Univerzitet u Sarajevu
Elektrotehnički fakultet
**Ugradbeni sistemi 2023 / 24**

# Izvještaj za laboratorijsku vježbu br. 5

Analogni izlazi i širinsko – impulsna modulacija (PWM)

Ime i prezime: **Ivona Jozić**
Broj indeks-a: **19357**

**09. april 2024.**

# Sadržaj

# 1 Pseudokod i / ili dijagram toka

U pseudokodovima za laboratorijsku vježbu 5, data je ideja kako pristupiti rješavanju zadataka, a komplentna implementacija, kako zbog svoje kompleksnosti, tako i duzine, data je kao prilog u odjeljku 5 ovog izvještaja.

## 1.1 Zadatak 1

```
AnalogIn potenciometar
PwmOut led

led.podesi_period(vrijeme)

while(1)
      led.podesi_duty_cycle(ocitanje_potencimetra)
      sacekaj(0.1 sekundu)

end_while
```

## 1.2 Zadatak 2

```
ADC fotootpornik
Pwm leds

leds.podesi_frekvenciju()
leds.podesi_duty_cycle()

delta -> int(65535/8)

while(1)

      osvjetljenje -> skaliraj_napon(ocitaj_fotootpornik())

      if (osvjetljenje < 0):
            ugasi_leds()
            sleep(0.1)
      else if (osvjetljenje >= 0 and osvjetljenje < delta):
            upali_leds(0)
            sleep(0.1)
      else if (osvjetljenje >= delta and osvjetljenje < 2*delta):
            upali_leds(1)
            sleep(0.1)
      else if (osvjetljenje >= 2*delta and osvjetljenje < 3*delta):
            upali_leds(2)
            sleep(0.1)
      else if (osvjetljenje >= 3*delta and osvjetljenje < 4*delta):
            upali_leds(3)
            sleep(0.1)
      else if (osvjetljenje >= 4*delta and osvjetljenje < 5*delta):
            upali_leds(4)
            sleep(0.1)
      else if (osvjetljenje >= 5*delta and osvjetljenje < 6*delta):
            upali_leds(5)
            sleep(0.1)
      else if (osvjetljenje >= 6*delta and osvjetljenje < 7*delta):
```

```
            upali_leds(6)
            sleep(0.1)
      else if (osvjetljenje >= 7*delta and osvjetljenje < 8*delta):
            upali_leds(7)
            sleep(0.1)
      else :
            upali_leds(8)
            sleep(0.1)
      end_if
      sleep(0.1)

end_while
```

## 1.3 Zadatak 3

Na raspolaganju je ukupno 5 signala koje je moguće generisati, sinusni, uzlaznu i silaznu rampu, triangularni, te apsolutni sinus.

```
AnalogOut signal

while(1)
      generisi_signal()
end_while
```

# 2 Analiza programskog rješenja

## 2.1 Zadatak 1

U prvom zadatku je bilo potrebno spojiti potenciometar na LPC1114ETF razvojni sistem, a zatim na digitalnom izlazu LED1 realizirati PWM signal sa periodom T. Vrijednost duty-cycle je povezana sa stanjem potenciometra, odnosno u zavisnosti od položaja potenciometra reguliše se jačina svjetla na LED1. Maksimalno i minimalno svjetlo na LED1 predstavljaju krajnje položaje potencimetra. Rješenje je realizirano na način da je LED1 deklarisan kao PwmOut objekat sa dodijeljenim periodom, a pri svakoj iteraciji while petlje dolazi do ažuriranja njogovog duty-cycle u zavisnosti od očitanog stanja na potenciometru.

U zadatku je traženo da se uporede rezultati za periode T1=50μs i T2=500ms. Nakon testiranja dolazi se do zaključka da je za promjenu intenziteta svjetlosti pri podešenom periodu T1 potrebna manja promjena stanja potencimetra u odnosu na podešeni period T2.

## 2.2 Zadatak 2

Cilj drugog zadatka je bio spojiti fotootpornik na razvojni sistem picoETF, te realizirati VU metar na 8 LED, na način da pored toga što se diode pale u zavisnosti od jačine osvjetljenja, intenzitet njihovog svijetla je također promjenljiv. U slučaju da je osvjetljenje minimalno sve diode trebaju biti upaljene, a LED7 treba imati najjači intenzitet, koji je na svakoj prethodnoj diodi proporcionalno smanjen.

Rješenje je realizirano na način da su diode deklarisane kao PWM objekti i svakoj je podešena frekvencija, kao i duty-cycle u zavisnosti od njenog položaja u poretku LED dioda. Procedura *update_leds(n_leds)* služi za paljenje LED dioda i podešavanje njihovog intenziteta u zavisnosti od pada napona na fotootporniku (0.0 – 3.3V), a parametar koji prima je broj dioda koje trebaju biti upaljene. Namjena procedure *scaleVoltage(readVoltage)* je adaptacija vrijednosti napona na fotootporniku u zavisnosti od minimalne i maksimalne jačine svjetla koja se može pojaviti u prostoriji. Adaptaciju je moguće izvršiti tako što se prvo izmjere pad napona pri minimalnom i maksimalnom osvjetljenju, a zatim se formira linearna funkcija. U svakoj iteraciji while petlje se očitava vrijednost napona na fotootporniku, zatim se vrši skaliranje očitane vrijednosti, te se u zavisnosti od skalirane vrijednosti ažurira stanje LED dioda.

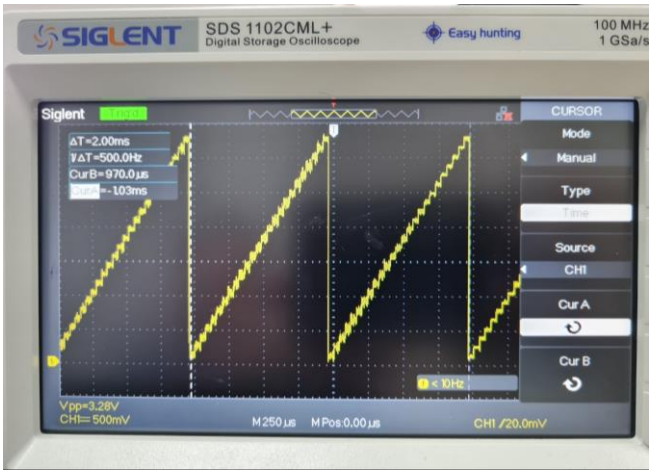
## 2.3 Zadatak 3

U trećem zadatku je bilo potrebno generisati zadane diskretne oblike signala na razvojnom sistemu FRDM-KL25Z i prikazati ih na osciloskopu uz zadanu frekvenciju (500Hz) koristeći objekat AnalogOut. Za vrijeme jednog perioda je potrebno realizirati 50 uzoraka, a vrijeme za jednu stepenicu je 40μs.

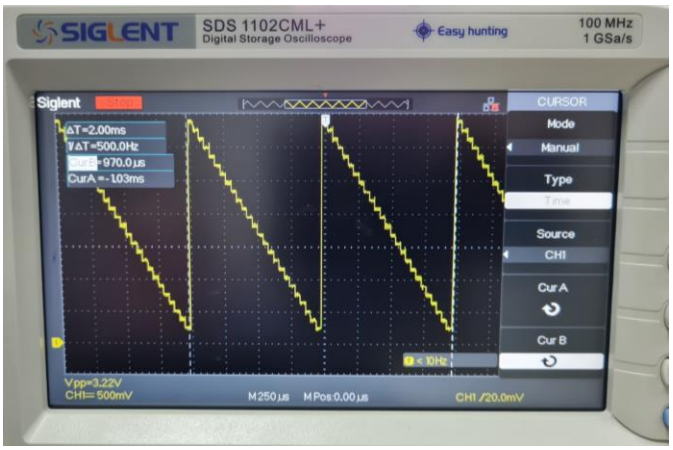
Rješenje je realizirano upotrebom objekta AnalogOut kojem je dodjeljivana vrijednost u zavisnosti od signala koji se želi generisati. Na raspolaganju je bilo 5 signala: sinusoida pomjerena za 0.5 po ordinati, uzlazna i silazna rampa, triangularni, te sinus apsolutni. Najveći izazov je bio podesiti odgovarajući wait() tako da se postigne tražena frekvencija.

Signali 2, 3 i 4 su ispravno generisani sa zahtijevanom frekvencijom od 500Hz. Vrijeme čekanja je drastično spušteno u odnosu na 40 μs jer se pri pokretanju programa u obzir treba uzeti i

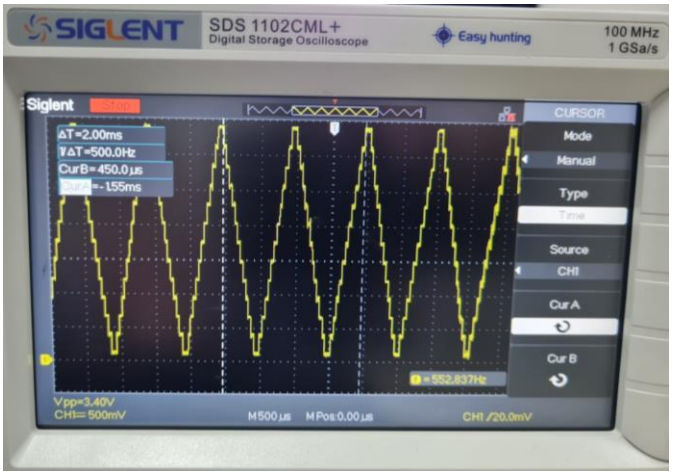
vrijeme potrebno za izvršavanje koda. Signali 1 i 5 se ispravno iscrtavaju, ali je zbog usporavanja programa stalnim pozivom funkcije za izračunavanje sinusa frekvencija neispravna. Moguće poboljšanje bi moglo biti implementirano na način da se vrijednosti izračunaju na početku i smjeste u niz, a da se prilikom generisanja signala samo čitaju vrijednosti iz tog niza jer bi na taj način samo jednom bilo utrošeno vrijeme potrebno za računanje vrijednosti.




Slika 1. – Prikaz signala 2 na osciloskopu, tzv. uzlazna rampa




Slika 2. – Prikaz signala 3 na osciloskopu, tzv. silazna rampa




Slika 3. – Prikaz signala 4 na osciloskopu, tzv. triangularni

# 3 Korišteni hardverski resursi

Za potrebe laboratorijske vježbe 5 korišteni su razvojni sistemi picoETF, LPC1114ETF i FRDM-KL25Z. Pored njih korištene su LED diode integrisane na razvojnim sistemima potenciometar, fotootpornik, te osciloskop.

Za sistem picoETF:

- 8 LED dioda

Za sistem LPC1114ETF:

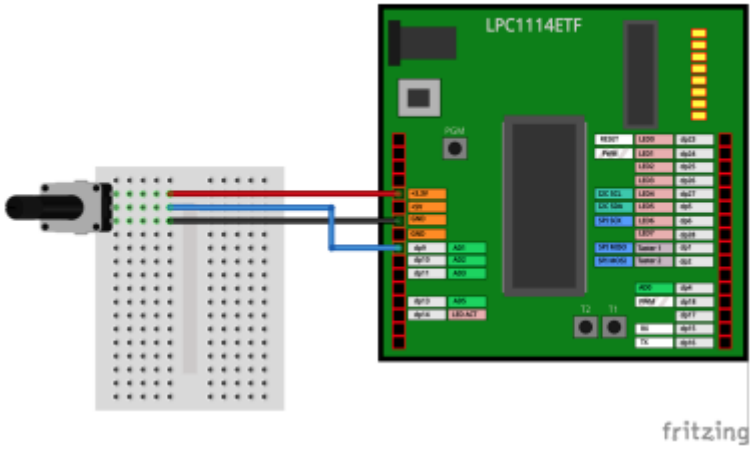
- 1 LED dioda

- rotacijski potenciometar

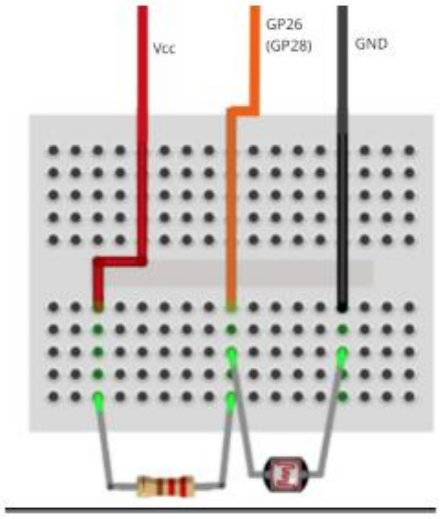
- fotootpornik

- osciloskop (korišten za ispitivanje analognog signala u zadatku 3, rađenom na FRDM-KL25Z)

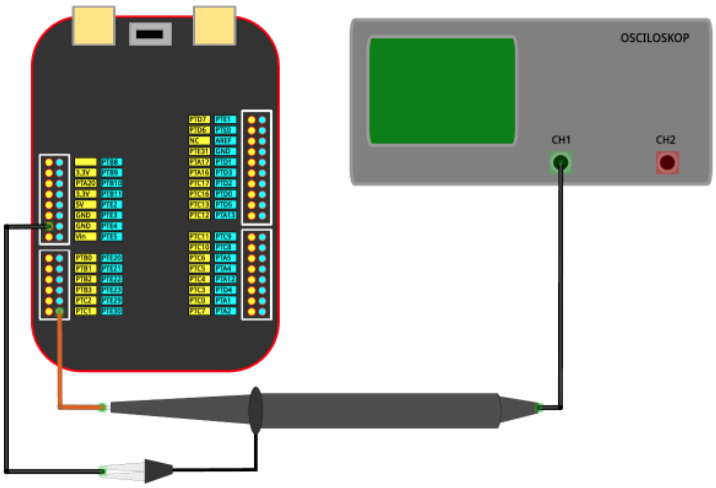
| ULAZI | IZLAZI |
|---|---|
| Potenciometar<br>(analogni; DP9 na LPC1114ETF) | LED1<br>(digitalni; LCP1114ETF) |
| Fotootpornik<br>(analogni, picoETF, GP28) | LED0 – LED7<br>(digitalni; picoETF GP4 – GP11) |




Slika 4. – Povezivanje potenciometra sa razvojnim sistemom LPC1114ETF (1. zadatak)




Slika 5. – Povezivanja fotootpornika sa razvojnim sistemom picoETF (2. zadatak)




Slika 6. – Povezivanje osciloskopa sa razvojnim sistemom FRDM-KL25Z (3. zadatak)

# 4 Zaključak

Prilikom izvođenja Laboratorijske vježbe 5 nije bilo poteškoća s obzirom da su sva tri razvojna sistema picoETF, FRDM-KL25Z i LPC1114ETF od ranije poznata. Također nije bilo problema sa povezivanjem osciloskopa, fotootpornika niti potenciometra. Jedina stvar na koju je bilo potrebno dodatno obratiti pažnju je skaliranje napona u 2. zadatku, te podešavanje wait() u 3. zadatku, kako bi oba rješenja radila u skladu sa očekivanjima. Cilj vježbe je bio upoznavanje sa analognim izlazima, te PWM-om, što je postignuto.

# 5 Prilog

U prilogu su dati kodovi koje se izvršavaju na razvojnim okruženjima LPC1114ETF (1. zadatak), picoETF (2. zadatak), te FRDM-KL25Z (3. zadatak). Date kodove, osim 3. zadataka zbog neadekvatnog simulatora za FRDM-KL25Z razvojni sistem, moguće je pokrenuti i u Mbed, odnosno Wokwi simulatoru u zavisnosti od korištenog razvojnog sistema, s tim da će za testiranje u Mbed simulatoru biti potrebno izmijeniti inicijalizaciju pinova.

## 5.1 Zadatak 1 / izvorni kod

Za ispravno izvršenje sljedećeg koda neophodno je upload-ati zaglavlje *lpc1114etf.h*.

```
#include "mbed.h"
#include "lpc1114etf.h"

AnalogIn pot(AD1); //promijeniti pin za simulator na p15
PwmOut led(LED1); //u simulatoru je p21
//za pokretanje u simulatoru potrebno dodati i LED diodu na p5

DigitalOut E(LED_ACT);

int main() {
    E=0;
    led.period_ms(500);
    //led.period_us(50);
    //kod perioda u mikros se intenzitet svjetla mijenja
    //sa manjim promjenama polozaja potenciometra

    while (1) {
        led.write(pot.read()); //podesen duty
        printf("%f \n",pot.read());

        wait_us(10000); //eventualno smanjiti
    }
}
```

## 5.2 Zadatak 2 / izvorni kod

```
from machine import Pin, PWM, ADC
from time import sleep
sleep(0.1) # Wait for USB

# Inicijalizacija fotootpornika
photoRes = ADC(Pin(28))

# Inicijalizacija LED dioda kao PWM izlaza
leds = [PWM(Pin(i)) for i in range(4, 12)]
```

```
# Podesavanje PWM frekvencije
for led in leds:
    led.freq(10000)

# 1/8 maksimalnog duty cycle
delta = int(65535/8)

# Postavljanje duty cycle vrijednosti za LED (gradient)
duty = [delta, 2*delta, 3*delta, 4*delta, 5*delta, 6*delta, 7*delta, 65535]
# Za mjerenje
print(duty)

# Paljenje ledica
def update_leds(n_leds):
  for i in range(0, 8):
    if i <= n_leds:
      leds[i].duty_u16(duty[i])
    else:
      leds[i].duty_u16(0)

# Skaliranje ulaznog napona
# Pad napona na fotootporniku nikada nije 0V, pa ga treba skalirati
# linearna transformacija
def scaleVoltage(readVoltage):
  k = 65535/(65535-40000)
  return int(k*readVoltage - 40000*k)

while True:
  # Očitavanje napona na otporniku, skaliranje i ispis za referencu
  lum = scaleVoltage(photoRes.read_u16())
  print(lum)

  # Zavisno od napona, pale se odredjene diode
  if lum < 0:
    update_leds(-1)
    sleep(0.1)
  elif lum >= 0 and lum < delta:
    update_leds(0)
    sleep(0.1)
  elif lum >= delta and lum < 2*delta:
    update_leds(1)
    sleep(0.1)
  elif lum >= 2*delta and lum < 3*delta:
    update_leds(2)
    sleep(0.1)
  elif lum >= 3*delta and lum < 4*delta:
    update_leds(3)
    sleep(0.1)
  elif lum >= 4*delta and lum < 5*delta:
```

```
    update_leds(4)
    sleep(0.1)
  elif lum >= 5*delta and lum < 6*delta:
    update_leds(5)
    sleep(0.1)
  elif lum >= 6*delta and lum <= 7*delta:
    update_leds(6)
    sleep(0.1)
  elif lum >= 7*delta and lum < 8*delta:
    update_leds(7)
    sleep(0.1)
  else :
    update_leds(8)
    sleep(0.1)

  sleep(0.1)
```

## 5.3 Zadatak 3 / izvorni kod

U zavisnosti od toga koji od signala se želi generisati potrebno je otkomentirati odgovarajuće dijelove programa u main-u.

(Pogledati odjeljak Analiza programskog rješenja za ovaj zadatak i ideju kako bi se rješenje moglo unaprijediti.)

```
#include "mbed.h"
#define PI 4*atan(1)

AnalogOut signal(PTE30);

//triangularni
void signal4() {
    for(float i = 1.0; i>0; i -= 1./12) {
        signal = i;
        wait_us(20);
    }
    for(float i = signal; i < 1; i += 1./12) {
        signal = i;
        wait_us(20);
    }
}
//sinusoida
void signal1() {
    for(float i = PI/6; i<=PI/2; i += float((PI/2-PI/6)/12)) {
        signal = sin(i);
        wait_us(20);
    }
    for(float i = PI/2; i<=PI; i += float(PI/50)) {
        signal = sin(i);
```

```
        wait_us(20);
    }
    for(float i = PI; i<7*PI/6; i += float(PI/66)) {
        signal = sin(i);
        wait_us(20);
    }
}
// |sin(x)|
void signal5() {
    for(float i = 0; i <= PI; i += float(PI/50)) {
        signal = sin(i);
        wait_ns(1); //podesiti odgovarajuci wait da se postigne zahtijevana frekvencija
    }
}

int main(){
    float i=1;
    const float incr=1./25;
    double fi=0;
    const float inc_sin=2*PI/50;
    while(true) {
        //SIGNAL 1 - sinusoida
        //signal1(); - kroz funkciju, a ispod je alternativa koja izbjegava poziv funkcije
        /*signal=sin(fi)/2+0.5;
        fi+=inc_sin;
        if(fi>=2*PI) fi=0;
        wait_ns(2); */ //podesiti odgovarajuci wait da se postigne zahtijevana frekvencija

        //SIGNAL 2 – uzlazna rampa
        //inicijalizirati i na 0 prije petlje
        /*signal = i;
        wait_ns(16000);
        i+=incr;
        if(i>1) i=0;*/

        //SIGNAL 3 – silazna rampa
        //inicijalizirati i na 1 prije petlje
        /*signal = i;
        wait_ns(15550);
        i-=incr;
        if(i<0) i=1;*/

        //SIGNAL 4 - triangularni
        //signal4();

        //SIGNAL 5 - |sin(x)|
        signal5();
    }
}
```