



Uvod

U ovoj laboratorijskoj vježbi ćemo se upoznati sa načinom realizacije komunikacije u okviru ugradbenih sistema. U vježbi će biti demonstrirano i kako se protokol MQTT koristi za upravljanje standardnim ulazima i izlazima razvojnog sistema. Za realizaciju zadataka ćemo koristiti Mbed simulator, te razvojni sistem picoETF, koji posjeduje WiFi modul.

Osnovni pojmovi vezani za komunikaciju u ugradbenim sistemima su pojašnjeni u knjizi [1], kao i na predavanjima [2] i [3], a dokumentacije za Mbed OS API i MicroPython su date na linkovima [4] i [5].

Komunikacija u ugradbenim sistemima

U nastavku će biti pojašnjen način korištenja aplikacije Putty, kao i struktura sistema koji koristi MQTT protokol za upravljanje standardnim ulazima i izlazima razvojnog sistema.

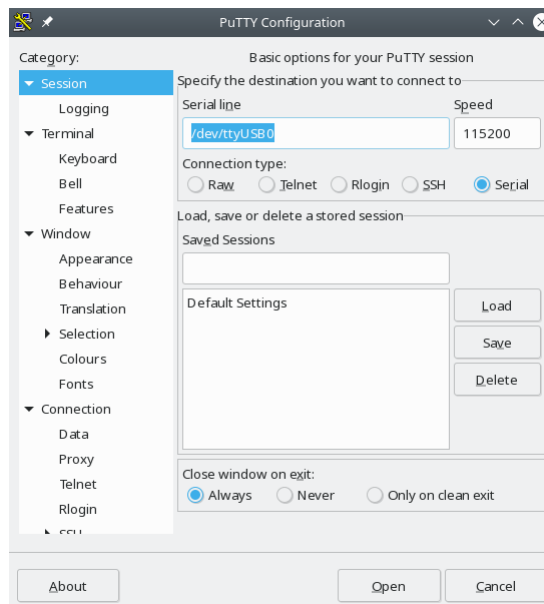
Putty

Putty predstavlja open source aplikaciju koja omogućava korištenje serijske komunikacije, kao i *ssh* (o *ssh* će biti riječi u jednoj od sljedećih laboratorijskih vježbi). Može se preuzeti sa linka <http://www.putty.org/>.

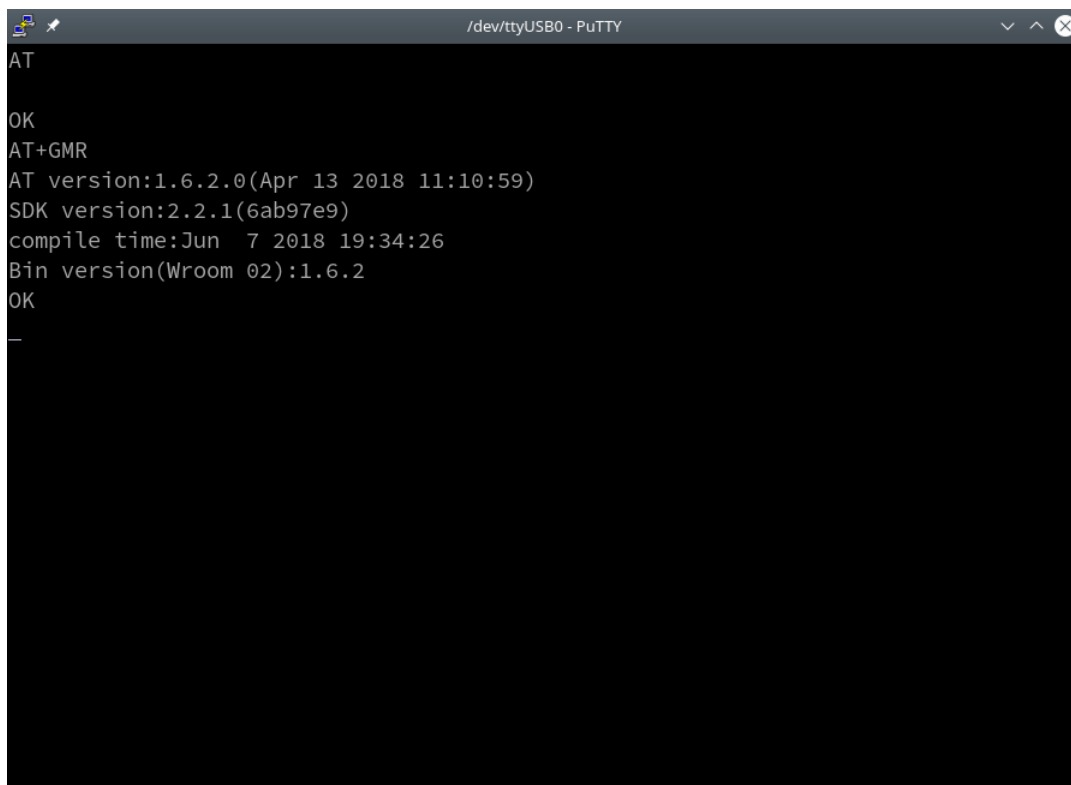
Nakon pokretanja aplikacije, bit će prikazan prozor u okviru koga je moguće podesiti parametre konekcije (slika 1). Voditi računa da se odabere **Serial**, te odgovarajući serijski port i brzina, u skladu sa podešenjima uređaja sa kojim je potrebno ostvariti komunikaciju. Nakon klika na dugme <Open> će biti prikazan prozor serijskog terminala, kao na slici 2.

Putem serijskog terminala se sada mogu slati ASCII karakteri, a u prozoru će biti prikazan odgovor koji pošalje sistem sa kojim se ostvaruje komunikacija.

Treba voditi računa da uređaji kao kraj komande očekuje par ASCII znakova <cr><lf> (ne samo <lf>), te je potrebno ili adekvatno podesiti *Putty*, ili ova dva znaka generirati pritiskom na kombinacije tastera <Ctrl>+<M>, te nakon toga <Ctrl>+<J>.



Slika 1: Izgled prozora za podešavanje parametara konekcije.



Slika 2: Izgled prozora serijskog terminala.

Korištenje MQTT protokola

MQTT protokol predstavlja jedan od najčešće korištenih protokola za prenos podataka u IoT sistemima danas. U ovoj vježbi će biti demonstrirano korištenje MQTT protokola za rad sa digitalnim ulazima, digitalnim izlazima, analognim ulazima i PWM izlazima na virtualnom sistemu u Mbed simulatoru, kao i na razvojnom sistemu picoETF. Kao klijentske aplikacije će se koristiti:

- MQTT-X - MQTT klijent za različite operativne sisteme,

- MQTT Dash - aplikacija za Android, koja omogućava kreiranje jednostavnih kontrolnih ploča za slanje i prijem MQTT poruka.

Radi toga je potrebno instalirati aplikaciju MQTT-X, te instalirati sa Google Play-a aplikaciju MQTT Dash. Za iPhone se mogu koristiti slične aplikacije (npr. MQTT Buddy ili MQTTTool).

O MQTT protokolu je bilo riječi u okviru predavanja [3].

MQTT broker

Za potrebe laboratorijske vježbe moguće je koristiti Mosquitto MQTT broker instaliran na ETF, a podaci za pristup su:

- adresa: 195.130.59.209
- username: ugradbeni
- password: laboratorija

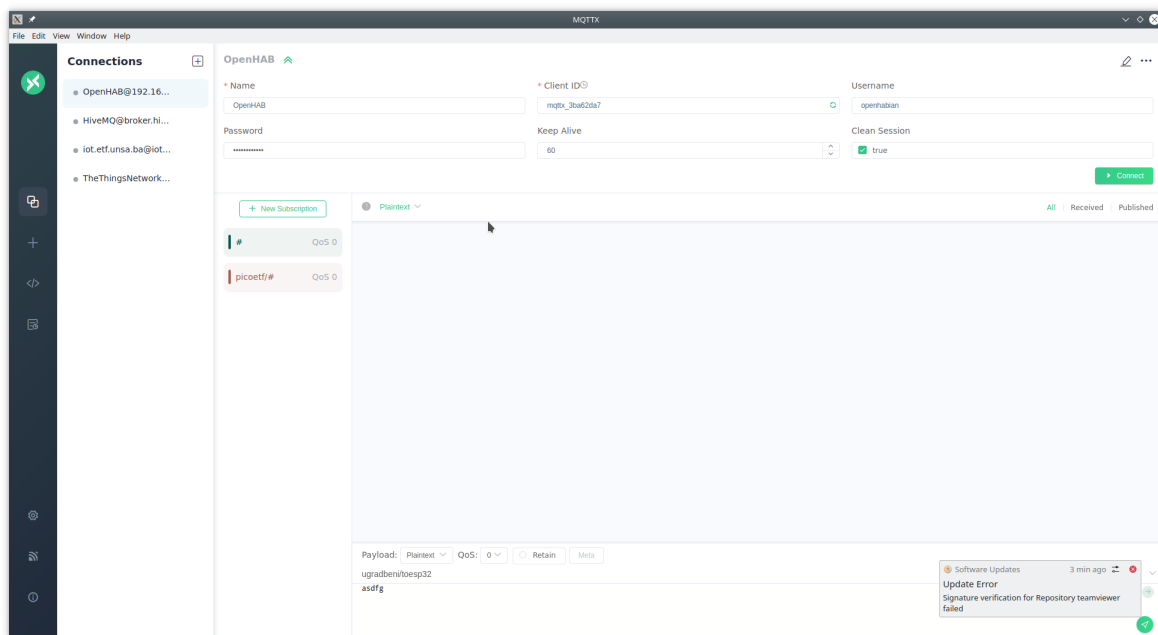
Moguće je koristiti i javni MQTT broker, kao npr. *HiveMQ*:

- adresa: broker.hivemq.com
- username: nema
- password: nema

MQTT-X

Aplikacija *MQTT-X* se može instalirati sa linka <https://mqttx.app/>.

Nakon pokretanja aplikacije *MQTT-X* je potrebno podesiti parametre konekcije (slika 3). Klikom na simbol <+> se omogućava dodavanje nove konekcije, a klikom na dugme <Connect> se uspostavlja konekcija sa MQTT brokerom.

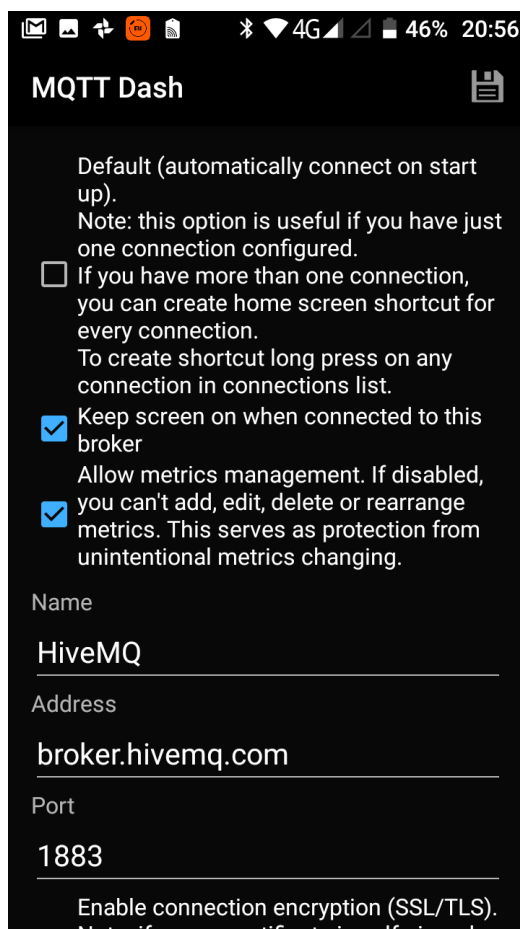


Slika 3: Podešavanje konekcije na MQTT broker u okviru *MQTT-X*.

MQTT Dash

MQTT Dash predstavlja aplikaciju za Android uređaje, koja se može instalirati sa Google Play Store-a. Za iPhone se mogu koristiti slične aplikacije (npr. MQTT Buddy ili MQTTTool).

Nakon instaliranja i pokretanja aplikacije *MQTT Dash*, potrebno je podesiti konekciju na MQTT broker. Klikom na simbol + će se pojaviti ekran za unos parametara konekcije, kao na slici 4.



Slika 4: Podešavanje konekcije na MQTT broker u okviru *MQTT Dash*.

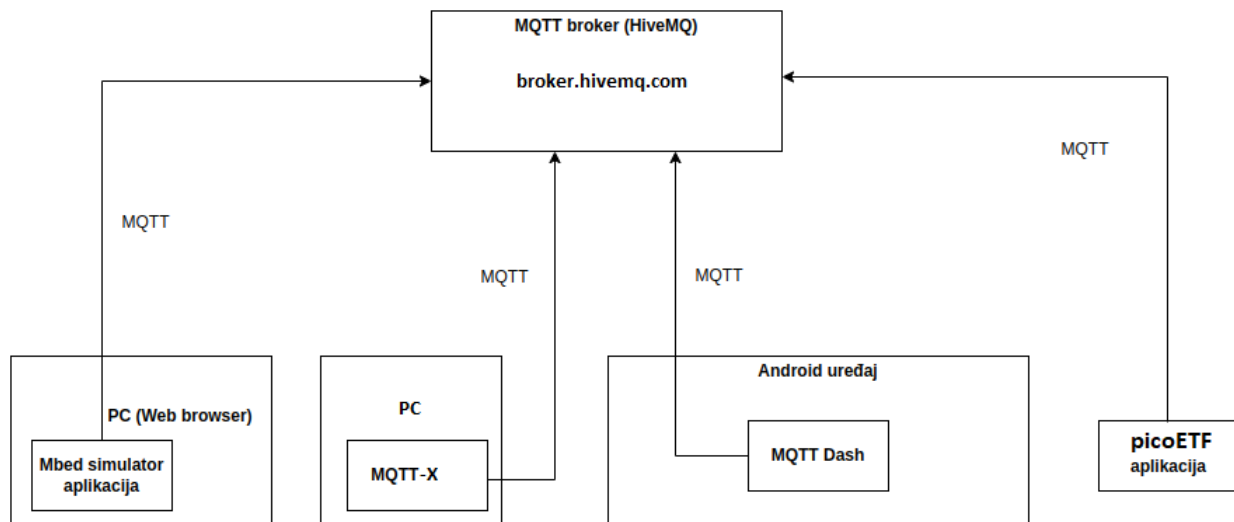
Nakon klika na simbol za snimanje konfiguracije i klika na konekciju HiveMQ, pojaviće se prazan prozor. U okviru ovog prozora se mogu dodavati grafičke kontrole za prikaz i zadavanje vrijednosti.

Pojašnjenje načina rada cjelokupnog sistema

Izgled cjelokupnog sistema je predstavljen na slici 5.

Aplikacija koja se izvršava na Mbed simulatoru, aplikacija koja se izvršava na razvojnom sistemu picoETF, *MQTT-X* i *MQTT Dash* predstavljaju klijente, koji će međusobno razmjenjivati poruke. Te poruke će omogućiti da se na virtualnom sistemu u Mbed simulatoru, kao i na razvojnom sistemu picoETF uključuju i isključuju LED diode (primjer upravljanja digitalnim izlazima), te postavlja intenzitet svjetlosti LED diode (primjer upravljanja PWM izlazom). Poruke koje će postavljati vrijednosti ovih izlaza će se generirati drugim klijentskim aplikacijama (*MQTT-X*, i/ili *MQTT Dash*). Isto tako, djelovanje na potencijometar u okviru Mbed simulatora i preko razvojnog sistema picoETF će rezultirati slanjem poruke koju će onda moći procesirati druga dva klijenta (primjer korištenja analognog ulaza), odnosno pritisak i otpuštanje tastera će rezultirati slanjem poruke na koju će reagirati druga dva klijenta

(primjer korištenja digitalnog ulaza).



Slika 5: Struktura cjelokupnog sistema.

Osnove MQTT protokola su predstavljene na predavanju, te je rečeno da MQTT klijenti međusobno razmjenjuju poruke posredstvom MQTT brokera. Radi toga je na svakom klijentu potrebno podesiti konekciju na MQTT broker, kako je prethodno opisano. Za potrebe ove vježbe će se radi jednostavnosti koristiti MQTT broker HiveMQ, za koji nije potrebno posjedovati korisničko ime i lozinku. Konekcija virtualnog sistema na MQTT broker je već podešena u okviru aplikacije koja će se izvršavati na Mbed simulatoru.

Rad sa WiFi i MQTT u MicroPython-u

MicroPython posjeduje pakete za korištenje WiFi modula, te za korištenje MQTT protokola, kako je već predstavljeno na predavanju.

Primjer koda koji omogućava povezivanje na WiFi je dat u dokumentaciji na linku https://docs.micropython.org/en/latest/esp8266/tutorial/network_basics.html?highlight=wifi, a primjer kôda koji omogućava povezivanje sa MQTT brokerom, pretplaćivanje na neku temu, kao i slanje poruke na temu su dati u nastavku. Dodatne informacije o korištenju ovih paketa se mogu pronaći na internetu.

```

# MQTT klijent
mqtt_conn = MQTTClient(client_id="JEDINSTVENI_ID", server=
    "ADRESA MQTT BROKERA",user="USERNAME",password="LOZINKA",
    port=1883)

# Funkcija koja se izvršava na prijem poruke
mqtt_conn.set_callback(sub)

# Povezivanje na MQTT broker
mqtt_conn.connect()

# Pretplata na temu
mqtt_conn.subscribe(TEMA)

# Čekanje na prijem poruke
mqtt_conn.wait_msg()

# Slanje MQTT poruke
mqtt_conn.publish(TEMA,PORUKA)

```

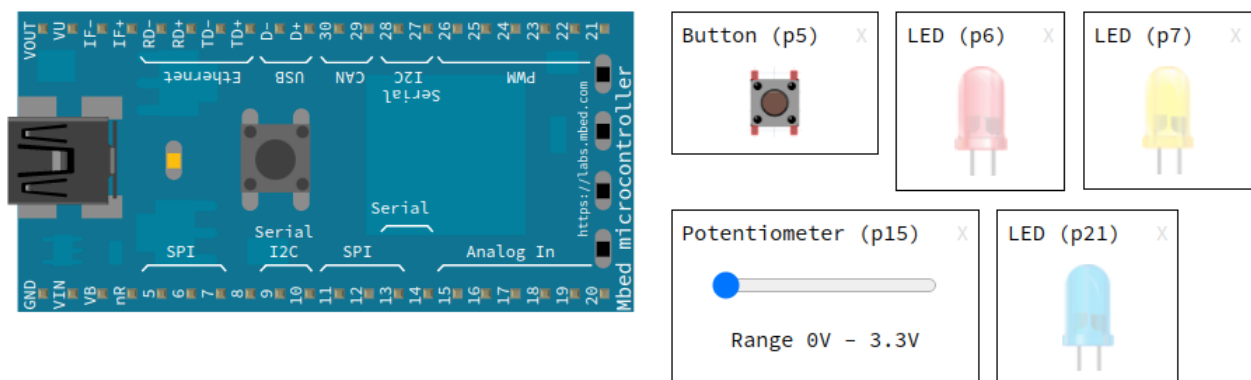
ZADACI

Zadatak 1

Mbed simulator

Dodavanje komponenti virtualnom sistemu

U Mbed simulatoru virtualnom sistemu dodati komponente kao na slici 6.



Slika 6: Izgled Mbed simulatora za zadatak 1.

LED na pinu p6 i LED na pinu p7 će biti korištene kao digitalni izlazi kojima se upravlja slanjem poruka sa klijenata *-X* i *MQTT Dash*. Intenzitet svjetla LED na pinu p21 će se postavljati putem PWM izlaza, a duty cycle će se postavljati klijentima *MQTT-X* i *MQTT Dash*. Klik na taster na pinu p5 će generirati poruku za druga dva klijenta. Treba napomenuti

da će pritisnuti taster rezultirati slanjem poruke sa vrijednošću 1, a otpušten taster slanjem poruke sa vrijednošću 0.

Hint

Da bi se na browseru simulirao pritisak i držanje tastera, potrebno je kliknuti na dugme, te držeći pritisnuto dugme, pointer miša izmaknuti sa tastera. Nakon toga će taster ostati u stanju 1. Za otpuštanje tastera je potrebno kliknuti izvan područja tastera, te držeći pritisnuto dugme, pointer miša pozicionirati na područje tastera i otpustiti dugme, nakon čega će taster preći u stanje 0.

Postavljanje vrijednosti potencijometra će rezultirati slanjem poruke koja sadrži postavljenu vrijednost.

Aplikacija za virtualni sistem u Mbed simulatoru

U Mbed simulator kopirati kôd koji se nalazi u fajlu <https://c2.etf.unsa.ba/mod/resource/view.php?id=89169>. Kako bi se izbjeglo da studenti jedni drugima upravljaju aplikacijama, potrebno je u okviru TEMASUBLED1, TEMASUBLED2, TEMASUBLED3, TEMAPUBPOT i TEMAPUBTAST promijeniti riječ “ugradbeni” u proizvoljni string:

```
#define TEMASUBLED1 "mojproizvoljnistring/led1"
#define TEMASUBLED2 "mojproizvoljnistring/led2"
#define TEMASUBLED3 "mojproizvoljnistring/led3"
#define TEMAPUBPOT "mojproizvoljnistring/potenciometar"
#define TEMAPUBTAST "mojproizvoljnistring/taster"
```

Isto tako, potrebno je u kodu postaviti jedinstveno ime klijenta u linijama `#define MQTT_CLIENT_NAME "MBED_SIMULATOR"` i `#define MQTT_CLIENT_NAME "FRDM_KL25Z"`. Razlog je to što MQTT broker zahtijeva da svaki klijent koji je povezan sa brokerom ima jedinstveno ime, te bi samo jedan od klijenata sa definiranim imenom mogao pristupiti klijentu u jednom trenutku.

Kliknuti na dugme Run, nakon čega će aplikacija biti kompajlirana, pokrenuta, te će se ostvariti konekcija sa MQTT brokerom.

Interakcija sa klijentom MQTT-X

Prikaz vrijednosti potencijometra

U okviru klijenta *MQTT-X*, u polje Subscribe upisati `mojproizvoljnistring/potenciometar` i kliknuti na dugme SUBSCRIBE. Time se klijent *MQTT-X* prijavio na pomenutu temu i sve poruke na tu temu će biti prikazane ispod Subscriptions.

Postaviti proizvoljnu vrijednost potencijometra i pročitati postavljenu vrijednost u okviru Topic: `mojproizvoljnistring/potenciometar`, gdje bi se trebao pojaviti JSON formatirani string kao npr.:

```
{
  "Potenciometar" : 0.45532
}
```

Svaka promjena vrijednosti potenciometra će rezultirati odgovarajućom porukom, koja će biti prikazana u *MQTTLens* klijentu.

Prikaz stanja tastera

U polje **Subscribe** upisati `mojproizvoljnistring/taster` i kliknuti na dugme **SUBSCRIBE**. Time će se klijent *MQTT-X* prijaviti na ovu temu, s tim da će ostati prijavljen i na prethodnu temu (sve dok se prijava na temu ne obriše). Na gore opisani način pritisnuti taster i pročitati stanje tastera u okviru **Topic**: `mojproizvoljnistring/taster`, gdje bi se trebao pojaviti JSON formatirani string, kao npr.:

```
{  
  "Taster" : 1  
}
```

Svaka promjena stanja tastera bi trebala rezultirati odgovarajućom porukom, čiji se sadržaj može vidjeti u klijentu *MQTT-X*.

Prikaz stanja LED dioda

U okviru klijenta *MQTT-X*, u polje **Publish** upisati `mojproizvoljnistring/led1` (slično za `led2`), a u polje **Message** tekst 1 (za uključivanje) ili 0 (za isključivanje), te kliknuti na dugme **PUBLISH**. U okviru Mbed simulatora bi se trebalo promijeniti stanje odgovarajuće LED. Svakim novim slanjem poruke će se stanje odgovarajuće diode promijeniti.

Postavljanje intenziteta osvjetljenja LED diode

U okviru klijenta *MQTTLens*, u polje **Publish** upisati `mojproizvoljnistring/led3`, a u polje **Message** vrijednost između 0.0 i 1.0, te kliknuti na dugme **PUBLISH**. U okviru Mbed simulatora bi se intenzitet osvjetljenja plave LED diode trebao postaviti u skladu sa poslanim Duty Cycle-om. Svaka nova poslana vrijednost će promijeniti intenzitet osvjetljenja plave LED diode.

Interakcija sa klijentom MQTT Dash

Za korištenje klijenta *MQTT Dash* je potrebno dodati i konfigurirati odgovarajuće grafičke kontrole, kao što je prikazano na slici 8.

Grafičke kontrole se dodaju klikom na dugme "+" u gornjem desnom dijelu prozora, a podešavanje svake od kontrola je prikazano na slikama 1.

Nakon što se sve grafičke kontrole konfiguriraju i snime, vrijednosti grafičkih kontrola "Potenciometar" i "Taster" će početi prikazivati one vrijednosti koje su zadane putem virtualnog sistema u Mbed s imulatoru. Isto tako, klik na grafičku kontrolu LED 1 i LED 2 će izazvati promjenu stanja odgovarajuće LED diode u okviru Mbed simulatora, dok će vrijednost zadana pomoću kontrole PWM LED postaviti intenzitet osvjetljenja plave LED u Mbed simulatoru.

Napomena

Zadatak 1 je moguće u potpunosti implementirati kod kuće, te je na laboratorijskoj vježbi potrebno samo demonstrirati isti.

Zadatak 2

picoETF

Razvojni sistem picoETF je potrebno USB kablom povezati sa računarom na uobičajeni način. Napisati kôd u MicroPython-u koji omogućava sljedeće:

- povezivanje sistema picoETF na WiFi mrežu u laboratoriji (mreža: Lab220, šifra: lab220lozinka)
- povezivanje na MQTT broker (broker.hivemq.com),
- pretplaćivanje na teme iz zadatka 1,
- slanje poruka na teme iz zadatka 1,

Na identičan način kao i sa virtualnim sistemom u Mbed simulatoru testirati razmjenu poruka sa klijentima *MQTT-X* i *MQTT Dash*.

Hint

Potrebno je povezati potencijometar sa razvojnim sistemom picoETF, kako bi se moglo testirati slanje podataka sa razvojnog sistema.

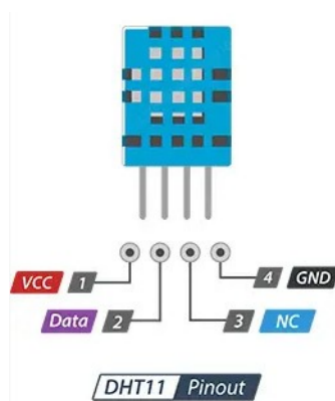
Zadatak 3

Potrebno je sljedeće:

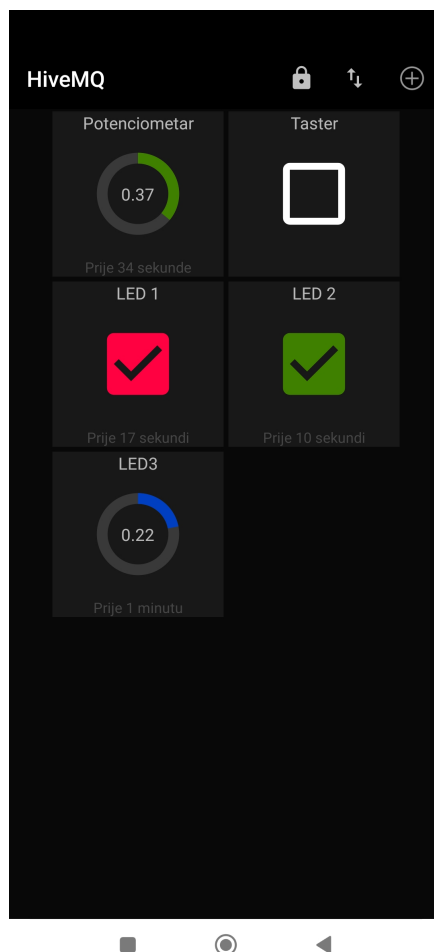
- Modificirati kôd za razvojni sistem picoETF, tako da se omogući postavljanje proizvoljne boje RGB led diode na sistemu picoETF.
- Na razvojni sistem picoETF dodati senzor za mjerenje temperature i vlažnosti DHT11, te napisati kod koji omogućuje slanje podataka o trenutnoj temperaturi i vlazi na odgovarajuću temu. Koristeći MQTT-X i MQTT Dash pretplatiti se na definisanu temu kako bi pratili mjerenja sa senzora. Mjerenja sa senzora slati svakih 2s.

Senzor DHT11

Senzor DHT11 (slika 7) omogućava mjerenje temperature u rasponu od 0 do 50 °C i vlažnosti u rasponu od 5% do 95%. DHT11 se sa picoETF povezuje tzv. Single Wire komunikacijom, te je potrebno povezati Data liniju sa nekim od raspoloživih pinova na picoETF (npr. GP28). Modul se napaja naponom od 3 do 5V, pa ga je najjednostavnije direktno povezati na pinove 3.3V i GND na razvojnem sistemu picoETF.



Slika 7: Senzor temperature i vlažnosti DHT11.



Slika 8: Izgled dashboard-a za zadatak 2 u klijentu *MQTT Dash*.

Literatura

- [1] S. Konjicija, E. Sokić (2019) *Ugradbeni sistemi: Hardverski aspekti, Elektrotehnički fakultet Univerziteta u Sarajevu, ISBN 978-9958-629-77-8*
- [2] S. Konjicija (2024) *Predavanje Ugradbeni sistemi: Komunikacija (1. dio)*
- [3] S. Konjicija (2024) *Predavanje Ugradbeni sistemi: Komunikacija (2. dio)*
- [4] ARM Holdings (2022) *Mbed OS API Documentation*
- [5] MicroPython projekat *Dokumentacija za MicroPython za RP2040*

21:53 74%

MQTT Dash

Name
Potenciometar

Topic (sub)
ugradbeni/potenciometar

Extract from JSON path (if payload is in JSON format), e.g.: \$.level.value. JSON path documentation at the URL below:
<https://github.com/jayway/JsonPath/blob/master/README.md>

\$.Potenciometar


☐ Enable publishing

Min 0.0 Max 1.0

Prefix _____ Postfix _____

Precision 2

☒ Display payload value instead of percentage

Progress color


Other settings
☒ QoS(0)
☐ QoS(1)
☐ QoS(2)

Blink tile to draw attention, if the expression

21:53 74%

MQTT Dash

Name
Taster

Topic (sub)
ugradbeni/taster



Extract from JSON path (if payload is in JSON format), e.g.: \$.level.value. JSON path documentation at the URL below:
<https://github.com/jayway/JsonPath/blob/master/README.md>



\$.taster

☐ Enable publishing

Payload and icons. If you need not a switch, but a simple button, just set the same payload values and the same icons for On and Off. This way the switch will never change icon and always send the same payload value.

On 1 Off 0

Other settings
☒ QoS(0)
☐ QoS(1)
☐ QoS(2)

Blink tile to draw attention, if the expression evaluates to 'true'.

21:54 74%

MQTT Dash

Name
LED 1

Topic (sub)
ugradbeni/led1

Extract from JSON path (if payload is in JSON format), e.g.: \$.level.value. JSON path documentation at the URL below:
<https://github.com/jayway/JsonPath/blob/master/README.md>



☒ Enable publishing



Topic (pub) - keep empty if the same as sub

☒ Update metric on publish immediately (do not wait for incoming message to update visual state)

Payload and icons. If you need not a switch, but a simple button, just set the same payload values and the same icons for On and Off. This way the switch will never change icon and always send the same payload value.

On 1 Off 0

Other settings

21:54 74%

MQTT Dash

Name
LED 2

Topic (sub)
ugradbeni/led2

Extract from JSON path (if payload is in JSON format), e.g.: \$.level.value. JSON path documentation at the URL below:
<https://github.com/jayway/JsonPath/blob/master/README.md>



☒ Enable publishing



Topic (pub) - keep empty if the same as sub

☒ Update metric on publish immediately (do not wait for incoming message to update visual state)

Payload and icons. If you need not a switch, but a simple button, just set the same payload values and the same icons for On and Off. This way the switch will never change icon and always send the same payload value.

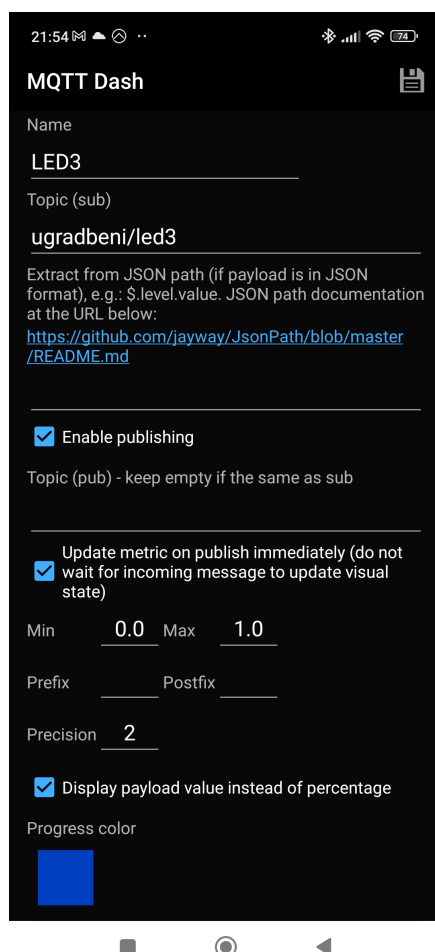
On 1 Off 0

Other settings

Tablica 1: Podešenja kontrola za zadatak 2.



Tablica 2: Podešenja kontrola za zadatak 2.