Univerzitet u Sarajevu
Elektrotehnički fakultet
Ugradbeni sistemi 2023/24

Izvještaj za laboratorijsku vježbu br. 4

Analogni ulazi i displeji

Ime i prezime: **Mirza Mahmutović**
Broj index-a: **19320**

2. april 2024. godine

Sadržaj

1 Pseudokod i/ili dijagram toka

1.1 Zadatak 1

```
leds = [Pin(),Pin(),...,Pin()]
adc = ADC(Pin())
counter = 0
increment = 1
while (true) do
	value = adc.read_u16()
	t = 0.1 + (value / 34492.1)
	if (counter == 0) then
		if (increment == -1) then
			leds[counter].off()
			increment = 1
			counter -= 1
		else
			leds[counter].on()
		end_if
	else_if (counter == 7 and increment == 1) then
		leds.allOn()
		increment = -1
		counter += 1
	else
		if (increment == 1) then
			leds[counter - 1].off()
			leds[counter].on()
		else
			leds[counter].off()
		end_if
	end_if
	counter += increment
	sleep()
end_while
```

1.2 Zadatak 2

```
adc = ADC(Pin())
display.setup()
draw_circle(xpos,ypos,rad,color)
      x = rad - 1
      y = 0
      dx = 1
      dy = 1
      err = dx - (rad << 1)
      while (x >= y) do
            display.pixel(xpos + x, ypos + y, col)
            display.pixel(xpos + y, ypos + x, col)
            display.pixel(xpos - y, ypos + x, col)
            display.pixel(xpos - x, ypos + y, col)
            display.pixel(xpos - x, ypos - y, col)
            display.pixel(xpos - y, ypos - x, col)
            display.pixel(xpos + y, ypos - x, col)
            display.pixel(xpos + x, ypos - y, col)
      end_while
      if (err <= 0) then
            y += 1
            err += dy
            dy += 2
      end_if
      if (err > 0) then
            x -= 1
            dx += 2
            err += dx - (rad << 1)
      end_if
linija(x0, y0, x1, y1, color)
      dx = abs(x1 - x0)
      dy = abs(y1 - y0)
      sx = -1 if x0 > x1 else 1
      sy = -1 if y0 > y1 else 1
      err = dx - dy
```

```
        while (true) do
                display.pixel(x0, y0, color)
                display.pixel(x0, y0+1, color)
        end_while
        if (x0 == x1 and y0 == y1) then
                break
        end_if
        e2 = 2 * err
        if (e2 > -dy) then
                err -= dy
                x0 += sx
        end_if
        if (e2 < dx) then
                err += dx
                y0 += sy
        end_if
t = 0
t_act = 0
vrijeme = 5
t=0
temp_tac = (12,200)
while (true) do
        t += 1
        t_act += 1
        volt = round(adc.read_u16()*3300/65535,2)
        tmp = round(volt/10,1)
        display.print('Temp: '+ str(tmp) + " C")
        display.print('Napon: ' + str(volt) + 'mV')
        display.print('Vrijeme: ' + str(t) + 's')
        x = 12 + t * 12
        y = 199 + (20 - int(tmp)) * 8
        linija(temp_tac[0], temp_tac[1], x, y, color565(255,0,0))
        for (i = 0; i < 5; i++) do
                draw_circle(temp_tac[0], temp_tac[1], i, color565(255,255,255))
        end_for
```

```
      for (i = 0; i < 5; i++) do
            draw_circle(x, y, i, color565(255,0,0))
      end_for
    temp_tac=(x,y)
    sleep()
end_while
```

1.3 Zadatak 3

```
BusOut leds()
AnalogIn adc()
while(true) do
      if(adc < 1./9) then
            leds = 255
      else_if (adc > 1./9 and adc < 2 * 1./9) then
            leds = 127;
      else_if (adc >= 2 * 1./9 and adc < 3 * 1./9) then
            leds = 63;
      else_if (adc >= 3 * 1./9 and adc < 4 * 1./9) then
            leds = 31;
      else_if (adc >= 4 * 1./9 and adc < 5 * 1./9) then
            leds = 15;
      else_if (adc >= 5 * 1./9 and adc < 6 * 1./9) then
            leds = 7;
      else_if (adc >= 6 * 1./9 and adc < 7 * 1./9) then
            leds = 3;
      else_if (adc >= 7 * 1./9 and adc < 8 * 1./9) then
            leds = 1;
      else
            leds = 0
      end_if
      sleep()
end_while
```

2 Analiza programskog rješenja

2.1 Zadatak 1

Zadatak 1 predstavlja prvi zadatak u radu sa analognim ulazima na ovoj laboratorijskoj vježbi. U zadatku se traži da se implementira trčeće svjetlo na LED diodama sistema, ali tako da

vrijeme za koje treba da svijetli neka dioda se podešava pomoću potenciometra povezanog na analogni ulaz. Raspon vremena treba da bude od 0,1s kada je potenciometar postavljen u jednu krajnju poziciju (0V), do 2s kada je potenciometar postavljen u drugu krajnju poziciju (3,3V). Za implementaciju su nam prije svega potrebni niz digitalnih izlaza povezanih na LED diode, te analogni ulaz povezan na pin označen u postavci zadatka. Za početak se varijabla *counter* postavlja na 0, a varijabla *increment* na vrijednost 1. U beskonačnoj *while* petlji se pomoću metode *read_u16()* učitava vrijednost na analognom ulazu, te se vrijeme postavlja na vrijednost $t = 0.1 + (\text{adc.}read\_u16()/x)$. Varijabla x predstavlja vrijednost za koju vrijedi sljedeća jednakost $\text{adc.}read\_u16()/x = 1.9$ da bi vrijeme bilo u potrebnim granicama. Zatim se po već opisanom algoritmu za implementaciju „trčećeg svjetla" provjerava koje LED diode treba upaliti, a koje ugasiti. Na kraju *while* petlje se još blokira program pomoću metode *sleep()* za očitano vrijeme *t.*

2.2 Zadatak 2

U drugom zadatku je potrebno očitavati vrijednost napona koju senzor temperature *LM35* daje na osnovu temperature. Potrebno je implementirati program koji na displeju *Banggood* prikazuje: mjerenu temperaturu, napon koji daje senzor, te dijagram promjene temperature u funkciji od vremena. Da bi se program mogao implementirati prije svega je potrebno uključiti sve potrebne datoteke vezane za rad sa tft displejem. Potrebno je isti taj displej konfigurisati, odnosno podesiti prije svega sve pinove, a zatim i njegove dimenzije, rotaciju, font... Za lakšu implementaciju iskorištene su i dvije funkcije koje smo nazvali *linija()* i *drawCircle()*. Funkcija *linija()* kao svoje parametre prima početne i krajnje koordinate, te odgovarajuću boju. Funkcija računa udaljenosti koordinata tačaka, te smjer u kojem bi se linija trebala kretati posmatrajući od početnih koordinata ka krajnjim. Zatim, na osnovu tih u vrijednosti, sve dok se tačke ne poklope, pomoću metode *pixel()* prikazuje liniju koja spaja te dvije tačke u navedenoj boji. Funkcija *draw_circle()* na osnovu zadanih koordinata, radijusa kruga, te podešene boje, prikazuje vrijednost temperature u potrebnim trenucima, a funkcioniše na sličnom principu kao prethodno opisana funkcija. Za rad programa, potrebno je postaviti početne vrijednosti varijablama koje nam služe za računanje vremena (t) i predstavljanja vrijednosti zadnje očitane temperature (*temp_act*). U suštini, vrijednost varijable *temp_act* nam je potrebna da bi smo na grafu mogli povezati vrijednost trenutno očitane temperature sa vrijednošću prethodno opisane temperature. U beskonačnoj *while* petlji je potrebno učitati vrijednost napona na analognom ulazu, na osnovu njega izračnuti vrijednost temperature, ispisati te vrijednosti, zajedno sa vremenom, na displej na unaprijed definisanoj poziciji. Zatim je potrebno na osnovu očitane

vrijednosti temperature, izračunati koordinate *(x,y)* sljedeće tačke koja će predstavljati tu temperaturu na grafu. Uz pomoć funkcije *linija()* se ta tačka povezuje sa zadnjom predstavljenom tačkom, te korištenjem funkcije *draw_circle()* se ista ta tačka prikazuje na displeju. Vrijednost varijable *temp_tac* se postavlja na *(x,y),* program se blokira 1 sekundu i prelazi se na sljedeću iteraciju.

2.3 Zadatak 3

U trećem zadatku je korištenjem 8 LED dioda na sistemu bilo potrebno realizovati UV metar koji prikazuje intenzitet osvjetljenja. Intenzitet osvjetljenja zavisi od vrijednosti napona na analognom ulazu na koji je povezan potenciometar. Treba obratiti pažnju da se radi o inverznoj logici, odnosno da je za 0V potrebno uključiti sve diode, a za 3,3V isključiti sve diode. Za realizaciju nam je potrebna BusOut varijabla *leds,* te AnalogIn varijabla *adc.* U ovoj situaciji imamo 9 mogućnosti (0 upaljenih LED dioda, 1 upaljena LED dioda, 2, 3, 4, 5, 6, 7 i 8 upaljenih LED dioda). Kako je vrijednost analognog ulaza predstavljena u opsegu od 0 do 1, potrebno je provjeriti u kojem intervalu se nalazi vrijednost, te na osnovu toga odrediti broj upaljenih dioda. Znači, potrebno je provjeriti kojem od sljedećih intervala pripada vrijednost *adc*:

- (0, 1/9) leds = 255
- [1/9, 2/9) leds = 127
- [2/9, 3/9) leds = 63
- [3/9, 4/9) leds = 31
- [4/9, 5/9) leds = 15
- [5/9, 6/9) leds = 7
- [6/9, 7/9) leds = 3
- [7/9, 8/9) leds = 1
- [8/9, 1] leds = 0

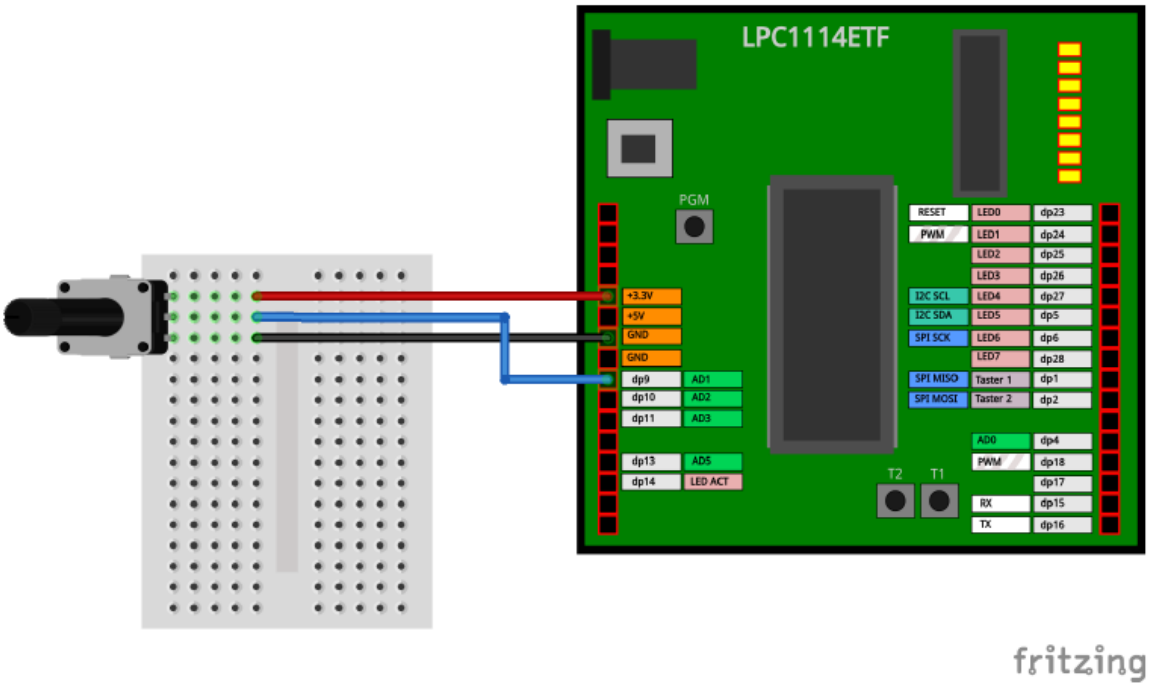
3 Korišteni hardverski resursi

Na ovoj vježbi su korišteni sljedeći razvojni sistemi:

- *LPC1114ETF,* baziran na mikrokontroleru *NXP LPC1114FN28*
- *picoETF*, baziran na mikrokontroleru *RP2040*
- potenciometar
- temperaturni senzor *LM35*
- TFT *Banggood* displej

Na sistemu *LPC1114ETF* su za ovu vježbu korišteni sljedeći elementi:

- 8x LED dioda



Slika 1: Povezivanje potenciometra na LPC1114ETF sistem

4 Zaključak

Ova vježba je za cilj imala da se studenti upoznaju sa načinom korištenja analognih ulaza pomoću Mbed OS-a i pomoću MicroPython-a, te sa dva različita tipa displeja. Kroz ovu laboratorijsku vježbu upoznali smo se i sa načinom povezivanja korištenih sistema sa potenciometrom, temperaturnim senzorom, te Banggood TFT i Nokia 5110 displejima. U Mbed OS-u smo se upoznali sa klasom AnalogIn, te njenim osnovnim metodama za očitavanje vrijednosti. U MicroPython-u za istu svrhu smo koristili klasu ADC. Korištenjem ovih klasa očitavali smo napon na pinovima te uz pomoć istog računali potrebne vrijednosti. Upoznali smo se sa osnovnim principima korištenja displeja, uređivanjem formata, ispisivanjem teksta, te crtanjem grafova. Zadaci nisu bili pretjerano programerski zahtjevni, ali je za njihovo rješavanje bilo potrebno temeljito analizirati dokumentaciju i dobijene primjere.

5 Prilog

5.1 Zadatak 1/izvorni kod

```
import time
```

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

leds = [Pin(4,Pin.OUT),Pin(5,Pin.OUT),Pin(6,Pin.OUT),Pin(7,Pin.OUT),
        Pin(8,Pin.OUT),Pin(9,Pin.OUT),Pin(10,Pin.OUT),Pin(11,Pin.OUT)]

adc = ADC(Pin(28))

counter = 0
increment = 1

while True:
    val = adc.read_u16()
    t = 0.1 + (val / 34492.1)
    print(t)
    if counter == 0:
        if increment == -1:
            leds[counter].off()
            increment = 1
            counter -= 1
        else:
            leds[counter].value(1)
    elif (counter == 7 and increment == 1):
        for i in range(8):
            leds[i].on()
        increment = -1
        counter += 1
    elif (counter == 7 and increment == -1):
        leds[counter].off()
    else:
        if increment == 1:
            leds[counter - 1].off()
            leds[counter].on()
        else:
            leds[counter].off()
    counter += increment
```

```
    time.sleep(t)
```

5.2 Zadatak 2/izvorni kod

```
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(28))

# 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 linija(x0, y0, x1, y1, color):
    dx = abs(x1 - x0)
    dy = abs(y1 - y0)
    sx = -1 if x0 > x1 else 1
    sy = -1 if y0 > y1 else 1
    err = dx - dy

    while True:
        display.pixel(x0, y0, color)
        display.pixel(x0, y0+1, color)

        if x0 == x1 and y0 == y1:
            break

        e2 = 2 * err
        if e2 > -dy:
            err -= dy
            x0 += sx
        if e2 < dx:
            err += dx
            y0 += sy

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 draw_circle(xpos0, ypos0, rad, col=color565(255, 255, 255)):
    x = rad - 1
    y = 0
    dx = 1
```

```
    dy = 1
    err = dx - (rad << 1)
    while x >= y:
        # Prikaz pojedinačnih piksela
        display.pixel(xpos0 + x, ypos0 + y, col)
        display.pixel(xpos0 + y, ypos0 + x, col)
        display.pixel(xpos0 - y, ypos0 + x, col)
        display.pixel(xpos0 - x, ypos0 + y, col)
        display.pixel(xpos0 - x, ypos0 - y, col)
        display.pixel(xpos0 - y, ypos0 - x, col)
        display.pixel(xpos0 + y, ypos0 - x, col)
        display.pixel(xpos0 + x, ypos0 - y, col)
        if err <= 0:
            y += 1
            err += dy
            dy += 2
        if err > 0:
            x -= 1
            dx += 2
            err += dx - (rad << 1)

def line(self, x0, y0, x1, y1, *args, **kwargs):
# Line drawing function. Will draw a single pixel wide line starting at
# x0, y0 and ending at 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


def lines():
    display.fill_rectangle(0,0,SCR_WIDTH,SCR_HEIGHT, color565(0,0,0))
    display.set_font(tt24)
    display.set_color(color565(255,255,255), color565(0,0,0))
    display.set_pos(14, 180)
    display.print("20")
    display.set_pos(14, 100)
    display.print("30")
    display.set_pos(14, 20)
    display.print("40")
    display.set_color(color565(255,255,255), color565(255,255,255))
    display.fill_rectangle(10,10,2,220, color565(255,255,255))
    display.fill_rectangle(5,200,285,2, color565(255,255,255))



graphics = gfx.GFX(240, 320, display.pixel, hline=fast_hline,
vline=fast_vline)


#graphics.line(0, 0, 239, 319, color565(255, 0, 0))
t = 0
t_act = 0
vrijeme = 5
```

```
t=0
temp_tac = (12,200)
while True :
    t += 1
    t_act += 1
    volt = round(adc.read_u16()*3300/65535,2)
    tmp = round(volt/10,1)
    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')
    x = 12 + t * 12
    y = 199 + (20 - int(tmp)) * 8

    #graphics.line(20,20,20,100)

    linija(temp_tac[0],temp_tac[1],x,y,color565(255,0,0))
    for i in range(5):
        draw_circle(temp_tac[0], temp_tac[1],i,color565(255,255,255))
    for i in range(5):
        draw_circle(x, y,i,color565(255,0,0))
    temp_tac=(x,y)
    #if t%23 == 0:
    #    t=0
    #    lines()
    #    temp_tac=(x-23*12,y)
    time.sleep(1)
```

5.3 Zadatak 3/izvorni kod

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

BusOut leds(LED0, LED1, LED2, LED3, LED4, LED5, LED6, LED7);
AnalogIn adc (dp9);
DigitalOut e(LED_ACT);
```

```
int main() {
    e = 0;
    while (1) {
        if(adc < 1./9) leds = 255;
        else if (adc > 1./9 && adc < 2 * 1./9) leds = 127;
        else if (adc >= 2 * 1./9 && adc < 3 * 1./9) leds = 63;
        else if (adc >= 3 * 1./9 && adc < 4 * 1./9) leds = 31;
        else if (adc >= 4 * 1./9 && adc < 5 * 1./9) leds = 15;
        else if (adc >= 5 * 1./9 && adc < 6 * 1./9) leds = 7;
        else if (adc >= 6 * 1./9 && adc < 7 * 1./9) leds = 3;
        else if (adc >= 7 * 1./9 && adc < 8 * 1./9) leds = 1;
        else leds = 0;
        wait_us(10);
    }
}
```