Univerzitet u Sarajevu
Elektrotehnički fakultet
Ugradbeni sistemi 2023 / 24

Izvještaj za laboratorijsku vježbu br. 6

Prekidi i tajmeri

Ime i prezime: **Ismar Muslić**
Broj index-a: **19304**

23. april 2024.

Sadržaj

1 Pseudokod 3
1.1 Zadatak 1 3
1.2 Zadatak 2 3
1.3 Zadatak 3 4
2 Analiza programskog rješenja 5
2.1 Zadatak 1 5
2.2 Zadatak 2a 5
2.3 Zadatak 2b 6
2.4 Zadatak 3 6
3 Korišteni hardverski resursi 7
4 Zaključak 7
5 Prilog 8
5.1 Zadatak 1a/Izvorni kod 8
5.2 Zadatak 1b/Izvorni kod 9
5.3 Zadatak 2a/Izvorni kod 10
5.4 Zadatak 2b/Izvorni kod 11
5.5 Zadatak 3/Izvorni kod 12

1 Pseudokod

Zbog genealne kompleksnosti implementacije svih zadataka, pseudokod je dat kao intuitivna ideja kako pristupiti rješavanju problema, dok su konkretne implementacije svih zadataka date u prilogu ovog izvještaja.

1.1 Zadatak 1

```
BusOut prikaz1
BusOut prikaz2

brojac1 = 0
brojac2 = 0

while(true)
      brojac1 = (brojac1 + 1) % 16
      prikaz1 = brojac1
      sleep(zadani_T)
      if(taster1 pritisnut)
        brojac2 = (brojac2 + 1) % 16

      prikaz2 = brojac2
```

1.2 Zadatak 2

```
BusOut prva_cifra
BusOut druga_cifra

DigitalOut signal_generator
AnalogIn input_signal

brojac = 0
while(true)
      signal_generator = 1
      wait(1ms)
      signal_generator = 0
      wait(1ms)

      if(input_signal)
        brojac++
        prva_cifra = brojac / 10
        druga_cifra = brojac % 10
        sacekajSilaznuIvicu()
```

1.3 Zadatak 3

```
brojac = 0
while(true)
      if(rotacija udesno)
        brojac++
      else if(rotacija ulijevo)
        brojac–

      if(pritisnut taster)
        brojac = 0
```

2 Analiza programskog rješenja

2.1 Zadatak 1

U prvom zadatku bilo je potrebno pružiti analizu datog koda koji se izvršava na LPC1114ETF. Zatim je bilo potrebno istu funkcionalnost implementirati korištenjem sistema prekida.

Dati program uključuje implementaciju dva brojača na po 4 LED diode ugrađene na sistemu, od kojih se prvi inkrementira svake vremenske jedinice (zadane varijablom T), dok se drugi inkrementira na pritisak tastera 1, s tim da se sa tastera unos očitava također svakih T vremenskih jedinica. Nedostatak ovog rješenja se ogleda u neispravnom funkcioniranju tastera, upravo zbog toga što se sa njega vrijednost očitava u pravilnim vremenskim intervalima (koji su usputno i predugi za ispravno funkcioniranje), a ne onda i samo onda kada je taster i pritisnut. Ukoliko se period očitavanja smanji, tada imamo problem s titranjem, dok ukoliko je period očitavanja veći, još je izraženiji problem neočitavanja kada je taster pritisnut. U praktičnom smislu, da bi se svaki pritisak na taster ispravno očitao, korisnik bi morao tačno da poznaje kada je program odbrojao period T, da bi u tom momentu držao taster pritisnutim. Ovo je očigledno besmisleno. Rješenje ovog problema se zasniva na implementaciji sistema prekida. Da bi se tražena implementacija provela u djelo sistemom prekida, potrebna su tri objekta: Ticker, Timer i InterruptIn.

Ticker u ovom slučaju služi za periodično inkrementiranje prvog brojača, što se postiže pozivanjem procedure prikaz_brojac1() kada Ticker otkuca zadani period.

Timer se koristi za debouncing na tasteru, da ne bi dolazilo do ponovljenog odlaska u prekidnu rutinu zbog titranja fizičkog tastera. Tek onda kada Timer za debouncing odbroji period od 200ms (unutar kojeg je malo vjerovatno da je taster zaista ponovno pritisnut), rutina za pritisak na taster će se ispravno izvesti na uređaju, te će se i brojač 2 inkrementirati. Ukoliko je prošlo manje od 200ms, smatra se da se na tasteru dogodio titraj, te se brojač ne inkrementira.

Dodatno, InterruptIn objekat je korišten za samo registrovanje pritiska na taster, gdje će se na uzlaznu ivicu sa tastera pokrenuti izvođenje prekidne rutine, te na osnovu debounce Timera brojač adekvatno inkrementirati. Detaljna implementacija ovog i ostalih rješenja je data u prilogu u odjeljku 5.

2.2 Zadatak 2a

Zadatak 2 podrazumijeva očitavanje ulaznog signala sa generatora koji stvara signal četvrtki promjenjive frekvencije, kao i generisanje izlaznog signala oblika četvrtki na pinu dp18 uređaja LPC1114ETF. Uslov zadatka je da se izlazni signal ispravno generiše, dok se sa ulaznog signala na svaku uzlaznu ivicu inkrementira brojač. Brojač je implementiran kao dvocifreni u BCD načinu kodiranja, na 8 LED dioda koje su integrisane u razvojni sistem.

Zadatak 2a podrazumijeva implementaciju ove funkcionalnosti bez korištenja sistema prekida, što je postignuto korištenjem BusOut objekata koji kontrolišu LED diode i ispisuju pravilno BCD cifre na istima, kao i AnalogIn objekta za očitavanje ulaznog signala, te DigitalOut objekta za kreiranje signala četvrtki frekvencije 500 Hz.

Signal se generiše na pinu dp18, koristeći spomenuti DigitalOut objekt kojem se vrijednost u periodu zadanom zadatkom mijenja sa 0 na 1.

Brojanje uzlaznih ivica ulaznog signala sa pina dp9 se realizira korištenjem spomenutog AnalogIn objekta, gdje se na svaku promjenu sa 0 na 1 brojač uvećava, a zatim, sve dok mu je vrijednost 1, u while petlji se čeka na sljedeću promjenu, odnosno na 0.

Ovo rješenje, međutim, ne funkcionira ispravno za velike frekvencije. U odnosu na to koja je funkcionalnost kodirana prva u while(true) petlji, ona će se ispravno izvršavati, dok će funkcionalnost koja je poslije iste, zbog vremena kašnjenja u izvođenju programa neispravno funkcionirati. Naime, za frekvencije ulaznog signala veće od 500 Hz, jedna od ove dvije funkcionalnosti, bilo brojač uzlaznih ivica ili generator signala, neispravno funkcionira.

Bitno je naglasiti da se korištenjem PWM-a za generisanje signala ovaj problem adekvatno rješava bez upotrebe sistema prekida. Razlog tome je što je PWM modul potpuno fizički neovisan od samog procesora, i na taj se način fizički odvajaju funkcionalnosti, te zbog toga ne dolazi do interferencije između istih. Međutim, ukoliko ne želimo generisati signal koristeći PWM, rješenje je koristiti sistem prekida.

2.3 Zadatak 2b

Korištenje sistema prekida za ovaj zadatak uključuje deklaraciju jednog Tickera za generisanje signala i jednog InterruptIn objekta za brojanje uzlaznih ivica ulaznog signala. Na svaki otkucaj perioda Tickera, vrijednost DigitalOut objekta na pinu dp18 se toggle-a, čime se generiše signal u zadanom periodu. S druge strane, na svaku uzlaznu ivicu signala na pinu dp9 se pokreće interrupt rutina koja inkrementira brojač i prikazuje vrijednost na LED diodama u BCD formatu. Ovakvo rješenje funkcionira sasvim ispravno za frekvencije koje su reda MHz.

2.4 Zadatak 3

U trećem zadatku bilo je potrebno programirati razvojni sistem picoETF za rad s inkrementalnim digitalnim enkoderom. Enkoder se koristi za kontrolisanje LED dioda integrisanih u razvojni sistem, a kompletan sklop funkcionira ponovno kao brojač. Za otklon enkodera udesno za jednu poziciju, brojač se inkrementira za 1, a za otklon ulijevo se umanji za 1. Pritiskom na dugme ugrađenu u ručicu enkodera, brojač se resetira na 0.

Implementacija sistemom prekida je postignuta korištenjem prvenstveno deklariranih objekata encoder_clk, encoder_dt, kao i encoder_click, koji predstavljaju respektivno signale clk i dt sa enkodera, kao i spomenuti taster na istom.

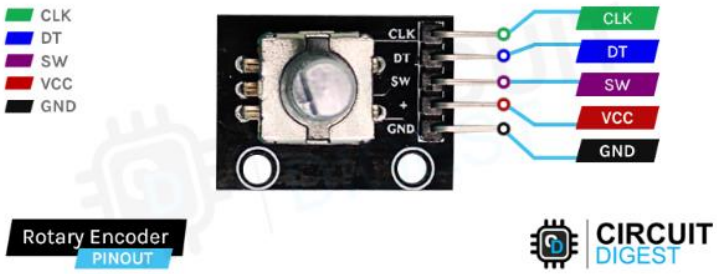
Enkoder funkcionira po principu izjednačavanja faze dva signala, na sljedeći način: kada je okinut impuls na 0 na signalu CLK, ukoliko je DT u istoj fazi sa njim, tada je enkoder okrenut ulijevo, a ukoliko je DT u suprotnoj fazi (ima vrijednost 1), enkoder je u tom slučaju okrenut udesno. S tim u vidu je implementirana funkcionalnost hardverskog prekida na ulazu encoder_clk, kao i na encoder_click. U momentu kada je encoder_clk na silaznoj ivici, pokreće se hardverski prekid i ispituje se stanje signala dt, te se brojač podešava prema prethodno navedenoj logici. Također, kada je taster za reset na uzlaznoj ivici, pokreće se rutina koje resetira brojač. S obzirom na dosta izražene titraje prilikom otklona enkodera, ubačen je sleep u trajanju od 100ms unutar glavne while(true) petlje.

3 Korišteni hardverski resursi

Za potrebe ove laboratorijske vježbe korišteni su sistemi LPC1114ETF, kao i picoETF, te ugrađene LED diode na istima. Osim toga, od dodatne opreme korišteno je sljedeće:

- Inkrementalni rotacijski enkoder
- Osciloskop (za ispitivanje analognog signala u zadatku 2, na sistemu LPC1114ETF)

ULAZI	IZLAZI
Rotacijski enkoder (digitalni; picoETF, GP0 (clk), GP1(dt), GP2(sw))	LED0 – LED7 (digitalni; LCP1114ETF P0_0 – P0_7)
	LED0 – LED7 (digitalni; picoETF GP4 – GP11)



Slika 1 – Pinout enkodera korištenog u zadatku 3

4 Zaključak

Prilikom izrade laboratorijske vježbe 6 nije bilo poteškoća. Jedina prepravka koja je bila potrebna jeste korištenje standardnog DigitalOut objekta u zadatku 2, umjesto PWM objekta, na šta je predmetni asistent skrenuo pažnju. Također, u zadatku 3 je bilo potrebno uvesti značajno dug sleep time u while(true) petlji, da bi se umanjio efekat titranja prilikom otklona enkodera. Cilj vježbe je bio upoznavanje sa sistemom prekida i tajmerima, što je uspješno i postignuto.

5 Prilog

5.1 Zadatak 1a/Izvorni kod

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

BusOut prikaz1(LED3, LED2, LED1, LED0);
BusOut prikaz2(LED7, LED6, LED5, LED4);
DigitalOut enable(LED_ACT);
DigitalIn taster(Taster_1);

const float T(0.2);
// const float T(2.0);
/*
Za T=2.0 se promjena na ledicama desava svako 2s, ali je nedostatak i to sto se
stanje tastera takodjer očitava na svako 2s i nije moguće da do izmjene dođe u
kraćem vremenskom periodu. Ako je period kraci, imamo problem sa titranjem.
*/

int brojac1(0);
int brojac2(0);

int main() {
    enable=0;

    prikaz1=brojac1;
    prikaz2=brojac2;

    while(1) {
        wait_us(T*1e6);
        brojac1=(brojac1+1)%16;
        if (taster) brojac2=(brojac2+1)%16;
        prikaz1=brojac1;
        prikaz2=brojac2;
    }
}
```

5.2 Zadatak 1b/Izvorni kod

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

BusOut prikaz1(LED3, LED2, LED1, LED0);
BusOut prikaz2(LED7, LED6, LED5, LED4);
DigitalOut enable(LED_ACT);
InterruptIn taster(Taster_1);

Ticker t;
Timer debounce;

const float T(0.2);
//const float T(2.0);

int brojac1(0);
int brojac2(0);

void prikaz_brojac1(){
    brojac1=(brojac1+1)%16;
    prikaz1=brojac1;
}

void prikaz_brojac2(){
    if(debounce.read_ms()>=200){
        brojac2=(brojac2+1)%16;
        prikaz2=brojac2;
        debounce.reset();
    }
}

int main() {
    enable=0;
    prikaz1=0;
    prikaz2=0;

    //Timer za otitravanje
    debounce.start();

    //procedura za kreiranje signala. Period 0.2s
    t.attach(&prikaz_brojac1, T);

    //rutina za inkrementiranje i prikaz brojaca
    taster.rise(&prikaz_brojac2);
    while(1) {
        wait_us(1);
    }
}
```

5.3 Zadatak 2a/Izvorni kod

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

DigitalOut E(LED_ACT);
BusOut prva_cifra(LED3, LED2, LED1, LED0);
BusOut druga_cifra(LED7, LED6, LED5, LED4);

DigitalOut generisani(dp18);
AnalogIn signal(dp9);

// Procedura za ispis cifara u BCD formatu
void ispisi(int &brojac) {
    if(brojac>99) brojac=0;
    druga_cifra=brojac%10;
    prva_cifra=brojac/10;
}

int main() {
    E=0;
    prva_cifra=0;
    druga_cifra=0;

    int brojac=0;

    while (true) {

        generisani=1;
        wait_us(1000);
        generisani=0;
        wait_us(1000);

        if(signal.read()==1){
            brojac++;
            ispisi(brojac);
            while(signal.read()==1) continue;
        }
    }
}

/*
radi OK do ~500Hz
nakon toga jedna od funkcija ne radi kako treba u zavisnosti od toga koja je
primarna
*/
```

5.4 Zadatak 2b/Izvorni kod

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

DigitalOut E(LED_ACT);
BusOut prva_cifra(LED3, LED2, LED1, LED0);
BusOut druga_cifra(LED7, LED6, LED5, LED4);
DigitalOut generisani(dp18);

InterruptIn signal(dp9);
Ticker t;

int brojac=0;

// Procedura za ispis cifara u BCD formatu
void ispisi() {
    brojac++;
    if(brojac>99) brojac=0;
    druga_cifra=brojac%10;
    prva_cifra=brojac/10;
}

void generisanje_toggle(){
    generisani = !generisani;
}

int main() {
    E=0;
    generisani=0;
    // Ticker za generisanje signala
    t.attach_us(&generisanje_toggle, 1000.0);

    // Na svaku uzlaznu ivicu inkrementiraj brojac
    signal.rise(&ispisi);

    while (true) {
        wait_ns(1);
    }
}

// Bez obzira na visinu frekvencije radi ispravno
```

5.5 Zadatak 3/Izvorni kod

```
from machine import Pin
from time import sleep

sleep(0.1)

leds = [Pin(i, Pin.OUT) for i in range(4, 12)]
encoder_clk = Pin(0, Pin.IN)
encoder_dt = Pin(1, Pin.IN)
encoder_click = Pin(2, Pin.IN)

brojac = 0

# Rutina za rotaciju enkodera
def interrupt_routine(pin):

    global brojac
    if encoder_dt.value() == 0:
        brojac -= 1
    else:
        brojac += 1

# Rutina za pritisak na taster
def click(pin):

    global brojac
    brojac = 0

# Definiranje rutine za rotaciju
encoder_clk.irq(handler=interrupt_routine, trigger=Pin.IRQ_FALLING)

#Definiranje rutine za pritisak na taster
encoder_click.irq(handler=click, trigger=Pin.IRQ_RISING)

# Glavna petlja, update-anje dioda
while True:
    for i in range(8):
        leds[i].value((brojac>>i)&1)
    sleep(0.1)
```