EEM 449 Embedded System Design
Read Heart Beat from
MAX30100

Written by Faruk Gökhan Sönmez

The project I have done contains information about I2C sensor readings, TCP and UDP servers on ek-tm4c1294xl launchpad. The data are obtained from the MAX30100 sensor and transmitted to the TCP server. At the same time, control of the system with 3 different commands can be provided via TCP server. Semaphore and mailbox structures were created for the transmission of data in the project.

I2C Sensor reading

First of all, I connected the SCL vs SDA legs correctly so that our sensor can communicate with the launchpad. Then I wrote the necessary codes with the device ID of the sensor to be able to communicate with the sensor. All that remains is to calibrate the sensor and extract the correct data. In order to decide in which mode the system will operate, we only provide HR measurement by making the capability of the 0x06 addressed mode. Later, we tackle issues such as Sampling Rate, Led Pulse, Led Current and filter with the help of datasheet.

Sending data to server

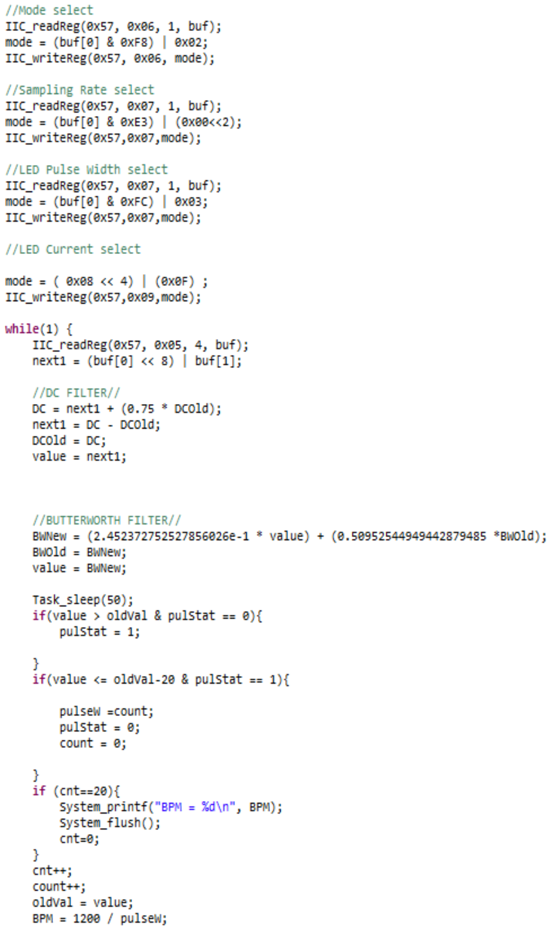
In this part of the project, I made use of the tcp_client_v4 file. First of all, we determine the ip address of our computer and the port numbers we will communicate with. We set the commands and responses we need to get from the system within the serverSocketTest function.

Reading time from ntp server

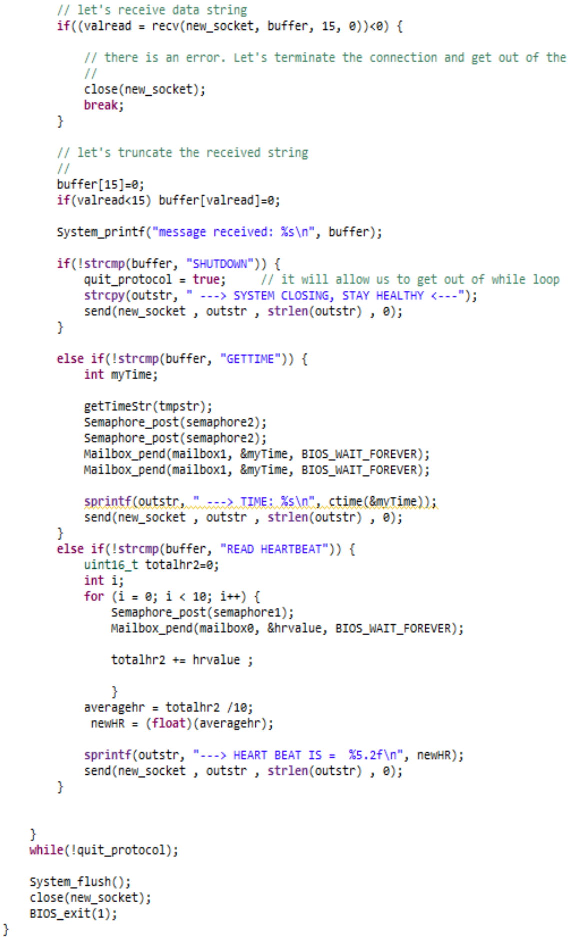
To read data from the Ntp server, we do the same as the sendData2server function for the ntp ip address. The data sent by our NTP server transmits us the second from a certain time to the current time. This is structured by dividing the second into the required units. The incoming data consists of 32 bits of data. I made a separation in 8-bit numbers to make data operations easier. Later, I made the configurations of these reserved numbers and made them transmit to the server when the GETTIME command is entered.

At the end of the project, a sensor that can measure heartbeat per minute with the command, and a launchpad structure that can receive data from the ntp server and send data to the server was established. With the command system, it has been provided to be easy to control over the server.

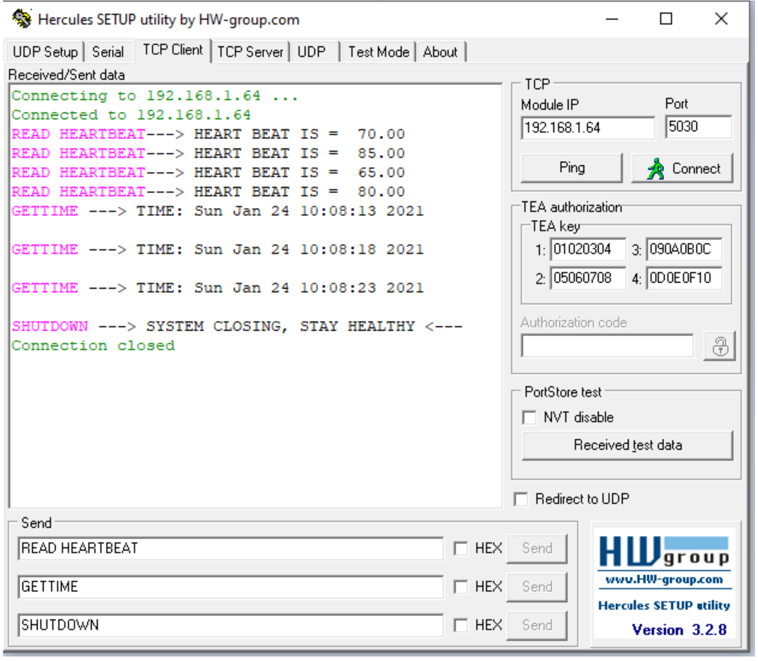
System Photos:



```
//Mode select
IIC_readReg(0x57, 0x06, 1, buf);
mode = (buf[0] & 0xF8) | 0x02;
IIC_writeReg(0x57, 0x06, mode);

//Sampling Rate select
IIC_readReg(0x57, 0x07, 1, buf);
mode = (buf[0] & 0xE3) | (0x00<<2);
IIC_writeReg(0x57,0x07,mode);

//LED Pulse Width select
IIC_readReg(0x57, 0x07, 1, buf);
mode = (buf[0] & 0xFC) | 0x03;
IIC_writeReg(0x57,0x07,mode);

//LED Current select

mode = ( 0x08 << 4) | (0x0F) ;
IIC_writeReg(0x57,0x09,mode);

while(1) {
    IIC_readReg(0x57, 0x05, 4, buf);
    next1 = (buf[0] << 8) | buf[1];

    //DC FILTER//
    DC = next1 + (0.75 * DCOld);
    next1 = DC - DCOld;
    DCOld = DC;
    value = next1;



    //BUTTERWORTH FILTER//
    BWNew = (2.452372752527856026e-1 * value) + (0.50952544949442879485 *BWOld);
    BWOld = BWNew;
    value = BWNew;

    Task_sleep(50);
    if(value > oldVal & pulStat == 0){
        pulStat = 1;

    }
    if(value <= oldVal-20 & pulStat == 1){

        pulseW =count;
        pulStat = 0;
        count = 0;

    }
    if (cnt==20){
        System_printf("BPM = %d\n", BPM);
        System_flush();
        cnt=0;
    }
    cnt++;
    count++;
    oldVal = value;
    BPM = 1200 / pulseW;
```



```
    // let's receive data string
    if((valread = recv(new_socket, buffer, 15, 0))<0) {

        // there is an error. Let's terminate the connection and get out of the
        //
        close(new_socket);
        break;
    }

    // let's truncate the received string
    //
    buffer[15]=0;
    if(valread<15) buffer[valread]=0;

    System_printf("message received: %s\n", buffer);

    if(!strcmp(buffer, "SHUTDOWN")) {
        quit_protocol = true;     // it will allow us to get out of while loop
        strcpy(outstr, " ---> SYSTEM CLOSING, STAY HEALTHY <---");
        send(new_socket , outstr , strlen(outstr) , 0);
    }

    else if(!strcmp(buffer, "GETTIME")) {
        int myTime;

        getTimeStr(tmpstr);
        Semaphore_post(semaphore2);
        Semaphore_post(semaphore2);
        Mailbox_pend(mailbox1, &myTime, BIOS_WAIT_FOREVER);
        Mailbox_pend(mailbox1, &myTime, BIOS_WAIT_FOREVER);

        sprintf(outstr, " ---> TIME: %s\n", ctime(&myTime));
        send(new_socket , outstr , strlen(outstr) , 0);
    }
    else if(!strcmp(buffer, "READ HEARTBEAT")) {
        uint16_t totalhr2=0;
        int i;
        for (i = 0; i < 10; i++) {
            Semaphore_post(semaphore1);
            Mailbox_pend(mailbox0, &hrvalue, BIOS_WAIT_FOREVER);

            totalhr2 += hrvalue ;

            }
        averagehr = totalhr2 /10;
         newHR = (float)(averagehr);

        sprintf(outstr, "---> HEART BEAT IS =  %5.2f\n", newHR);
        send(new_socket , outstr , strlen(outstr) , 0);
    }


}
while(!quit_protocol);

System_flush();
close(new_socket);
BIOS_exit(1);
}
```



```
Connecting to 192.168.1.64 ...
Connected to 192.168.1.64
READ HEARTBEAT---> HEART BEAT IS =  70.00
READ HEARTBEAT---> HEART BEAT IS =  85.00
READ HEARTBEAT---> HEART BEAT IS =  65.00
READ HEARTBEAT---> HEART BEAT IS =  80.00
GETTIME ---> TIME: Sun Jan 24 10:08:13 2021

GETTIME ---> TIME: Sun Jan 24 10:08:18 2021

GETTIME ---> TIME: Sun Jan 24 10:08:23 2021

SHUTDOWN ---> SYSTEM CLOSING, STAY HEALTHY <---
Connection closed
```

Module IP: 192.168.1.64
Port: 5030

TEA key 1: 01020304
TEA key 2: 05060708
TEA key 3: 090A0B0C
TEA key 4: 0D0E0F10

□ NVT disable

□ Redirect to UDP

READ HEARTBEAT □ HEX

GETTIME □ HEX

SHUTDOWN □ HEX