LV6-IZVJESTAJ

2. U ovom zadatku je bilo potrebno samo promijeniti ime u define. Ovaj program sluzi za upravljanje sistemom putem MQTT.

#define TEMASUBLED1 "selma/led1"

#define TEMASUBLED2 "selma/led2"

#define TEMASUBLED3 "selma/led3"

#define TEMAPUBPOT "selma/potenciometar"

#define TEMAPUBTAST "selma/taster"

#include "mbed.h"

#define MQTTCLIENT\_QOS2 0

#include "MQTTNetwork.h"

#include "MQTTmbed.h"

#include "MQTTClient.h"

#include <string.h>

//==============DIO KODA KOJI TREBA BITI KOMENTARISAN ZA MBED SIMULATOR============

#define MBED\_CONF\_APP\_WIFI\_SSID "ETF-WiFi-Guest"

#define MBED\_CONF\_APP\_WIFI\_PASSWORD "ETF-WiFi-Guest"

#include "ESP8266Interface.h"

ESP8266Interface wifi(PTE0, PTE1);

//==============KRAJ DIJELA KODA KOJI TREBA BITI KOMENTARISAN ZA MBED SIMULATOR============

int arrivedcount = 0;

//==============DIO KODA KOJI TREBA BITI KOMENTARISAN ZA MODUL FRDM-KL25Z============

//#define MQTT\_CLIENT\_NAME "MBED\_SIMULATOR"

//DigitalIn taster(p5);

//DigitalOut led1(p6);

//DigitalOut led2(p7);

//AnalogIn pot(p15);

//PwmOut led3(p21);

//==============KRAJ DIJELA KODA KOJI TREBA BITI KOMENTARISAN ZA MODUL FRDM-KL25Z============

//==============DIO KODA KOJI TREBA BITI KOMENTARISAN ZA MBED SIMULATOR============

#define MQTT\_CLIENT\_NAME "selma"

DigitalIn taster(D8);

DigitalOut led1(LED\_RED);

DigitalOut led2(LED\_GREEN);

AnalogIn pot(A0);

PwmOut led3(LED\_BLUE);

//==============KRAJ DIJELA KODA KOJI TREBA BITI KOMENTARISAN ZA MBED SIMULATOR============

char\* str;

double pot\_value=-1;

bool taster\_state=1;

void messageArrived\_led1(MQTT::MessageData& md)

{

    MQTT::Message &message = md.message;

    printf("Message arrived: qos %d, retained %d, dup %d, packetid %d\r\n", message.qos, message.retained, message.dup, message.id);

    printf("Payload %.\*s\r\n", message.payloadlen, (char\*)message.payload);

    ++arrivedcount;

    str=(char\*)message.payload;

    led1=atoi(str);

}

void messageArrived\_led2(MQTT::MessageData& md)

{

    MQTT::Message &message = md.message;

    printf("Message arrived: qos %d, retained %d, dup %d, packetid %d\r\n", message.qos, message.retained, message.dup, message.id);

    printf("Payload %.\*s\r\n", message.payloadlen, (char\*)message.payload);

    ++arrivedcount;

    str=(char\*)message.payload;

    led2=atoi(str);

}

void messageArrived\_led3(MQTT::MessageData& md)

{

    MQTT::Message &message = md.message;

    printf("Message arrived: qos %d, retained %d, dup %d, packetid %d\r\n", message.qos, message.retained, message.dup, message.id);

    printf("Payload %.\*s\r\n", message.payloadlen, (char\*)message.payload);

    ++arrivedcount;

    str=(char\*)message.payload;

    led3=atof(str);

}

int main(int argc, char\* argv[])

{

    printf("Ugradbeni sistemi\r\n");

    printf("Demonstracija korištenja MQTT protokola\r\n\r\n");

    SocketAddress a;

//==============DIO KODA KOJI TREBA BITI KOMENTARISAN ZA MODUL FRDM-KL25Z============

//    NetworkInterface \*network;

//    network = NetworkInterface::get\_default\_instance();

//

//    if (!network) {

//        return -1;

//    }

//    MQTTNetwork mqttNetwork(network);

//==============KRAJ DIJELA KODA KOJI TREBA BITI KOMENTARISAN ZA MODUL FRDM-KL25Z============

//==============DIO KODA KOJI TREBA BITI KOMENTARISAN ZA MBED SIMULATOR============

    printf("\r\nConnecting to WiFi...\r\n");

    int ret = wifi.connect(MBED\_CONF\_APP\_WIFI\_SSID, MBED\_CONF\_APP\_WIFI\_PASSWORD, NSAPI\_SECURITY\_WPA\_WPA2);

    if (ret != 0) {

        printf("\r\nConnection error\r\n");

        return -1;

    }

    printf("Success\r\n\r\n");

    printf("MAC: %s\r\n", wifi.get\_mac\_address());

    wifi.get\_ip\_address(&a);

    printf("IP: %s\r\n", a.get\_ip\_address());

    wifi.get\_netmask(&a);

    printf("Netmask: %s\r\n", a.get\_ip\_address());

    wifi.get\_gateway(&a);

    printf("Gateway: %s\r\n", a.get\_ip\_address());

    printf("RSSI: %d\r\n\r\n", wifi.get\_rssi());

    MQTTNetwork mqttNetwork(&wifi);

//==============KRAJ DIJELA KODA KOJI TREBA BITI KOMENTARISAN ZA MBED SIMULATOR============

    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);

    else

        printf("Connected to broker!\r\n");

    MQTTPacket\_connectData data = MQTTPacket\_connectData\_initializer;

    data.MQTTVersion = 3;

    data.clientID.cstring = MQTT\_CLIENT\_NAME;

    data.username.cstring = "";

    data.password.cstring = "";

    if ((rc = client.connect(data)) != 0)

        printf("rc from MQTT connect is %d\r\n", rc);

    if ((rc = client.subscribe(TEMASUBLED1, MQTT::QOS0, messageArrived\_led1)) != 0)

        printf("rc from MQTT subscribe is %d\r\n", rc);

    else

        printf("Subscribed to %s\r\n", TEMASUBLED1);

    if ((rc = client.subscribe(TEMASUBLED2, MQTT::QOS0, messageArrived\_led2)) != 0)

        printf("rc from MQTT subscribe is %d\r\n", rc);

    else

        printf("Subscribed to %s\r\n", TEMASUBLED2);

    if ((rc = client.subscribe(TEMASUBLED3, MQTT::QOS0, messageArrived\_led3)) != 0)

        printf("rc from MQTT subscribe is %d\r\n", rc);

    else

        printf("Subscribed to %s\r\n", TEMASUBLED3);

    MQTT::Message message;

    // QoS 0

    char buf[100];

    while(1) {

        if (taster\_state!=taster) {

            taster\_state=taster;

            sprintf(buf, "{\"Taster\": %d}", taster.read());

            message.qos = MQTT::QOS0;

            message.retained = false;

            message.dup = false;

            message.payload = (void\*)buf;

            message.payloadlen = strlen(buf);

            rc = client.publish(TEMAPUBTAST, message);

        }

        if (pot\_value!=pot) {

            pot\_value=pot;

            sprintf(buf, "{\"Potenciometar\": %f}", pot\_value);

            message.qos = MQTT::QOS0;

            message.retained = false;

            message.dup = false;

            message.payload = (void\*)buf;

            message.payloadlen = strlen(buf);

            rc = client.publish(TEMAPUBPOT, message);

        }

        rc = client.subscribe(TEMASUBLED1, MQTT::QOS0, messageArrived\_led1);

        rc = client.subscribe(TEMASUBLED2, MQTT::QOS0, messageArrived\_led2);

        rc = client.subscribe(TEMASUBLED3, MQTT::QOS0, messageArrived\_led3);

        wait\_us(100);

    }

}

3. U ovom zadtaku smo dodali tri potenciometra pomocu kojih upravljemo ledicama.

//

// Ugradbeni sistemi

// Demonstracija korištenja MQTT protokola

//

// Napomena: Svaki student treba koristiti jedinstvenu riječ

//           umjesto "ugradbeni" u okviru tema

//

#define TEMASUBLED1 "selma/led1"

#define TEMASUBLED2 "selma/led2"

#define TEMASUBLED3 "selma/led3"

#define TEMAPUBPOT "selma/potenciometar"

#define TEMAPUBPOTT "selma/potenciometar2"

#define TEMAPUBPOTTT "selma/potenciometar3"

#define TEMAPUBTAST "selma/taster"

#include "mbed.h"

#define MQTTCLIENT\_QOS2 0

#include "MQTTNetwork.h"

#include "MQTTmbed.h"

#include "MQTTClient.h"

#include <string.h>

//==============DIO KODA KOJI TREBA BITI KOMENTARISAN ZA MBED SIMULATOR============

#define MBED\_CONF\_APP\_WIFI\_SSID "ETF-WiFi-Guest"

#define MBED\_CONF\_APP\_WIFI\_PASSWORD "ETF-WiFi-Guest"

#include "ESP8266Interface.h"

ESP8266Interface wifi(PTE0, PTE1);

//==============KRAJ DIJELA KODA KOJI TREBA BITI KOMENTARISAN ZA MBED SIMULATOR============

int arrivedcount = 0;

//==============DIO KODA KOJI TREBA BITI KOMENTARISAN ZA MODUL FRDM-KL25Z============

//#define MQTT\_CLIENT\_NAME "MBED\_SIMULATOR"

//DigitalIn taster(p5);

//DigitalOut led1(p6);

//DigitalOut led2(p7);

//AnalogIn pot(p15);

//PwmOut led3(p21);

//==============KRAJ DIJELA KODA KOJI TREBA BITI KOMENTARISAN ZA MODUL FRDM-KL25Z============

//==============DIO KODA KOJI TREBA BITI KOMENTARISAN ZA MBED SIMULATOR============

#define MQTT\_CLIENT\_NAME "FRDM\_KL25Z"

DigitalIn taster(D8);

PwmOut led1(LED\_RED);

PwmOut led2(LED\_GREEN);

AnalogIn pot(A0);

PwmOut led3(LED\_BLUE);

//==============KRAJ DIJELA KODA KOJI TREBA BITI KOMENTARISAN ZA MBED SIMULATOR============

char\* str;

double pot\_value=-1;

bool taster\_state=1;

void messageArrived\_led1(MQTT::MessageData& md)

{

    MQTT::Message &message = md.message;

    printf("Message arrived: qos %d, retained %d, dup %d, packetid %d\r\n", message.qos, message.retained, message.dup, message.id);

    printf("Payload %.s\r\n", message.payloadlen, (char)message.payload);

    ++arrivedcount;

    str=(char\*)message.payload;

    led1=atof(str);

}

void messageArrived\_led2(MQTT::MessageData& md)

{

    MQTT::Message &message = md.message;

    printf("Message arrived: qos %d, retained %d, dup %d, packetid %d\r\n", message.qos, message.retained, message.dup, message.id);

    printf("Payload %.s\r\n", message.payloadlen, (char)message.payload);

    ++arrivedcount;

    str=(char\*)message.payload;

    led2=atof(str);

}

void messageArrived\_led3(MQTT::MessageData& md)

{

    MQTT::Message &message = md.message;

    printf("Message arrived: qos %d, retained %d, dup %d, packetid %d\r\n", message.qos, message.retained, message.dup, message.id);

    printf("Payload %.s\r\n", message.payloadlen, (char)message.payload);

    ++arrivedcount;

    str=(char\*)message.payload;

    led3=atof(str);

}

int main(int argc, char\* argv[])

{

    printf("Ugradbeni sistemi\r\n");

    printf("Demonstracija korištenja MQTT protokola\r\n\r\n");

    SocketAddress a;

//==============DIO KODA KOJI TREBA BITI KOMENTARISAN ZA MODUL FRDM-KL25Z============

//    NetworkInterface \*network;

//    network = NetworkInterface::get\_default\_instance();

//

//    if (!network) {

//        return -1;

//    }

//    MQTTNetwork mqttNetwork(network);

//==============KRAJ DIJELA KODA KOJI TREBA BITI KOMENTARISAN ZA MODUL FRDM-KL25Z============

//==============DIO KODA KOJI TREBA BITI KOMENTARISAN ZA MBED SIMULATOR============

    printf("\r\nConnecting to WiFi...\r\n");

    int ret = wifi.connect(MBED\_CONF\_APP\_WIFI\_SSID, MBED\_CONF\_APP\_WIFI\_PASSWORD, NSAPI\_SECURITY\_WPA\_WPA2);

    if (ret != 0) {

        printf("\r\nConnection error\r\n");

        return -1;

    }

    printf("Success\r\n\r\n");

    printf("MAC: %s\r\n", wifi.get\_mac\_address());

    wifi.get\_ip\_address(&a);

    printf("IP: %s\r\n", a.get\_ip\_address());

    wifi.get\_netmask(&a);

    printf("Netmask: %s\r\n", a.get\_ip\_address());

    wifi.get\_gateway(&a);

    printf("Gateway: %s\r\n", a.get\_ip\_address());

    printf("RSSI: %d\r\n\r\n", wifi.get\_rssi());

    MQTTNetwork mqttNetwork(&wifi);

//==============KRAJ DIJELA KODA KOJI TREBA BITI KOMENTARISAN ZA MBED SIMULATOR============

    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);

    else

        printf("Connected to broker!\r\n");

    MQTTPacket\_connectData data = MQTTPacket\_connectData\_initializer;

    data.MQTTVersion = 3;

    data.clientID.cstring = MQTT\_CLIENT\_NAME;

    data.username.cstring = "";

    data.password.cstring = "";

    if ((rc = client.connect(data)) != 0)

        printf("rc from MQTT connect is %d\r\n", rc);

    if ((rc = client.subscribe(TEMAPUBPOT, MQTT::QOS0, messageArrived\_led1)) != 0)

        printf("rc from MQTT subscribe is %d\r\n", rc);

    else

        printf("Subscribed to %s\r\n", TEMAPUBPOT);

    if ((rc = client.subscribe(TEMAPUBPOTT, MQTT::QOS0, messageArrived\_led2)) != 0)

        printf("rc from MQTT subscribe is %d\r\n", rc);

    else

        printf("Subscribed to %s\r\n", TEMAPUBPOTT);

    if ((rc = client.subscribe(TEMAPUBPOTTT, MQTT::QOS0, messageArrived\_led3)) != 0)

        printf("rc from MQTT subscribe is %d\r\n", rc);

    else

        printf("Subscribed to %s\r\n", TEMAPUBPOTTT);

    MQTT::Message message;

    // QoS 0

    char buf[100];

    while(1) {

        if (taster\_state!=taster) {

            taster\_state=taster;

            sprintf(buf, "{\"Taster\": %d}", taster.read());

            message.qos = MQTT::QOS0;

            message.retained = false;

            message.dup = false;

            message.payload = (void\*)buf;

            message.payloadlen = strlen(buf);

            rc = client.publish(TEMAPUBTAST, message);

        }

        if (pot\_value!=pot) {

            pot\_value=pot;

            sprintf(buf, "{\"Potenciometar\": %f}", pot\_value);

            message.qos = MQTT::QOS0;

            message.retained = false;

            message.dup = false;

            message.payload = (void\*)buf;

            message.payloadlen = strlen(buf);

            rc = client.publish(TEMAPUBPOT, message);

        }

        rc = client.subscribe(TEMAPUBPOT, MQTT::QOS0, messageArrived\_led1);

        rc = client.subscribe(TEMAPUBPOTT, MQTT::QOS0, messageArrived\_led2);

        rc = client.subscribe(TEMAPUBPOTTT, MQTT::QOS0, messageArrived\_led3);

        wait\_us(100);

    }

}