Univerzitet u Sarajevu
Elektrotehnički fakultet
Ugradbeni sistemi 2023 / 24

Izvještaj za laboratorijsku vježbu br. 3

Višebitni digitalni ulazi i izlazi

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

26. mart 2024.

Sadržaj

1 Pseudokod i / ili dijagram toka

1.1 Zadatak 1

```
leds[]
while(1)
      broj_na_tastaturi=ocitajUnos()
      if(broj_na_tastaturi >=1 and broj_na_tastaturi<=8)
            leds[broj_na_tastaturi].ukljuci()

      else if(broj_na_tastaturi == 'C')
            ugasi_sve_diode()
      end if
end while
```

1.2 Zadatak 2

```
brojac -> 0

while(1)
      if(taster1 pritisnut)
            brojac -> brojac + 1

      else if(taster2 pritisnut)
            brojac -> brojac - 1

      else if(taster3 pritisnut)
            brojac -> 0

      else if(taster4 pritisnut)
            aktiviraj_automatski_brojac()

      end if

      if(brojac_izvan_opsega)
            podesi_ispravan_opseg()
      end if

      prikazi_brojac_na_ekranu()

end while
```

1.3 Zadatak 3 – izbor 1

Podrazumijevamo da je period zadan u milisekundama, a pod varijablom taster_broj se smatra očitanje bilo kojeg broja u rasponu 0 do 9. Varijabla ocitani_broj odgovara brojčanoj vrijednosti pritisnutog tastera taster_broj.

```
while(1)
      ucitaj_unos_sa_tastature()

      if(tasterA pritisnut)
            ukljuci_signal()

            while(tasterA nije pritisnut)

                  ocitaj_unos_sa_tastature()

                  if(taster_broj pritisnut)
                        period -> ocitani_broj + 1

                  if(tasterC pritisnut)
                        period -> period + 1

                  else if(tasterD pritisnuti)
                        period -> period – 1

                  end if

            end while

            iskljuci_signal()

      end if

end while
```

2 Analiza programskog rješenja

2.1 Zadatak 1

U prvom zadatku je bilo potrebno implementiranje paljenja LED dioda na LPC1114ETF u zavisnosti od pritisnute cifre na matričnoj tastaturi. Na samom početku su sve diode ugašene, a nakon što se sa tastature očita pritisak nekog od tastera 1 - 8 potrebno je upaliti odgovarajuću LED diodu. Pritiskom na taster C na matričnoj tastaturi dolazi do gašenja svih LED dioda. Ovo rješenje je implementirtano korištenjem 3 niza čiji su elementi redom LED diode, redovi i kolone matrične tastature. Također, korišten je i objekat tipa BusOut kako bi se gašenje LED dioda pojednostavilo. U rješenju je implementirana funkcija koja očitava pritisak na neki od tastera na tastaturi na način da prolazi kroz redove i ispituje da li je neki od tastera pritisnut. U slučaju da jeste, iz matrice znakova na tastaturi se očitava odgovarajući char, na osnovu kojeg se procjenjuje koja radnja će uslijediti, to jeste da li će doći do paljenja neke od LED dioda, gašenja svih dioda ili će stanje ostati nepromijenjeno u slučaju pritiska na neodgovarajući taster.

2.2 Zadatak 2

Cilj drugog zadatka je bio implementacija brojača koji je upravljan pritiskom na neki od tastera na picoETF-u. U slučaju pritiska na taster1 brojač se inkrementira za 1, dok se pritiskom na taster2 on dekremetira za 1. Pritisak na taster3 treba izazvati resetovanje brojača na 0, a pritisak na taster4 treba uključiti automatsko inkrementiranje brojača za 1 svake sekunde sve do ponovnog pritiska na taster4. Stanje brojača se treba prikazivati na četverocifrenom 7-segmentnom displeju.

Kontrola 7 – segmentnog displeja izvedena je pomoću lookup tabele u kojoj su smještene vrijednosti segmenata za svaku cifru od 0 do 9, u binarnoj reprezentaciji. Pojedinačne cifre od 0 do 4 i segmenti cifara, te tasteri pohranjeni su u odgovarajuće nizove Pin objekata. Prikaz cifara na displeju je implementiran kroz niz pomoćnih funkcija, od kojih funkcija *prikazi_cifru(cifra)* služi za aktivaciju odgovarajućih segmenata A – G. Ona iz lookup tabele očitava binarnu reperezentaciju proslijeđenog broja u obliku upaljenih (0) i ugašenih (1) segmenta, a zatim elemente niza *segmenti[]* u for petlji postavlja na odgovarajuće vrijednosti. Funkcija *prikazi_poziciju(pozicija)* vrši odabir jedne od 4 cifre na display-u na osnovu prosljeđenog parametra. Na početku su sve četiri ciftre ugašenje, a zatim se pali ona koja odgovara proslijeđenom parametru. Funkcija *daj_cifru(broj, n)* korištena je kako bi se izvukla n-ta cifra proslijeđenog broja. Funkcija *prikazi_broj(broj)* koristi prethodne tri pomoćne funkcije kako bi prikazala broj na displeju. Kako je hardverski nemoguće da istovremenu budu prikazane sve četiri cifre na displeju, neophodno je osvježavati prikaz na displeju dovoljno brzo da bi promjene bile neuočljive ljudskom oku. Ova funkcija služi upravo za to, odnosno izdvaja po jednu cifru iz brojača i pali jednu po jednu cifru na displeju dovoljnom brzinom.

2.3 Zadatak 3 – izbor 1

U trećem zadatku je implementiran „generator četvrtki“ koristeći picoETF na koji su povezani 7-segmentni displej i matrična tastatura, te osciloskop koji očitava signal. U implementaciji je korišten PWM objekat za generisanje signala, to jeste za početak i prekid generisanja signala. Pritiskom na taster A na matričnoj tastaturi, potrebno je započeti generisanje signala koje će trajati sve do ponovnog pritiska na taster A. Na samom početku period generisanja iznosi 1ms,

dok se pritiskom na neki od tastera sa ciframa 0-9 on postavlja na za jedan veću milisekundu u odnosu na pritisnuti taster (npr. za taster 2 će iznositi 3ms). U slučaju pritiska na taster C period se uvećava za 1ms, dok se pritiskom na taster D umanjuje za 1ms. Na 7-segmentnom displeju se treba prikazivati trenutni period. Frekvencija oscilovanja je dobijena na način 1/period. Za kontrolu i očitavanje pritisnutih tastera na matričnoj tastaturi iskorištena je prezentirana logika u prvom zadatku, dok je za prikaz perioda na 7-segmentom displeju korištena logika objašnjena u drugom zadatku.

3 Korišteni hardverski resursi

Za potrebe laboratorijske vježbe 3 korišteni su razvojni sistemi LPC1114ETF i picoETF. Dodatno su od opreme korišteni:

- 8 LED dioda (integrisane na sistemu LPC1114ETF)

- 4 fizička tastera (integrisani na sistemu LPC1114ETF)

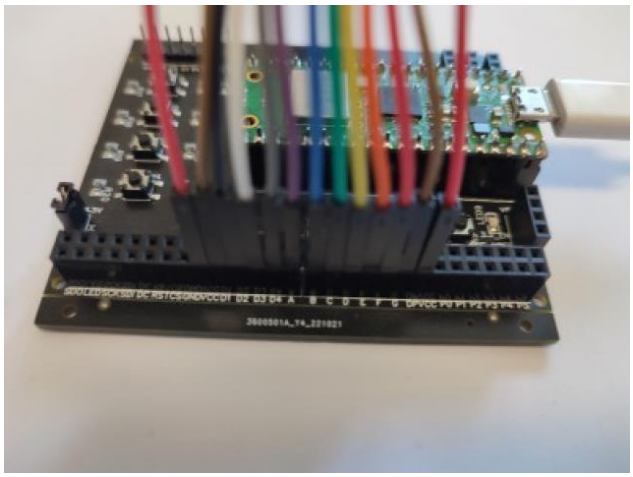
- četverocifreni 7-segmentni displej sa zajedničkom anodom

- matrična tastatura 4x4

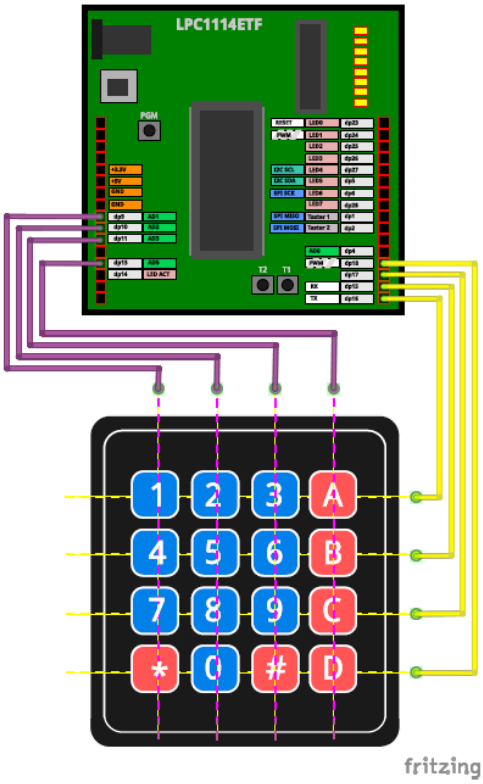
- osciloskop

ULAZI	**IZLAZI**
Tasteri 1 – 4 (GP0 – GP3 na picoETF)	LED (GP4 – GP11 na picoETF)
Kolone matrične tastature (DP9, DP10, DP11 i DP13 na LPC1114ETF; GP0-GP4 na picoETF)	Redovi matrične tastature (DP16, DP15, DP17, DP18 na LPC1114ETF; GP21, GP22, GP26, GP27 na picoETF)
	Segmenti display – a (GP8 – A, GP9-B, GP10-C, GP11-D, GP12-E, GP13-F, GP14-G, GP15-DP na picoETF)
	Cifre display – a (GP4-DIG1, GP5-DIG2, GP6-DIG3, GP7-DIG4 na picoETFu)
	PWM signal (GP16 na picoETF)

Signal prikazan na osciloskopu je square – wave PWM signal.



Slika 1. - način priključivanja 7 – segmentnog display-a na razvojni sistem LPC1114ETF



Slika 1. - shema spajanja matrične tastature na razvojni sistem LPC1114ETF

4 Zaključak

Prilikom izvođenja Laboratorijske vježbe 3 upoznali smo se na višebitnim digitalnim izlazima (7-segmentni displej) i višebitnim digitalnim ulazima (matrična tastatur). S obzirom da su rješenja zadataka pripremana u simulatoru Wokwi, bilo je neophodno prilagoditi način paljenja 7-segmentnog displeja kako bi se rješenje moglo testirati na picoETF-u. Pored toga bilo je potrebno usklađivanje neki od perioda sleep-a, ponovno zbog neusklađenosti simulatora i razvojnog sistema, ali su na kraju sva rješenja uspješno testirana.

5 Prilog

U prilogu su dati kodovi koje se izvršavaju na razvojnim okruženjima picoETF i LPC1114ETF (prvi zadatak). Vrijedi napomenuti da u zadacima 2 i 3 treba obratiti pažnja na način paljenja pojedinačnih cifara (pinovi 4 – 7) na 7 segmentnom displeju, s obzirom da se iste u Wokwi simulatoru pale logičkom 1, dok se na picoETF-u pale logičkom 0.

5.1 Zadatak 1 / izvorni kod

Zadatak 1 je pripremljen za pokretanje na razvojnom sistemu LPC1114ETF, te je programski kod u nastavku rađen u programskom jeziku C++.

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

DigitalOut leds[]={LED0, LED1, LED2, LED3, LED4, LED5, LED6, LED7};
BusOut leds_off(LED0, LED1, LED2, LED3, LED4, LED5, LED6, LED7);  //objekat kreiran zbog
lakseg gasenja LED dioda

char keypad[4][4]={
    {'1','2','3','A'},
    {'4','5','6','B'},
    {'7','8','9','C'},
    {'*','0','#','D'}};

DigitalOut rows[4]={dp16, dp15, dp17, dp18};
DigitalIn cols[4]={dp9, dp10, dp11, dp13};

DigitalOut E(LED_ACT);

// funkcija koja sluzi za ocitavanje pritisnutog tastera na matricnoj tastaturi
char readKeypad(){
    for(int i=0; i<4; i++){
        rows[i].write(1);

        for(int j=0; j<4; j++){
            if(cols[j].read()){
                rows[i].write(0);
                return keypad[i][j];
            }
        }
        rows[i].write(0);
    }
    return ' ';
}

int main(){
    E=0;
    leds_off=0;
```

```
    char button=' ';
    bool releassed=true;

    while (true){
        button=readKeypad(); //ocitavanje trenutno pritisnutog tastera
        if(button==' ') releassed=true; //indikator da su svi tasteri otpusteni

        if(button!=' ' && releassed){
            if(button>='1' && button<='8') leds[int(button)-'0'].write(1); //upali LED
            else if (button=='C') leds_off=0; //ugasi sve LED diode

            releassed=false; //ponisti indikator prije sljedeceg ocitavanja
        }
    }
}
```

5.2 Zadatak 2 / izvorni kod

Zadatak 2 je pripremljen za pokretanje na razvojnom sistemu picoETF, te je programski kod u nastavku rađen u programskom jeziku Micro Python.

```
from machine import Pin
import time
time.sleep(0.1) # Wait for USB to become ready

cifre = {
    0: 0b1000000,
    1: 0b1111001,
    2: 0b0100100,
    3: 0b0110000,
    4: 0b0011001,
    5: 0b0010010,
    6: 0b0000010,
    7: 0b1111000,
    8: 0b0000000,
    9: 0b0010000,
    10: 0b1111111
}

mjesto = [Pin(m, Pin.OUT) for m in range(4, 8)]
segmenti = [Pin(s, Pin.OUT) for s in range(8, 15)]
tasteri = [Pin(i, Pin.IN) for i in range(4)]

# prikazi cifru na jednom mjestu
def prikazi_cifru(cifra):
    binarno = cifre[cifra]
    for i in range(7):
        segmenti[i].value(binarno & 1)
```

```
        binarno = binarno >> 1

# biranje pozicije ručno
def prikazi_poziciju(pozicija):
    for i in range(4):
        mjesto[i].value(1)
    mjesto[pozicija].value(0)

# prikazivanje kompletnog broja
def prikazi_broj(broj):
    n = 3
    cifra = daj_cifru(broj, n)
    while (n>=0):
        prikazi_cifru(10)
        prikazi_poziciju(3-n)
        prikazi_cifru(cifra)
        n -= 1
        cifra = daj_cifru(broj, n)
        time.sleep(0.0001)

# izvuci n-tu cifru broja
def daj_cifru(broj, n):
    return broj // 10 ** n % 10

# inicijalizacija
auto = False
broj = 0

# main loop
while True:
    # tasteri
    if tasteri[0].value():
        broj += 1
        while (tasteri[0].value()):
          continue

    elif tasteri[1].value():
        broj -= 1
        while (tasteri[1].value()):
            continue

    elif tasteri[2].value():
        broj = 0

    elif tasteri[3].value():
        auto = not auto
        count = 0
        while(tasteri[3].value()):
            continue
```

```
    # ukoliko je pritisnut taster za automatsko inkrementiranje
    if auto:
        if count == 720:
            broj += 1
            count = 0
        count += 1

    # broj mora ostati u rasponu 0 - 9999
    if broj > 9999:
        broj = 0

    if broj < 0:
        broj = 9999

    # prikaži na display-u
    prikazi_broj(broj)
    time.sleep(0.0001)
```

5.3 Zadatak 3 - izbor 1 / izvorni kod

Zadatak 3 je pripremljen za pokretanje na razvojnom sistemu picoETF, te je programski kod u nastavku rađen u programskom jeziku Micro Python.

Implementacija se razlikuje u odnosu na postavku zadatka jer se za umjesto pina gp16 kao „generatora četvorki“, koristi pin gp28 zbog lakšeg povezivanja osciloskopa i picoETF-a.

```
from machine import Pin, PWM
import time
time.sleep(0.1) # Wait for USB to become ready

# matricna tastatura
tipke = [['1', '2', '3', 'A'],
         ['4', '5', '6', 'B'],
         ['7', '8', '9', 'C'],
         ['*', '0', '#', 'D']]

# pinovi na koje je povezana
keypad_rows = [21, 22, 26, 27]
keypad_cols = [0, 1, 2, 3]

# prazni nizovi u koje će se dodijeliti Pin(i, Pin.IN/OUT)
col_pins = []
row_pins = []

# dodjela fizickih pinova
for x in range(0,4):
  row_pins.append(Pin(keypad_rows[x], Pin.OUT))
  row_pins[x].value(1)
```

```
  col_pins.append(Pin(keypad_cols[x], Pin.IN, Pin.PULL_DOWN))
  col_pins[x].value(0)

# scankeys() - vraca char koji je procitan s tastature
def scankeys():
  for row in range(4):
    for col in range(4):
      row_pins[row].high()
      key = None

      if col_pins[col].value() == 1:
        print("Tipka: ", tipke[row][col])
        key_press = tipke[row][col]
        return key_press
        time.sleep(0.3)

    row_pins[row].low()
  return None

# 7 - segment display
cifre = {
    0: 0b1000000,
    1: 0b1111001,
    2: 0b0100100,
    3: 0b0110000,
    4: 0b0011001,
    5: 0b0010010,
    6: 0b0000010,
    7: 0b1111000,
    8: 0b0000000,
    9: 0b0010000,
    10: 0b1111111
}

mjesto = [Pin(m, Pin.OUT) for m in range(4, 8)]
segmenti = [Pin(s, Pin.OUT) for s in range(8, 15)]

# prikazi cifru na jednom mjestu
def prikazi_cifru(cifra):
    binarno = cifre[cifra]
    for i in range(7):
        segmenti[i].value(binarno & 1)
        binarno = binarno >> 1

# biranje pozicije ručno
def prikazi_poziciju(pozicija):
    for i in range(4):
        mjesto[i].value(1)
    mjesto[pozicija].value(0)
```

```
# prikazivanje kompletnog broja
def prikazi_broj(broj):
    n = 3
    cifra = daj_cifru(broj, n)
    while (n>=0):
        prikazi_cifru(10)
        prikazi_poziciju(3-n)
        prikazi_cifru(cifra)
        n -= 1
        cifra = daj_cifru(broj, n)
        time.sleep(0.00001)

# izvuci n-tu cifru broja
def daj_cifru(broj, n):
    return broj // 10 ** n % 10

T_current = 0.001
pwm_on = False

while True:

  unos = scankeys()

  prikazi_broj(int(T_current * 1000))

  if unos == 'A' :
    if pwm_on:
      pwm.deinit()
      pwm_on = False
    else:
      pwm = PWM(Pin(28))
      pwm.freq(int(1.0 / T_current))
      pwm.duty_u16(32767) # 50% duty cycle za simetrican talas
      pwm_on = True
    time.sleep(1)

  elif unos == '0':
    T_current = 0.001
    time.sleep(1)

  elif unos == '1':
    T_current = 0.002
    time.sleep(1)

  elif unos == '2':
    T_current = 0.003
    time.sleep(1)
```

```
elif unos == '3':
  T_current = 0.004
  time.sleep(1)

elif unos == '4':
  T_current = 0.005
  time.sleep(1)

elif unos == '5':
  T_current = 0.006
  time.sleep(1)

elif unos == '6':
  T_current = 0.007
  time.sleep(1)

elif unos == '7':
  T_current = 0.008
  time.sleep(1)

elif unos == '8':
  T_current = 0.009
  time.sleep(1)

elif unos == '9':
  T_current = 0.01
  time.sleep(1)

elif unos == 'C':
  T_current += 0.001
  if T_current > 0.01:
    T_current = 0.001
  time.sleep(1)

elif unos == 'D':
  T_current -= 0.001
  if T_current < 0.001:
    T_current = 0.01
  time.sleep(1)

if pwm_on:
  pwm.freq(int(1.0/T_current))

prikazi_broj(int(T_current * 1000))

time.sleep(0.0001)
```