figure 17-30: Acknowledge Flowchart

1. Idle mode
2. Set ACKEN
3. Force SCL = 0
4. SCL = 0?
   - Yes: Drive ACKDT bit (SSPCON2<5>) onto SDA pin, Load BRG with SSPADD<6:0>, start count.
   - No: BRG rollover?
     - Yes: Force SCL = 1
     - No: SCL = 0?
       - Yes: Reset BRG, Force SCL = 0, Clear ACKEN, Set SSPIF
       - No: SCL = 1?
         - Yes: ACKDT = 1?
           - Yes: Load BRG with SSPADD<6:0>, start count.
           - No: SDA = 1?
             - Yes: Bus collision detected, Set BCLIF, Release SCL, Clear ACKEN
             - No: (Clock Arbitration)
2. No: BRG rollover?
   - Yes: Force SCL = 1
   - No: SCL = 1?
     - Yes: Load BRG with SSPADD<6:0>, start count.
     - No: (Clock Arbitration)
17.4.14 Stop Condition Timing

A stop bit is asserted on the SDA pin at the end of a receive/transmit by setting the Stop sequence enable bit, PEN (SSPCON2<2>). At the end of a receive/transmit the SCL line is held low after the falling edge of the ninth clock. When the PEN bit is set, the master will assert the SDA line low. When the SDA line is sampled low, the baud rate generator is reloaded and counts down to 0. When the baud rate generator times out, the SCL pin will be brought high, and one \( T_{BRG} \) (baud rate generator rollover count) later, the SDA pin will be de-asserted. When the SDA pin is sampled high while SCL is high the P bit (SSPSTAT<4>) is set. A \( T_{BRG} \) later, the PEN bit is cleared and the SSPIF bit is set (Figure 17-31).

Whenever the firmware decides to take control of the bus, it will first determine if the bus is busy by checking the S and P bits in the SSPSTAT register. If the bus is busy, then the CPU can be interrupted (notified) when a Stop bit is detected (i.e. bus is free).

17.4.14.1 WCOL Status Flag

If the user writes the SSPBUF when a STOP sequence is in progress, then the WCOL bit is set and the contents of the buffer are unchanged (the write doesn’t occur).

Figure 17-31: Stop Condition Receive or Transmit Mode

Write to SSPCON2
Set PEN

Falling edge of 9th clock

SCL

SDA

ACK

\( T_{BRG} \)

\( T_{BRG} \)

\( T_{BRG} \)

\( T_{BRG} \)

\( T_{BRG} \)

\( T_{BRG} \)

SCL brought high after \( T_{BRG} \)

SDA asserted low before rising edge of clock to setup stop condition.

Note: \( T_{BRG} \) = one baud rate generator period.
Figure 17-32: Stop Condition Flowchart

Idle Mode, SSPEN = 1, SSPCON1<3:0> = 1000

PEN = 1

Force SDA = 0 SCL doesn't change

SDA = 0?

No

Yes

Start BRG

BRG rollover?

No

Yes

BRG rollover?

Release SDA, Start BRG

No

Yes

P bit Set?

No

Yes

Bus Collision detected, Set BCLIF, Clear PEN

SDA going from 0 to 1 while SCL = 1, Set SSPIF, Stop Condition done

PEN Cleared

Start BRG

BRG rollover?

No

Yes

(P bit Set?)
17.4.15 Clock Arbitration

Clock arbitration occurs when the master, during any receive, transmit, or Repeated Start/stop condition de-asserts the SCL pin (SCL allowed to float high). When the SCL pin is allowed to float high, the baud rate generator (BRG) is suspended from counting until the SCL pin is actually sampled high. When the SCL pin is sampled high, the baud rate generator is reloaded with the contents of SSPADD<6:0> and begins counting. This ensures that the SCL 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 17-33).

![Figure 17-33: Clock Arbitration Timing in Master Transmit Mode](image)

17.4.16 Sleep Operation

While in sleep mode, the I2C module can receive addresses or data, and when an address match or complete byte transfer occurs wake the processor from sleep (if the MSSP interrupt is enabled).

17.4.17 Effect of a Reset

A reset disables the MSSP module and terminates the current transfer.
17.4.18 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 SDA pin, arbitration takes place when the master outputs a ‘1’ on SDA by letting SDA float high and another master asserts a ‘0’. When the SCL pin floats high, data should be stable. If the expected data on SDA is a ‘1’ and the data sampled on the SDA pin = ‘0’, then a bus collision has taken place. The master will set the Bus Collision Interrupt Flag, BCLIF and reset the I^2C port to its IDLE state. (Figure 17-34).

If a transmit was in progress when the bus collision occurred, the transmission is halted, the BF flag is cleared, the SDA and SCL lines are de-asserted, and the SSPBUF 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 SDA and SCL lines are de-asserted, and the respective control bits in the SSPCON2 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 SDA and SCL pins, and if a STOP condition occurs, the SSPIF bit will be set.

A write to the SSPBUF will start the transmission of data at the first data bit, regardless of where the transmitter left off when 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 SSPSTAT register, or the bus is idle and the S and P bits are cleared.

Figure 17-34: Bus Collision Timing for Transmit and Acknowledge

<table>
<thead>
<tr>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data changes while SCL = 0</td>
</tr>
<tr>
<td>SDA line pulled low by another source</td>
</tr>
<tr>
<td>Sample SDA. While SCL is high data doesn’t match what is driven by the master. Bus collision has occurred.</td>
</tr>
<tr>
<td>SDA released by master</td>
</tr>
<tr>
<td>Set bus collision interrupt (BCLIF).</td>
</tr>
<tr>
<td>SDA</td>
</tr>
<tr>
<td>SCL</td>
</tr>
<tr>
<td>BCLIF</td>
</tr>
</tbody>
</table>
17.4.18.1 Bus Collision During a START Condition

During a START condition, a bus collision occurs if:

a) SDA or SCL are sampled low at the beginning of the START condition (Figure 17-35).

b) SCL is sampled low before SDA is asserted low (Figure 17-36).

During a START condition both the SDA and the SCL pins are monitored.

If:

the SDA pin is already low
or the SCL pin is already low,

then:

the START condition is aborted,
and the BCLIF flag is set,
and the SSP module is reset to its IDLE state (Figure 17-35).

The START condition begins with the SDA and SCL pins de-asserted. When the SDA pin is sampled high, the baud rate generator is loaded from SSPADD<6:0> and counts down to 0. If the SCL pin is sampled low while SDA is high, a bus collision occurs, because it is assumed that another master is attempting to drive a data ‘1’ during the START condition.

If the SDA pin is sampled low during this count, the BRG is reset and the SDA line is asserted early (Figure 17-37). If however a ‘1’ is sampled on the SDA pin, the SDA pin is asserted low at the end of the BRG count. The baud rate generator is then reloaded and counts down to 0, and during this time, if the SCL pins is sampled as ‘0’, a bus collision does not occur. At the end of the BRG count the SCL pin is asserted low.

**Note:** The reason that bus collision is not a factor during a START condition is that no two bus masters can assert a START condition at the exact same time. Therefore, one master will always assert SDA before the other. This condition does not cause a bus collision because the two masters must be allowed to arbitrate the first address following the START condition, and if the address is the same, arbitration must be allowed to continue into the data portion, Repeated Start, or STOP conditions.
Figure 17-35: Bus Collision During Start Condition (SDA only)

- SDA goes low before the SEN bit is set. Set BCLIF.
- S bit and SSPIF set because SDA = 0, SCL = 1
- SDA sampled low before SDA goes low before the SEN bit is set.
- S bit and SSPIF set because SDA = 0, SCL = 1
- SCL = 0 before SDA = 0, SCL = 1
- SSPIF and BCLIF cleared in software.

Figure 17-36: Bus Collision During Start Condition (SCL = 0)

- SDA = 0, SCL = 1
- SCL = 0 before SDA = 0, SCL = 1
- SSPIF and BCLIF cleared in software.
Figure 17-37: BRG Reset Due to SDA Arbitration During Start Condition

- **SDA**: SDA pulled low by other master. Reset BRG and assert SDA.
- **SCL**: SCL pulled low after BRG Timeout.
- **SEN**: Set SEN, enable start sequence if SDA = 1, SCL = 1.
- **BCLIF**: Interrupts cleared in software.
- **SSPIF**: SDA = 0, SCL = 1. Set SSPIF.
17.4.18.2 Bus Collision During a Repeated Start Condition

During a Repeated Start condition, a bus collision occurs if:

a) A low level is sampled on SDA when SCL goes from low level to high level.
b) SCL goes low before SDA is asserted low, indicating that another master is attempting to transmit a data ‘1’.

When the user de-asserts SDA and the pin is allowed to float high, the BRG is loaded with SSPADD<<6:0>>, and counts down to zero. The SCL pin is then de-asserted, and when sampled high, the SDA pin is sampled. If SDA is low, a bus collision has occurred (i.e. another master, Figure 17-38, is attempting to transmit a data ‘0’). If, however, SDA is sampled high then the BRG is reloaded and begins counting. If SDA goes from high to low before the BRG times out, no bus collision occurs, because no two masters can assert SDA at exactly the same time.

If, however, SCL goes from high to low before the BRG times out and SDA has not already been asserted, then a bus collision occurs. In this case, another master is attempting to transmit a data ‘1’ during the Repeated Start condition, Figure 17-39.

If at the end of the BRG time out both SCL and SDA are still high, the SDA pin is driven low, the BRG is reloaded, and begins counting. At the end of the count, regardless of the status of the SCL pin, the SCL pin is driven low and the Repeated Start condition is complete.

Figure 17-38: Bus Collision During a Repeated Start Condition (Case 1)

<table>
<thead>
<tr>
<th>SDA</th>
<th>SCL</th>
<th>RSEN</th>
<th>BCLIF</th>
<th>S</th>
<th>SSPIF</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>'0'</td>
</tr>
</tbody>
</table>

Sample SDA when SCL goes high.
If SDA = 0, set BCLIF and release SDA and SCL

Cleared in software

'0'
Figure 17-39: Bus Collision During Repeated Start Condition (Case 2)

- SDA
- SCL
- BCLIF: SCL goes low before SDA, Set BCLIF, Release SDA and SCL
- RSEN
- S: '0'
- SSPIF: '0'

Interrupt cleared in software

TBRG 

TBRG
17.4.18.3 Bus Collision During a STOP Condition

Bus collision occurs during a STOP condition if:

a) After the SDA pin has been de-asserted and allowed to float high, SDA is sampled low after the BRG has timed out.

b) After the SCL pin is de-asserted, SCL is sampled low before SDA goes high.

The STOP condition begins with SDA asserted low. When SDA is sampled low, the SCL pin is allow to float. When the pin is sampled high (clock arbitration), the baud rate generator is loaded with SSPADD<6:0> and counts down to 0. After the BRG times out SDA is sampled. If SDA is sampled low, a bus collision has occurred. This is due to another master attempting to drive a data '0' (Figure 17-40). If the SCL pin is sampled low before SDA is allowed to float high, a bus collision occurs. This is another case of another master attempting to drive a data '0' (Figure 17-41).

Figure 17-40: Bus Collision During a STOP Condition (Case 1)

![Diagram of Case 1]

Figure 17-41: Bus Collision During a STOP Condition (Case 2)

![Diagram of Case 2]
17.5 Connection Considerations for I2C Bus

For standard-mode I2C bus devices, the values of resistors Rp and Rs in Figure 17-42 depends on the following parameters:

- Supply voltage
- Bus capacitance
- Number of connected devices (input current + leakage current)

The supply voltage limits the minimum value of resistor Rp due to the specified minimum sink current of 3 mA at Volmax = 0.4V for the specified output stages. For example, with a supply voltage of Vdd = 5V±10% and Volmax = 0.4V at 3 mA, Rpmin = (5.5-0.4)/0.003 = 1.7 kΩ. Vdd as a function of Rp is shown in Figure 17-42. The desired noise margin of 0.1Vdd for the low level, limits the maximum value of Rs. Series resistors are optional, and used to improve ESD susceptibility.

The bus capacitance is the total capacitance of wire, connections, and pins. This capacitance limits the maximum value of Rp due to the specified rise time (Figure 17-42).

The SMP bit is the slew rate control enabled bit. This bit is in the SSPSTAT register, and controls the slew rate of the I/O pins when in I2C mode (master or slave).

Figure 17-42: Sample Device Configuration for I2C Bus

NOTE: I2C devices with input levels related to Vdd must have one common supply line to which the pull up resistor is also connected.
17.6  Initialization

Example 17-2:  SPI Master Mode Initialization

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLRF STATUS</td>
<td>Bank 0</td>
</tr>
<tr>
<td>CLRF SSPSTAT</td>
<td>SMP = 0, CKE = 0, and clear status bits</td>
</tr>
<tr>
<td>BSF SSPSTAT, CKE</td>
<td>CKE = 1</td>
</tr>
<tr>
<td>MOVWF 0x31</td>
<td>Set up SPI port, Master mode, CLK/16, Data xmit on falling edge (CKE=1 &amp; CKP=1)</td>
</tr>
<tr>
<td>MOVWF SSPCON</td>
<td>Data sampled in middle (SMP=0 &amp; Master mode)</td>
</tr>
<tr>
<td>BSF STATUS, RP0</td>
<td>Bank 1</td>
</tr>
<tr>
<td>BSF PIE, SSPIE</td>
<td>Enable SSP interrupt</td>
</tr>
<tr>
<td>BCF STATUS, RP0</td>
<td>Bank 0</td>
</tr>
<tr>
<td>BSF INTCON, GIE</td>
<td>Enable, enabled interrupts</td>
</tr>
<tr>
<td>MOVLW DataByte</td>
<td>Data to be Transmitted</td>
</tr>
<tr>
<td>MOVWF SSPBUF</td>
<td>Start Transmission</td>
</tr>
</tbody>
</table>

17.6.1  Master SSP Module / Basic SSP Module Compatibility

When changing from the SPI in the Basic SSP module, the SSPSTAT register contains two additional control bits. These bits are:

- SMP, SPI data input sample phase
- CKE, SPI Clock Edge Select

To be compatible with the SPI of the Master SSP module, these bits must be appropriately configured. If these bits are not at the states shown in Table 17-4, improper SPI communication may occur.

Table 17-4:  New bit States for Compatibility

<table>
<thead>
<tr>
<th>Basic SSP Module</th>
<th>Master SSP Module</th>
</tr>
</thead>
<tbody>
<tr>
<td>CKP</td>
<td>CKP</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
17.7 Design Tips

Question 1: Using SPI mode, I do not seem able to talk to an SPI device.
Answer 1:
Ensure that you are using the correct SPI mode for that device. This SPI supports all 4 SPI modes so you should be able to get it to function. Check the clock polarity and the clock phase.

Question 2: Using I2C mode, I write data to the SSPBUF register, but the data did not transmit.
Answer 2:
Ensure that you set the CKP bit to release the I2C clock.
## 17.8 Related Application Notes

This section lists application notes that are related to this section of the manual. These application notes may not be written specifically for the Mid-Range MCU family (that is they may be written for the Base-Line, or High-End families), but the concepts are pertinent, and could be used (with modification and possible limitations). The current application notes related to the Master SSP module are:

<table>
<thead>
<tr>
<th>Title</th>
<th>Application Note #</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use of the SSP Module in the (I^2C) Multi-Master Environment.</td>
<td>AN578</td>
</tr>
<tr>
<td>Using Microchip 93 Series Serial EEPROMs with Microcontroller SPI Ports</td>
<td>AN613</td>
</tr>
<tr>
<td>Interfacing PIC16C64/74 to Microchip SPI Serial EEPROM</td>
<td>AN647</td>
</tr>
<tr>
<td>Interfacing a Microchip PIC16C92x to Microchip SPI Serial EEPROM</td>
<td>AN668</td>
</tr>
</tbody>
</table>
17.9 Revision History

Revision A

This is the initial released revision of the Master SSP module description.