Dokumentacija implementacije

Za potrebe projekta korištena su dva mikrokontrolera:

- Raspberry Pi Pico (spojena 2 RFID čitača, matrična tastatura i LCD displej)
- ESP8266 (spojen servomotor za rampu).

Planirane funkcionalnosti smo implementirali na sljedeći način:

1) Uvezivanje kartice s brojem sobe na recepciji

    a. Unos broja sobe

    Unos se započinje pritiskom na taster `button`. Za unos broja sobe se koristi matrična tastatura, a prilikom unosa informativne poruke se prikazuju na LCD displeju. Za potrebe obrade unosa se koriste klase `Tastatura`, pomoću koje se bilježi unos, te `TastController`, koji sadrži instancu klase `Tastatura` i vraća odgovarajuće vrijednosti u ovisnosti od validnosti unosa.

    Očitavanje unosa s tastature je realizovano na sljedeći način:

    - Zasebni redovi (u kodu `rows`, deklarisani kao digitalni izlazi) se pomoću Timer-a periodično pale. Trenutno aktivni red (sačuvan u varijabli `currentRow`) se tako mijenja svakih 50ms. Timer se inicijalizira pritiskom na gore navedeni `button`, u sklopu funkcije `ocitaj()`:

      ```
      self.timer=Timer(period=50, mode=Timer.PERIODIC, callback=self.rowFun)
      ```

    - Na pin-ove za kolone (u kodu `cols`, deklarisane kao digitalni ulazi) su postavljeni hardverski interrupt-i, čija je callback funkcija `colPress()`:

      ```
      for i in range(4):
          self.cols[i].irq(handler = self.colPress, trigger = Pin.IRQ_RISING)
      ```

    - U okviru iste, nakon debouncing-a, se u varijablu `currentCol` pohranjuje broj pritisnute kolone.

    Zatim se na osnovu vrijednosti `currentRow` i `currentCol` određuje pritisnuti taster, te akcija koju je potrebno izvršiti. Kao signalizacija neispravnog unosa ili potvrde se koriste bool varijable `pogresanUnos` i `potvrdjeno`. Unos se obrađuje na sljedeći način:

- Za pritisnute tastere 'A', 'B', 'D' i '*' se unos završava deinicijalizacijom `timer`-a i varijabla `pogresanUnos` postavlja na True.

- Pritiskom na taster '#' se potvrđuje četverocifreni unos te varijabla `potvrdjeno` postavlja na True. U slučaju da je '#' pritisnut prije nego što su unesene 4 cifre, unos nije validan te se ponovno `pogresanUnos` postavlja na True. U oba slučaja se unos završava deinicijalizacijom `timer`-a.

- Ukoliko je pritisnut taster 'C', briše se posljednji uneseni karakter.

- Tasteri s ciframa od '0' do '9' se smatraju validnim unosom, izuzev u situaciji kada se dotadašnji unos već sastoji od 4 cifre, jer se prelazi maksimalna dozvoljena veličina.

Ukoliko je unos ispravan, cifra se dodaje na `string`

```
self.string += self.matrica[self.currentRow][self.currentCol]
```

koji se ispisuje na displej

```
self.lcd.clear()
self.lcd.putstr(self.string).
```

U okviru metode `getUnos()` klase `TastController`, a koja se koristi za dobivanje unosa s tastature, se vraćaju odgovarajuće vrijednosti. Povratna vrijednost -1 označava neispravan unos (u slučaju da je `pogresanUnos` postavljen na True), a uneseni `string` označava ispravan unos (u slučaju da je `potvrdjeno` postavljeno na True).

b. Uvezivanje unesene sobe i RFID kartice prislonjene čitaču

Nakon što je završen unos pomoću matrične tastature, mogu nastupiti dva slučaja. Ukoliko je unos neispravan, pomoću LCD displeja se korisniku signalizira greška, te on ima opciju da ponovno započne unos pritiskom na `button`. U slučaju kada je unos broja sobe ispravan prelazi se na konfiguraciju, tj. uvezivanje unesene sobe s RFID karticom kojom se želi omogućiti ulazak u nju. Najprije se korisniku na displeju ispisuje poruka da RFID čitaču `reader1` prinese karticu koja se želi uvezati sa sobom, te se očitava njegovo stanje.

```
(stat1, tag_type1) = reader1.request(reader1.REQIDL)
```

Kada se prinese kartica koja se želi konfigurisati, čita se njen ID

```
(stat, uid) = reader1.SelectTagSN()
card = int.from_bytes(bytes(uid),"little", False)
```

i u mapi `karticeISobe` se unesenoj sobi pridružuje vrijednost prislonjene kartice

```
karticeISobe[int(soba)] = card.
```

2) Pristup sobi pomoću kartice i RFID čitača

Čitanjem vrijednosti iz mape `karticeISobe` pri pokušaju ulaska u sobe se vrši provjera da li je dozvoljen pristup prislonjenom karticom. Ova funkcionalnost je demonstrirana pomoću RFID čitača `reader2`, koji predstavlja čitač za sobu 1000. Naime, nakon što se na recepciji na prethodno opisani način uveže soba 1000 s određenom karticom čiji se ID pohranjuje u mapu, prilikom prislanjanja kartice čitaču `reader2`, njen ID se poredi s vrijednošću u mapi.

```
if karticeISobe[1000] == card
```

Ukoliko su ove vrijednosti jednake, paljenjem zelene LED se signalizira da je kartica ispravna i njom se može pristupiti sobi. U suprotnom, paljenjem crvene LED se signalizira da prislonjena kartica nije povezana sa sobom kojom se pristupa, te ulazak u nju nije moguć.

3) Povezivanje i kontrola parking – rampe

Kontrola parking rampe je implementirana koristeći MQTT komunikacijski protokol. Na uređaju ESP8266 je inicijalizirana konekcija na WiFi koristeći SSID i password lokalne WiFi mreže, nakon čega se MQTT klijent može uspješno povezati na cloud brokera. Istovremeno je na kontrolnom uređaju (Android smartphone) podešena „MQTT Dash“ aplikacija sa podešenim opcijama za sve funkcionalnosti sistema. Moguće je jednim tasterom aktivirati rampu, te imati uvid u trenutno stanje sistema i konekcije, kao i stanje semafora (zeleno i crveno svjetlo).

Pozivom metode

```
client.subscribe("USprojekat/Servo", [](const String & payload)
```

podešava se supskripcija na temu „USprojekat/Servo“ koja se koristi za kontrolu servo motora, odnosno same rampe. Sve ostale teme se koriste samo za publishing statusnih poruka o sistemu, tako da za njih nije ni potrebna supskripcija.

Lambda funkcija koja je proslijeđena kao parametar metodi `subscribe()` se bavi svim funkcijama vezanim za kontroliranje same rampe. U slučaju da je sa korisničke strane na temi registrovana poruka „1“, za aktiviranje rampe, najprije se publish-aju poruke o stanju LED dioda na odgovarajuće teme, a zatim se i podiže sama rampa podešenjem ugla otklona servo motora.

```
int pos = 0;
      for(pos; pos <= 180; pos += 5)
      servo.write(pos);
```

Na isti način se rampa ponovno i spušta nakon čekanja od 10s da bi automobil mogao da prođe. Tim želi naglasiti da je kvalitetnije rješenje implementirati u sklopu rampe i ultrazvučni mjerač udaljenosti ili IR blaster i reciever sa suprotnih strana kako bi sistem „znao“ kada je automobil ispod rampe. Time bi se izbjeglo da rampa udari u automobil u slučaju da za 10s on ne prođe ispod iste. Nažalost, u sklopu ovog projekta nije bilo dovoljno resursa za takvu implementaciju.
Statusne poruke o stanju rampe se pravovremeno šalju na svoje odgovarajuće teme, a također se i na serijski port šalju iste poruke, kako bi se uz pomoć monitora serijske veze uvidjelo funkcioniranje mikrokontrolera i olakšalo eventualno servisiranje, odnosno debugging. Poruke koje se ispisuju su u formatu:

```
Serial.println("Podizem rampu...");
Serial.println("Rampa podignuta!");
Serial.println("Spustam rampu...");
Serial.println("Rampa spustena.");
```

Iste statusne poruke se šalju i na odgovarajuće teme metodom `publish()` kako bi se i na korisničkoj strani mogle vidjeti u okviru MQTT Dash aplikacije.