**INTRODUCTION**

There are many different microcontrollers on the market today that are being used in embedded control applications. Many of these embedded control systems need nonvolatile memory. Because of their small footprint, byte level flexibility, low I/O pin requirement, low power consumption, and low cost, serial EEPROMs are a popular choice for nonvolatile storage.

Microchip Technology has addressed these needs by offering a full line of serial EEPROMs covering industry standard serial communication protocol for two-wire (I$^2$C™), three-wire (Microwire), and SPI communication.

Serial EEPROM devices are available in a variety of densities, operational voltage ranges and packaging options.

This application note provides assistance and source code to ease the design process of interfacing a Microchip mid-range PIC18F4520 microcontroller to a Microchip Microwire serial EEPROM. The Master Synchronous Serial Port (MSSP) provides a simple three-wire connection to the EEPROM and no external “glue” logic is required.

Figure 1 depicts the hardware schematic for the interface between Microchip’s Microwire devices and the Microchip PIC18F4520 Microcontroller. The schematic shows the necessary connections to interface the microcontroller and the serial EEPROM (software was written assuming these connections).

**FIGURE 1: CIRCUIT FOR PIC18F4520 AND 93XXXXX (MICROWIRE) DEVICE**

Note 1: CS should always have a pull-down resistor to protect against data corruption during power-up or power-down of the Microcontroller.
FIRMWARE DESCRIPTION

The purpose of the program is to show individual features of the Microwire protocol and give code samples of the Start bit, opcodes and addressing schemes so that the basic building blocks of a program can be shown. The waveforms provided will be shown from CS active to CS disable so an entire instruction can be seen. To ease the interpretation of the serial data, the data sheet waveform will be provided below the oscilloscope screen shot. A graphic similar to that of Figure 2 will be shown with the values being programmed by the firmware to also assist in ease of reading.

THEORY OF OPERATION

To use an SPI port (MSSP) to communicate with Microchip's Microwire Serial EEPROMs, the bytes to be output to the 93XXXX must be aligned such that the LSB of the address is the 8th bit (LSB) of a byte to be output. From there the bits should fill the byte from right to left consecutively. If more that 8 bits are required, then two bytes will be required to be output. This same method will work for any 93XXXX series device but the data sheet must be referenced for these because density and organization will change the number of bits sent for each command. Since more than 8 bits are required to control a 93LC66C, two consecutive bytes are required.

FIGURE 2: COMMAND ALIGNMENT

<table>
<thead>
<tr>
<th>High Byte</th>
<th>Low Byte</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0 0 0 0</td>
<td>SB OP1 OP2 A7 A6 A5 A4 A3 A2 A1 A0</td>
</tr>
</tbody>
</table>

Leading 0's here must be 0's, otherwise the device will see a Start bit with an invalid command following.

High Byte (Where the Start bit, opcode bits, and address MSb reside)

The High Byte is configured in the following format: SB is the Start bit. OP1 is the MSb of the opcode and OP0 is the opcode LSB. The CS line can be set before the byte is output because the leading 0’s output to the 93XXXX prevent a Start bit from being recognized by the 93XXXX until the first high bit is sent.

Low Byte (8 Address bits)

The Low Byte contains A7-A0, which are the address bits required to access 256 bytes in x16 mode.
INITIALIZATION

In order to configure the MSSP module to work for the Microwire protocol, several key registers on the PICmicro® Microcontroller need to be properly initialized. Code examples are shown for each. Since the Microwire protocol is not native to the MSSP module, a version of SPI mode 0,0 has been implemented and works within the data sheet specifications for Microwire.

MSSP STATUS Register (SSPSTAT)

SSPSTAT holds all of the Status bits associated with the MSSP module. For Microwire, the SMO bit of the register needs to be set for data to be sampled at the end of the data output time. The CKE bit also needs to be set so that data is transmitted on the rising edge of SCK when CKP (SSPCON1) is cleared.

EXAMPLE 1: SSPSTAT CONFIGURATION

```c
SSPSTAT = 0xC0;  //SPI Bus mode 0,0
```

SSP Control Register 1 (SSPCON1)

SSPCON1 is another register for the MSSP module. For Microwire communication, the upper two bits of the SSPCON1 are indicator bits and should be cleared initially. The SSP Enable bit (SSPEN) needs to be set in order to enable the SSP module and the Clock Polarity Select bit needs to be cleared to set the IDLE state of the clock to be a low level. The lower four bits of the SSPCON1 set the mode and speed of communications, in this case we are setting this to Master mode and Fosc/16.

EXAMPLE 2: SSPCON1 CONFIGURATION

```c
SSPCON1 = 0x21;  //Enable SSP,Fosc/16
```

TRISC Register

In order to be properly controlled by the MSSP module, the CLK, DI and DO pins must be configured properly. We have also chosen RC7 for Chip Select (CS) control so it must be configured as an output. This is done by setting their respective bits in TRISC to ‘1’ for inputs and ‘0’ for outputs, as shown in Example 3.

EXAMPLE 3: TRISC CONFIGURATION

```c
DDRCbits.RC7 = 0;  //CS as Output
DDRCbits.RC3 = 0;  //CLK as Output
DDRCbits.RC4 = 1;  //DI as Input
DDRCbits.RC5 = 0;  //DO as Output
```
WRITE ENABLE

Figure 3 shows an example of the Erase/Write Enable (EWEN) command. This command consists of a Start bit and the four bit opcode (0000). Because this command doesn't require addressing, the two high order address bits (A7 and A6) are used for opcode. The remaining address bits (set to zeros in this example) are “don’t cares”.

Chip Select is brought high (active), the Start bit and opcode are sent out through the MSSP port. The EWEN command must be given before a write is attempted. The device will be enabled for writes until a Erase/Write Disable command is given or the device is powered down.

**FIGURE 3: ERASE/WRITE ENABLE (EWEN)**

![Diagram showing the EWEN command and its timing](image)

<table>
<thead>
<tr>
<th>High Byte</th>
<th>Low Byte</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0 0 0 0 0</td>
<td>hibyte = 0x04h</td>
</tr>
<tr>
<td>0 0 0 0 1 0 0 0</td>
<td>lobyte = 0xC0h</td>
</tr>
</tbody>
</table>
WRITE COMMAND (START BIT, OPCODE, ADDRESS AND DATA)

Figure 4 shows an example of the Write command. The device is selected and the high byte is sent out which contains the Start bit, and opcode. The second low byte is sent which contains the address bits A7-A0.

Finally, the data is clocked in, in this case, 0xA55A. When the Chip Select is toggled at the end of this, the internal write cycle is initiated. Once the internal write cycle has begun the READY/Busy signal can be polled on the DO pin to check when the write finishes. A 6 ms delay needs to be added if the READY/Busy status is not being polled. This code uses READY/Busy polling.

FIGURE 4: WRITE COMMAND, ADDRESS AND DATA

<table>
<thead>
<tr>
<th>High Byte</th>
<th>Low Byte</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0 0 0 0</td>
<td>SB OP1 OP0 A7 A6 A5 A4 A3 A2 A1 A0</td>
</tr>
<tr>
<td>hibyte = 0x05h</td>
<td>lobyte = 0x10h</td>
</tr>
</tbody>
</table>
READY/BUSY POLLING

After a valid Write command is given, the DO line of the 93XXXX can be monitored to check if the internal write cycle has been initiated and it can continuously be monitored to look for the end of the write cycle. The oscilloscope plot below shows that the device is selected and the DO line is low for approximately 3.2 ms before the device brings the DO line high indicating that the write cycle is complete.

FIGURE 5: READY/BUSY POLLING
READ COMMAND (START BIT, OPCODE, ADDRESS AND DATA)

Figure 6 shows an example of the Read command. The device is selected and the high byte is sent out which contains the Start bit and opcode.

The second low byte is sent which contains the address bits A7-A0 (0x10). At this point the device gets ready to send data out, the controller needs to send a dummy byte in order for the clock signals to be sent so the data can be read out of the device and into the microcontroller. In this case, data being read is 0xA55A.

FIGURE 6: READ COMMAND

<table>
<thead>
<tr>
<th>High Byte</th>
<th>Low Byte</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0 0 0</td>
<td>SB OP1 OP0</td>
</tr>
<tr>
<td>0 0 0 0 0</td>
<td>A7 A6 A5 A4 A3 A2 A1 A0</td>
</tr>
<tr>
<td>hibyte = 0x06h</td>
<td>lobyte = 0x10h</td>
</tr>
</tbody>
</table>
ERASE/WRITE DISABLE COMMAND

Once the device write is finished, the Write Disable (EWDS) command should be given (see Figure 7). This command consists of a Start bit and the four bit opcode (0000). Because this command doesn't require addressing, the two high order address bits (A7 and A6) are used for opcode. The remaining address bits (set to zeros in this example) are “don’t cares”.

The EWDS command should always be sent to the device after completing a write or prior to powering down the device/system.

FIGURE 7: ERASE/WRITE DISABLE COMMAND
CONCLUSION

These are some of the basic features of Microwire communications using the MSSP module on the PIC18F4520. The code is highly portable and can be used on many devices that have the MSSP module with very minor modifications. Using the code provided, designers can begin to build their own Microwire libraries to be as simple or as complex as needed.

The code was tested on Microchip’s PICDEM™ 2 Plus Demonstration Board with a 4 MHz crystal and the connections shown in Figure 1.
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.

Information contained in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION, INCLUDING BUT NOT LIMITED TO ITS CONDITION, QUALITY, PERFORMANCE, MERCHANTABILITY OR FITNESS FOR PURPOSE. Microchip disclaims all liability arising from this information and its use. Use of Microchip’s products as critical components in life support systems is not authorized except with express written approval by Microchip. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights.

Trademarks

The Microchip name and logo, the Microchip logo, Accuron, dsPIC, KEELOC, microID, MPLAB, PIC, PICmicro, PICSTART, PRO MATE, PowerSmart, rFIC, and SmartShunt are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

AmpLab, FilterLab, Migratable Memory, MXDEV, MXLAB, PICMASTER, SEEVAL, SmartSensor and The Embedded Control Solutions Company are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Analog-for-the-Digital Age, Application Maestro, dsPICDEM, dsPICDEM.net, dsPICworks, ECAN, ECONOMONITOR, FanSense, FlexROM, fuzzyLAB, In-Circuit Serial Programming, ICSP, ICEPIC, Linear Active Thermostor, MPASM, MPLIB, MPLINK, MPSIM, PICkit, PICDEM, PICDEM.net, PICLAB, PICtail, PowerCal, PowerInfo, PowerMate, PowerTool, Real ICE, rLAb, rFICDEM, Select Mode, Smart Serial, SmartTel, Total Endurance, UNI/O, WiperLock and Zena are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

SQTP is a service mark of Microchip Technology Incorporated in the U.S.A.

All other trademarks mentioned herein are property of their respective companies.

© 2006, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

Printed on recycled paper.

QUALITY MANAGEMENT SYSTEM CERTIFIED BY DNV

ISO/TS 16949:2002

Microchip received ISO/TS-16949:2002 quality system certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona and Mountain View, California in October 2003. The Company’s quality system processes and procedures are for its PICmicro® 8-bit MCUs, 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.