Application Note: EAS-AN001 (Jan '98)

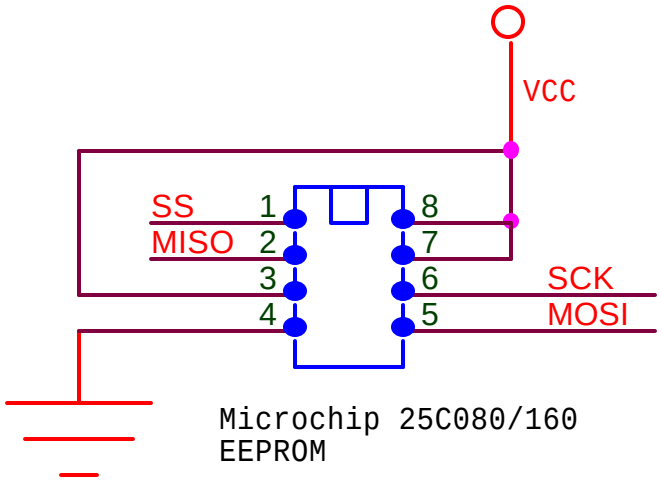
Using SPI EEPROM memory

By Jim Fong

EEPROM (electrically erasable programmable read-only memory) are memory devices that retain their data contents when power is off. These devices can be read and written to like standard RAM. They can be useful for data logging applications where storage of data is critical. Microchip Technology makes a series of very inexpensive EEPROM devices that are SPI compatible. You can purchase them from DigiKey for around $3 each. Microchip 25C080 is a 1K x 8 device and the 25C160 is a 2K x 8 device. They also make a 4K and 8K size EEPROMs but we will focus on the 25C080/160 versions.

EEPROMs will eventually wear out after repeated writes but these have an endurance of 10million cycles. It will take a very long time before you will need to replace one of these chips.

Hooking up the 25C080/160 to the SPI port is pretty simple. The following schematic diagram shows which pins to connect to the SPI port. Pin 1 of the 25C080/160 is the chip select and we will be using the SPI I/O pin SS to control access to the device. You can use any available I/O pin as the chip select. For more technical details of the chip, download the PDF datasheet from www.microchip.com.



The following Interactive C routines are procedures to read and write to the EEPROM. Before using the EEPROM routines, first initialize the SPI port with the init_spi() procedure. This will setup the SPI for 1MHz sclock and also set SS as an output pin.

This example program will initialize the SPI port and store the value 128 to memory address location 1000 on the EEPROM. It will then print out the contents of memory address location 150 to the LCD screen. The maximum value that can be stored to a memory location is 255.

```
void main()
{
init_spi();
eeprom_write(1000, 128);
printf("%d\n", eeprom_read(150));
}
```

Interactive C routines to access 25Cxxx EEPROMs by Microchip Technology.

```
/* C subprograms to access a Microchip 25Cxxx EEPROM via the SPI port.
    Interactive C Version 3.1

   January 23, 1998

   Hook up diagram, 25Cxxx EEPROM to Finger Board SPI port
   25Cxxx           Finger Board SPI port JP4
   Pin 1            Vcc +5V
   Pin 2            MISO
   Pin 3            Vcc
   Pin 4            Vss ground
   Pin 5            MOSI
   Pin 6            SCK
   Pin 7            Vcc
   Pin 8            Vcc

Embedded Acquisition Systems
Web: http://www.embeddedtronics.com
Email: jimf@embeddedtronics.com

*/

void init_spi()
/* init_spi will initialize the SPI port for 1MHz SCK rate and set PD5 (SS) as
an output bit.*/
{
/* set PORTD ($1008) SS* high, SCK-Lo, MOSI-Hi when DDRD bit 5 is set
(xx10111)*/
poke(0x1008,0x2f);

/* set DDRD ($1009) register  (xx1110xx) */
poke(0x1009,0x38);

/* set SPCR ($1028) register (01010000)
SPIE = 0 , no interrupt
SPE = 1 , set SPI on
DWOM = 0, port D outputs are push-pull
MSTR = 1 , SPI is configured as master
CPOL = 0 , active high clocks selected
CPHA = 0, select phase
SPR1, SPR1 = 0 , select E/2 clock
*/
poke(0x1028,0x50);
}

void  out_spi(int outchar)
/* out _spi will write an ASCII character to the SPI port.*/
{
       int test=0;
       poke(0x102A, outchar);
       while (test == 0)                      /*loop until data is sent*/
       {
              test = peek(0x1029);
              test = test & 0x80;
       }
}
```

```
void eeprom_write_enable()
/*This routine will set the write enable latch of the 25Cxxx eeprom*/
{
bit_clear(0x1008, 0b00100000);          /*set PD5 CS* low*/
out_spi(0x6);                           /*send out write enable instruction*/
bit_set(0x1008, 0b00100000);            /*set PD5 CS* hi*/
}

int eeprom_read(int address)
/*This routine will read a eeprom memory cell at location address

25C80 EEPROM, address is 0-1027
25C160 EEPROM, address is 0-2047
*/
{
int HiByte, LoByte,mem_data;

HiByte=address>>8;                      /*mask off lower 8 bits*/
LoByte=address & 0b0000000011111111;    /*mask off 8 higher bits*/


bit_clear(0x1008, 0b00100000);          /*set PD5 CS* low*/
out_spi(0x3);                           /*send out read instruction*/
out_spi(HiByte);                        /*send out first 8 bits of address*/
out_spi(LoByte);                        /*send out last 4 bits of address*/
out_spi(0x0);                           /*clock out the memory data*/
mem_data=peek(0x102A);                  /*read in memory data*/
bit_set(0x1008, 0b00100000);            /*set PD5 CS* hi*/

return mem_data;
}

void eeprom_write(int address, int mem_data)
/*This routine will write mem_data to location address

25C80 EEPROM, address is 0-1027
25C160 EEPROM, address is 0-2047
*/
{
int HiByte, LoByte;

HiByte=address>>8;                      /*mask off lower 8 bits*/
LoByte=address & 0b0000000011111111;    /*mask off 8 higher bits*/

eeprom_write_enable();                  /*set eeprom so can write to it*/

bit_clear(0x1008, 0b00100000);          /*set PD5 CS* low*/
out_spi(0x2);                           /*send out write instruction*/
out_spi(HiByte);                        /*send out first 8 bits of address*/
out_spi(LoByte);                        /*send out last 4 bits of address*/
out_spi(mem_data);                      /*clock out the memory data*/
bit_set(0x1008, 0b00100000);            /*set PD5 CS* hi*/
}
```