Univerzitet u Sarajevu
Elektrotehnički fakultet
Ugradbeni sistemi 2023/24

Izvještaj za laboratorijsku vježbu br. 1

Razvojni sistem picoETF. MicroPython.

Ime i prezime: **Haris Mališević**
Broj Indeksa: **19328**

12.03.2024.

Sadržaj

1 Pseudokod / dijagram toka

Kako su samo redovni zadatak broj 5 i zadatak za dodatne bodove broj 2 tražili programsko rješenje, samo će za njih biti ponuđen pseudokod.

1.1 Zadatak 5

```
upaliSve()
spavaj(1 sek)

  while True:
      If (T1 pritisnuto):
            inkrementiraj()
            blokirajTastere()

      If (T2 pritisnuto):
            dekrementiraj()
            blokirajTastere()

      If (T3 pritisnuto):
            ugasiSVe()
            blokirajTastere()


      If (T4 pritisnuto):
            upaliSve()
            blokirajTastere()

    spavaj(0.25 sek)
```

Funkcija blokirajTastere ima za cilj blokiranje unosa pritisnutim ili bilo kojim drugim tasterom, dok se pritisnuti taster ne pusti. Cilj ove funkcije je onemogućiti kontinuriano ponavljanje jedne radnje.

1.1 Zadatak za dodatne bodove 2

```
while True:
    pauza = 0.1 sek
        while pauza <= 1.0 sek:
            permutacijaRGB_saPauzom(pauza)
            pauseTime += 0.1 sek
        while pauza>= 0.1 sek:
            permutacijaRGB_saPauzom(pauza)
            pauseTime -= 0.1 sek
```

2 Analiza programskog rješenja

2.1 Zadatak 1

Zadatak je bio obisati kompletan sadržaj memorije mikrokontrolera. Prateći upute date postavkom zadatka, prekopirali smo *flash_nuke.uf2* datoteku putem USB kabla u memoriju mikrokonrolera. Naziv datoteke upućuje da je njen efekt destruktivan za sadržaj memorije.

To je omogućilo da na *čist* uređaj prekopiramo firmware potreban za rad sa MicroPython programskim jezikom. Konkretno, radilo se o verziji *v1.22.2* preuzete sa zvanične internet stranice MicroPython projekta [1].

2.2 Zadatak 2

Drugi zadatak zahtjevao je upoznavanje sa Thonny razvojnim okruženjem i interakciju sa Pico W razvojnom pločom putem *REPL* prompta. Zbog ličnih preferencija, odlučio sam se da koristim *Visual Studio Code* za koji je dostupna ekstenzija *MicroPico*[2] koja donosi sve potrebne funkcionalnosti. U ovom trenutku ne vidim nedostatak ovog okruženja te ga toplo preporučujem.

Nakon pokretanja u postavci navedenih komandi, upoznali smo se sa osnovnom sintaksom za interakciju sa pinovima na razvojnoj ploči.

2.3 Zadatak 3

Zadatak broj tri je imao za cilj upoznavanje mogućih interakcija sa pinovima u *output* načinu rada. Pomoću jednostavnih komandi, putem *REPL* prompta upalili smo LED-ove i omogućili paljenje istih putem tastera koji su ugrađeni na *picoETF*.

2.4 Zadatak 4

Ovaj zadatak tražio je pisanje Python skripte koja će na osnovu tastera T4 paliti ili gasiti LED7. Ovo se postiglo *while* petljom u čijem radnom bloku koda se provjerava stanje tastera i ažurira stanje LED-a.

Upoznali smo se sa postojanje mogućnosti prekidanja rada programa kombinacijom tipki *Ctrl + C*.

2.5 Zadatak 5

Ovaj programerski zadatak tražio je reprezentaciju binarnog inkrementiranja (taster T1) i dekrementiranja (taster T2) tako da osam LED-ova predstavlja bite u stanjima 1 ili 0 (upaljeno ili ugašeno). Dodatno, potrebno je da se tasterima T3 i T4 omogući gašenje ili paljenje svih LED-ova.

---

[1] https://micropython.org/download/RPI_PICO_W/

[2] https://marketplace.visualstudio.com/items?itemName=paulober.pico-w-go

Za svaku od četri opisane funkcionalnosti, napisana je odgovarajuća funkcija koja provodi odgovarajuće radnje nad nizom pinova. Tako imamo sljedeće relacije:

T1 increment(outputLEDs)
T2 decrement(outputLEDs)
T4 setZeroes(outputLEDs)
T5 setOnes(outputLEDs)

Slušanje pritiska tastera i poziv funkcija provodi se u main funkciji koja inicijalno postavlja sve diode na upaljeno stanje. Ovo za cilj ima da se odmah pri pokretanju može ustanoviti da li je program ispravno učitan, mada ne postoji obaveza za tim - može početno stanje biti uz ugašene diode.

Dodatno, provodi se blokiranje svih radnji, sve dok se drži neki taster. Ovime se spriječava npr. kontinuirano inkrementiranje držanjem tastera T1. Ovo blokiranje se provodi tako što se poziva *sleep* funkcija sve dok je T1 na vrijednosti *True*. Za više detalja, pogledati priloženu implementaciju.

2.6 Zadatak za dodatne bodove 1

REPL promptom pokrenute su komande kojima se pale i gase R, G i B pinovi za RGB LED koji je ugrađen na *picoETF* ploču. Prvo kreiramo objekte *red, green* i *blue* tipa Pin kojima upravljamo izlaznim pinovima redom 14, 12 i 13.

```
>>> red = Pin(14, Pin.OUT)
>>> green = Pin(12, Pin.OUT)
>>> blue = Pin(13, Pin.OUT)
```

Onda je potrebno paliti i gasiti pinove metodama *.on()* i *.off()* različitim kombinacijama kako bi postigli različite boje. Na primjer, dobijene boje su:

crvena + zelena = žuta
crvena + plava = ljubičasta
zelena + plava = tirkizna
crvena + zelena + plava = bijela

Konkretno, za postizanje žute boje koristimo komande:

```
>>> red.on()
>>> green.on()
```

2.7 Zadatak za dodatne bodove 2

Posljednji za datak tražio je da automatiziramo proces mijenjanja boja na RGB LED-u uz određnu pauzu između boja. Dodatno se tražilo da se pomenuta pauza inkrementira od 0.1 sekunde do 1 sekunde, te dekrementira od 1 sekunde do 0.1 sekunde nakon što se prikažu sve permutacije crvene, zelene i žute, te da se opisana promjena pauze ponavlja beskonačno.

Za rješavanje zadatka na pregledan način, uvode se pomoćne funkcije *setRGB(r, g, b)* i *permuteRGB(pauseTime)*. Prva funkcija postavlja stanje za svaku od tri osnovne boje na osnovu proslijeđenih parametara. Druga, za svaku permutaciju boja, poziva *setRGB* uz vremensku pauzu koja se proslijedi kao parametar.

Konačno, u *main* funkcij provodi se beskonačna *while* petlja koja radi inkrementiranje te dekrementiranje vremena pauze i poziva *permuteRGB*. Dodatno mimo specifikacije zadatke, u ovoj petlji se ispisuju stringovi *"max"* i *"min"* koji kroz terminal ukazuju na to da se maksimalno ili minimalno vrijeme pauze dostiglo.

3 Korišteni hardverski resursi

Za ovu laboratorijsku vježbu korišten je picoETF razvojni sistem koji ima integrisane sve elemente koji su korišteni.

Korišteni integrisani elementi su:

- LED (7x)
- RGB LED (1x)
- Tasteri (4x)

Kako su navedeni elementi dio same ploče razvojnog sistema, ovdje se neće ulaziti u detalje njihove implementacije.

4 Zaključak

Usput je bilo manjih problema sa implementacijom rješenja u MicroPython-u obzirom da Python dosta odudara od ostatka programskih jezika koje smo do sada radili. Konkretno, sintaksa je dosta slobodnija za različite konstrukcije koje suštinski postižu istu stvar ali nije slično jezicima kao što je C/C++, Java, C# ili Kotlin koje radimo na drugim predmetima koji su mnogo strožiji oko sintakse.

Prva laboratorijska vježba za cilj je imala upoznavanje sa laboratorijom, načinom rada, novim programskim jezikom i opremom. U tom cilju smo bili uspješni te su svi ponuđeni zadaci urađeni.

5 Prilog

5.1 Zadatak 5/izvorni kod

```
from machine import Pin
from time import sleep

T1 = Pin(0, Pin.IN)  # Increment
T2 = Pin(1, Pin.IN)  # Decrement
T3 = Pin(2, Pin.IN)  # Set sum to 0x00
T4 = Pin(3, Pin.IN)  # Set sum to 0xFF

LED0 = Pin(4, Pin.OUT)
LED1 = Pin(5, Pin.OUT)
LED2 = Pin(6, Pin.OUT)
LED3 = Pin(7, Pin.OUT)
LED4 = Pin(8, Pin.OUT)
LED5 = Pin(9, Pin.OUT)
LED6 = Pin(10, Pin.OUT)
LED7 = Pin(11, Pin.OUT)

outputLEDs = [LED0, LED1, LED2, LED3, LED4, LED5, LED6, LED7]

def setZeroes(outputLEDs):
    for pin in outputLEDs:
        pin.value(0)

def setOnes(outputLEDs):
    for pin in outputLEDs:
        pin.value(1)

def increment(outputLEDs):
    carry = 1

    for pin in outputLEDs:
        current = pin.value()

        if current == 1 and carry == 1:
            pin.value(0)
```

```
        else:
            new = current + carry
            pin.value(new)
            carry = 0

def decrement(outputLEDs):
    index = 0

    while index < len(outputLEDs):
        if outputLEDs[index].value() == 1:
            break
        else:
            index = index + 1

    if index != len(outputLEDs):
        outputLEDs[index].value(0)

    for i in range(index):
        outputLEDs[i].value(1)

def main():
    setOnes(outputLEDs)
    sleep(1)

    while True:

        if(T1.value()):
            increment(outputLEDs)
            while T1.value():
                sleep(0.1)

        elif(T2.value()):
            decrement(outputLEDs)
            while T2.value():
                sleep(0.1)

        elif(T3.value()):
            setZeroes(outputLEDs)
            while T3.value():
                sleep(0.1)
```

```
        elif(T4.value()):
            setOnes(outputLEDs)
            while T4.value():
                sleep(0.1)

        sleep(0.25)

if __name__ == "__main__":
    main()
```

5.1 Zadatak za dodatne bodove 1/izvorni kod

```
from machine import Pin
from time import sleep

red = Pin(14, Pin.OUT)
green = Pin(12, Pin.OUT)
blue = Pin(13, Pin.OUT)

def setRGB(r, g, b):  # True / False (1 / 0)
    red.value(r)
    green.value(g)
    blue.value(b)

def permuteRGB(pauseTime):
    for r in [False, True]:
        for g in [False, True]:
            for b in [False, True]:
                setRGB(r, g, b)
                sleep(pauseTime)

def main():

    while True:
        pauseTime = 0.1

        while pauseTime <= 1.0:
            permuteRGB(pauseTime)
```

```
            pauseTime += 0.1
        print("max")

        while pauseTime >= 0.1:
            permuteRGB(pauseTime)
            pauseTime -= 0.1
        print("min")

if __name__ == "__main__":
    main()
```