TB3192

Using the SPI Module on 8-Bit PIC® Microcontrollers

Introduction

Author: Jason Layton, Microchip Technology Inc.

The Serial Peripheral Interface (SPI) module is a synchronous serial interface useful for communicating with other peripheral or microcontroller devices. These peripheral devices can be serial EEPROMs, shift registers, display drivers, Analog-to-Digital Converters and so on. The SPI module is compatible with Motorola® SPI and Serial Input/Output Port (SIOP) interfaces.

The Serial Peripheral Interface (SPI) has long been part of the PIC® MCU set of core peripherals. One of the newest features introduced with the PIC18F2x/4xK42 microcontroller series is a stand-alone module for the SPI functionality. Previous devices have integrated SPI capabilities with the other serial communications protocols and implemented them in the Master Synchronous Serial Port (MSSP) module, where SPI shared resources such as registers and interrupt flags.

The dedicated module for SPI allows significant improvement in the implementation of traditional SPI capabilities and expansions by adding new features for more flexibility and control. This technical brief discusses the implementation, modes of operation and other additional features of the SPI module

Table of Contents

1. SPI Module Overview

The SPI module can be implemented in a variety of different configurations to fit the scope of an application. In addition to being configured as a basic master or slave in the SPI protocol, this SPI module offers additional configurations within those basic options that give the user even more control, allowing them to tailor it to their needs. In the SPI protocol, devices communicate in a single master/multiple slave environment, where the master device initiates the data exchange. When a device is configured as a master, it provides and controls the serial clock (SCK) signal. In addition to the modes that the SPI feature of the MSSP module supported, the following features are now available with this dedicated SPI module:

- SDI, SDO, and $\overline{SS}$ Polarity Control
- Separate Transmit and Receive Enables
- Separate Transmit and Receive Buffers with respective 2-Byte FIFOs
- DMA Bus Connection and Capabilities with Transmit and Receive Buffers
- Transfer Counter Integrated into the SPI Control Module

1.1 Serial Communication

To begin a serial communication in the SPI protocol, the master must send out the first clock signal (SCK) to the slave at a frequency that both devices support. The master device always controls the SPI timing and provides the clock signal to all of the slave devices on the bus. Slave devices are not capable of manipulating the SPI clock in any way. No data can be transferred unless a clock signal is present. The slave devices on the SPI bus will always receive the SPI clock signal, but will not respond to the SPI master until their respective Slave Select line has been activated. Once a slave device has been initiated by the SPI master, the slave can respond and data can be transferred one bit at a time, sequentially over the SPI bus.

1.2 Data Transfers

Data transfers between master and slave devices on the SPI bus are handled by separate transmit and receive FIFO buffers. Refer to the data sheet for more information about the separate transmit and receive FIFO bufferss found on this SPI module. The SPI module works in Full-Duplex mode, meaning that during each clock cycle, the SPI module is simultaneously transmitting and receiving data from the activated slave. This bit-wise exchange of data will continue until there is no more data to be exchanged or an interrupt occurs, at which point the master will stop sending clock signals to the slave.

1.3 Interface with Slave Devices

The SPI module supports interfacing with one or more slave devices on the bus. It is important to note that when multiple slave devices are being used, each slave should have an independent Slave Select connection from the master. Each slave should have their own Slave Select line, and should be connected so that they all receive the SPI clock signal from the master device. Although the master always controls the SPI timing and provides the clock signal to all the slaves on the bus, only the slave that is selected will respond to the master. Once a slave has been selected by activating the Slave Select line, which can be active-high or active-low depending on the configuration, then it will respond to the master and can begin transferring data. The master device will have three signal lines that are common to all of the slave devices on the bus, and then a slave select line for each of the slave devices. It is important to refer to the documentation for the slave devices on the SPI bus to determine their requirements. Refer to the device data sheet for more information about connecting devices on the SPI bus. The four signal connections that are used by this SPI module are:

- Serial Clock (SCK)
- Serial Data Out (SDO)
- Serial Data In (SDI)
- Slave Select ($\overline{SS}$)

2. Configuration of the SPI Peripheral

Operation of the SPI module is controlled by the following registers:

- SPIxCON0: Used to enable/disable the SPI module, select LSb or MSb first data exchange, specify whether the device is a master or slave, and contains the bit length mode select bit (BMODE)
- SPIxCON1: Used to configure the polarity of the $\overline{SS}$, SDI, SDO, SDK lines, enable fast start, select leading clock edge on CKE, and set the sample phase control bit.
- SPIxCON2: Used to configure slave select settings, and contains the TXR and RXR control bits.
- SPIxTWIDTH: SPI Transfer Width Register
- SPIxBAUD: SPI Baud Rate Generator Control Register
- SPIxINTE: Interrupt Enable Register
- SPIxINTF: Interrupt Flag Register
- SPIxTCTH/L: SPI Transfer Counter Register Pair
- SPIxSTATUS: FIFO Status Register
- SPIxRxB: Receive Buffer Register
- SPIxTxB: Transmit Buffer Register
- SPIxCLK: Clock Source Selection Register

Table 2-1. Registers Associated with SPI

Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
SPIxINTF	SRMTIF	TCZIF	SOSIF	EOSIF	-	RXOIF	TXUIF	-
SPIxINTE	SRMTIE	TCZIE	SOSIE	EOSIE	-	RXOIE	TXUIE	-
SPIxTCNTH	-	-	-	-	-	TCNT10	TCNT9	TCNT8
SPIxTCNTL	TCNT7	TCNT6	TCNT5	TCNT4	TCNT3	TCNT2	TCNT1	TCNT0
SPIxTWIDTH	-	-	-	-	-	TWIDTH2	TWIDTH1	TWIDTH0
SPIxBAUD	BAUD7	BAUD6	BAUD5	BAUD4	BAUD3	BAUD2	BAUD1	BAUD0
SPIxCON0	EN	-	-	-	-	LSBF	MST	BMODE
SPIxCON1	SMP	CKE	CKP	FST	-	SSO	SDIP	SDOP
SPIxCON2	BUSY	SSFLT	-	-	-	SSET	TXR	RXR
SPIxSTATUS	TXWE	-	TXBE	-	RXRE	CLRBF	-	RXBF
SPIxRXB	RXB7	RXB6	RXB5	RXB4	RXB3	RXB2	RXB1	RXB0
SPIxTXB	TXB7	TXB6	TXB5	TXB4	TXB3	TXB2	TXB1	TXB0
SPIxCLK	-	-	-	-	CLK3	CLK2	CLK1	CLK0

Table 2-2. Registers Associated with IO Control

Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
SPIxSCKPPS	-	-	-	SPIxSCKPPS4	SPIxSCKPPS3	SPIxSCKPPS2	SPIxSCKPPS1	SPIxSCKPPS0
SPIxSSPPS	-	-	-	SPIxSSPPS4	SPIxSSPPS3	SPIxSSPPS2	SPIxSSPPS1	SPIxSSPPS0
SPIxSDIPPS	-	-	-	SPIxSDIPPS4	SPIxSDIPPS3	SPIxSDIPPS2	SPIxSDIPPS1	SPIxSDIPPS0

2.1 Configuration Overview

When configuring the SPI module on a PIC® device, the first step should be to ensure that all devices have been connected correctly in hardware on the SPI bus. Once the master device has been connected to all slave devices in hardware, the next step is to assign necessary pins for SPI communication in software. The SPI signals can be rerouted to different pins using the Peripheral Pin Select (PPS) feature on newer microcontrollers. Refer to the device data sheet for rerouting options. Configuration of these pins, along with the configuration of the microcontroller, should all be a part of the initial device setup.

After the pins and system have been configured, the next step is to initialize the SPI module in software. The example below shows how to initialize the system oscillator, configure the SPI pins using Peripheral Pin Select and initialize the SPI module. The main SPI module control registers are SPIxCON0, SPIxCON1, and SPIxCON2. In addition to those registers, the SPIxBAUD register and the SPIxCLK registers are initialized in this example to demonstrate setting the frequency of the SCK output signal.

Note: This example was implemented using the PIC18F25K42 microcontroller; please note that minor changes may need to be made for this example to work on different PIC devices with this SPI module.

Example 2-1. BASIC SETUP AND INITIALIZATION OF THE SPI MODULE

```
// Oscillator Configuration
OSCCON1 = 0x60;   // HFINTOSC; NDIV = 1;
OSCCON3 = 0x00;   // CSWHOLD may proceed; SOSCPWR Low power;
OSCEN = 0x00;     // Oscillator Manual Enable Register;
OSCFRQ = 0x03;    // HFINTOSC = 8 MHz;
OSCTUNE = 0x00;   // // Calibrated Frequency (Default);
// Port Configuration
TRISA = 0x00;     // RA3 = output;
TRISB = 0x10;     // RB1 = input, RB3 = output;
ANSELA = 0x00;    // Digital I/O;
ANSELB = 0x00;    // Digital I/O;
SPI1SDIPPSbits.SPI1SDIPPS = 0x0A; // RB2->SPI1:SDI1; -- input;
RB1PPS = 0x1E;    // RB1->SPI1:SCK1; -- output;
RB3PPS = 0x1F;    // RB3->SPI1:SDO1; -- output;
RA3PPS = 0x20;    // RA3->SPI1:SS1; -- output;
// SPI Configuration
SPI1CON1 = 0x64;  // CKE; CKP; SDI/SDO Polarity;
SPI1CON2 = 0x07;  // TXR; RXR;
SPI1BAUD = 0x07;  // SPI Baud Pre-Scaler;
SPI1CLK = 0x00;   // SPI Clock Select;
SPI1CON0 = 0x83;  // SPI Enable; BMODE; Master/Slave; MSb/LSb;
```

2.2 Clock Sources

The SPIxCLK register allows the user to select different clock sources. The SPIxBAUD register is used for dividing the clock source selected to generate the SCK output signal. The frequency of the SCK output signal is defined by the equation below.

Equation 2-1. Frequency of SCK Output Signal Calculation

$$F_{BAUD} = \frac{F_{CSEL}}{((2 \times BAUD) + 1)}$$

where F_{BAUD} is that baud rate frequency output on the SCK pin, F_{CSEL} is the frequency of the input clock selected by the SPIxCLK register, and BAUD is the value of the SPIxBAUD register.

In the SPI protocol, there are four different configurations that control the timing of data transmissions. The configuration of the Clock Edge Select bit (CKE) and the Clock Polarity Select bit (CKP) are used to configure the SPI master to match the mode of the other devices on the SPI bus. The most common mode for SPI slave communication is when data is sampled at the leading rising edge of the clock (CKP = 0 / CKE = 0). It is important to check the SPI slave device specifications to ensure that the proper polarity and clock edge settings have been made for the SPI master. Refer to the device data sheet for more information about how the CKP and CKE bits affect SPI clocking.

The complete list of registers associated with this peripheral, along with their descriptions, can be found in the device data sheet.

3. Transfer Counter

The transfer counter is a feature that has been added to the SPI module and provides the capability to determine how many data transfers that the SPI will send/receive. It is important to configure the transfer counter and the BMODE (Bit-Length Mode Selection) bit of the SPIxCON0 register appropriately to ensure that the SPI module operates as intended. The functionality of the transfer counter depends on the value of the BMODE bit of the SPIxCON0 register.

3.1 Total Bit Count Mode (BMODE = 0)

In this mode, SPIxTCTH, SPIxTCTL and SPIxTWIDTH<2:0> are combined to create a 14-bit transfer counter. When BMODE = 0 the SPI master communicates with the slave by transferring 8-bit data packets. As the data packets are sent from the SPI buffer, the transfer counter will decrement by eight with each transfer. The master will continue to send packets of data until the transfer counter has decremented to zero. Once the value stored in the transfer counter has reached zero, data transfer will cease until the transfer counter is reloaded. The SPI master will only begin sending data once the transfer counter has a non-zero value written to it.

3.2 Variable Transfer Size Mode (BMODE = 1)

In this mode, SPIxTCTH<2:0> and SPIxTCTL are combined to form an 11-bit transfer counter that specifies the number of data transfers that are to occur. When BMODE = 1, the SPIxTWIDTH register signifies the width of the data packets that will be transferred by the SPI master. Every time a data transfer occurs, the 11-bit transfer counter will decrement by the width of the packet, which is dependent on the configuration of the SPIxTWIDTH register. When the SPI master is configured in Transmit-Only mode, the transfer counter will decrement with each packet being transferred, however, it will not cease communication once the transfer counter has reached zero.

4. Master Mode

There are four different configurations for the SPI module while in Master mode that are selected by setting the TXR and RXR bits of the SPIxCON2 register. The four modes of operation available, while the SPI device is operating as a master, are illustrated in Table 4-1. There are typically three types of scenarios in a SPI data transaction:

- The master sends useful data and the slave sends useful data.
- The master sends useful data and the slave sends dummy data.
- The master sends dummy data and the slave sends useful data.

This SPI module allows the user to handle these different scenarios without software involvement by configuring the TXR and RXR pins of the SPI1xCON2 register accordingly.

Table 4-1. Master Mode TXR/RXR Settings

	TXR = 1	**TXR = 0**
RXR = 1	Full-Duplex (legacy) mode If BMODE = 1, transfer when RXFIFO is not full and TXFIFO is not empty. If BMODE = 0, transfer when RXFIFO is not full, TXFIFO is not empty and the transfer counter is non-zero.	Receive-Only mode Transfer when RxFIFO is not full and the transfer counter is non-zero. Transmitted data is either the top of the most-recently received data.
RXR = 0	Transmit-Only mode If BMODE = 1, transfer when TXFIFO is not empty. If BMODE = 0, transfer when TXFIFO is not empty and the transfer counter is non-zero. Received data is not stored.	No Transfers

4.1 Full-Duplex Mode

Full-Duplex mode most closely matches the operation of the SPI incorporated into the MSSP module of older PIC devices, allowing for simultaneous exchange of data between the master and slave devices. To configure the SPI module as a master in Full-Duplex mode, the RXR and TXR registers of the SPIxCON2 register must both be set high. Depending on the status of the BMODE bit, the transfer counter may need to be loaded to initiate the SPI module to begin transferring data. The example below demonstrates the basic configuration of the SPI module in Full-Duplex mode.

Example 4-1. CONFIGURATION OF A SPI MASTER IN FULL DUPLEX MODE

```
uint8_t SPI1_Master_FullDuplex (uint8_t data)
{
SPI1CON2bits.TXR = 1; //Transmit Data-Required Bit;
SPI1CON2bits.RXR = 1; // Receive FIFO Space-Required Bit;
SPI1TCNT = 1; // Load SPI Transfer Counter;
SPI1TXB = data; // Load data into SPI transmit buffer;
while (PIR2bits.SPI1RXIF == 0); // Check for any SPI Receive Interrupts;
return (SPI1RXB); // Return data from SPI Receive Buffer Register;
}
```

4.2 Transmit-Only Mode

A feature unique to this SPI module is the ability to configure the master to operate in Transmit-Only mode. While in this mode, any data that is received will not be stored in the RXFIFO and the SPI master will continue transmitting data via the transmit buffer. To configure the SPI module as a master in Transmit-Only mode, the TXR bit of the SPIxCON2 register must be set, and the RXR bit of the SPIxCON2 register must be cleared. Depending on the status of the BMODE bit, a value may need to be written to the transfer counter to initiate the SPI to begin transferring data. The example below demonstrates the basic configuration of the SPI module in Transmit-Only mode.

Example 4-2. CONFIGURATION OF A SPI MASTER IN TRANSMIT ONLY MODE

```
uint8_t SPI1_Master_TransmitOnly (uint8_t data)
{
SPI1CON2bits.TXR = 1; //Transmit Data-Required Bit;
SPI1CON2bits.RXR = 0; // Receive FIFO Space-Required Bit;
SPI1TCNT = 1; // Load SPI Transfer Counter;
SPI1TXB = data; // Write Data to SPI Transmit Buffer;
}
```

4.3 Receive-Only Mode

This SPI module also provides the ability to configure the master to operate in Receive-Only mode. For the SPI to operate in this mode and begin receiving data from the slave, a non-zero value must be written to the transfer counter. In this mode, the master will continually receive data until the transfer counter is empty or the SPI is disabled. To configure the SPI module in this mode of operation, the RXR bit of the SPIxCON2 register must be set and the TXR bit of the SPIxCON2 register must be cleared. Regardless of the BMODE status, the transfer counter must have a non-zero value written to initiate the data transfer. The example below demonstrates the basic configuration of the SPI module in Receive-Only mode.

Example 4-3. CONFIGURATION OF A SPI MASTER IN RECEIVE ONLY MODE

```
uint8_t SPI1_Master_RecieveOnly (uint8_t data)
{
SPI1CON2bits.TXR = 0; //Transmit Data-Required Bit;
SPI1CON2bits.RXR = 1; // Receive FIFO Space-Required Bit;
SPI1TCNT = 1; // Load SPI Transfer Counter;
SPI1TXB = data; // Load data into SPI transmit buffer;
while (PIR2bits.SPI1RXIF == 0); // Check for any SPI Receive Interrupts;
return (SPI1RXB); // Return data from SPI Receive Buffer Register;
}
```

4.4 Transfer-Off Mode

In this mode, the SPI master clock signal (SCK) will not toggle and no data will be exchanged between the master and slave devices on the SPI bus. Writes will continue to be transferred to the TXFIFO, which will be transmitted if the TXR bit of SPIxCON2 is set.

Example 4-4. CONFIGURATION OF A SPI MASTER IN TRANSFER OFF MODE

```
void SPI1_Master_NoTransmit (void)
{
// Note: SCK will NOT toggle in this mode of operation;
SPI1CON2bits.TXR = 0; //Transmit Data-Required Bit;
SPI1CON2bits.RXR = 0; // Receive FIFO Space-Required Bit;
}
```

4.5 Clear Buffer/FIFOS

As previously mentioned, the transmission FIFO (TXFIFO) is written by software and the receive FIFO (RXFIFO) is written by the SPI module as data is shifted in from the slave. This SPI module allows the user to reset the occupancy for both FIFOs and clear the SPI buffers. To clear the SPI buffers, the CLRBF bit of the SPIxSTATUS register must be set high. Please note that the transmit and receive FIFOs are also reset when the SPI module is disabled. Clearing the buffers is an optional step, but can be useful in ensuring that the SPI does not have any existing data sitting in the buffer at the start of a data exchange.

5. Using the SPI with DMA

This SPI module has separate receive and transmit FIFO buffers that both allow for Direct Memory Access (DMA) bus connections. This can be useful in certain applications where data transfers need to be made without CPU intervention. The SPI is capable of triggering interrupts under several different conditions. The three top-level SPI interrupts, SPI Transmit, SPI Receive and SPI Module Status, can be found in the device's PIR registers

The DMA module can be used to implement fully interrupt driven operation with the SPI module. Instead of loading and clearing the TXFIFO and RXFIFO in software, the DMA can be configured to perform these tasks when certain triggers occur. The system arbitration decides memory access allocation, depending on priority and can be configured to give the DMA priority over the CPU. In the default case where the CPU has top priority, the DMA will wait for unused CPU cycles to perform DMA transfers. When the DMA is given top priority, the CPU will be stalled until the DMA has completed its transfers. The example below demonstrates how the system arbiter can be used to give the DMA top priority.

Example 5-1. ASSIGNING DMA1 HIGHEST PRIORITY USING THE SYSTEM ARBITER

```
// System Arbiter Configuration
ISRPR = 0x01; // Interrupt Service Routine Priority;
MAINPR = 0x02; // Main Routine Priority;
DMA1PR = 0x00; // DMA1 Priority;
DMA2PR = 0x03; // DMA2 Priority;
SCANPR = 0x04; // Scanner Priority;
asm (”BCF INTCON0, 7”); // disable Global Interrupts
asm (”BANKSEL PRLOCK”);
asm (”MOVLW 0x55”);
asm (”MOVWF PRLOCK”); // Arbiter Priority lock
asm (”MOVLW 0XAA”);
asm (”MOVWF PRLOCK”);
asm (”BSF PRLOCK, 0”);
asm (”BSF INTCON0, 7”); // enable Global Interrupts
```

5.1 SPI Receiver Data Interrupt

The SPI receiver data interrupt is set when the receive FIFO contains data, and cleared when the FIFO is empty. For DMA-based operation the user does not need to enable the receive interrupt required for another operation. The status of the SPI receiver data interrupt flag can be monitored by reading the SPIxRXIF bit located in the device peripheral interrupt registers.

The DMA module can be configured to shift data from the receive FIFO to another memory region on the PIC microcontroller. This allows the SPI to keep receiving data without software involvement to keep the receive FIFO from filling up. The example below shows how to configure the DMA to take data from the RXFIFO and store it in Flash memory as it is being received.

Example 5-2. USING DMA TO CLEAR SPI RECEIVE BUFFER

```
// DMA Configuration (Source SPI1RXB)
DMA1CON0 = 0xC0; // DMA Enable; SIRQEN;
DMA1CON1 = 0x0B; // DMODE; SMR; SMODE; SSTP;
DMA1SSA = &SPI1RXB; // DMA Source Start Address - SPI1RXB;
DMA1SSZ = 0x01; // DMA Source Size ();
DMA1DSA = 0x1000; // DMA Destination Start Address ();
DMA1DSZ = 0x32; // DMA Destination Size ();
DMA1SIRQ = 0x14; // DMA Start Interrupt Request Source (SPI1RXB:20);
```

5.2 SPI Transmitter Data Interrupt

The SPI Transmitter data interrupt is set when the transmit FIFO is not full, and is cleared when the transmit FIFO is full. For DMA-based operation the user does not need to enable the transmit interrupt required for another operation. The status of the SPI transmitter data interrupt flag can be monitored by reading the SPI transmit interrupt flag.

The DMA module can also be configured to take data from another memory region on the PIC microcontroller and move that to the TXFIFO to be transmitted by the SPI. This allows the SPI to continually transmit data without software involvement. Instead of needing to write to the transmit buffer every time a message needs to be sent, the DMA can move data into the transmit buffer as it empties out. The example below demonstrates how to configure the DMA to move data from Flash memory into the SPI transmit FIFO.

Example 5-3. USING DMA TO LOAD SPI TRANSMIT BUFFER

```
// DMA Configuration (Destination SPI1TXB)
DMA1CON0 = 0xC0; // DMA Enable; SIRQEN;
DMA1CON1 = 0x0B; // DMODE; SMR; SMODE; SSTP;
DMA1SSA = 0x1000; // DMA Source Start Address - PFM;
DMA1SSZ = 0x32; // DMA Source Size (50);
DMA1DSA = &SPI1TXB; // DMA Destination Start Address (SPI1TXB);
DMA1DSZ = 0x01; // DMA Destination Size (1);
DMA1SIRQ = 0x15; // DMA Start Interrupt Request Source (SPI1TXB:21);
```

6. Slave Mode

Table 6-1. Slave Mode TXR/RXR Settings

Setting	Description
TXR = 1	Data from the TX buffer is transmitted and the write FIFO pointer is incremented. If the TX buffer is empty, the most recently received data is transmitted and the Transmit Underflow Interrupt Flag (TXUIF) bit is set to indicate that this type of error occurred.
TXR = 0	Data in the TX buffer is transmitted if available, but the write FIFO pointer is not incremented. If the TX buffer is empty, the most recently received data is transmitted, but the TXUIF bit will not be set.
RXR = 1	Data will be stored in the RX buffer if it is not full and the read FIFO pointer is incremented. If data is received and the RX buffer is full, the Receive Overflow Interrupt Flag (RXOIF) bit is set to indicate the error and the data received is discarded.
RXR = 0	All received data will be ignored and not stored in the RX register.

The TXR and RXR bits control how data is transferred when a device is configured as a slave on the SPI bus. Table 6-1 summarizes how the configuration of the RXR and TXR bits affect the operation of the SPI module while in Slave mode. For more information about how data is transmitted/received while in Slave mode, refer to the device data sheet.

In the event where the Slave Select ($\overline{SS}$) line transitions to an inactive state while a data transfer is ongoing, the Slave Select Fault (SSFLT) bit in the SPIxCON2 register will be set. The SSP bit of the SPIxCON1 register controls slave select polarity. When the SSP bit is set, the Slave Select ($\overline{SS}$) line is active-low. Conversely, when the SSP bit is cleared, the Slave Select ($\overline{SS}$) line is active-high. In addition to this, the SCK pin must always be an input and configured to the same clock polarity and edge as the master device. Clock polarity is controlled by the CKP bit and the clock edge is set by the CKE bit both found in the SPIxCON1 register.

7. Conclusion

This technical brief gives a brief overview of the new Serial Peripheral Interface (SPI) module, described and walked through basic modes of operation, and briefly discussed other key features. The code examples included in this document serve as a walk through on getting started in building an application that uses this SPI module. For more information or specifications pertaining to this module, refer to the device datasheet.

8. Revision A (4/2018)

Initial release of this document.

The Microchip Web Site

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

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

Customer Change Notification Service

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

To register, access the Microchip web site at http://www.microchip.com/. Under "Support", click on "Customer Change Notification" and follow the registration instructions.

Customer Support

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support

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

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

Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.

- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as “unbreakable.”

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip’s code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Legal Notice

Trademarks

ISBN: 978-1-5224-2965-4

Quality Management System Certified by DNV

ISO/TS 16949

Microchip received ISO/TS-16949:2009 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.

Worldwide Sales and Service

AMERICAS	ASIA/PACIFIC	ASIA/PACIFIC	EUROPE
Corporate Office 2355 West Chandler Blvd. Chandler, AZ 85224-6199 Tel: 480-792-7200 Fax: 480-792-7277 Technical Support: http://www.microchip.com/support Web Address: www.microchip.com **Atlanta** Duluth, GA Tel: 678-957-9614 Fax: 678-957-1455 **Austin, TX** Tel: 512-257-3370 **Boston** Westborough, MA Tel: 774-760-0087 Fax: 774-760-0088 **Chicago** Itasca, IL Tel: 630-285-0071 Fax: 630-285-0075 **Dallas** Addison, TX Tel: 972-818-7423 Fax: 972-818-2924 **Detroit** Novi, MI Tel: 248-848-4000 **Houston, TX** Tel: 281-894-5983 **Indianapolis** Noblesville, IN Tel: 317-773-8323 Fax: 317-773-5453 Tel: 317-536-2380 **Los Angeles** Mission Viejo, CA Tel: 949-462-9523 Fax: 949-462-9608 Tel: 951-273-7800 **Raleigh, NC** Tel: 919-844-7510 **New York, NY** Tel: 631-435-6000 **San Jose, CA** Tel: 408-735-9110 Tel: 408-436-4270 **Canada - Toronto** Tel: 905-695-1980 Fax: 905-695-2078	**Australia - Sydney** Tel: 61-2-9868-6733 **China - Beijing** Tel: 86-10-8569-7000 **China - Chengdu** Tel: 86-28-8665-5511 **China - Chongqing** Tel: 86-23-8980-9588 **China - Dongguan** Tel: 86-769-8702-9880 **China - Guangzhou** Tel: 86-20-8755-8029 **China - Hangzhou** Tel: 86-571-8792-8115 **China - Hong Kong SAR** Tel: 852-2943-5100 **China - Nanjing** Tel: 86-25-8473-2460 **China - Qingdao** Tel: 86-532-8502-7355 **China - Shanghai** Tel: 86-21-3326-8000 **China - Shenyang** Tel: 86-24-2334-2829 **China - Shenzhen** Tel: 86-755-8864-2200 **China - Suzhou** Tel: 86-186-6233-1526 **China - Wuhan** Tel: 86-27-5980-5300 **China - Xian** Tel: 86-29-8833-7252 **China - Xiamen** Tel: 86-592-2388138 **China - Zhuhai** Tel: 86-756-3210040	**India - Bangalore** Tel: 91-80-3090-4444 **India - New Delhi** Tel: 91-11-4160-8631 **India - Pune** Tel: 91-20-4121-0141 **Japan - Osaka** Tel: 81-6-6152-7160 **Japan - Tokyo** Tel: 81-3-6880- 3770 **Korea - Daegu** Tel: 82-53-744-4301 **Korea - Seoul** Tel: 82-2-554-7200 **Malaysia - Kuala Lumpur** Tel: 60-3-7651-7906 **Malaysia - Penang** Tel: 60-4-227-8870 **Philippines - Manila** Tel: 63-2-634-9065 **Singapore** Tel: 65-6334-8870 **Taiwan - Hsin Chu** Tel: 886-3-577-8366 **Taiwan - Kaohsiung** Tel: 886-7-213-7830 **Taiwan - Taipei** Tel: 886-2-2508-8600 **Thailand - Bangkok** Tel: 66-2-694-1351 **Vietnam - Ho Chi Minh** Tel: 84-28-5448-2100	**Austria - Wels** Tel: 43-7242-2244-39 Fax: 43-7242-2244-393 **Denmark - Copenhagen** Tel: 45-4450-2828 Fax: 45-4485-2829 **Finland - Espoo** Tel: 358-9-4520-820 **France - Paris** Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79 **Germany - Garching** Tel: 49-8931-9700 **Germany - Haan** Tel: 49-2129-3766400 **Germany - Heilbronn** Tel: 49-7131-67-3636 **Germany - Karlsruhe** Tel: 49-721-625370 **Germany - Munich** Tel: 49-89-627-144-0 Fax: 49-89-627-144-44 **Germany - Rosenheim** Tel: 49-8031-354-560 **Israel - Ra'anana** Tel: 972-9-744-7705 **Italy - Milan** Tel: 39-0331-742611 Fax: 39-0331-466781 **Italy - Padova** Tel: 39-049-7625286 **Netherlands - Drunen** Tel: 31-416-690399 Fax: 31-416-690340 **Norway - Trondheim** Tel: 47-7289-7561 **Poland - Warsaw** Tel: 48-22-3325737 **Romania - Bucharest** Tel: 40-21-407-87-50 **Spain - Madrid** Tel: 34-91-708-08-90 Fax: 34-91-708-08-91 **Sweden - Gothenberg** Tel: 46-31-704-60-40 **Sweden - Stockholm** Tel: 46-8-5090-4654 **UK - Wokingham** Tel: 44-118-921-5800 Fax: 44-118-921-5820