Section 16. Basic Synchronous Serial Port (BSSP)

HIGHLIGHTS

This section of the manual contains the following major topics:

16.1 Introduction ................................................................. 16-2
16.2 Control Registers ........................................................... 16-3
16.3 SPI™ Mode ................................................................. 16-6
16.4 SSP I²C Operation ......................................................... 16-15
16.5 Initialization .............................................................. 16-23
16.6 Design Tips ............................................................... 16-24
16.7 Related Application Notes ............................................. 16-25
16.8 Revision History ........................................................ 16-26

Note: Please refer to Appendix C.2 or the device data sheet to determine which devices use this module.

SPI is a trademark of Motorola Corporation.
I²C is a trademark of Philips Corporation.
16.1 Introduction

The Basic Synchronous Serial Port (BSSP) 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 BSSP module can operate in one of two modes:

- Serial Peripheral Interface (SPI™)
- Inter-Integrated Circuit (I²C™)
  - Slave mode
  - I/O slope control, Start and Stop bits to ease software implementation of Master and Multi-master modes

I²C is a trademark of Philips Corporation.
### 16.2 Control Registers

<table>
<thead>
<tr>
<th>Register 16-1: SSPSTAT: Synchronous Serial Port Status Register</th>
</tr>
</thead>
<tbody>
<tr>
<td>U-0</td>
</tr>
<tr>
<td>---</td>
</tr>
<tr>
<td>---</td>
</tr>
</tbody>
</table>

| bit 7:6 | Unimplemented: Read as ‘0’ |
| 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) |
|        | 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) |
|        | 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. |
|        | 1 = Read |
|        | 0 = Write |

| 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 |

| bit 0  | BF: Buffer Full Status bit |
|        | Receive (SPI and I²C modes) |
|        | 1 = Receive complete, SSPBUF is full |
|        | 0 = Receive not complete, SSPBUF is empty |
|        | Transmit (I²C mode only) |
|        | 1 = Transmit in progress, SSPBUF is full |
|        | 0 = Transmit complete, SSPBUF is empty |

**Legend**

- R = Readable bit
- W = Writable bit
- U = Unimplemented bit, read as ‘0’
- n = Value at POR reset
Register 16-2: SSPCON: Synchronous Serial Port Control Register

<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
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. 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 I²C 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.
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 I²C 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 I²C mode:
SCK release control
1 = Enable clock
0 = Holds clock low (clock stretch) (Used to ensure data setup time)
Register 16-2:  SSPCON: Synchronous Serial Port Control Register  (Cont’d)

bit 3:0  SSPM3:SSPM0: Synchronous Serial Port Mode Select bits

0000 = SPI master mode, clock = Fosc/4
0001 = SPI master mode, clock = Fosc/16
0010 = SPI master mode, clock = Fosc/64
0011 = SPI master mode, clock = TMR2 output/2
0100 = SPI slave mode, clock = SCK pin. SS pin control enabled.
0101 = SPI slave mode, clock = SCK pin. SS pin control disabled. SS can be used as I/O pin
0110 = I²C slave mode, 7-bit address
0111 = I²C slave mode, 10-bit address
1000 = Reserved
1001 = Reserved
1010 = Reserved
1011 = I²C Firmware controlled Master mode (slave idle)
1100 = Reserved
1101 = Reserved
1110 = I²C Firmware controlled Multi-Master mode,
    7-bit address with start and stop bit interrupts enabled
1111 = I²C Firmware controlled Master mode,
    10-bit address with start and stop bit interrupts enabled

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

The SPI mode allows 8-bits of data to be synchronously transmitted and received simultaneously. 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)

16.3.1 Operation

When initializing the SPI, several options need to be specified. This is done by programming the appropriate control bits in the SSPCON register (SSPCON<5:0>). 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 (Output/Input data on the Rising/Falling edge of SCK)
- Clock Rate (Master mode only)
- Slave Select Mode (Slave mode only)

Figure 16-1 shows the block diagram of the SSP module, when in SPI mode.

Figure 16-1: 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 previously written to the SSPSR, until the received data is ready. Once the 8-bits of data have been received, that information is moved to the SSPBUF register. Then the buffer full detect bit, BF (SSPSTAT<0>), and 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 received. Any write to the SSPBUF register during transmission/reception of data will be ignored, and the write collision detect bit, WCOL (SSPCON<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 can then be read (if data is meaningful) and/or the SSPBUF (SSPSR) can be 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 16-1 shows the loading of the SSPBUF (SSPSR) for data transmission. The shaded instruction is only required if the received data is meaningful (some SPI applications are transmit only).

**Example 16-1: 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 from addressing the SSPBUF register. Additionally, the SSP status register (SSPSTAT) indicates the various status conditions.
16.3.2 Enabling SPI I/O

To enable the serial port, SSP enable bit, SSPEN (SSPCON<5>), must be set. To reset or reconfigure SPI mode, clear the SSPEN bit which re-initializes the SSPCON register, 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, they must have their data direction bits (in the TRIS register) appropriately programmed. That is:

- SDI must have the TRIS bit set
- SDO must have the TRIS bit cleared
- SCK (Master mode) must have the TRIS bit cleared
- SCK (Slave mode) must have the TRIS bit set
- SS must have the 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. An example would be in master mode where you are only sending data (to a display driver), then both SDI and SS could be used as general purpose outputs by clearing their corresponding TRIS register bits.
16.3.3 Typical Connection

Figure 16-2 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 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 16-2: SPI Master/Slave Connection](image-url)
16.3.4 Master Operation

The master can initiate the data transfer at any time because it controls the SCK. The master determines when the slave (Processor 2) wishes 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 (SSPCON<4>). This then would give waveforms for SPI communication as shown in Figure 16-5 and Figure 16-5 where the MSb is transmitted first. In master mode, the SPI clock rate (bit rate) is user programmable to be one of the following:

- Fosc/4 (or Tcy)
- Fosc/16 (or 4 • Tcy)
- Fosc/64 (or 16 • Tcy)
- Timer2 output/2

This allows a maximum data rate of 5 Mbps (at 20 MHz).

![Figure 16-3: SPI Mode Waveform (Master Mode)](image)
16.3.5 Slave Operation

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.

The clock polarity is selected by appropriately programming the CKP bit (SSPCON<4>). This then would give waveforms for SPI communication as shown in Figure 16-5 and Figure 16-5 where the MSb is transmitted first. When in slave mode the external clock must meet the minimum high and low times.

In sleep mode, the slave can transmit and receive data and wake the device from sleep if the interrupt is enabled.

Figure 16-4: SPI Mode Waveform (Slave Mode w/o SS Control)
16.3.6 Slave Select Mode

The SS pin allows a synchronous slave mode. The SPI must be in slave mode (SSPCON<3:0> = 04h) and the TRIS bit must be set for the synchronous slave mode to be enabled. 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. If the SS pin is taken low without resetting SPI mode, the transmission will continue from the point at which it was taken high. To clear the bit counter the Basic SSP module must be disabled and then re-enabled. External pull-up/pull-down resistors may be desirable, depending on the application.

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.

Figure 16-5: SPI Mode Waveform (Slave Mode with SS Control)
Figure 16-6: Slave Synchronization Waveform
16.3.7 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 SSP interrupt flag bit will be set and if enabled will wake the device from sleep.

16.3.8 Effects of a Reset

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

Table 16-1: 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>RBIF(2)</td>
<td>0000 000x</td>
<td>0000 000u</td>
</tr>
<tr>
<td>PIR</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>SSPIE (1)</td>
<td></td>
<td></td>
<td></td>
<td>00 00</td>
<td></td>
</tr>
<tr>
<td>PIE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>SSPIE (1)</td>
<td></td>
<td></td>
<td></td>
<td>00 00</td>
<td></td>
</tr>
<tr>
<td>SSPBUF</td>
<td>Synchronous Serial Port Receive Buffer/Transmit Register</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>xxxx xxxx</td>
<td>xxxxxxx xxxxxxx</td>
</tr>
<tr>
<td>SSPCON</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>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>D/A</td>
<td>P</td>
<td>S</td>
<td>R/W</td>
<td>UA  BF</td>
<td>--00 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.
Note 2: These bits can also be named GPIE and GPIF.
16.4  SSP I\textsuperscript{2}C Operation

The SSP module in I\textsuperscript{2}C mode fully implements all slave functions, except General Call Support, and provides interrupts on start and stop bits in hardware to facilitate software implementations of the master functions. The SSP module implements the standard and fast mode specifications as well as 7-bit and 10-bit addressing. Appendix A gives an overview of the I\textsuperscript{2}C bus specification.

Two pins are used for data transfer. These are the SCL pin, which is the clock, and the SDA pin, which is the data. The user must configure these pins as inputs through the TRIS bits. The SSP module functions are enabled by setting SSP Enable bit, SSPEN (SSPCON<5>).

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 16-7: SSP Block Diagram (I\textsuperscript{2}C Mode)
The SSP module has five registers for I\textsuperscript{2}C operation. They are:

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

The SSPCON register allows control of the I\textsuperscript{2}C operation. Four mode selection bits (SSPCON<3:0>) allow one of the following I\textsuperscript{2}C modes to be selected:

- I\textsuperscript{2}C Slave mode (7-bit address)
- I\textsuperscript{2}C Slave mode (10-bit address)
- I\textsuperscript{2}C Firmware controlled Multi-Master mode, 7-bit address (start and stop bit interrupts enabled)
- I\textsuperscript{2}C Firmware controlled Multi-Master mode, 10-bit address (start and stop bit interrupts enabled)
- I\textsuperscript{2}C Firmware controlled Master mode, slave is idle

Before selecting any I\textsuperscript{2}C mode, the SCL and SDA pins must be programmed to inputs by setting the appropriate TRIS bits. Selecting an I\textsuperscript{2}C mode, by setting the SSPEN bit, enables the SCL and SDA pins to be used as the clock and data lines in I\textsuperscript{2}C 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 SSPSTAT register is read only.

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 doubled 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 the SSPIF flag bit is set. If another complete byte is received before the SSPBUF register is read, a receiver overflow has occurred and bit SSPOV (SSPCON<6>) is set.

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

In slave mode, the SCL and SDA pins must be configured as inputs (TRIS bits set). The SSP module will override 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 (SSPCON<6>), was set before the transfer was received.

In this case, the SSPSR register value is not loaded into the SSPBUF, but bit SSPIF and SSPOV bits are set. Table 16-2 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 the SSPOV bit 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 I²C specification as well as the requirement of the SSP module are given in parameter 100 and parameter 101 of the “Electrical Specifications” section.
16.4.1.1 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 eight SCL pulse.
b) The buffer full bit, BF, is set on the falling edge of the eight 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 the SSPIF flag bit.
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.

Note: Following the RESTART 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 16-2: 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>No</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.
16.4.1.2 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 (SSPCON<6>) is set.

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.

Figure 16-8: I²C Waveforms for Reception (7-bit Address)
16.4.1.3 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 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 (SSPCON<4>). The master must 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 16-9).

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 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 ACK), then the data transfer is complete. When the not 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 (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 16-9: I2C Waveforms for Transmission (7-bit Address)

16.4.1.4 Clock Arbitration

Clock arbitration has the SCL pin to inhibit the master device from sending the next clock pulse. The SSP module in I2C slave mode will hold the SCL pin low when the CPU needs to respond to the SSP interrupt (SSPIF bit is set and the CKP bit is cleared). The data that needs to be transmitted will need to be written to the SSPBUF register, and then the CKP bit will need to be set to allow the master to generate the required clocks.
16.4.2 Master Mode (Firmware)

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 clearing the corresponding TRIS bit(s). The output level is always low, irrespective of the value(s) in PORT. So when transmitting data, a ‘1’ data bit must have the TRIS bit set (input) and a ‘0’ data bit must have the TRIS bit cleared (output). The same scenario is true for the SCL line with the TRIS bit.

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

- START condition
- STOP condition
- Data transfer byte transmitted/received

Master mode of operation can be done with either the slave mode idle (SSPM3:SSPM0 = 1011) or with the slave active (SSPM3:SSP0 = 1110 or 1111). When the slave modes are enabled, the software needs to differentiate the source(s) of the interrupt.

16.4.3 Multi-Master Mode (Firmware)

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 to see if the signal level is the expected output level. This check only needs to be done when a high level is output. If a high level is expected and a low level is present, the device needs to release the SDA and SCL lines (set the TRIS bits). There are two stages where this arbitration can be lost, they are:

- Address Transfer
- Data Transfer

When the slave logic is enabled, the slave continues to receive. If arbitration was lost during the address transfer stage, communication to the device may be in progress. If addressed an ACK pulse will be generated. If arbitration was lost during the data transfer stage, the device will need to re-transfer the data at a later time.
16.4.4 Sleep Operation

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

16.4.5 Effect of a Reset

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

Table 16-3: Registers Associated with I^2C 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>RBIF(2)</td>
<td>0000 000x</td>
<td>0000 000u</td>
</tr>
<tr>
<td>PIR</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></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></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>SSPBUF</td>
<td>Synchronous Serial Port Receive Buffer/Transmit Register</td>
<td>0000 0000</td>
<td>0000 0000</td>
<td>0000 0000</td>
<td>0000 0000</td>
<td>0000 0000</td>
<td>0000 0000</td>
<td>0000 0000</td>
<td>0000 0000</td>
<td>0000 0000</td>
</tr>
<tr>
<td>SSPADD</td>
<td>Synchronous Serial Port (I^2C mode) Address Register</td>
<td>0000 0000</td>
<td>0000 0000</td>
<td>0000 0000</td>
<td>0000 0000</td>
<td>0000 0000</td>
<td>0000 0000</td>
<td>0000 0000</td>
<td>0000 0000</td>
<td>0000 0000</td>
</tr>
<tr>
<td>SSPSTAT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Legend: $x =$ unknown, $u =$ unchanged, $-$ = unimplemented read as '0'.

Shaded cells are not used by SSP in I^2C mode.

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

2: These bits can also be named GPIE and GPIF.
16.5 Initialization

Example 16-2: SPI Master Mode Initialization

```assembly
   CLRF STATUS       ; Bank 0
   CLRF SSPSTAT      ; Clear status bits
   MOVLW 0x31         ; Set up SPI port, Master mode, CLK/16,
   MOVWF SSPCON      ;   Data xmit on rising edge
                      ;   Data sampled in middle
   BSF STATUS, RP0   ; Bank 1
   BSF PIE1, SSPIE   ; Enable SSP interrupt
   BCF STATUS, RP0   ; Bank 0
   BSF INTCON, GIE   ; Enable, enabled interrupts
   MOVLW DataByte    ; Data to be Transmitted
                      ;   Could move data from RAM location
   MOVWF SSPBUF      ; Start Transmission
```

16.5.1 SSP Module / Basic SSP Module Compatibility

When changing from the SSP Module to 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 Basic SSP module, these bits must be appropriately configured. If these bits are not at the states shown in Table 16-4, improper SPI communication should be expected. If the SSP module uses a different configuration then shown in Table 16-4, the Basic SSP module can not be used to implement that mode. That mode may be implemented in software.

<table>
<thead>
<tr>
<th>Basic SSP Module</th>
<th>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>

Table 16-4: New Bit States for Compatibility
16.6 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 two of the four SPI modes so ensure that the SPI device that you are trying to interface to is compatible with one of these two modes. Check the clock polarity and the clock phase.

If the device is not compatible, switch to one of the Microchip devices that has the SSP module, and that should solve this.

**Question 2:** *Using I²C mode, I do not seem able to make the master mode work.*

**Answer 2:**
This SSP module does not have master mode fully automated in hardware, see Application Note AN578 for software which uses the SSP module to implement master mode. If you require a fully automated Hardware implementation of I²C master mode, please refer to the Microchip Line Card for devices that have the Master SSP module.

**Note:** At the time of printing only the High-end family of devices (PIC17CXXX) have devices with the Master SSP module implemented.

**Question 3:** *Using I²C mode, I write data to the SSPBUF register, but the data did not transmit.*

**Answer 3:**
Ensure that you set the CKP bit to release the I²C clock.
16.7 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 this section are:

<table>
<thead>
<tr>
<th>Title</th>
<th>Application Note #</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use of the SSP Module in the I^2CO 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>Software Implementation of I^2C Bus Master</td>
<td>AN554</td>
</tr>
<tr>
<td>Use of the SSP module in the Multi-Master Environment</td>
<td>AN578</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>
16.8 Revision History

Revision A

This is the initial revision of the Basic SSP module description.