|  |
| --- |
| Univerzitet u Novom Sadu  Fakultet tehničkih nauka  Novi Sad  Departman za računarstvo i automatiku  Odsek za računarsku tehniku i računarske komunikacije  **Servo ctrl in Multitasking OS**  **Predmet: Operativni sistemi za rad u realnom vremenu**  Autori: Mentor:  Luka Šarenac - RA36/2020 Miloš Subotić  Tomislava Nikolić – RA235/2020 Miloš Pilipović  Lea Bujas - RA198/2020  Zoran Vidović – RA195/2020  **Novi Sad, 2022.** |
|  |

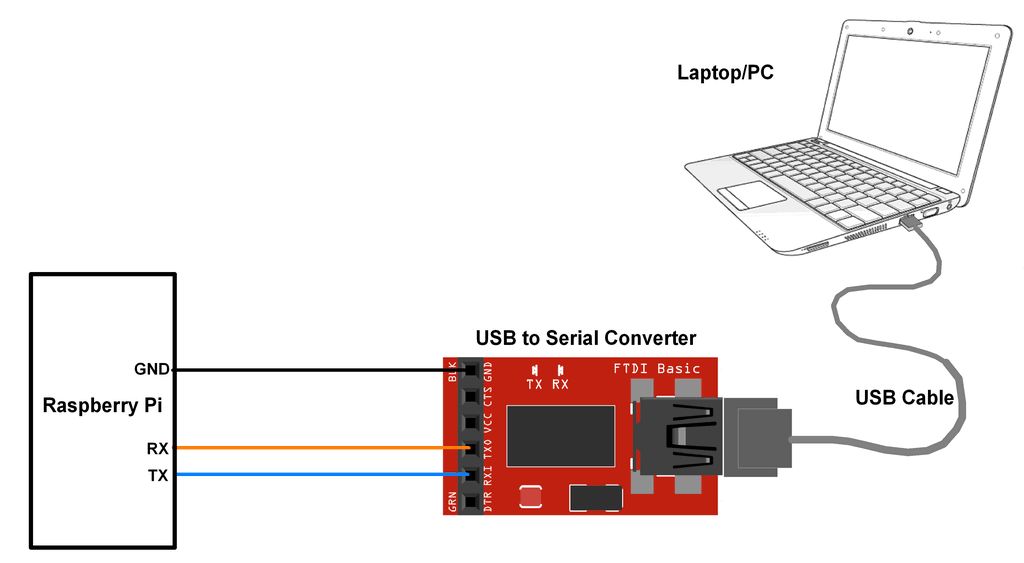
**1.UVOD**

Cilj našeg projektnog zadataka bio je rešiti problem kontrole servo motora u multiprocesorskom okruženju implementacijom rešenja unutar kernela, tako što ćemo generisati PWM signale pomoću tajmera. Te signale ćemo slati preko odabranog GPIO pina na Raspberry Pi-ju.

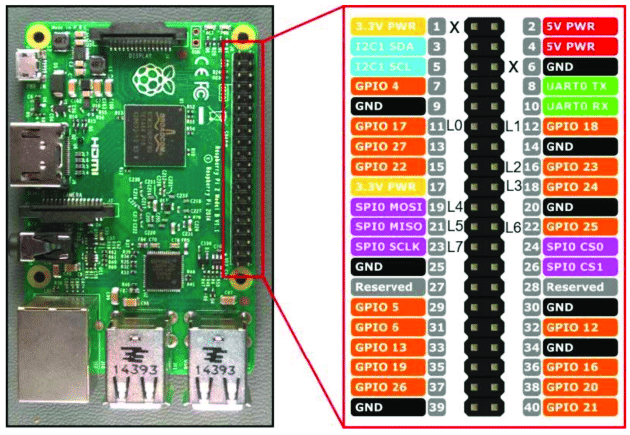
**2. REŠAVANJE PROBLEMA**

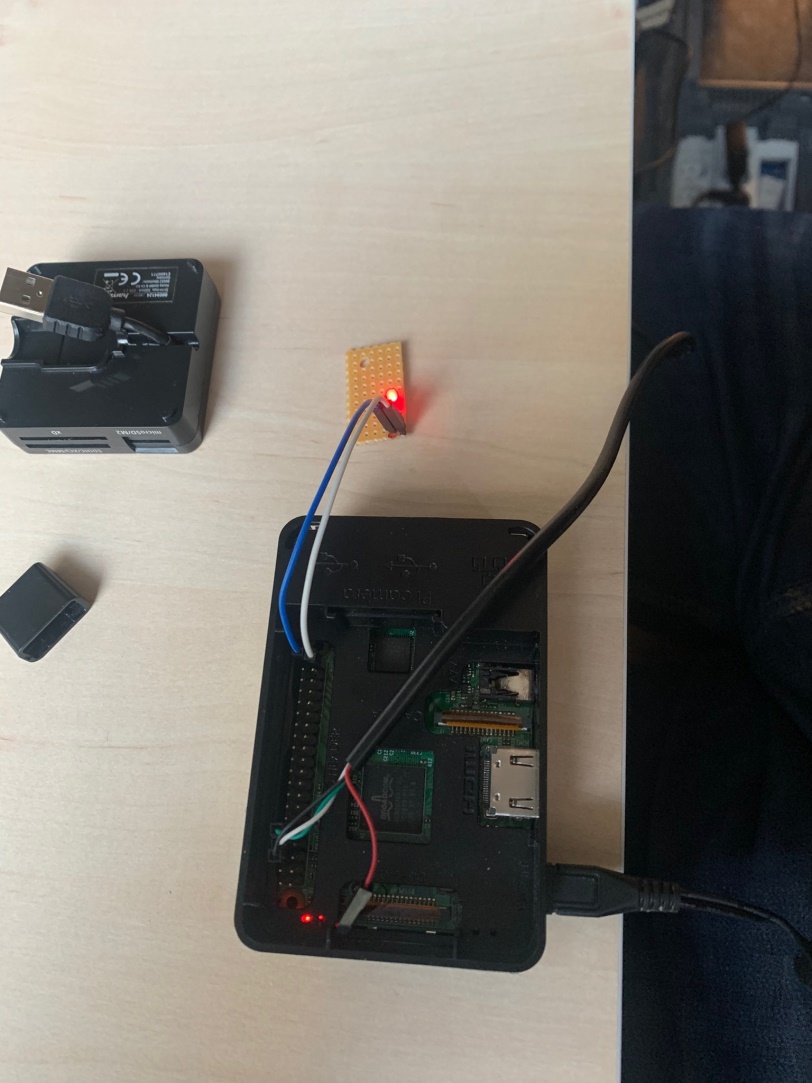
**2.1. Okruženje**

Na početku projekta bilo je potrebno namestiti samo okruženje za rad koje se sastojalo iz više koraka, kao primer koristili smo materijal iz vežbe 8 i na sledeći način podešavali okruženje:

1. Da bi se omogućilo prevođenje koda pisanog u programskim jezicima C i C++ neophodno je bilo instalirati skup alata za prevođenje koda za ARM procesore.
2. Nakon što smo to uspešno instalirali, za samo pokretanje koristili smo skriptu build.sh zajedno sa našom ciljanom arhitekturom rpi2 kao i primerom koji nam je koristio samo kao aktivacija.
3. Nakon uspešnog prevođenja u direktorijumu out napravljena je datoteka kernel.img, koja predstavlja preveden kod višeprocesorskog okruženja. Da bismo isti pokrenuli potrebno je ovu datoteku kopirati na memorijsku karticu.
4. Kartica je pre prvog pokretanja formatirana u formatu FAT32 i na nju je prekopiran sadržaj direktorijuma bootfiles, nakon čega smo posle svake izmene mogli kopirati samo kernel.img datoteku.
5. Ubacivanjem kartice u RPI i spajanjem napajanja kao i spajanjem sa računarom preko UART-a koji smo povezali na odgovarajuće pinove naše okruženje je spremno.
6. Kada je adapter povezan novi serijski prolaz /dev/ttyUSB0 isti ćemo koristiti da bismo komunicirali sa pločom kroz serijski prolaz i to uz pomoć picocom-a, kao jednog od programa za serijsku komunikaciju.

**2.2 Rad sa LED diodom**

 Prvo smo proverili rad diode setovanjem odgovarajućeg pina na High pozivanjem funkcije *RPI\_SetGpioHi( RPI\_GPIO21 );*



Kako bismo odradili simulaciju našeg rešenja koristili smo LED diodu na kojoj smo simulirali paljenje i gašenje na određenu periodu u funkciji *void TimeInt (struct STACK\_FRAME \* reg),* implementiranjem softverskog brojača koji je brojao do 100 i koji se inkrementovao prilikom svakog poziva funkcije. Setovanjem pina na High vrednost 50 puta i na Low vrednost 50 puta, dobili smo rezultat da se naša dioda pali i gasi na period od 1 sekunde, ovim smo zaključili da naša funkcija radi na 20ms, što je pokazalo da je početna pretpostavka mentora Miloša bila pogrešna.

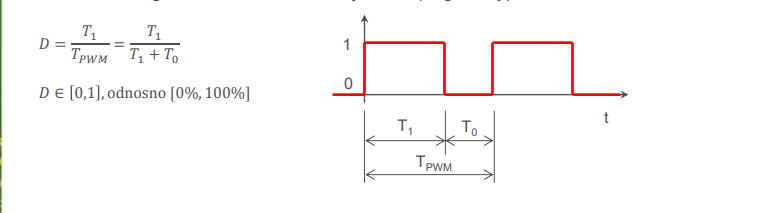
**2.3 Generisanje PWM signala**

Dalje je bilo potrebno ubrzati rad sistema, kako bismo isti mogli da usporimo kako bi dobili odgovarajuće vrednosti za generisanje signala. Ubrzali smo ga tako što smo *RPI\_ARMTIMER\_CTRL\_PRESCALE\_256;* setovali na *RPI\_ARMTIMER\_CTRL\_PRESCALE\_1;* Nakon toga usporavanje smo postigli tako što smo izmenili da brojač broji po modulu 2. I ovu izmenu testirali smo na diodi.

**2.4 Osciloskop**

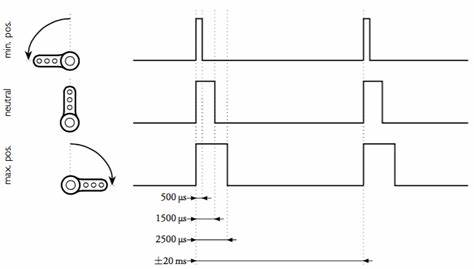
Za dalji rad, nije nam bila dovoljna dioda, jer nam je bilo potrebno da merimo frekvenciju kao i faktor ispune (Duty) kako bismo iste mogli da podesimo za pokretanje motora.

Tražena frekvencija bila je 50Hz, dok je faktor ispune za kretanje motora u pravcu kazaljke na satu bio 3, a za smer suprotan od pravca kazaljke na satu faktor ispune bio je 10.

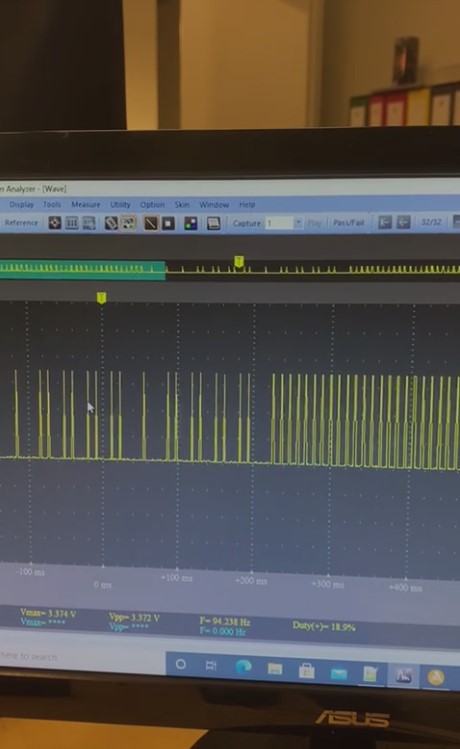


**2.5 Pokretanje motora**

Nakon uspešno dobijenih vrednosti, više nam nije bilo potrebno da nam se vrednosti signala smenjuju na 50, već je bilo potrebno da signal oko 3% periode bude High, dok je ostatak periode odnosno oko 97% na Low, za kretanje motora u smeru suprotnom od kazaljke na satu, dok je 6% periode na High i 94% periode na Low rezultovalo kretanjem motora u smeru kazaljke na satu.

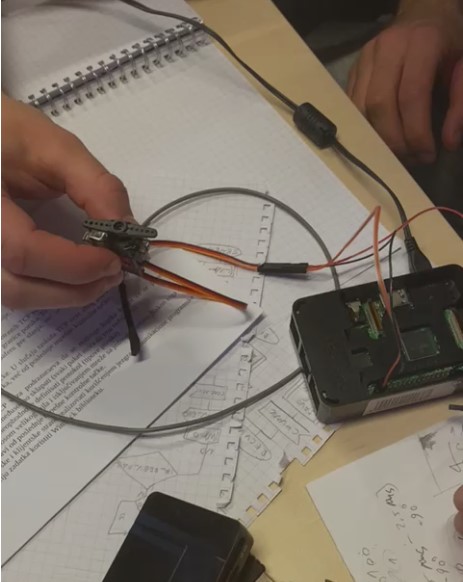


Ovim smo uspeli da pokrenemo motor da se kreće samo u jednu ili drugu stranu. Nakon ovoga bilo je potrebno omogućiti da se motor konstantno kreće u obe strane.

To smo obezbedili pomoću trećeg brojača koji je omogućio da se perioda za kretanje u desno ili levo šalje veći broj puta, odnosno 60 puta u našem slučaju, takođe menjanjem ove vrednosti omogućili smo i usporavanje motora.

Na slici sa osciloskopa vidimo više uzastupnih istih perioda, koje se smenjuju.

Još jedan način za usporavanje motora jeste spajanje na pin od 3.3V umesto na pin od 5V, obaranjem napona oborićemo i samu brzinu motora.



<https://drive.google.com/file/d/1l5_MFVSEspAiY4itsrsNO1azu_qhXCsC/view?usp=sharing>

Kao finalni rezultat projekta dobili smo motor koji pokretanjem izvršava traženu „koreografiju“.

**3. ZAKLJUČAK**

Tokom izrade ovog projekta naučili smo rad unutar multitasking OS-a, način na koji se generišu PWM signali i kako se pomoću njih upravlja servo motorom. Takođe, razumeli smo komunikaciju između RPI-ja i periferijskih uređaja sa kojima se spaja preko GPIO pinova kao i kako se istima upravlja.

Rad na ovom projektu naučio nas je kako da analitički, odnosno step-by-step prilazimo rešavanju samog problema, pre svega razumevajući samu materiju, a tek onda primenjujući naučeno.