```
/*-------------------------------------------------------------------------------+
|                                                                               |
|          Interfacing the DS1620 for AT89C52 that via the comport control      |
|                                                                               |
|                                                                               |
| Name: DemoMain.c                                                              |
| Purpose:                                                                      |
|    Interfacing the DS1620 via the comport, measure temperature form -55 to +125   |
|    This note introduces the software for interfacing a 8052 microcontroller to the|
|    DS1620 digital temperature sensor. The DS1620 communicates via a 3-wire serial |
|    digital interface. Software code is provide that reads the DS1620 temperaure   |
|    register and calculates high-resolution results.                           |
|                                                                               |
| By Dillian Wong, last modify data 4/10/03                                     |
+-------------------------------------------------------------------------------*/

/*_____________ I N C L U D E S _____________________________________________*/

#include <reg51.h>
#include <stdio.h>
#include "DS1620\DS1620.h"
#include "rPrintf\rprintf.h"


/*_____________ M A I N  P R O G R A M _____________________________________*/

char code menu[] = // menu of programmer
    "\n--Digital Thermometer-2-"
    "\ns set temperature "
    "\nr read temperature"
    "\nh read TH"
    "\nl read TL"
    ;

// com port with 9600 baud with crystal 11.0592MHz.
void init_uart(void)
{
    SCON  = 0x50;
    TMOD |= 0x20;
    TH1   = 253;
    TR1   = 1;
    TI    = 1;
}

void main(void)
{
    int i;
    char cmd;       // selection
    float fh;       // high temperature limit
    float fl;       // low temperature limit

    init_uart();    // utilize rs232 port

    printf(menu);
    while(1){
        printf( "\n[Enter selection:] " );
        scanf("%c", &cmd);
        switch(cmd)
```

```
        {
            case 's':
                        // enter higher limit.
                        printf("\nEnter Higher Temperature limit  :");
                        scanf("%f",&fh);
                        while(fh>125)
                        {
                            printf("\nhigh temperature limited to +125! ");
                            printf("\nEnter Higher Temperature limit again :");
                            scanf("%f",&fh);
                        }
                        // enter lower limit.
                        printf("Enter Lower Temperture limit  :");
                        scanf("%f",&fl);
                        while(fl<-55)
                        {
                            printf("\nLower temperature limited to -55");
                            printf("\nEnter Lower Temperture limit again :");
                            scanf("%f",&fl);
                        }
                        // start setting.
                        printf("\nstart program Ds1620\n");
                        ds1620_init(fh,fl);
                        printf("\nSet Temperature complete!");

                        break;
            case 'r':
                        for(i=0;i<200;i++);
                        printf("\nCurrent temperature is: ");
                        printf("%f",ds1620_ReadTemperature());
                        break;
            case 'h':
                        for(i=0;i<200;i++);
                        printf("\nHigh temperature limit: %f",ds1620_ReadTH());
                        break;
            case 'l':
                        for(i=0;i<200;i++);
                        printf("\nLow temperature limit: %f",ds1620_ReadTL());
                        break;
            case '?':
                        printf(menu);
                        break;
            default:
                        printf("\nInvilad key!");
                        break;
        }


    }

}
```

```
/*-------------------------------------------------------------------------------+
|                                                                                |
|                    Ditital Thermometer and Thermostat (DS1620)                 |
|                                                                                |
| Name: DS1620.c                                                                 |
| Purpose:                                                                       |
|                                                                                |
| By Dillian Wong, last modify data 4/10/03                                      |
+-------------------------------------------------------------------------------*/

/*_____________ I N C L U D E S _____________________________________________*/

#include <reg51.h>
#include <stdio.h>
#include "rPrintf\rprintf.h"

/*_____________ M A C R O S _____________________________________________*/

#define READ_TEMPERATURE    0xAA        // reads last converted temperature value
#define START_CONVERT_T     0xEE        // initates temperature conversion
#define STOP CONVERT_T      0x22        // halts temperature conversion
#define WRITE_TH            0x01        // write high temperature limit
#define WRITE_TL            0x02        // write low temperature limit
#define READ_TH             0xA1        // read high temperature limit
#define READ_TL             0xA2        // read low temperature limit
#define WRITE_CONFIG        0x0C        // write configuration data
#define READ_CONFIG         0xAC        // read configuration data
#define CONTINUOUS          0x02        // set for continuous conversion
#define ONESHOT             0x01        // set for one temperature conversion
#define TEMPSTEP            0.5         // digit step of the temperature
#define TOOLOW              0x00
#define GOOD                0x01
#define TOOHIGH             0x02

/*_____________ H A R D W A R E _____________________________________________*/

sbit DQ     = P3^5;
sbit CLK    = P3^6;
sbit RST    = P3^7;
/* set or clr of important PIN */
#define setDQ       DQ=1;while (DQ!=1);                    // data in
#define clrDQ       DQ=0;
#define setCLK      CLK=1;while (CLK!=1);                  // clock
#define clrCLK      CLK=0;
#define setRST      RST=1;while (RST!=1);                  // chip reset
#define clrRST      RST=0;

/*_____________ P R O T O C O L _____________________________________________*/

void ds1620_init(float TH, float TL);
void  ds1620_sendByte(unsigned char dat);
unsigned char ds1620_readByte(void);
float ds1620_toDegree(unsigned char dat);
float ds1620_ReadTemperature(void);
float ds1620_ReadTH(void);
float ds1620_ReadTL(void);

/*_____________ F U N C T I O N _____________________________________________*/
```

```
/*
    initalize the digital thermometer with the High temperature trigger and Low
    temperature trigger.
*/
void ds1620_init(float TH, float TL)
{
    int numTH,numTL;                    // no. step
    unsigned char thH,thL,tlH,tlL;

    /* convert the float input to digit step */
    numTH=TH/TEMPSTEP;                  // the no. step of TH
    numTL=TL/TEMPSTEP;                  // the no. step of TL

    // high temperature limit
    thH=((numTH&0xff00)/0x00ff);        // high byte of TH
    if(!(TH>0))
    {
        thH = 0x01;
    }
    thL=((numTH&0x00ff));               // low byte of TH

    // low temperature limit
    tlH=((numTL&0xff00)/0x00ff);        // high byte of TH
    if(!(TL>0))
    {
        tlH = 0x01;
    }
    tlL=((numTL&0x00ff));               // low byte of TH

    printf("Digital Code is : ");
    printHex8(thH);
    printHex8(thL);
    printHex8(tlH);
    printHex8(tlL);

    /* start the configuration */
    ds1620_sendByte(WRITE_CONFIG);
    ds1620_sendByte(CONTINUOUS);
    clrRST;
    setRST;
    ds1620_sendByte(WRITE_TH);          // set TH
    ds1620_sendByte(thL);
    ds1620_sendByte(thH);
    clrRST;
    setRST;
    ds1620_sendByte(WRITE_TL);          // set TL
    ds1620_sendByte(tlL);
    ds1620_sendByte(tlH);
    clrRST;
    setRST;
    ds1620_sendByte(START_CONVERT_T);           // issues start convert T command

}


/*
    send 8 bit data to the digital thermometer.
```

```
*/
void  ds1620_sendByte(unsigned char dat)
{
    unsigned char i,mask=1;
    setRST;
    for (i=0; i<8; i++)
    {
        clrCLK;
        DQ = (dat & mask);                      // send bit to 1620
        setCLK;
        mask=(mask<<1);                         // setup to send next bit
    }
}

/*
    get 8 bit data from the digital thermometer.
*/
unsigned char ds1620_readByte(void)
{
    unsigned char i,rcv=0,mask=1;
    for (i=0; i<8; i++)
    {
        clrCLK;
        if (DQ) rcv|=mask;                      // read bit and or if = 1
        setCLK;
        mask=(mask<<1);                         // setup for next read and or
    }
    return rcv;
}

/*
    the binary data represend a interger step. For 0x01, it may represent 5, for 0x03,
    it may represnet 15. So. this function is for the conversation of the binary data
    to the refered interger.
*/
float ds1620_toDegree(unsigned char dat)
{
    float fdeg=0;
    unsigned int i;

    // only for positive data, if dat is negative, the 2 complement will required.
    for(i=0;i<dat;i++)
    {
        fdeg +=0.5;
    }

    return fdeg;
}

/*
    issues Ds1620 measures the current temperature and then read data from the
    thermometer,convert digit ouputs and return the decimal interger of temperature.
*/
float ds1620_ReadTemperature(void)
{
    unsigned char datH,datL;
    float deg;
```

```
    clrRST;
    setRST;
    ds1620_sendByte(READ_TEMPERATURE);          // read temperature T
    datL = ds1620_readByte();
    datH = ds1620_readByte();

    if(datH&0x01)
    {
        deg = -ds1620_toDegree(~datL+1 );       // if the temperature is negative
    }                                           // take complement
    else
    {
        deg = ds1620_toDegree(datL);            // if the temperature is positive
    }

    return deg;                                 // return T in interger form
}

/*
    reads stored valuse of high temperature limit from TH register.
*/
float ds1620_ReadTH()
{
    unsigned char datH,datL;
    float deg;

    clrRST;
    setRST;
    ds1620_sendByte(READ_TH);               // read TH
    datL = ds1620_readByte();
    datH = ds1620_readByte();

    if(datH&0x01)
    {
        deg = -ds1620_toDegree(~datL+1 );       // if the temperature is negative
    }                                           // take complement
    else
    {
        deg = ds1620_toDegree(datL);            // if the temperature is positive
    }

    return deg;                                 // return T in interger form

}

/*
    reads stored valuse of low temperature limit from TL register.
*/
float ds1620_ReadTL()
{
    unsigned char datH,datL;
    float deg;

    clrRST;
    setRST;
    ds1620_sendByte(READ_TL);               // read TH
    datL = ds1620_readByte();
    datH = ds1620_readByte();
```

```
    if(datH&0x01)
    {
        deg = -ds1620_toDegree(~datL+1 );       // if the temperature is negative
    }                                           // take complement
    else
    {
        deg = ds1620_toDegree(datL);            // if the temperature is positive
    }

    return deg;                                 // return T in interger form

}
```

```
/*
  Name :  DS1620.h

  Header file for DS1620.c
*/


/*_____________ P R O T O C O L _____________________________________________*/


void ds1620_init(float TH, float TL);
float ds1620_ReadTemperature(void);
float ds1620_ReadTH(void);
float ds1620_ReadTL(void);
```

```
/*-------------------------------------------------------------------------------+
|                                                                               |
|                          Print Hex8 or Hex16                                  |
|                                                                               |
| Name: printHex.c                                                              |
                                                        |
| Purpose:                                                                      |
|   print the hex8 or hex16 data the streaming output.                          |
+-------------------------------------------------------------------------------*/

/*_____________ I N C L U D E S _____________________________________________*/

#include <reg51.h>
#include <stdio.h>

/*_____________ P R O T O C O L _____________________________________________*/
void printHex8(unsigned char hexdata);
void printHex16(int hex16);
unsigned char hex2ascii(unsigned char bits_data);

/*_____________ F U N C T I O N _____________________________________________*/

/*
    print 8 bit hex data
*/
void printHex8(unsigned char hex8)
{
    char hex8H,hex8L;
    hex8L = hex2ascii(hex8&0x0f);                       // convert low byte
    hex8H = hex2ascii((hex8&0xf0)/0x0f);                // convert high byte
    putchar(hex8H);                                     // print out
    putchar(hex8L);
}

/*
    print 16 bit hex data
*/
void printHex16(int hex16)
{
    char hex16H,hex16L;
    hex16H=((hex16&0xff00)/0x00ff);                     // high byte
    hex16L=((hex16&0x00ff));                            // low byte
    printHex8(hex16H);
    printHex8(hex16L);
}

/*
    convert the hex data(4 bits) to ascii code(8 bits)
*/
unsigned char hex2ascii(unsigned char bits_data)
{
    if(bits_data >=0x0f)                                // all invalid data return 'F'
    {
        return 0x46;
    }
    if (bits_data >=0x00 & bits_data <=0x09)            // return '0'-'9'
    {
        bits_data +=0x30;
```

```
        return bits_data;
    }
    if (bits_data >=0x0a & bits_data <=0x0f);           // return 'A'-'F'
    {
        bits_data +=0x37;
        return bits_data;
    }
}
```

```
/*
  Name :  rprintf.h

  Header file for rprintf.c
*/


/*_____________ P R O T O C O L _____________________________________________*/

void printHex8(unsigned char hexdata);
void printHex16(int hex16);
void printInt(int num);
void printFloat(float flt);
void printStr(char str[]);
void rprintf(char type[],char content[]);
```