Univerzitet u Sarajevu

Elektrotehnički fakultet
Odsjek za automatiku i elektroniku

Laboratorijska vjezba 7

Komunikacija - Zapisivanje podataka u bazu i vizualizacija podataka

---

Bazdar Faris
Predmet: Ugradbeni sistemi
Akademska godina: 2019/2020

Sarajevo, april 2020.

1 Zadatak 1

mojejedinstvenovlastitoime = fbazdar

2 Zadatak 2

```
#include "mbed.h"
#include "http_request.h"

#define USER_NAME "fbazdar"


#define TEMAPUBPOT1 "ugradbeni/fbazdar/potenciometar1"
#define TEMAPUBPOT2 "ugradbeni/fbazdar/potenciometar2"

#include "mbed.h"
#include "C12832.h"

#define MQTTCLIENT_QOS2 0

#include "easy-connect.h"
#include "MQTTNetwork.h"
#include "MQTTmbed.h"
#include "MQTTClient.h"
#include <string.h>

double pot_value1=-1;
double pot_value2=-1;

Ticker t;
AnalogIn pot1(p15);
AnalogIn pot2(p16);


void write_db(void) {

    NetworkInterface* network;
    network=NetworkInterface::get_default_instance();
    if (!network) {
        printf("Cannot connect to the network, see serial output
           \n");
        return;
    }
    MQTTNetwork mqttNetwork(network);

    MQTT::Client<MQTTNetwork, Countdown> client(mqttNetwork);

    const char* hostname = "broker.hivemq.com";
```

```
    int port = 1883;
    printf("Connecting to \%s:\%d\r\n", hostname, port);
    int rc = mqttNetwork.connect(hostname, port);
    if (rc != 0)
        printf("rc from TCP connect is \%d\r\n", rc);

    MQTTPacket_connectData data =
       MQTTPacket_connectData_initializer;
    data.MQTTVersion = 3;
    data.clientID.cstring = "ugradbeni";
    data.username.cstring = "";
    data.password.cstring = "";
    if ((rc = client.connect(data)) != 0)
        printf("rc from MQTT connect is \%d\r\n", rc);

   MQTT::Message message;

    // QoS 0
    char buf[100];

        if (pot_value2!=pot2) {
            pot_value2=pot2;
            sprintf(buf, "\%f", pot_value2);
            message.qos = MQTT::QOS0;
            message.retained = false;
            message.dup = false;
            message.payload = (void*)buf;
            message.payloadlen = strlen(buf);
            rc = client.publish(TEMAPUBPOT1, message);
        }
        if (pot_value1!=pot1) {
            pot_value1=pot1;
            sprintf(buf, "\%f", pot_value1);
            message.qos = MQTT::QOS0;
            message.retained = false;
            message.dup = false;
            message.payload = (void*)buf;
            message.payloadlen = strlen(buf);
            rc = client.publish(TEMAPUBPOT2, message);
        }
}

int main() {
    t.attach(write_db,2);
    while(1) {
        wait(0.5);
    }
}
```

Izmjena koda za drugi zadatak
U ovom zadatku bilo je potrebno podesiti kod tako da se komunikacija vrši preko MQTT protokola sa grafan-om. To je urađeno tako što je prvo ostvarena veza sa MQTT hive-om putem komade *rc = mqttNetwork.connect(hostname, port);*.
Postavljene su dvije metode za ažuriranje promjene stanja unutar MQTT što bi izazvalo promjenu stanja na grafani. Vrijednost otklona potenciometra se pohranjivala privremeno u string *buf*.
Potom se ta poruka šalje putem komandi *rc = client.publish(TEMAPUBPOT1,message)* ili *rc = client.publish(TEMAPUBPOT2,message)* u ovisnosti koji je potenciometar promijenio vrijednost.
Funkcija *write_db* se poziva putem tickera, svake dvije sekunde. Na taj način je ostvareno periodično upisivanje u bazu koristeći programiranje vođeno događajima.

3 Zadatak 3

```
#include "mbed.h"
#include "http_request.h"

#define USER_NAME "fbazdar"

#define TEMAPUBSIN  "ugradbeni/fbazdar/sinusoida"

#include "mbed.h"
#include "C12832.h"

#define MQTTCLIENT_QOS2 0

#include "easy-connect.h"
#include "MQTTNetwork.h"
#include "MQTTmbed.h"
#include "MQTTClient.h"
#include <string.h>
#include <cmath>

Ticker sine;


float i = 0;
void draw_sine(void){
    NetworkInterface* network;
    network=NetworkInterface::get_default_instance();
    if (!network) {
        printf("Cannot connect to the network, see serial output
           \n");
        return;
    }
    // Create a TCP socket
    printf("\n----- Setting up TCP connection -----\n");
    TCPSocket* socket = new TCPSocket();
    nsapi_error_t open_result = socket->open(network);
    if (open_result != 0) {
        printf("Opening TCPSocket failed... \%d\n", open_result)
           ;
        return;
    }

    nsapi_error_t connect_result = socket->connect
       ("195.130.59.222", 8086);
    if (connect_result != 0) {
        printf("Connecting over TCPSocket failed... \%d\n",
           connect_result);
```

```
        return;
    }

    printf("Connected over TCP\n");

    MQTTNetwork mqttNetwork(network);

    MQTT::Client<MQTTNetwork, Countdown> client(mqttNetwork);

    const char* hostname = "broker.hivemq.com";
    int port = 1883;
    printf("Connecting to \%s:\%d\r\n", hostname, port);
    int rc = mqttNetwork.connect(hostname, port);
    if (rc != 0)
        printf("rc from TCP connect is \%d\r\n", rc);

    MQTTPacket_connectData data =
       MQTTPacket_connectData_initializer;https
       ://195.130.59.222:3000/dashboard/script/ugradbeni_zad3.js
       ?panelId=2&fullscreen&orgId=1&rows=1&from=now-5m&to=now&
       name=fbazdar&refresh=1s
    data.MQTTVersion = 3;
    data.clientID.cstring = "ugradbeni";
    data.username.cstring = "";
    data.password.cstring = "";
    if ((rc = client.connect(data)) != 0)
        printf("rc from MQTT connect is \%d\r\n", rc);

    MQTT::Message message;

    char buf[100];
    sprintf(buf, "\%f", sin(i)*5);
    message.qos = MQTT::QOS0;
    message.retained = false;
    message.dup = false;
    message.payload = (void*)buf;
    message.payloadlen = strlen(buf);
    rc = client.publish(TEMAPUBSIN, message);
   if(i<=6.28){ i+=((6.28)/(12)); }else i = 0;
}

int main() {
    sine.attach(draw_sine,5);
    while(1) {
        wait(0.5);
    }
}
```

Izmjena koda za treći zadatak

U ovom zadatku bilo je potrebno omogućiti prikaz sinusoidalnog signala na grafani koristeći se MQTT protokolom. Konekcija sa samim MQTT klijentom je ostvarena na isti način kao i u prethodnom zadatku, jedina izmjena u smislu konekcije je tema na koju se sada šalje poruka iz simulatora. Zatim bilo je potrebno podesiti sinusni signal perioda 60s i amplitude $\pm 5V$. To je ostvarno tako što se u string *buf* slala vrijednost signala sinusa određenog koraka pomnožena sa 5. Taj korak je bio $\frac{2 \cdot \pi}{12}$. Vrijednost varijable i je poprimala vrijednost do 0 do 6.28 i resetovala bi se pri dostizanju vrijednosti od 6.28 kako bi se željena sinusoida nastavila pravilno prikazivati. Razlog zašto vrijednost i nije imala 60 promjena već 12, je zbog traženog osvježavanja u grafani, naime u prilogu je dat link sa osvježavanjem od 5 sekundi a $60/5 = 12$, pa je potrebno 12 puta uzeti uzorak vrijednosti sinusa.

Student: Bazdar Faris
Broj indeksa: 18413

Profesor: Red. prof. dr. Samim Konjicija
Demonstrator: Kemal Altwlkany