Univerzitet u Sarajevu

Elektrotehnički fakultet

**Ugradbeni sistemi 2023 / 24**

**Izvještaj za laboratorijsku vježbu br. 4**

Analogni ulazi i displeji

Ime i prezime: **Ivona Jozić**

Broj indeks-a: **19357**

**02. april 2024.**

**Sadržaj**

[1 Pseudokod i / ili dijagram toka 3](#_Toc163082885)

[1.1 Zadatak 1 3](#_Toc163082886)

[1.2 Zadatak 2 3](#_Toc163082887)

[1.3 Zadatak 3 3](#_Toc163082888)

[1.4 Zadaci za dodatne bodove 4](#_Toc163082889)

[1.4.1 Zadatak 4 4](#_Toc163082890)

[2 Analiza programskog rješenja 5](#_Toc163082891)

[2.1 Zadatak 1 5](#_Toc163082892)

[2.2 Zadatak 2 5](#_Toc163082893)

[2.3 Zadatak 3 5](#_Toc163082894)

[2.4 Zadaci za dodatne bodove 6](#_Toc163082895)

[2.4.1 Zadatak 4 6](#_Toc163082896)

[3 Korišteni hardverski resursi 6](#_Toc163082897)

[4 Zaključak 8](#_Toc163082898)

[5 Prilog 9](#_Toc163082899)

[5.1 Zadatak 1 / izvorni kod 9](#_Toc163082900)

[5.2 Zadatak 2 / izvorni kod 10](#_Toc163082901)

[5.3 Zadatak 3 / izvorni kod 14](#_Toc163082902)

[5.4 Zadaci za dodatne bodove 15](#_Toc163082903)

[5.4.1 Zadatak 4 / izvorni kod 15](#_Toc163082904)

# Pseudokod i / ili dijagram toka

U pseudokodovima za laboratorijsku vježbu 4, 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.

## Zadatak 1

adc potenciometar

while(1)

t -> podesi\_vrijeme(potenciometar)

for i in 0..8

prikazi\_na\_diodama(2^i)

sacekaj(t)

end\_for

prikazi\_na\_diodama(255)

sacekaj(t)

led -> 255

for i in 8..0, step -1

led -> led – 2^i

prikazi\_na\_diodama(led)

sacekaj(t)

end\_for

end\_while

## Zadatak 2

while(1)

ocitaj\_senzor()

temperatura -> izracunaj\_temperaturu()

napon -> izracunaj\_napon()

prikazi\_na\_displeju(temperatura, napon, vrijeme)

iscrtaj\_graf(temperatura)

sleep(1 sekunda)

vrijeme -> vrijeme + 1

end\_while

## Zadatak 3

BusOut led

AnalogIn potenciometar

while(1)

led.upali(potenciometar)

sleep(0.01)

end\_while

## 1.4 Zadaci za dodatne bodove

### 1.4.1 Zadatak 4

ucitaj\_displej()

prikaz - > otpor

while(1)

if(taster1 pritisnut) prikaz -> !prikaz

end\_if

if(prikaz = otpor)

prikazi\_otpor()

else

prikazi\_napon()

end\_if

sleep(0.01)

# Analiza programskog rješenja

## 2.1 Zadatak 1

U prvom zadatku je bilo potrebno spojiti potenciometar otpornosti 10K na picoETF razvojni sistem. Osim toga bilo je potrebno realizirati „trčeće“ svjetlo, što podrazumijeva paljenje po jedne od LED dioda u smjeru sa lijeva na desno, zatim paljenje svih dioda, te postepeno gašenje po jedne LED diode u smjeru sa desna na lijevo. Ovaj algoritam je detaljno opisan u sklopu izvještaja za laboratorijsku vježbu 2, s tim da postoji jedna razlika. Ovaj put, period između dvije promjene se podešava u zavisnosti od očitanja potenciometra, gdje u slučaju da potenciometar očita vrijednost 0V vrijeme treba iznositi 0.1s, dok u slučaju očitanja 3.3V treba iznositi 2s. Periodi između ova dva trebaju biti srazmjernog trajanja u odnosu na raspon očitanja potenciometra.

## 2.2 Zadatak 2

Cilj drugog zadatka je bio spojiti Banggood displej, temperaturni senzor LM35 i voltmetar na razvojni sistem picoETF. Temperaturni senzor na svom izlazu daje napon koji ovisi o temperaturi, a preslikavanje je linearno, pri čemu 0V odgovara 0°C dok 1V odgovara 100°C (inkrement je konstantan i iznosi 10mV/°C). Program implementiran u ovom zadatku na displeju prikazuje mjerenu temperaturu (sa tačnošću od 0.1°C), napon koji se očitava sa senzora (tačnost 1mV), te dijagram promjene temperature u funkciji od vremena na svaku sekundu.

Da bi program radio očekivano bilo je neophodno učitati datoteke *gfx* (za iscrtavanje elemenata na ekranu), *glcdfont* (za korištenje različitih fontova), *ili934xnew* (za kontrolu TFT displeja), *tt14*, *tt24*, *tt32*.

Za izračunavanje temperature u °C korišten je izraz var/260, dok je za izračunavanje napona u mV korišten var/26. Izrazi su povezani sa tehničkim karakteristikama temperaturnog senzora koji ima stalni inkrement napona za 1mV kad god dođe do povećanja temperature za 1°C. Za očitanje izlaznog napona na senzoru je korišten A/D pretvarač sa mikrokontrolera, koji ima raspon od 0 do 65535, te je i ovaj raspon potrebno adekvatno preračunati u mV.

## Zadatak 3

U trećem zadatku je implementiran tzv. VU metar na razvojnom sistemu LPC1114ETF, a koji prikazuje intenzitet osvjetljenja, pri čemu su LED diode korištene za indikaciju napona na potenciometru (0V – sve diode upaljene, a 3.3V – sve diode ugašene). Zadatak je realiziran tako što je raspon očitanja potenciometra podijeljen na 9 (8 dioda i stanje kada su sve ugašene) jednakih dijelova i u zavisnosti od toga šta je očitano, određene diode se pale i gase. Paljenje i gašenje dioda realizirano je pomoću objekta klase BusOut, dok je očitanje sa senzora vršeno pomoću objekta klase AnalogIn.

## 2.4 Zadaci za dodatne bodove

### 2.4.1 Zadatak 4

U ovom zadatku je na razvojni sistem LPC1114ETF spojen Nokia displej, potenciometar otpornosti 100K, te voltmetar. Zahtijevana je implementacija programa koji mjeri trenutni otpor potenciometra, te napon na potenciometru, a zatim očitane vrijednosti prikazuje na Nokia displeju. U ovom zadatku funkcionalnost je data i tasteru1 čijim pritiskom se utječe na to da li se prikazuje vrijednost otpora ili napona. Uloga voltmetra je bila čisto usporedna, kako bi se usporedile vrijednosti očitane voltmetrom i one koje su prikazane na displeju.

Rješenje je implementirano korištenjem objekta klase AnalogIn koji nudi funkcije za očitanje stanja potenciometra, te pomoćnih funkcija za ispis vrijednosti napona i otpora. Također je implementirana i pomoćna funkcija koja očitani realni broj pretvara u string s obzirom da funkcija sprintf ne radi u skladu sa očekivanjima sa realnim varijablama.

# 3 Korišteni hardverski resursi

Za potrebe laboratorijske vježbe 4 korišteni su razvojni sistemi picoETF i LPC1114ETF. Pored njih korišteni su TFT (Banggood) displej, Nokia displej, temperaturni senzor LM35, potenciometar i voltmetar.

Za sistem picoETF:

- 8 LED dioda

Za sistem LPC1114ETF:

- 8 LED dioda

- 1 fizički taster

- Nokia N5110 display

- ILI9314 TFT LCD color display

- Rotacijski potenciometar (totalne otpornosti 100kΩ)

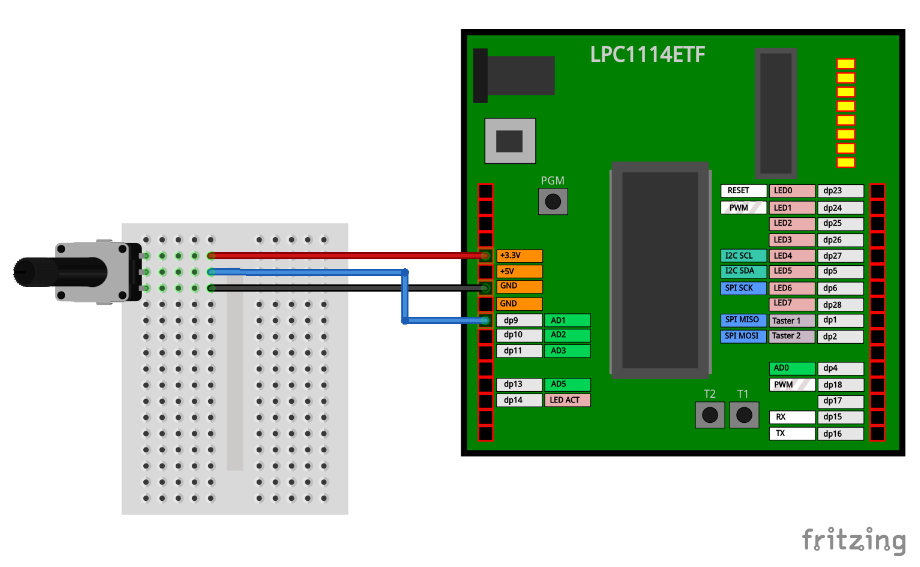
- LM35 analogni temperaturni senzor

- voltmetar

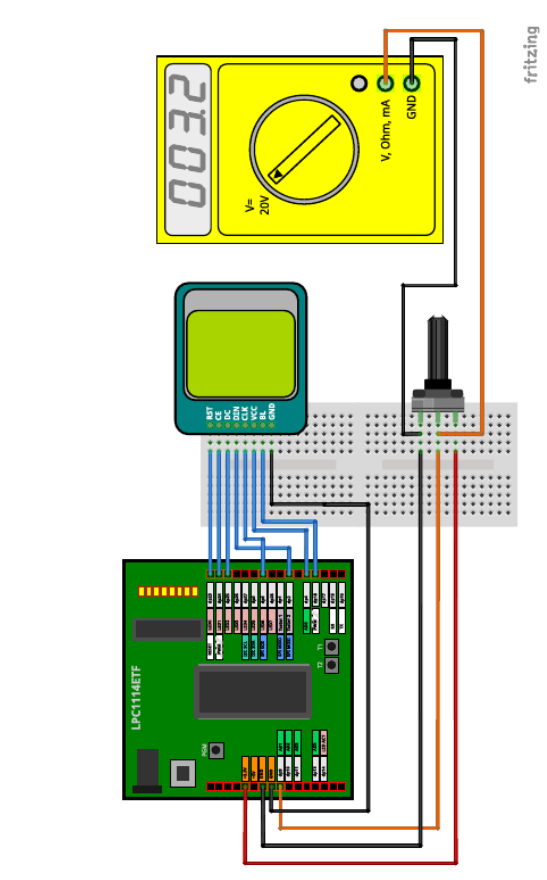
|  |  |
| --- | --- |
| **ULAZI** | **IZLAZI** |
| Taster1  (digitalni, LPC1114ETF) | Nokia display  (digitalni output pinovi) |
| LM35  (analogni, picoETF, GP26) | ILI9314 TFT display  (digitalni output pinovi) |
| Potenciometar  (analogni, oba sistema; DP9 na LPC1114ETF, GP28 na picoETF) |  |



Slika 1. – Način povezivanja TFT displeja i LM35 senzora na sistem picoETF (2. zadatak)



Slika 2. – Način povezivanja potenciometra na LPC1114ETF (3. zadatak)



Slika 3. – Način povezivanja Nokia displeja sa sistemom LPC1114ETF (4. zadatak)

# 4 Zaključak

Prilikom izvođenja Laboratorijske vježbe 4 nije bilo poteškoća s obzirom da su oba razvojna sistema picoETF i LPC1114ETF od ranije poznata. Također nije bilo problema ni sa radom sa TFT i Nokia displejima, potenciometrom niti temperaturnim senzorom. Jedini primijećeni problem je neočekivani rad funkcije sprintf koja je korištena u 4. zadatku, a koja sa formatom za cijele brojeve radi sasvim u skladu sa očekivanjima, dok sa formatom realnih brojeva ne vrši ispis. Problem je prevaziđen implementacijom pomoćne funkcije koja realni broj pretvara u string. Cilj vježbe je bio upoznavanje sa analognim ulazima i izlazima, spomenutim u odjeljku 3 ovog izvještaja, što je postignuto.

# 5 Prilog

U prilogu su dati kodovi koje se izvršavaju na razvojnim okruženjima picoETF (1. i 2. zadatak) LPC1114ETF (3. i 4. zadatak). Date kodove, osim 4. zadataka zbog nedostatka komponenti, je moguće pokrenuti i u Wokwi, te Mbed simulatorima, s tim da će za testiranje u Mbed simulatoru biti potrebno izmijeniti inicijalizaciju pinova.

## 5.1 Zadatak 1 / izvorni kod

import time

from machine import Pin, ADC

time.sleep(0.1) # Wait for USB to become ready

leds = [Pin(i, Pin.OUT) for i in range(4, 12)]

adc = ADC(Pin(28))

def decimal\_to\_binary\_8bits(n):

    # Konverzija decimalnog u binarni broj i uklanjanje prefiksa '0b'

    binary\_str = bin(int(n))[2:]

    # Dopuna nulama do dužine 8

    while len(binary\_str) < 8:

        binary\_str = '0' + binary\_str

    # Konverzija binarnog stringa u listu integera

    binary\_list = [int(bit) for bit in binary\_str]

    return binary\_list

def update\_leds(num, leds): # Funkcija za paljenje LED

    num\_binList = decimal\_to\_binary\_8bits(num)

    for i in range(0, 8):

        leds[i].value(num\_binList[i])

def set\_t(adc\_val):

    return 0.1 + 1.9\*adc\_val/65535

brojac = 0

t = 0.1

adc\_val = adc.read\_u16()

while True:

    print(adc.read\_u16())

    t = set\_t(adc.read\_u16())

    brojac += 1

    update\_leds(brojac, leds)

    time.sleep(t)

    for i in range(0, 8): # Svjetlo sa lijeva na desno

        brojac \*= 2

        update\_leds(brojac, leds)

        t = set\_t(adc.read\_u16())

        time.sleep(t)

    brojac = 255

    update\_leds(brojac, leds)

    t = set\_t(adc.read\_u16())

    time.sleep(t)

    pomocna = 256

    for i in range(0, 8): # Svjetlo sa desna na lijevo

        brojac -= pomocna/2

        update\_leds(brojac, leds)

        t = set\_t(adc.read\_u16())

        time.sleep(t)

        pomocna /= 2

    brojac = 0

## 5.2 Zadatak 2 / izvorni kod

Za ispravno funkcioniranje ovog koda, neophodno je upload-ati i sljedeće datoteke: *gfx*, *glcdfont*, *ili934xnew*, *tt14*, *tt24*, *tt32*.

from ili934xnew import ILI9341, color565

from machine import Pin, SPI

from micropython import const

import os

import glcdfont

import tt14

import tt24

import tt32

import time

import gfx

from machine import ADC

adc = ADC(Pin(26))

# Dimenzije displeja

SCR\_WIDTH = const(320)

SCR\_HEIGHT = const(240)

SCR\_ROT = const(2)

CENTER\_Y = int(SCR\_WIDTH/2)

CENTER\_X = int(SCR\_HEIGHT/2)

#print(os.uname())

# Podešenja SPI komunikacije sa displejem

TFT\_CLK\_PIN = const(18)

TFT\_MOSI\_PIN = const(19)

TFT\_MISO\_PIN = const(16)

TFT\_CS\_PIN = const(17)

TFT\_RST\_PIN = const(20)

TFT\_DC\_PIN = const(15)

# Fontovi na raspolaganju

fonts = [glcdfont, tt14, tt24, tt32]

text = 'RPi Pico/ILI9341'

print(text)

print("Fontovi:")

for f in fonts:

    print(f.\_\_name\_\_)

spi = SPI(0,

          baudrate=62500000,

          miso=Pin(TFT\_MISO\_PIN),

          mosi=Pin(TFT\_MOSI\_PIN),

          sck=Pin(TFT\_CLK\_PIN))

print(spi)

display = ILI9341(spi,

                  cs=Pin(TFT\_CS\_PIN),

                  dc=Pin(TFT\_DC\_PIN),

                  rst=Pin(TFT\_RST\_PIN),

                  w=SCR\_WIDTH,

                  h=SCR\_HEIGHT,

                  r=SCR\_ROT)

# Brisanje displeja i odabir pozicije (0,0)

display.erase()

display.set\_pos(0,0)

# Ispis teksta različitim fontovima, počevši od odabrane pozicije

for ff in fonts:

    display.set\_font(ff)

    display.print(text)

# Ispis teksta u drugoj boji

display.set\_font(tt14)

display.set\_color(color565(150, 200, 0), color565(0, 0, 0))

time.sleep(1)

#Brisanje displeja

display.erase()

# Dodatna funkcija za crtanje kružnice ispisom pojedinačnih piksela

display.set\_font(tt14)

display.erase()

# Različita orijentacija teksta na displeju

display.set\_pos(10,100)

display.rotation=0

display.erase()

display.set\_pos(10,100)

display.rotation=1

display.init()

display.erase()

display.set\_pos(10,100)

display.rotation=2

display.init()

display.erase()

display.set\_pos(10,100)

display.rotation=3

display.init()

display.erase()

display.set\_pos(10,100)

display.rotation=4

display.init()

display.erase()

display.set\_pos(10,100)

display.rotation=5

display.init()

display.erase()

display.set\_pos(10,100)

display.rotation=6

display.init()

display.erase()

display.set\_pos(10,100)

display.rotation=7

display.init()

display.erase()

def fast\_hline(x, y, width, color):

    display.fill\_rectangle(x, y, width, 1, color)

def fast\_vline(x, y, height, color):

    display.fill\_rectangle(x, y, 1, height, color)

def line(self, x0, y0, x1, y1, \*args, \*\*kwargs):

# Funkcija za crtanje linija. Iscrtati ce po jedan piksel linije krenuvsi od (x0, y0)

# pa do (x1, y1)

    steep = abs(y1 - y0) > abs(x1 - x0)

    if steep:

        x0, y0 = y0, x0

        x1, y1 = y1, x1

    if x0 > x1:

        x0, x1 = x1, x0

        y0, y1 = y1, y0

    dx = x1 - x0

    dy = abs(y1 - y0)

    err = dx // 2

    ystep = 0

    if y0 < y1:

        ystep = 1

    else:

        ystep = -1

    while x0 <= x1:

        if steep:

            self.\_pixel(y0, x0, \*args, \*\*kwargs)

        else:

            self.\_pixel(x0, y0, \*args, \*\*kwargs)

        err -= dy

        if err < 0:

            y0 += ystep

            err += dx

        x0 += 1

gfx = gfx.GFX(240, 320, display.pixel, hline=fast\_hline, vline=fast\_vline)

#graphics.line(0, 0, 239, 319, color565(255, 0, 0))

vrijeme = 5

t=0

while True :

    var = adc.read\_u16()

    tmp = var/260

    volt = var/26

    display.set\_pos(180,10)

    display.rotation=1

    display.init()

    display.print('Temp: '+ str(tmp) + " C")

    display.print('Napon: ' + str(volt) + 'mV')

    display.print('Vrijeme: ' + str(t) + 's')

    #graphics.line(20,20,20,100)

    gfx.line(20, 20, 20, 220, color565(255, 255, 255))

    gfx.line(20, 220, 260, 220, color565(255, 255, 255))

    display.set\_pos(14, 220)

    display.print("20")

    display.set\_pos(14, 160)

    display.print("25")

    display.set\_pos(14, 100)

    display.print("30")

    display.set\_pos(14, 0)

    display.print("40")

    gfx.fill\_circle(20+vrijeme,220-int((tmp-22)\*18),4, color565(150, 200, 0) )

    vrijeme = vrijeme + 5

    if(20 + vrijeme) > 260:

        vrijeme = 5

        display.erase()

    t+=1

    #display.erase()

## 5.3 Zadatak 3 / izvorni kod

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

#include "mbed.h"

#include "lpc1114etf.h"

#define nula   0xff  //sve diode ugašene

#define jedan  0x7f

#define dva    0x3f

#define tri    0x1f

#define cetiri 0x0f

#define pet    0x07

#define sest   0x03

#define sedam  0x01

#define osam   0x00  //sve diode upaljene

BusOut leds(LED0, LED1, LED2, LED3, LED4, LED5, LED6, LED7);

AnalogIn potMeter (dp9);

DigitalOut e(LED\_ACT);

int main() {

    e = 0;

    while(true) {

    if(potMeter < 1./9) leds = nula;

    else if(potMeter >=1./9 && potMeter < 2\*1./9) leds = jedan;

    else if(potMeter >=2\*1./9 && potMeter < 3\*1./9) leds = dva;

    else if(potMeter >=3\*1./9 && potMeter < 4\*1./9) leds = tri;

    else if(potMeter >=4\*1./9 && potMeter < 5\*1./9) leds = cetiri;

    else if(potMeter >=5\*1./9 && potMeter < 6\*1./9) leds = pet;

    else if(potMeter >=6\*1./9 && potMeter < 7\*1./9) leds = sest;

    else if(potMeter >=7\*1./9 && potMeter < 8\*1./9) leds = sedam;

    else if(potMeter > 8\*1./9) leds = osam;

    wait\_us(10);

    }

}

## 5.4 Zadaci za dodatne bodove

### 5.4.1 Zadatak 4 / izvorni kod

#include "mbed.h"

#include "N5110.h"

#include "lpc1114etf.h"

#include <stdio.h>

AnalogIn potMeter(dp9);

DigitalIn taster(Taster\_1);

N5110 lcd(dp4,dp24,dp23,dp25,dp2,dp6,dp18);

char\* floatToString(float num) {

    // Alociranje memorije za string

    char\* str = (char\*)malloc(20 \* sizeof(char));

    // U slucaju da je broj negativan

    if (num < 0) {

        \*str++ = '-';

        num = -num;

    }

    // Cijeli dio

    int intPart = (int)num;

    int index = 0;

    do {

        str[index++] = intPart % 10 + '0';

        intPart /= 10;

    } while (intPart != 0);

    // Okretanje poretka

    for (int i = 0; i < index / 2; i++) {

        char temp = str[i];

        str[i] = str[index - i - 1];

        str[index - i - 1] = temp;

    }

    // Decimalna tacka

    str[index++] = '.';

    // Necijeli dio

    float fracPart = num - (int)num;

    for (int i = 0; i < 6; i++) { // 6 decimala

        fracPart \*= 10;

        int digit = (int)fracPart;

        str[index++] = digit + '0';

        fracPart -= digit;

    }

    // Zavrsi string

    str[index] = '\0';

    return str;

}

void prikaziOtpor() {

    char otporString[10];

    float otpor = 100000 \* potMeter.read();

    sprintf(otporString, "%d", (int)otpor);

    lcd.clear();

    lcd.printString("Otpor (Ohm): ", 0, 0);

    lcd.printString(otporString, 0, 2);

}

void prikaziNapon() {

    float refNapon = 3.3;

    float napon = refNapon \* potMeter.read();

    char\* naponString;

    //sprintf(naponString, "%d", (int)napon);

    naponString = floatToString(napon);

    lcd.clear();

    lcd.printString("Napon (V): ", 0, 0);

    lcd.printString(naponString, 0, 2);

free(naponString);

}

int main() {

    //false za prikaz otpora, true za prikaz napona

    lcd.init();

    bool viewMode = false;

    while (true) {

        if(taster == 1) viewMode = !viewMode;

        if(viewMode == false) prikaziOtpor();

        else prikaziNapon();

        wait\_us(100);

    }

}