UNIVERZITET U NOVOM SADU
FAKULTET TEHNIČKIH NAUKA
NOVI SAD
Departman za računarstvo i automatiku
Odsek za računarsku tehniku i računarske komunikacije

Software PWM over busy wait

PROJEKTNI ZADATAK

Kandidat: Aleksandar Ćetković RA 24/2021

Kandidat: Vuk Antović RA 52/2021

Kandidat: Ana Mujić RA 58/2021

Kandidat: Stefan Mitić RA 216/2021

Predmet: Operativni sistemi u realnom vremenu

Mentor rada: dr Miloš Subotić

Novi Sad, decembar, 2023.

Sadržaj

1 Uvod

1.1 Opis problema

Naš zadatak se svodi da napravimo softversko rešenje za tvrdi operativni sistem u realnom vremenu. Konkretnije, naši zadaci su:

- Logovanje PWM signala i vremenskih oznaka događaja
- Forsiranje operativnog sistema da u slučaju da ne stiže da završi izvršavanje datog događaja na vreme, da isti ne započinje.
- Iscrtavanje PWM signala, kašnjenja i distribucije kašnjenja pomoću skripte u Juliji

1.2 Teorijske osnove

Da bismo mogli da uspešno uradimo projekat, prvo moramo da razumemo neke osnovne pojmove i šta se zapravo dešava "ispod haube".

Prvo krećemo od pojma tvrdi operativni sistem u realnom vremenu, odnosno *hard real time operative system.*

Operativni sistemi u realnom vremenu postavljaju određene vremenske rokove do kad je potrebno da se neki zadatak uradi. U zavisnosti od načina rukovanja zadacima u slučaju kašnjenja, operativne sisteme u realnom vremenu možemo podeliti na "tvrde"(*hard real time*) i "meke"(*soft real time*). Tvrdi operativni sistem u realnom vremenu ne dopušta kašnjenje i u slučaju da zna da neki zadatak neće stići da se završi, on ga neće ni započeti. Meki operativni sistem u realnom vremenu dopušta određeno kašnjenje i dopušta početak zadataka koji se potencijalno neće završiti na vreme.

Mi ćemo se baviti *hard real time* operativnim sistemom i implementirati rukovanje zadacima u slučaju njihovog kašnjenja.

Da bismo to uradili, potrebno je takođe da razumemo šta je PWM signal.
PWM (*Pulse Width Modulation*) odnosno širinsko-impulsna modulacija jeste tehnika dobijanja analognih vrednosti pomoću digitalnih impulsa konstantne amplitude. Radi jednostavnosti, u nastavku teksta ćemo koristiti isključivo pojam PWM.

Uz pomoć PWM signala, mi kontrolišemo hardver koji se pomera na određeni način (ugao pomeraja i slično).

1.3 Analiza problema

Problem smo podelili na tri dela.

Prvo,logujemo vremenske oznake (*timestamps*) da bismo mogli da ispratimo vremena zadataka i da li kasne.

Dalje, ograničavamo operativni sistem da ignoriše svaki zadatak čiji početak kasni i svaki zadatak čiji bi završetak bio nakon roka.

I na kraju potrebno je prikazati na grafiku sve vremenske oznake kada se sve implementira.

2 Rešenje problema

Rešenje ćemo prikazati u tri etape, koje su navedene u analizi problema.

Poenta zadatka jeste da sprečimo izvršavanje bilo kakvih zadataka koji kasne sa početkom ili završetkom.

Da bismo to postigli, potrebno je prvo da ispišemo vremenske oznake događaja, odnosno zadataka. Zatim je potrebno da implementiramo same "blokade"koje sprečavaju da se događaj desi ako ne može da se izvrši do kraja do zadatog roka, a da bismo proverili da li sve to radi i da bismo lakše otklonili potencijalne greške, potrebno je da izvršimo vizuelizaciju vremenskih oznaka i događaja.

Sve relevantne informacije (vremenske oznake, stanja, da li kasni ili ne, koliko kasni ako dođe do kašnjenja...) možemo prikazati uz pomoć grafika u okviru *Plots* paketa u Juliji.

2.1 Logovanje vremenskih oznaka PWM signala

Da bismo uopšte mogli da upisujemo vremenske oznake u fajl, pre svega moramo da inicijalizujemo logovanje pozivom funkcije *log__init* u funkciji *main* datoteke *main.c*.

Funkcija nema argumente, povratna vrednost joj je samo potvrda uspešnosti (0, 1 ili -1).

U *log__init* pravi se fajl u koji će se upisati vremenske oznake.

Dalje, u funkciji *busy_pwm_loop* u datoteci *busy_pwm.c* u kojoj se vrti petlja i određuje vremena promene stanja PWM signala, pozivamo funkciju *log__add* koja doda razliku vremena buđenja niti i zadatog vremena u log fajl. U log fajlu imamo i da li je vrednost PWM 1 ili 0 i imamo podatak sa kog kanala se upisuje.

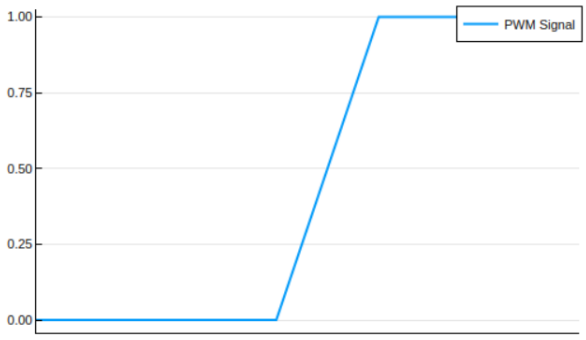
2.2 Hard Real Time

U fajlu *sw_pwm.c* u funkciji *busy_pwm_loop* imamo beskonačnu petlju koja obrađuje događaje. Tu je bitan dodatak uslov ukoliko je vreme događaja, odnosno krajnji rok izvršavanja događaja, manje od trenutnog vremena (na primer rok događaja je 700ms od početka programa a trenutno vreme je 1000 ms) onda ignorišemo taj događaj i prelazimo na sledeći jer smo zakasnili.

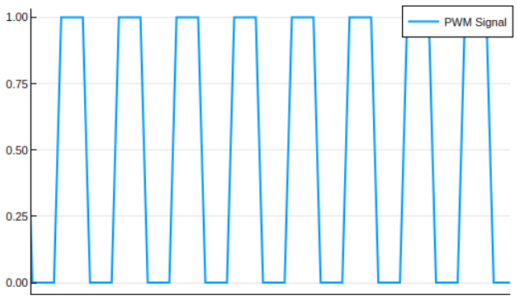
Ukoliko događaj zakasni, nit se uspava i čeka do roka sledećeg događaja.

2.3 Vizuelizacija pwm signala u Juliji

Pre svega, možemo prikazati kako izgleda PWM signal i njegova rastuća ivica.



Slika 1: Rastuća ivica PWM signala



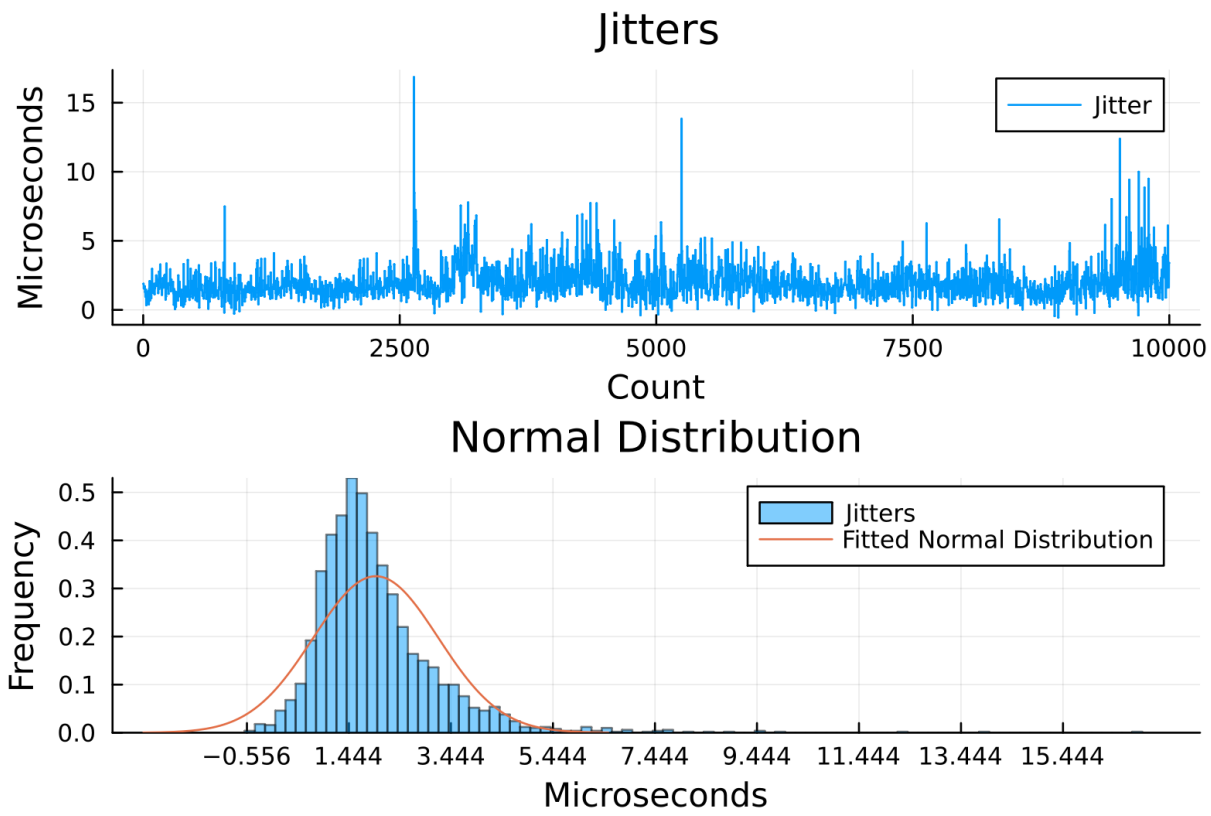
Slika 2: PWM signal, ispune 50

Da bismo od datoteke koja sadrži vremenske oznake dobili nešto čitljivo, potrebno je u Juliji otvoriti fajl i parsirati liniju po liniju.

Koristimo tri niza, jedan za vremenske oznake kašnjenja, jedan za stanja signala (0 ili 1) i jedan za pin za koji je opisan događaj.

U svakoj liniji odvajamo podatak u odgovarajuć niz i plotovaćemo isključivo kašnjenje da bismo videli kašnjenja kroz vreme i statistički obradili.

Na kraju plotujemo kašnjenje, i vršimo aproksimaciju krive distribucije kašnjenja.



Slika 3: Grafici kašnjenja i distribucije

3 Pokretanje rešenja

Potrebno je navigirati se u folder *SW/Driver/motor_ctrl* i u terminalu ukucati komande:

```
sudo make

sudo make start
```

Listing 1: Pokretanje programa

Dalje, da bi se pokrenula skripta za iscrtavanje, potrebno je navigirati se u folder *SW/Scripts* i u pokrenuti Juliu. Za pokretanje skripte koja radi u realnom vremenu:

```
include("plot_real_time.jl")
```

Listing 2: Pokrtanje Julia skripte za grafik kašnjenja

A za pokretanje skripte koja se ne menja kroz vreme:

```
include("plot.jl")
```

Listing 3: Pokrtanje Julia skripte za iscrtavanje distribucije kašnjenja

4 Zaključak i završna reč

Za kraj, ceo proces projekta je iziskivao mnogo više vremena razmišljajući o načinu rada i implementiranju algoritma nego samoj implementaciji.

Korisni linkovi:

- GitHub repozitorijum: https://github.com/OSSuRV-PWM-Busy-wait/OSuRV_2023/tree/main/PWM/SW
- YouTube link: https://youtu.be/mqTBjuDcG8A