Application Note: EAS-AN002 (January 30,1998)

MiniDaq to Finger Board SPI interface

By Jim Fong

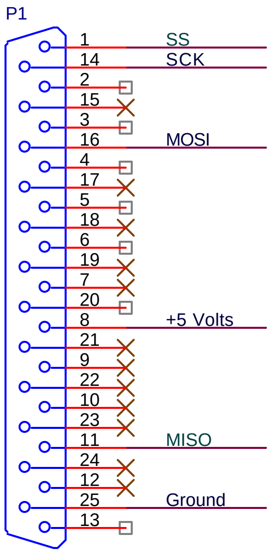
The 68HC11 processor has a built in 8 bit, 8 channel analog to digital (A/D) converter that can be used to read analog voltages from a variety of different sensors. Some applications need to measure voltages with higher degree of resolution than the 256 steps that an 8bit A/D gives us. By using a 12bit A/D, analog signals can be acquired with a resolution of 4096 steps.

This application note will focus on how to hookup our MiniDaq A/D module to the Finger Board SPI port. The Minidaq is a 12bit, 8 channel analog to digital (A/D) data acquisition module. It is based on the Maxim MAX186 A/D integrated circuit. The MAX186 has a SPI compatible 4 wire interface that makes it easy to hook up to any micro-controller that has a built in SPI interface. The MiniDaq was originally designed to hook up to the parallel port of an IBM PC type computer. Since the SPI interface I/O pins of the MAX186 are available on the MiniDaq RS232 connector we can easily connect it to the Finger Board also.

Note: When the MiniDaq is connected to the IBM PC parallel port, 7 digital inputs and 4 digital outputs are available. We are only interfacing the A/D portion of the MiniDaq to the Finger Board so none the extra I/O ports will be available.

The following is the hook up diagram for the MiniDaq to Finger Board SPI interface. Header JP4 on the Finger Board is the SPI connector. You can purchase a female solder cup type DB25 connector and solder a short length of ribbon cable to connect it to the Finger Board JP4 header. On the other end, use an 8 pin IDC type connector.

MiniDaq P1 Male DB25	Finger Board SPI JP4 header
Pin 1	Pin 3 SS*
Pin 8	Pin 2 +5volt
Pin 11	Pin 4 MISO
Pin 14	Pin 5 SCK
Pin 16	Pin 6 MOSI
Pin 25	Pin 1 Ground



MiniDaq DB25 (male) connector

The following Interactive C routines are procedures to acquire data from the A/D. Before using the A/D routines, first initialize the SPI port with the init_spi() procedure. This will setup the SPI for 1MHz Sclock and also setup SS as an output pin, which is used as the MAX186 chip select. The function bpse12() will read in a bipolar single ended A/D data value and unise12() will read in a unipolar single ended A/D data value.

The following example program will initialize the SPI port and print a bipolar A/D value from channel 1 to the LCD.

```
void main()
{
init_spi();
printf("%d\n", bpse12(1));
}
```

Since other processors have SPI interfaces, you are not limited on only using the MiniDaq with the Finger Board. The C subroutines are commented and can be modified for other different C compilers and processor combinations.

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

Interactive C routines to access the MiniDaq A/D.

```
/* C subprograms to access EAS MiniDaq data acquisition board .
   Finger Board SPI to MiniDaq transfer
   Interactive C Version 3.1

   January 30, 1998

   Hook up diagram, MiniDaq to Finger Board
   MiniDaq P1 Male DB25       Finger Board SPI port JP4
   Pin 1                      Pin 3  SS*
   Pin 8                      Pin 2  +5volt
   Pin 11                     Pin 4  MISO
   Pin 14                     Pin 5 SCK
   Pin 16                     Pin 6 MOSI
   Pin 25                     Pin 1 Ground

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;
      }
}

int bpse12(int channel)
/* bpse12 Version 1.0 by EMBEDDED ACQUISITION SYSTEMS

   Finger Board to MiniDaq A/D routine

   Function will start a bipolar single ended conversion process
     and return a integer value.

   The returned value is between -2048 and +2047
      +2047 = Vref/2 - 1LSB
      -2048 = -Vref/2

   -2.048 to 2.047 Volts maximum input

   channel is a input integer from 0 to 7
*/
{  /*begin function bpse12*/

   int HiByte, LoByte, data_out, din;

   data_out=0;

 if  (channel==0)             /*Set din to correct A/D channel*/
      din=0x86;               /*10000110*/
else if (channel==1)
      din=0xC6;               /*11000110*/
else if (channel==2)
      din=0x96;               /*10010110*/
else if (channel==3)
      din=0xD6;               /*11010110*/
else if (channel==4)
      din=0xA6;               /*10100110*/
else if (channel==5)
      din=0xE6;               /*11100110*/
else if (channel==6)
      din=0xB6;               /*10110110*/
else if (channel==7)
      din=0xF6;               /*11110110*/

/* MSB sent first
   This din word config the MAX186 for CH# respect
   to COM, BIPOLAR, CLOCK
   data_out is decimal rep of MAX186 DOUT
*/
```

```
bit_clear(0x1008, 0b00100000);      /*set PD5 CS* low*/
out_spi(din);                       /*send out Control-Byte*/
HiByte = peek(0x102A);              /*read in 8bits and discard*/
out_spi( 0x0);                      /*send out Control-Byte of 0*/
HiByte = peek(0x102A);              /*read in first 8 bits*/
out_spi(0x0);                       /*send out Control-Byte of 0*/
LoByte = peek(0x102A);              /*read in second 8 bits*/
bit_set(0x1008, 0b00100000);        /*set PD5 CS* hi*/

HiByte=HiByte<<1;                   /*shift left HiByte  1*/
LoByte=LoByte>>3;                   /*shift right LoByte 3*/
data_out = ((HiByte*16)+LoByte);    /*calculate data_out*/

if (data_out >= 2048) {             /*scale for +2048 to -2048*/
    data_out=data_out-4096; }

  return (data_out);                /*return value to main*/

} /*end function bpse12*/

int unise12(int channel)
/* unise12 Version 1.0 by EMBEDDED ACQUISITION SYSTEMS

   Finger Board to MiniDaq A/D routine

   Function will start a unipolar single ended conversion process
     and return a integer value

   The returned value is between 0 and +4095
      +4095 = Vref - 1LSB
       0 = 0 volts

   0 to +4.095 Volts maximum input

   channel is a input integer from 0 to 7
*/
{  /*begin function unise12*/

   int HiByte, LoByte, data_out, din;

   data_out=0;

 if  (channel==0)             /*Set din to correct A/D channel*/
      din=0x8E;               /*10001110*/
else if (channel==1)
      din=0xCE;               /*11001110*/
else if (channel==2)
      din=0x9E;               /*10011110*/
else if (channel==3)
      din=0xDE;               /*11011110*/
else if (channel==4)
      din=0xAE;               /*10101110*/
else if (channel==5)
      din=0xEE;               /*11101110*/
else if (channel==6)
      din=0xBE;               /*10111110*/
```

```
else if (channel==7)
      din=0xFE;               /*11111110*/

/* MSB sent first
   This din word config the MAX186 for CH# respect
   to COM, UNIPOLAR, CLOCK
   data_out is decimal rep of MAX186 DOUT
*/

bit_clear(0x1008, 0b00100000);      /*set PD5 CS* low*/
out_spi(din);                       /*send out Control-Byte*/
HiByte = peek(0x102A);              /*read in 8bits and discard*/
out_spi( 0x0);                      /*send out Control-Byte of 0*/
HiByte = peek(0x102A);              /*read in first 8 bits*/
out_spi(0x0);                       /*send out Control-Byte of 0*/
LoByte = peek(0x102A);              /*read in second 8 bits*/
bit_set(0x1008, 0b00100000);        /*set PD5 CS* hi*/

HiByte=HiByte<<1;                   /*shift left HiByte  1*/
LoByte=LoByte>>3;                   /*shift right LoByte 3*/
data_out = ((HiByte*16)+LoByte);    /*calculate data_out*/

  return (data_out);                /*return value to main*/

} /*end function unise12*/
```