Univerzitet u Sarajevu
Elektrotehnički fakultet u Sarajevu
Odsjek za računarstvo i informatiku

Dokumentacija projekta
Kasa

Ugradbeni sistemi

Mentor:
prof. dr. Samim Konjiicija

Studenti:
Tarik Beganović
Vedad Grbo
Elvir Vlahovljak

Svako stanje kase ima svoju funkciju za koju je vezano. Te funckije glase:

```
    pocetno_stanje();
    kupovina_stanje();
    placanje_stanje();
    unos_stanje();
    brisanje_stanje();
```

Listing 1. Funkcije vezana za stanja kase

Kasa je u početku u početnom stanju, a na to stanje se uvijek moze vratiti zahvaljujući tasteru na mikrokontroleru: Svako stanje kase ima svoju funkciju za koju je vezano. Te funckije glase:

```
InterruptIn nazadNaPocetno(BUTTON1);
nazadNaPocetno.fall(&pocetno_stanje);
```

Listing 2. Vraćanje na početno stanje

Vraćanjem u početno stanje svi dotadašnji podaci o računu kupca se gube:

```
gasi_ledice();
trenutnoStanje=POCETNO;
iznosRacuna=0;
kolicinaArtikla = 1.f;
decimalnaKolicinaArtikla = 0.f;
skeniraniArtikal = Artikal();
```

Listing 3. Efekti vraćanja u početno stanje

Također, primarna uloga tih funkcija je da crtaju na ekran korisnički interfejs za to stanje te postavljanje varijabli na određene vrijednosti (npr. vraćanjem u početno stanje iznosRacuna se resetuje na nulu). U svako stanje se prelazi pritiskom na tastere zahvaljujući prekidima. Samim tim, ne postoji neka petlja čekanja na pritisak tastera. Primjer prekida u početnom stanju:

```
taster_p5.fall(&kupovina_stanje);
taster_p6.fall(&brisanje_stanje);
taster_p7.fall(&unos_stanje);
```

Listing 4. Prekidi u početnom stanju

Problem nastaje sa promjenom količine. Pošto se iz rutina za prekide ne mogu vršiti ”teške” funkcionalnosti, ne možemo crtati na ekran iz prekidnih rutina za promjenu količine. To je riješeno tako što u main-u provjeravamo stanje logičke varijable ”promijenjenaKoličina” koja se postavlja na ”true” iz prekidnih rutina. Kada se u main funkciji primijeti promjena te varijable na ”true”, na ekranu se ažurira količina, kao i cijena artikla sa zadanom količinom.

```
void povecaj_kolicinu() {
    if (trenutnoStanje != KUPOVINA) return;
    promijenjenaKolicina = true;
    kolicinaArtikla += 1.f;
}

if (promijenjenaKolicina) {
    kupovina_stanje();
    promijenjenaKolicina = false;
}
```

Listing 5. Promjena količine

Glavna logika kase smještena je u "**mqtt_**" funkcijama. Svaka od ovih funkcija ce izvršavati svoju funkcionalnost samo ako je kasa u odgovarajućem stanju (npr. u "bazu podataka" se neće dodati novi artikal ako kasa nije u stanju **UNOS**).

mqtt_stigao_skenirani_artikal() se poziva kada se preko mobilne aplikacije na broker pošalje barkod skeniranog artikla. Ta funkcija provjerava je li artikal dostupan u "bazi podataka" te, shodno postojanju u bazi, ažurira varijablu "skeniraniArtikal", ažurira ekran i trenutno stanje računa na odgovarajući način.

mqtt_stigao_novi_artikal() se poziva kada se preko mobilne aplikacije na broker pošalju podaci o novom artiklu. Podaci su u formatu "barkod,naziv,cijena". U zavisnosti postoji li artikal već u "bazi podataka", novi artikal će se spasiti u "bazu" i ekran će se ažurirati na odgovarajući način.

mqtt_stigao_barkod_za_brisanje() se poziva kada se preko mobilne aplikacije na broker pošalje barkod artikla koji želimo da se obriše. Pod uslovom postoji li taj artikal u bazi podataka, artikal će se obrisati i ekran će se ažurirati na odgovarajući način.

Funkcija **pali_ledice()** za svakih 20 KM na računu PWM-om pali po jednu ledicu. Za svaki iznos između $n \cdot 20$ i $(n+1) \cdot 20$ odgovarajućim duty-cycle-om pali sljedeću ledicu. Npr. neka je račun 30 KM. Pali se prva ledica i 50% duty-cycle druga ledica, jer $30 = 20 + 0.5 \cdot 20$.

Na kraju, main funkcija je jako jednostavna. Poziva se **pali_ledice()** funkcija i testira se je li bilo promjene količine artikla.

```
pali_ledice();
if (promijenjenaKolicina != 0) {
    kupovina_stanje();
    promijenjenaKolicina = 0;
};
```

Listing 6. Funkcija pali_ledice()