```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/socket.h>
#include <arpa/inet.h>

#include <wiringPi.h>

#define PORT 9000

#define SCK             11      // 온습도 모듈의 TEMP의 SCK를 GPIO 11번에 연결
#define SDA             9  // 온습도 모듈의 TEMP의 SDA를 GPIO 9번에 연결

#define NOACK 0
#define ACK   1
                                    // Addr     Code(command)   r/w
#define MEASURE_TEMP                0x03    // 000          0001
        1
#define MEASURE_HUMI                0x05    // 000          0010
        1
#define READ_STATUS_REG             0x07    // 000          0011
        1
#define WRITE_STATUS_REG    0x06    // 000          0011                    0
#define RESET                       0x1e    // 000          1111
        0

enum { TEMP, HUMI }; // 온도와 습도를 나타내는 문자열 변수 선언

#define P_PIN  6
#define N_PIN  12  // fan전원은 5v에 연결

//////////////////////////////////////

void SHT11_Init (void);
void Connection_reset (void);
void Transmission_start (void);
float get_SHT11_data (unsigned char type);
unsigned char Write_byte (unsigned char value);
unsigned char Read_byte (unsigned char ack);
unsigned char Measure (unsigned short *p_value, unsigned short *p_checksum,
```

```
        unsigned char mode);
void calc_SHT11 (unsigned short p_humidity ,unsigned short p_temperature);

float    val_temp, val_humi;
unsigned short SHT11_humi, SHT11_temp;
unsigned short error, checksum;

// SHT11 센서의 TWI(I2C) 사용을 위한 초기화 작업
void SHT11_Init (void)
{
        pinMode(SDA, OUTPUT);
        pinMode(SCK, OUTPUT);
        Connection_reset ();
}

// TWI통신 연결
//------------------------------------------------------------------------------
-------
// communication reset: DATA-line=1 and at least 9 SCK cycles followed by
transstart
//                        ---------------------------------------------------
--------
// DATA:                                                    |_______|
//          _    _    _    _    _    _    _    _    _        ___     ___
// SCK : __| |__| |__| |__| |__| |__| |__| |__| |__| |______|   |___|   |______
//------------------------------------------------------------------------------
-------
void Connection_reset (void)
{
        unsigned char i;
        digitalWrite(SDA, 1);                           // Initial state
        digitalWrite(SCK, 0);                           // Initial state
        delayMicroseconds(1);
        for (i=0; i<9; i++) {               // 9 SCK cycles
                digitalWrite(SCK, 1);
                delayMicroseconds(1);
        digitalWrite(SCK, 0);
                delayMicroseconds(1);
        }
}
```

```
// 전송 시작 신호
//-----------------------------------------------------------------------------
-------
// generates a transmission start
//       _____         ________
// DATA:      |_______|
//           ___     ___
// SCK : ___|   |___|   |______
//-----------------------------------------------------------------------------
-------
void Transmission_start (void)
{
        digitalWrite(SDA, 1);                    //Initial state
        digitalWrite(SCK, 0);                     //Initial state
        delayMicroseconds(1);

        digitalWrite(SCK, 1);
        delayMicroseconds(1);

        digitalWrite(SDA, 0);
        delayMicroseconds(1);

        digitalWrite(SCK, 0);
        delayMicroseconds(1);

        digitalWrite(SCK, 1);
        delayMicroseconds(1);

        digitalWrite(SDA, 1);
        delayMicroseconds(1);

        digitalWrite(SCK, 0);
}

//온/습도 데이터 획득
float get_SHT11_data (unsigned char type)
{
        error   =       0;

        if (type == HUMI) { // 습도 수치 측정
                // measure humidity
```

```
                error   +=      Measure (&SHT11_humi, &checksum, HUMI);
                if (error != 0)         // 에러 발생 시, Connection reset
                        Connection_reset ();
                else                            //- 측정된 값을 바탕으로 온/습도 값
계산
                        calc_SHT11 (SHT11_humi, SHT11_temp);
                return  val_humi;
        }
        else if (type == TEMP) { // 온도 수치 측정
                // measure temperature
                error   +=      Measure (&SHT11_temp, &checksum, TEMP);
                if (error != 0)         // 에러 발생 시, Connection reset
                        Connection_reset ();
                else                            //- 측정된 값을 바탕으로 온/습도 값
계산
                        calc_SHT11 (SHT11_humi, SHT11_temp);
                return  val_temp;
        }
        else {
                return  0;
        }
}

// makes a measurement (humidity/temperature) with checksum
unsigned char Measure (unsigned short *p_value, unsigned short *p_checksum,
        unsigned char mode)
{
        unsigned short error    =       0;
        unsigned short SHT11_msb, SHT11_lsb;

        switch (mode)                           //send command to sensor
        {
                case TEMP : // 온도 요청 신호 송신(센서 디바이스 주소 및 온도 Read
명령)
                        error   +=      Write_byte (MEASURE_TEMP);
                        break;
                case HUMI : // 습도 요청 신호 송신(센서 디바이스 주소 및 습도 Read
명령)
                        error   +=      Write_byte (MEASURE_HUMI);
                        break;
                default :
```

```
                    break;
        }
        if (error != 0)
                return  error;

        pinMode(SDA, INPUT);  // 데이터 수신 대기

        while (digitalRead(SDA));  // 센서가 센싱한 데이터를 보내 줄 때까지 대기
                                                        //
        pinMode(SDA, INPUT);

        // 데이터 수신
        SHT11_msb     =       Read_byte (ACK);              //  read  the  first
byte (MSB)
        SHT11_lsb     =       Read_byte (ACK);              // read the second
byte (LSB)
        *p_value      =       (SHT11_msb * 256) + SHT11_lsb;
        *p_checksum   =       Read_byte (NOACK);    // read checksum

        return error;
}

//데이터 송신
// writes a byte on the Sensibus and checks the acknowledge
unsigned char Write_byte (unsigned char value)
{
        unsigned char i, error  =       0;
        pinMode(SDA, OUTPUT);
        for (i=0x80; i>0; i/=2)   {  // shift bit for masking
                if (i & value)    digitalWrite(SDA, 1); // masking value with i , write
to SENSI-BUS
                else              digitalWrite(SDA, 0);

                delayMicroseconds(1);
                digitalWrite(SCK, 1);  // clk for SENSI-BUS
                delayMicroseconds(1);
                digitalWrite(SCK, 0);
                delayMicroseconds(1);
        }
        digitalWrite(SDA, 1);   // release DATA-line
        pinMode(SDA, INPUT);
```

```
        delayMicroseconds(1);
        digitalWrite(SCK, 1);     // clk #9 for ack
        error   =       digitalRead(SDA); // check ack (DATA will be pulled down
by SHT11)

        digitalWrite(SCK, 0);
        pinMode(SDA, OUTPUT);

        return error;  // error=1 in case of no acknowledge
}

//데이터 수신
// reads a byte from the Sensibus and gives an acknowledge in case of "ack=1"
unsigned char Read_byte(unsigned char ack)
{
        unsigned char i, val    =       0;

        digitalWrite(SDA, 1);    // release DATA-line
        pinMode(SDA, INPUT);
        delayMicroseconds(1);

        for (i=0x80; i>0; i/=2)   {  // shift bit for masking
                digitalWrite(SCK, 1);  // clk for SENSI-BUS
                delayMicroseconds(1);
                if (digitalRead(SDA))  // read bit
                        val = (val | i);
                digitalWrite(SCK, 0);
                delayMicroseconds(1);
        }
        pinMode(SDA, OUTPUT);

        if (ack) digitalWrite(SDA, 0);  // in case of "ack==1" pull down DATA-Line
        else            digitalWrite(SDA, 1);

        digitalWrite(SCK, 1);     // clk #9 for ack
        delayMicroseconds(1);
        digitalWrite(SCK, 0);
        delayMicroseconds(1);
        digitalWrite(SDA, 1);     // release DATA-line

        return val;
```

```
}

 //온/습도 값 계산 함수(센서로부터 수신한 값은 변환이 필요)
void calc_SHT11 (unsigned short humidity, unsigned short temperature)
{
        const float C1  =       -2.0468;        // for 12 Bit
        const float C2  =       0.0367;         // for 12 Bit
        const float C3  =       -0.0000015955; // for 12 Bit
        const float T1  =       0.01;                   // for 12 Bit
        const float T2  =       0.00008;                // for 12 Bit

        float rh_lin;               // Relative Humidity
        float rh_true;              // Humidity Sensor RH/Temperature compensation
        float t_C;                          // Temperature
        float rh =      (float)humidity;
        float t         =       (float)temperature;

        t_C                     =       ((t * 0.01) - 40.1) - 5;
        rh_lin          =       (C3 * rh * rh) + (C2 * rh) + C1;
        rh_true         =       (t_C - 25) * (T1 + (T2 * rh)) + rh_lin;

        if (rh_true > 100)      rh_true = 100;
        if (rh_true < 0.1)      rh_true = 0.1;

        val_temp        =       t_C;
        val_humi        =       rh_true;
}


float temp;

int main (void)
{
        if(wiringPiSetupGpio() == -1)
                return 1;

        SHT11_Init(); //온/습도 센서 초기 설정

        pinMode(P_PIN, OUTPUT);
        pinMode(N_PIN, OUTPUT);
```

```
int  s_socket, c_socket;
struct sockaddr_in s_addr, c_addr;

int n;
int len;
char rcvBuffer[BUFSIZ];

s_socket = socket(PF_INET, SOCK_STREAM, IPPROTO_TCP);

memset(&s_addr, 0, sizeof(s_addr));
s_addr.sin_addr.s_addr = htonl(INADDR_ANY);
s_addr.sin_family = AF_INET;
s_addr.sin_port = htons(PORT);

if(bind(s_socket, (struct sockaddr*)&s_addr, sizeof(s_addr)) == -1){
        printf("Can not Bind!!!\n");
        return -1;
}

if(listen(s_socket, 5) == -1){
        printf("Listen Fail!!!\n");
        return -1;
}

printf("Temp Server started...\n");

while(1){
        len = sizeof(c_addr);
        c_socket = accept(s_socket, (struct sockaddr*)&c_addr, &len);
        printf("Connected IP : %s\n", inet_ntoa(c_addr.sin_addr));

        while((n = read(c_socket, rcvBuffer, sizeof(rcvBuffer))) > 0){
                rcvBuffer[n] = '\0';
                //printf("%s", rcvBuffer);
                if(strncmp(rcvBuffer, "temp", 4) == 0){

                        Transmission_start(); // 전송 시작
                        temp = get_SHT11_data (TEMP);// Sensing Temp
                        delay(10);

                        printf("Temp = %2.2f [C]\n", temp);
```

```
                        sprintf(rcvBuffer, "%2.2f", temp);

                        rcvBuffer[5] = '\n';
                        delay(1);
                        write(c_socket, rcvBuffer, 6);
                    }else if(strncmp(rcvBuffer, "on", 2) == 0){
                        digitalWrite(P_PIN, HIGH);
                        digitalWrite(N_PIN, LOW);
                        rcvBuffer[2] = '\n';

                        delay(1);
                        write(c_socket, rcvBuffer, 3);
                    }else if(strncmp(rcvBuffer, "off", 2) == 0){
                        digitalWrite(P_PIN, LOW);
                        digitalWrite(N_PIN, LOW);
                        rcvBuffer[3] = '\n';
                        delay(1);
                        write(c_socket, rcvBuffer, 4);
                    }
                }

        }
        return 0;
}
```