

Raspberry Pi

Živković, Nikola

Undergraduate thesis / Završni rad

2017

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Department of Mathematics / Sveučilište Josipa Jurja Strossmayera u Osijeku, Odjel za matematiku**

Permanent link / Trajna poveznica: <https://um.nsk.hr/um:nbn:hr:126:616288>

Rights / Prava: [In copyright](#) / [Zaštićeno autorskim pravom.](#)

Download date / Datum preuzimanja: **2024-11-07**



mathos

Repository / Repozitorij:

[Repository of School of Applied Mathematics and Informatics](#)



Sveučilište J.J. Strossmayera u Osijeku
Odjel za matematiku

Nikola Živković
Raspberry Pi
Završni rad

Osijek, 2017

Sveučilište J.J. Strossmayera u Osijeku
Odjel za matematiku

Nikola Živković
Raspberry Pi

Završni rad

Mentor: izv. prof. dr. sc. Domagoj Matijević

Osijek, 2017

Sadržaj

Uvod	viii
1 Raspberry Pi	1
1.1 Hardver	2
1.2 Procesor	4
1.3 GPIO iglice	5
1.4 Operacijski sustav	8
1.4.1 Linux	8
1.5 Python	9
2 Projektni zadatak	11
2.1 Vrste motora	11
2.1.1 DC motor	13
2.1.2 Servo motor	14
2.1.3 Stepper motor	15
2.2 Upravljanje motorima	17
Literatura	26

Popis slika

1	Raspberry Pi 1 Model B, prva verzija Raspberry Pi	ix
1.1	BBC Micro	1
1.2	Shema hardvera Raspberry Pi modela B	2
1.3	Shema hardvera Raspberry Pi modela A i Zero	2
1.4	Donja strana Raspberry Pi-a	3
1.5	Gornja strana Raspberry Pi-a	4
1.6	Opći prikaz sustav na čipu	5
1.7	Shema GPIO iglice	6
1.8	Shema električnog kruga	6
1.9	Električni krug sa Raspberry Pi-om	7
1.10	Dijagram GPIO iglica sa svim njihovim imenima i legendom što koja boja označava	7
1.11	Tux - maskota Linuxa	8
1.12	Logo Raspbiana	9
1.13	Python 3.4.2 Shell	10
2.1	Pametni telfon kojeg pokreće Raspberry Pi	11
2.2	Vrste motora	12
2.3	DC motor	13
2.4	Princip rada DC motora	14
2.5	Servo motor	15
2.6	Shematski prikaz principa rada servo motora	16
2.7	Stepper motor	17
2.8	PWM prikazan grafički	18
2.9	Upravljanje DC motorima	20
2.10	Kako spojiti DC motore da bi mogli upravljat sa oba istovremeno	20
2.11	Upravljanje servo motorom	22
2.12	Kako spojiti stepper motor	24
2.13	Upravljanje stepper motorom	25

Popis tablica

1.1	Razlike modela	3
-----	--------------------------	---

Popis programskih kodova

2.1	Enkapsulacija klase Motor	20
2.2	Enkapsulacija klase ABMotors	21
2.3	Glavna skripta pomoću koje pokrećemo DC motore	22
2.4	Skripta pomoću koje pokrećemo servo motor	23
2.5	Skripta pomoću koje pokrećemo stepper motor	25

Title

Raspberry Pi

Sažetak

Raspberry Pi je jeftino mikroročunalo veličine bankovne kartice. Kreirala ga je Raspberry Pi fondacija s ciljem da poboljša i olakša početnicima učenje u računarstvu. Hardver se razvijao i poboljšavao sa svakom novom verzijom a trenutni procesor je Broadcomov SoC koji sadrži ARM-ove jezgre procesora koje također možemo pronaći i kod pametnih telefona. Najvažniji dio njegovog hardvera su svakako GPIO iglice koje mu i donose praktičnu važnost jer pomoću njih može kontrolirati različite uređaje. Kako je Raspberry Pi računalo to znači da mu je potreban i operacijski sustav. Specijalno u njegovom slučaju potreban mu je operacijski sustav na čijem se izvornom kodu mogu raditi modifikacije i dodavati nove stvari. Operacijski sustav takvog tipa je Linux i njegove distribucije, a najpopularnija Linuxova distribucija za Raspberry Pi je Raspbian koji dolazi u paketu sa hrpom dodatnih mogućnosti. Jedna od tih mogućnosti je i programski jezik Python u kojemu se mogu napisati programski kodovi za kontrolu uređaja. Upravo sve ove značajke, njegova mala veličina, mala cijena i mogućnost pokretanja operacijskog sustava čine ga savršenim za kontrolu manjih uređaja poput motora kod mobilnih robota.

Ključne riječi

Raspberry Pi, Mikoročunalo, Linux, SoC, Mikroprocesor, Broadcom, ARM jezgra procesora, GPIO iglice, Operacijski sustav, Linux, Raspbian, Python, IDLE, RPi.GPIO, Pigiopio, Mikrokontroler, Motor, DC motor, Servo motor, Stepper motor

Abstract

Raspberry Pi is a microcomputer size of a bank card and very low price. Its created by the Raspberry Pi Foundation to improve and facilitate beginners learning in computer science. Hardware Raspberry Pi has developed and improved with every new version, and its processor is Broadcom's SoC that contains ARM processor cores, ARM processor cores can be found in smartphones. The most important part of its hardware is certainly GPIO pins that give it great importance, with which they can control some devices. As Raspberrypi is a computer that means it needs an operating system, in particular, it needs an open source operating system. The operating system of this type is Linux and its distributions, the most popular Linux distribution for Raspberry Pi is Raspbian, that comes packed with many additional features. One of these feature is also the Python programming language in

which you can write program codes to control some devices. All these features; its small size, low cost, the ability to run operating systems make it perfect for controlling some smaller devices, such as robot engines.

Key word

Raspberry Pi, Microcomputer, Linux, SoC, Microprocessor, Broadcom, ARM processor core, GPIO pins, Operating system, Linux, Raspbian, Python, IDLE, RPi.GPIO, Pigiopio, Microcontroller, Motor, DC motor, Servo motor, Stepper motor

Uvod

U ovom radu će biti opisan Raspberry Pi, maleno računalo razvijeno u Ujedinjenom Kraljevstvu, a proizvodi se u Sony-evoj tvornici u Wales-u, u gradu Pencoed. Razvila ga je Raspberry Pi zaklada s jednim ciljem, a to je da pomoću njega u školama omogući učenje sadržaja vezanog za računarstvo, znanosti koja se u to vrijeme razvijala velikom brzinom. Međutim, taj njihov prvi cilj vrlo brzo se našao u drugom planu. Razlog tomu je što se Raspberry Pi, osim za učenje, počeo koristiti i u robotici, te područjima sličnim njoj. Od trenutka, kada su ga kao platformu za rad prihvatili mnogi inženjeri, proizvođači te ambiciozni ljudi koji se bave projektima s područja elektronike prodaja mu je naglo porasla čemu svjedoče i podatci koje je Raspberry Pi zaklada objavila. Podaci govore da je do veljače 2015. godine ukupno prodano 5 miliona primjeraka, što mu je donijelo titulu najprodavanijeg britanskog računala. U studenom prošle godine ova brojka je iznosila već vrtoglavih 11 miliona. Priči, što se tiče broja prodanih primjeraka tu nije bio kraj i u iduća četiri mjeseci prodano je još 1,5 miliona komada čime je ukupan broj prodanih primjeraka Raspberry Pi-a iznosio 12.5 miliona. Zato je u ožujku 2017. godine postao treće najprodavanije računalo.

Do danas je promovirano nekoliko modela i verzija Raspberry Pi-a, a izlazile su sljedećim redoslijedom:

- Raspberry Pi 1 Model B, Travanja 2012
- Raspberry Pi 1 Model A, Travanja 2013
- Raspberry Pi 1+ Model B, Srpanj 2014
- Raspberry Pi 1+ Model A, Studeni 2014
- Raspberry Pi 2 Model B, Veljača 2015
- Raspberry Pi PCB 1.2 Zero, Studeni 2015
- Raspberry Pi 3 Model B, Veljača 2016
- Raspberry Pi PCB 1.3 Zero, Svibanj 2016
- Raspberry Pi 2 verzija 1.2 Model B, Listopad 2016
- Raspberry Pi W Zero, Veljača 2017

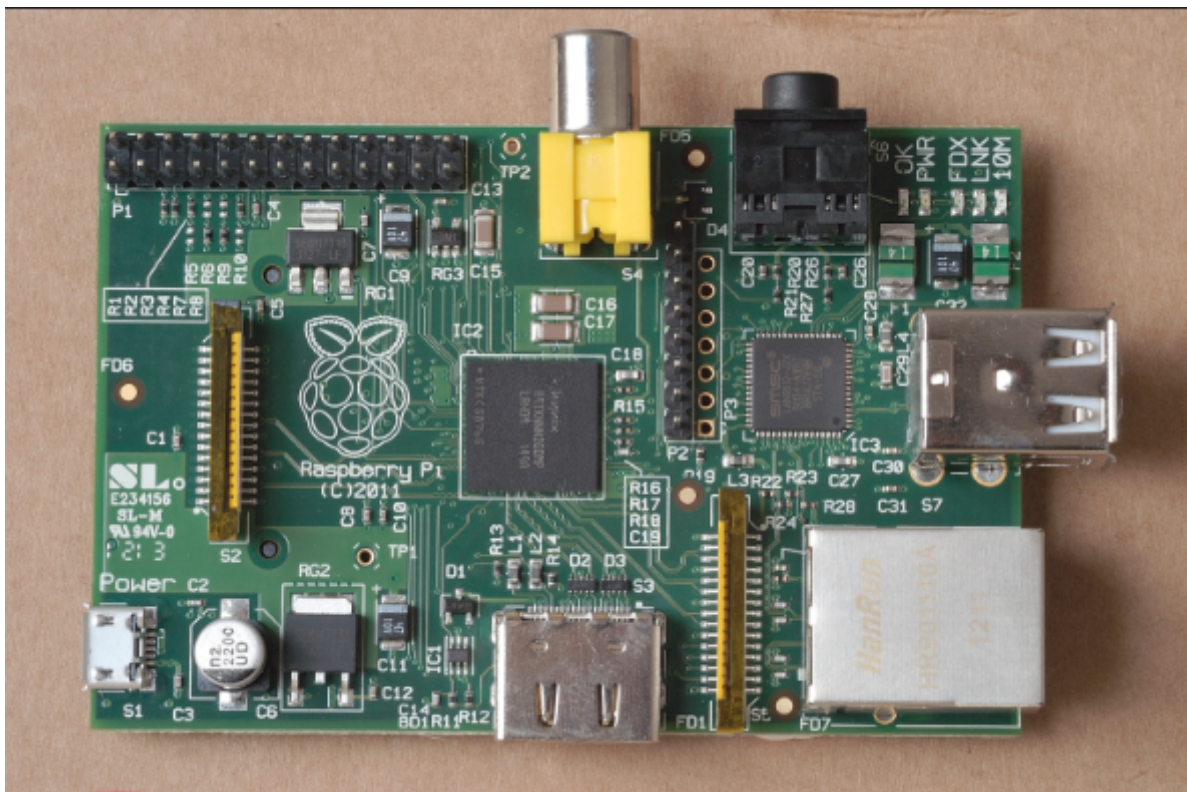
Svi gore nabrojani modeli i njihove verzije imaju jednu zajedničku značajku, a to je Broadcomov SoC¹, čiju osnovu čine ARM-ov CPU² i Broadcomov GPU³ VideoCore. Ostale značajke,

¹eng. System on a Chip

²eng. Central Processing Unit

³eng. Graphics Processing Unit

poput brzine CPU¹-u, količine RAM⁴-a, veličine utora za SD karticu, broja USB ulaza itd., su im različite od modela do modela. U želji da ostanu konkurentni na tržištu i odgovore na neke zahtjeve ljudi koji koriste Raspberry Pi u značajnim projektima grupa zadužena za razvoj u Raspberry Pi zakladi sa svakim novim modelom ili verzijom unaprijeđivala je ove ostale značajke. Stoga iz specifikacija pojedinih modela možemo vidjeti kako se recimo povećala brzina CPU¹-a sa 700MHz na 1.2GHz, veličina RAM⁴-a od početnih 256MB do 1 GB, broj USB ulaza sa jednog na četiri, te GPIO⁵ iglica sa 26 na 40. Od ostalih unaprijeđenja treba spomenuti da je smanjen utor za SD⁶ karticu, koji kod novijih modela prima samo MicroSD⁶ kartice. Dodan je ethernet ulaz, te je omogućeno spajanje preko Wi-Fi i Bluetootha. I još, treba spomenuti da je kod model Zero pored svih ovih unaprijeđenja rađeno i na njegovoj minimizaciji.



Slika 1: Raspberry Pi 1 Model B, prva verzija Raspberry Pi

U prvoj sekciji prvog poglavlja ovoga rada govoriti ćemo detaljnije o hardveru Raspberry Pi-a te u slikama predočiti koje se sve komponente nalaze na njemu. Zatim ćemo u drugoj sekciji reći nešto o njegovom procesoru, o tome od čega je sastavljen, gdje se proizvodi i kakve su mu performanse. Treća sekcija ukratko govori što su GPIO iglice, čemu one služe, te koliku važnost one donose Raspberry Pi-u dok četvrta obrađuje temu o operacijskom sustavu s posebnim naglaskom na Linux i njegovu distribuciju Raspbian. Razlog tomu je što ću upravo tu distribuciju pokrenuti na svom Raspberry Pi-u kojeg koristim za izradu ovoga rada. Na kraju ovog poglavlja, u petoj sekciji, govori se o programskom jeziku Python

⁴eng. Random Access Memory

⁵eng. General-purpose Input/Output

⁶eng. Secure digital

u kojem ću pisati programski kod za praktični projekt koji dolazi uz ovaj završni rad, te o RPi.GPIO i RPi.GPIO Python bibliotekama i kako pomoću njih pristupati GPIO iglicama. U prvoj sekciji drugog poglavlja govorit ćemo općenito o motorima i njihovim značajkama, te kroz tri podsekcije posebno objasniti DC motor, servo motor i stepper motor. U drugoj sekciji uz pomoć slika i programskih kodova ćemo demonstrirati kako se može upravljati motorima pomoću Raspberry Pi-a i programskog jezika Python.

I za kraj želio bih još istaknuti da sve opise koje ću raditi u ovom radu bit će isključivo vezani za Raspberry Pi 3 Model B.

Poglavlje 1

Raspberry Pi

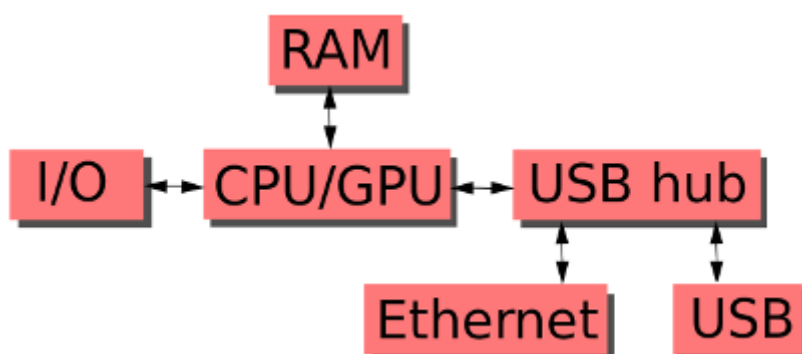
Raspberry Pi je uređaj koji posjeduje gotovo sve komponente koje posjeduje i obično računalo, ali sve one su kod njega smještene na samo jednoj pločici veličine bankovne kartice. Inspiracija za kreiranje ovakvog uređaja bilo je mikroručunalo BBC Micro iz 1981. (slika 1.1), a glavna ideja jednog od njegovih tvoraca, Ebena Uptona koji je ujedno najviše i pridonio kreiranju, bila je stvoriti jeftin uređaj koji će poboljšati vještine programiranja i razumijevanja hardvera kod ljudi koji se tek počinju baviti računarstvom. No međutim, zahvaljujući upravo pristupačnoj cijeni, maloj veličini, te pružanju nešto malo većih mogućnosti od onih koje možemo dobiti od običnog mikrokontrolera, kao što je naprimjer Arduino, ova ideja se vrlo brzo proširila i njegova upotreba se počela primjenjivati u još mnogim područjima. Uspoređujući ga sa stolnim i prijenosnim računalima sporiji je od njih, što ne znači da neće moći odgovoriti na sve zahtjeve koji se postavljaju pred današnja moderna računala. Ono što mu donosi prednost nad ostalim računalima je mala potrošnja električne energije i zbog toga je danas uključen u procese stvaranje napredne robotike.



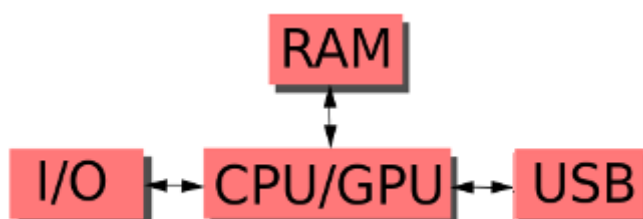
Slika 1.1: BBC Micro

1.1 Hardver

Hardver Raspberry Pi-a je pravi mali računalni svijet smiješten na samo jednoj pločici čija je veličina gotovo u svim modelima i verzijama jednaka. Ono što se mijenjalo kod generacija Raspberry Pi-a su komponente koju su dodavane na njega poput dodatnih USB ulaza, ethernet ulaz, Wi-Fi antene ili čipa koji podržaje Bluetooth i Wi-Fi. Na slici 1.2 prikazana je shema pomoću koje se ugrubo opisuje način na koji su spojene komponente kod svih verzija Model B, jedina razlika između verzije 1 i ostalih u pogledu fizičkog izgleda vezana je za blok u kojemu piše "USB hub", što je zapravo integrirani USB/ethernet razdjelnik. Kod verzije 1 on može podržati samo 3 priključka od kojih je jedan ethernet, a preostali su USB, dok kod ostalih verzija on podržava 5 priključaka što omogućuje još dodatna 2 USB priključka. Slika 1.3 prikazuje isto što i slika 1.2, ali za Model A i Zero, iz nje je vidljivo da oni nemaju integrirani USB razdjelnik, što znači da ukoliko njih želimo spojiti na mrežu potreban nam je vanjski USB razdjelnik kojeg priključujemo na jedini USB priključak koji imaju ovi modeli, a zatim na njega adapter za ethernet priključak. Napomenimo još da kod Raspberry Pi-a Zero taj jedini USB priključak je mikro oblika, pa treba biti pažljiv pri kupnji USB razdjelnika.



Slika 1.2: Shema hardvera Raspberry Pi modela B



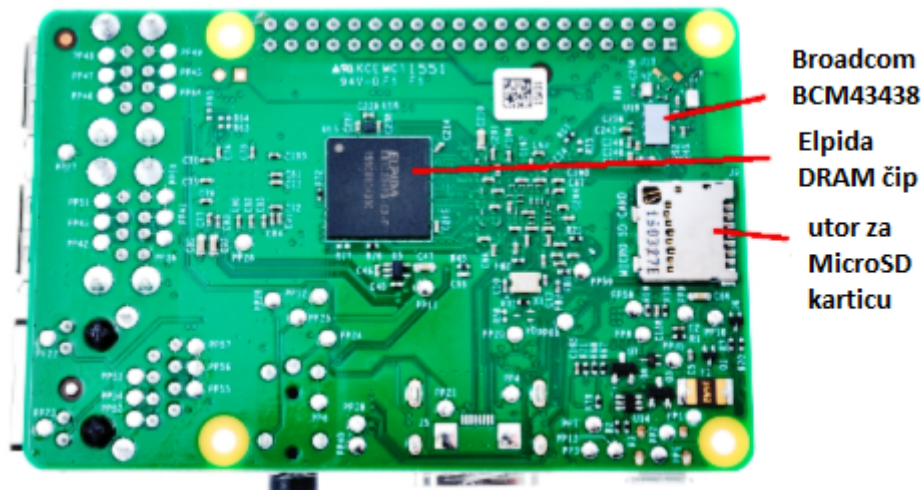
Slika 1.3: Shema hardvera Raspberry Pi modela A i Zero

Međutim, treba naglasiti da se Raspberry Pi grubo može podijeliti na tri modela: Model A, Zero (model sa slabijim hardverom), te Model B (koji je puni hardverski model, odnosno sadrži gotovo sve komponente kao i računalno). Glavne razlike između ova tri modela dane su u tablici 1.1

Resurs	Model A	Model B	Zero
RAM	256 ili 512 MB	512 MB ili 1 GB	512 MB
USB priključak	1	2 ili 4	1
Ethernet priključak	Nema	10/100 Ethernet (RJ45)	Nema
Potrošnja energije	200 - 300 mA (1 - 1.5 W)	700 mA - 1.34 A (1 - 6.7 W)	100 - 350 mA (0.5 - 1.75 W)
Cijena	\$25.00	\$35.00	\$10.00

Tablica 1.1: Razlike modela

Specifikacije Raspberry Pi-a pomoću kojega sam izrađivao ovaj rad su sljedeće (vidi sliku 1.4 i 1.5). Sa gornje strane, gotovo na sredini pločice, nalazi se Broadcomov čip BCM2837 koji je napravljen posebno za ovo verziju 3, zatim duž jednog ruba 40 GPIO¹ iglica što je za razliku od prethodnih modela koji su imali po 26 iglica značajan napredak. Tu su još 4 USB priključka i ethernet RJ45 priključak iza njih je SMSC LAN9514 USB/ethernet IC². Nasuprot GPIO iglica smjestili su se 3.5mm audio izlaz, CSI³ priključak za kameru, HDMI izlaz te mikro USB priključak za struju, koji služi samo za napajanje električnom energijom ali ne i za prijenos podataka. Također imamo DSI⁴ priključak za zaslon koji je alternativa za HDMI i antenu za Wi-Fi i Bluetooth. S donje strane imamo još utor za MicroSD karticu, Elpida B8132B4OB-8D-F DRAM⁵ čip, te Broadcomov BCM43438 čip koji omogućuje 2.4 GHz 801.11n Wi-Fi i 4.1 Bluetooth.



Slika 1.4: Donja strana Raspberry Pi-a

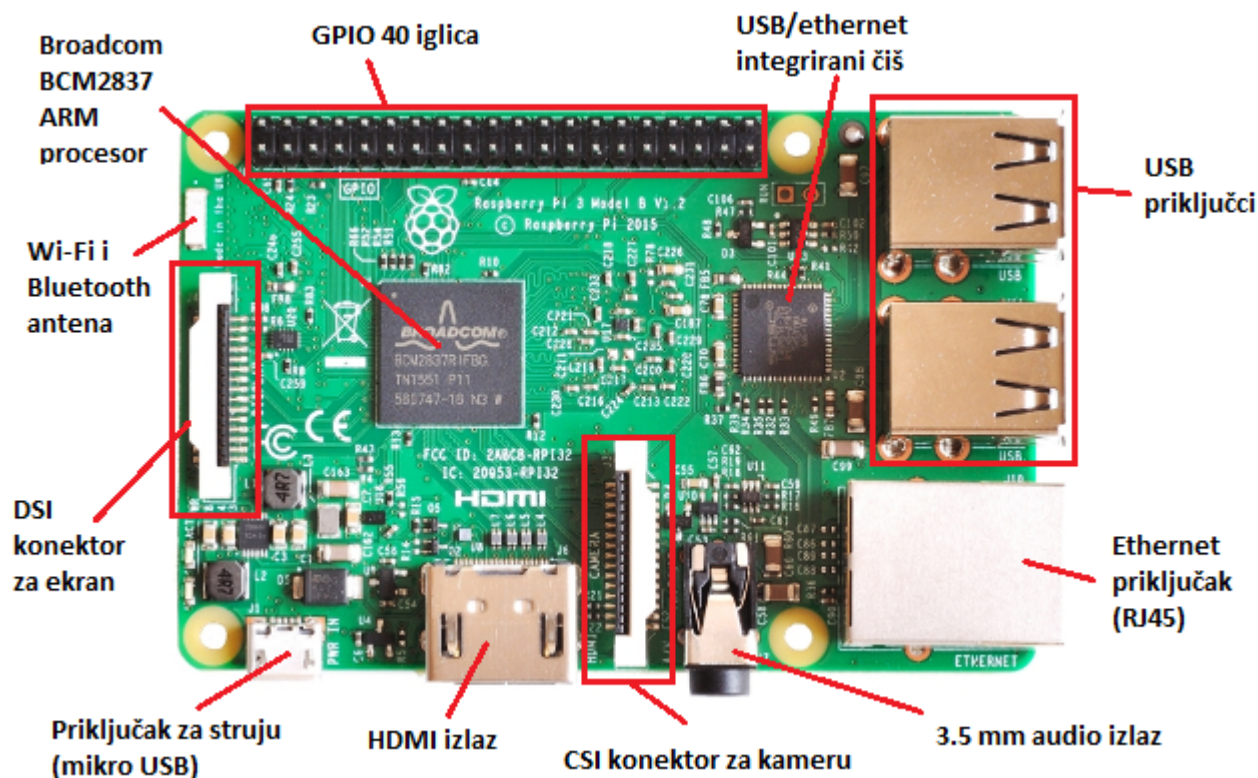
¹eng. General-purpose Input/Output

²eng. Integrated Chip

³eng. Camera Serial Interface

⁴eng. Display Serial Interface

⁵eng. Dynamic Random Access Memory



Slika 1.5: Gornja strana Raspberry Pi-a

1.2 Procesor

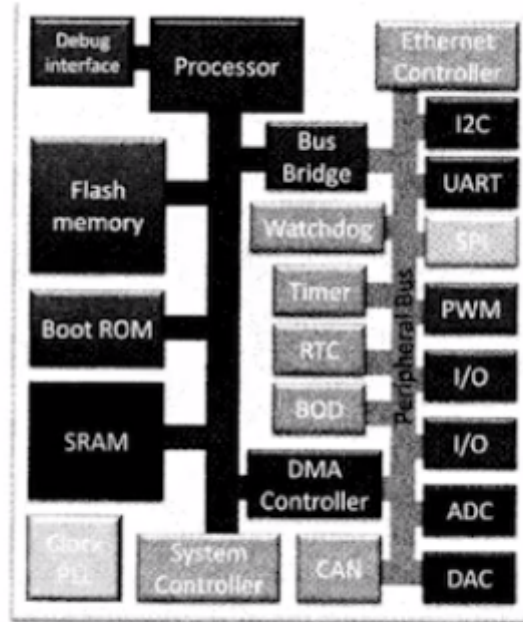
Procesor Raspberry Pi-a je složena struktura sastavljena od puno manjih dijelova (kao npr. struktura na slici 1.6), zajedno se zovu SoC⁶, koje komponira i sastavlja u jedan čip tvrtka Broadcom, sjedište joj je u gradu Irvine, savezna država Kalifornija, SAD. Srce te složene strukture je ARM procesorska jezgra, ARM je Britanska kompanija koja posjeduje prava samo na intelektualno vlasništvo. Dakle, oni ne proizvode fizički opipljive jezgre procesora, nego dizajniraju strukturu i daju teoretski opis procesora ili jezgre procesora i takav proizvod prodaju tvrtkama koje proizvode čipove. Na slici 1.6 nije prikazan naš procesor, ali jest jedna vrsta SoC⁶-a i ona će nam poslužiti samo da bi mogli pokazati što je to ARM procesor. To je upravo ovaj blok u kojemu piše "Processor". Sve ostale stavke koje možemo vidjeti na slici 1.6 većinom proizvodi i sastavlja u jednu cijelinu tvrtka koja proizvodi same čipove, a u našem slučaju to je Broadcom. Koliko je zapravo moćan Broadcomov SoC⁶ procesor koji se nalazi u Raspberry Pi-u govori nam i činjenica da ih također možemo pronaći u našim pametnim telefonima i računalima te su korišteni u različite svrhe. Upravo te dvije činjenice, da je SoC⁶ i da korsiisti drugačiju ISA⁷-u, je ono što čini Broadcomove čipove različitim u odnosu na procesorske čipove koje možemo pronaći u našim stolnim ili prijenosnim računalima. Verzije Broadcomovih čipova koji se koriste u obitelji Raspberry Pi su BCM2835, BCM2836 i BCM2837.

Upravo ovaj posljednji, Broadcom BCM2837, je procesor koji pokreće Raspberry Pi 3 Model B, sastoji se od 4 ARM Cortex-A53 jezgre visokih performansi čija je brzina 1.2

⁶eng. System on a Chip

⁷eng. Instruction set architecture

GHz sa 32kB razine 1 (L1) i 512kB razine 2 (L2) cache memorije, VideoCore IV procesora za grafiku i povezan je sa 1GB LPDDR2 radne memorije. On je 10 puta učinkovitiji od procesora Raspberry Pi 1, a također neke reference pokazuju da je otprilike 80% brži od svog prethodnika kada je u pitanju paralelno izvršavanje.



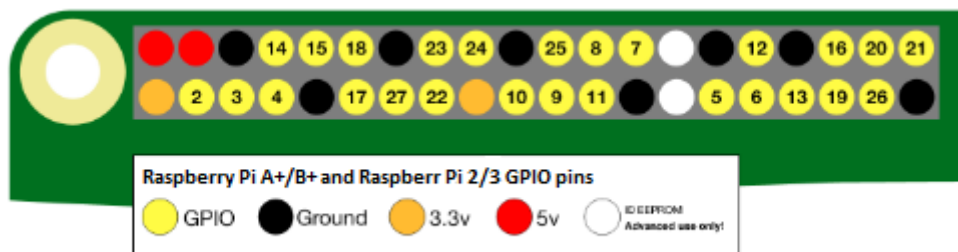
Slika 1.6: Opći prikaz sustav na čipu

1.3 GPIO iglice

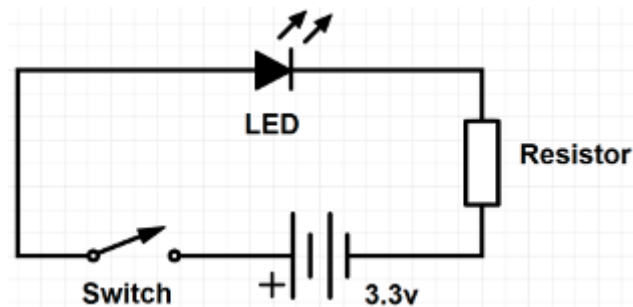
GPIO⁸ iglice su uz mogućnost podržavanja operacijskog sustava najvažnija stavka u specifikacijama Raspberry Pi-a. Znamo da je on mikroročunalno i da kao takav služi učenju programiranja, ali smo također spominjali i njegovu primjenu u robotici te u nekim elektrotehničkim granama, upravo zbog toga ove iglice dobijaju toliku pozornost. Ono što GPIO⁸ iglice omogućuju Raspberry Pi-u jest da u tim projektima igra ulogu mikrokontrolera, dakle Raspberry Pi-a preko svojih iglica može komunicirati s drugim uređajima i sustavima. Prve generacije Raspberry Pi-a imale su 26 GPIO⁸ iglica dok novije imaju 40, što je povećanje broja iglica za nešto više od 50% to je očiti pokazatelj koliko je široka njegova primjena kao mikrokontrolera te koliki su zahtjevi za većim mogućnostima kontrole i kontrola većeg broj uređaja. Kod svih modela i verzija Raspberry Pi-a GPIO⁸ iglice čine kompaktnu cijelinu smještenu uz jedan rub pločice, raspoređene su u dva reda po 13, odnosno 20 iglica, u kojega Raspberry Pi-a one se rasprostiru duž cijele jedne strane nasuprot one na kojoj se nalze audio i HDMI izlazi te priključak za struju. Ono što je specifično za ove iglice jesta da ih sve zajedno zovemo GPIO⁸ iglice, ali od njih 40 samo je 26 pravih GPIO⁸ iglica, tj. samo tih 26 omogućuje Raspberry Pi-u komunikaciju s drugim uređajima. Preostalih 14 su naponske,

⁸eng. General Purpose Input/Output

uzmeljenje i posebna vrsta iglica, ID EEPROM⁹ iglice koje ne bi trebali dirati ako ne znamo za što točno služe. Raspodjeljene su na sljedeći način, dvije su iglice koje propuštaju napon od 5V, dvije koje propuštaju napon od 3.3V, dvije su ove ID EEPROM⁹, te 8 iglica služi kao uzmeljenje (vidi sliku 1.7).

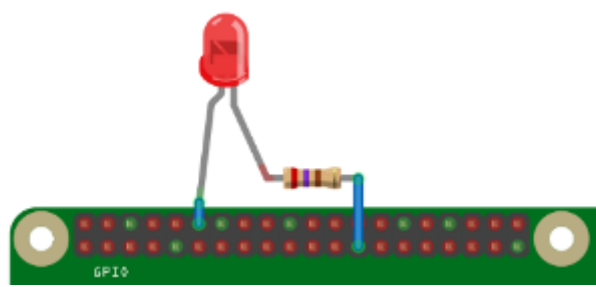


Slika 1.7: Shema GPIO iglice



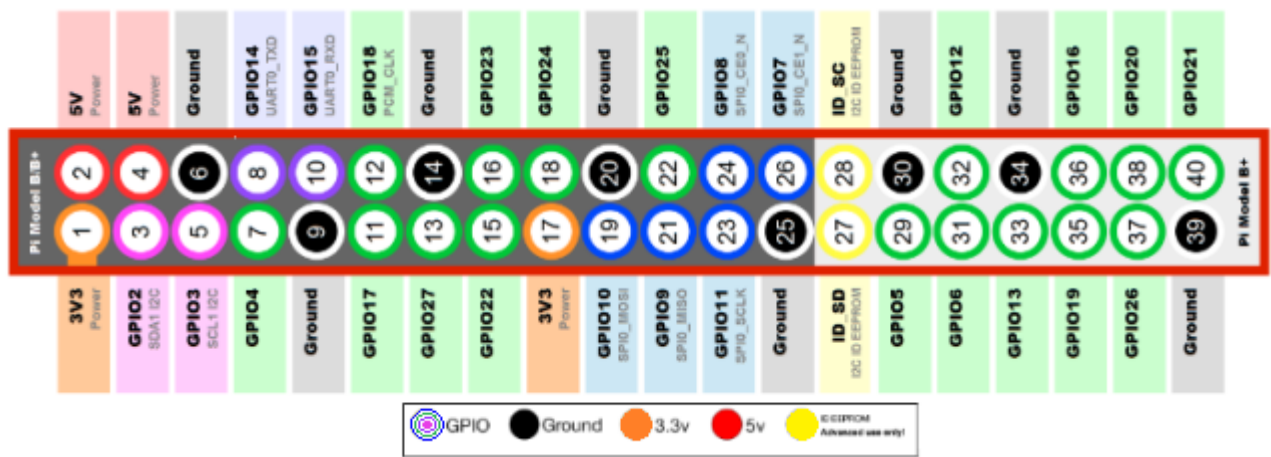
Slika 1.8: Shema električnog kruga

Najlakši način za objasniti kako Raspbrry Pi radi preko GPIO⁸ iglica jest da povučemo paralelu sa jednostvanim električnim krugom kojega čine baterija, lampica i prekidač (vidi sliku 1.8). Dakle, vidimo da je lampica priključena na napon na bateriji i da prekidač kontrolira da li će ona svijetliti, ili ne. Ukoliko ovakav strujni krug želimo napraviti pomoću Raspberry Pi-a dovoljan nam je sam uređaj i lampica (vidi sliku 1.9). Kako možemo vidjeti on je zamijenio čak dvije komponente iz strujnog kruga prikazanog na slici 1.8. Dakako, ako spojimo lampicu na GPIO⁸ iglice 1 i 3 (vidi na slici 1.10 koje su to) ona će samo svijetliti i nećemo imati kontrolu nad njom, a ukoliko želimo i kontrolu potrebno ju je priključiti na iglice 6 i 15 te je tada Raspberry Pi i baterija i prekidač. Prilikom rada sa pravim GPIO⁸ iglicama potrebno je naglasiti u koju svrhu ih koristimo, za primanje podataka ili slanje. Igljice rade na samo dva načina, ili šalju signal, ili ga primaju. Ukoliko iglice koristimo za slanje signala tada kroz njih puštamo struju ili od 3.3V ili od 0V. U terminima bitova, ili je bit postavljen na 1 ili 0. Razloga zašto se šalje baš 3.3V, a ne 5V koliko možemo propustit kroz iglice 2 i 4 (vidi sliku 1.10) i služe samo za neke uređaje kojima je potrebno više napona, jest taj što cijeli sustav u Raspberry Pi funkcionira na logici 3.3V. Ako iglice koristimo za primanje signala, onda treba voditi računa o tome kako Raspberry Pi može primiti samo digitalni signal, što znači da ne može primiti analogni signal i pretvoriti ga u digitalni jer u njemu nema pretvarača koji će analogni signal pretvoriti u digitalni.



Slika 1.9: Električni krug sa Raspberry Pi-om

Gledajući slike u ovoj sekciji primjetimo da postoje dva načina označavanja iglica, a kod nekih čak i tri, što možemo preciznije vidjeti i na slici 1.10. Na prva dva načina označene su sve iglice te njih i koristimo prilikom rada sa iglicama. Jedan način je fizičko brojanje iglica te se upravo i zove fizička numeracija, s tim da treba voditi računa o tome da su iglice u vanjskom redu označene s parnim brojevima, npr. gledajući od Wi-Fi antene prema USB priključcima oznake su 2,4,6,... (vidi sliku 1.5), a iglice u unutarnjem redu s neparnim, npr. 1,3,5,... gledajući na isti način kao i one u gornjem redu. Drugi način se zove GPIO⁸ numeracija, a iglice se označavaju tako da na riječ “GPIO” dodamo neki broj, koji točno možemo vidjeti na slici 1.6. Iz slike 1.10 vidimo da se na ovaj način označavaju samo prave GPIO⁸ iglice i to je zapravo način na koji Broadcomov procesor vidi te iglice. Treći način označavanja nije zastupljen kod svih i on se odnosi na svrhu za koju te iglice služe, prema toj svrsi neke prave GPIO iglice možemo i grupirati u disjunktne grupe. Jedna od tih grupa je UART¹⁰ kojoj pripadaju iglice 8 i 10, druga je I²C¹¹ kojoj pripadaju iglice 3 i 5, i treća je SPI¹² grupa u koju spadaju iglice 19, 21,23, 24 i 26.



Slika 1.10: Dijagram GPIO iglica sa svim njihovim imenima i legendom što koja boja označava

⁹eng. Electrically Erasable Programmable Read-Only Memory
¹⁰eng. Universal Asynchronous Receiver-Transmitter
¹¹eng. Inter-Integrated Circuit
¹²eng. Serial Peripheral Interface

1.4 Operacijski sustav

Na početku ovoga rada definarali smo Raspberry Pi kao mikroračunalo, a kao takvoga su ga zamišljali i ljudi koji su ga dizajnirali. Iz dosadašnjeg iskustva znamo da svako računalo kako bi radilo treba operacijski sustav. Već pri dizajniranju ideja je bila da se na Raspberry Pi-u vrti sustav otvorenog tipa (eng. open source), tj. sustav kod kojega je moguće mijenjati izvorni kod i prilagođavati ga svojim potrebama, a sama ta ideja postajala je sve jača primjenom ovog uređaja u raznim eksperimentalnim projektima koji su bili jako specifični i zahtjevali neke vlastite standarde. Ova činjenica da pokreće operacijski sustav otvorenog tipa je još jedna stavka u kojoj se razlikuje od uobičajenih računala, jer većina njih ima na sebi pokrenuti različite verzije Microsoft Windows i Apple OS X operacijskih sustava koji su operacijski sustavi zatvorenog tipa (eng. closed source). Zatvoreni tip operacijskog sustava ne dopušta kranjim korisnima izmjene izvornog koda prema vlastitim željama. Jedino što oni mogu jest instalirati operacijski sustav i koristiti ga prema već zadanim uputama bez da znaju što se u pozadni događa. Operacijski sustav otvorenog tipa koji je najviše prihvaćen od strane korisnika Raspberry Pi-a je Linux, jedan od razloga zašto baš on je taj što je besplatan, ali najveći razlog je zato što ARM-ova jezgra procesora može podržavati Linux, za razliku od većine ostalih operacijskih sustava koji su razvijani isključivo za Intelovu procesorsku arhitekturu. Međutim, treba napomenuti da ARM-ova jezgra procesora ne podržaje baš svaku Linux distribuciju, jedan od takvih primjeta je Ubuntu Linux. Treba naglasiti da postoji još nekoliko operacijskih sustava koji se mogu instalirati na Raspberry Pi, a to su: OpenElec, Pidora, ROSC OS, Snappy Ubuntu Core, Android, Windows 10, Ubuntu MATE 15.04, Minibian, Hypriot, Arch Linux, te PiPlay.

1.4.1 Linux



Slika 1.11: Tux - maskota Linuxa

Linux je prvenstveno naziv za jezgru operacijskog sustava koju je načinio Linus Torvalds za svoje osobno računalo kojega je pokretao procesor sa Intelovom x86 arhitekturom. Nedugo nakon toga odlučio je programski kod jezgre objaviti na internetu i pozvati ljude da zajedno s njima sudjeluju u nastajanju operacijskog sustava. Ono što danas zovemo Linux ili Linux distribucije je zapravo punog imena GNU/Linux i to je ustvari ime za jednu familiju operacijskih sustava, a svaki član te familije ima svoje posebne značajke, poput mnoštva biblioteka koje su dodane na Linux jezgru. Dakle, ono što Linux čini različitim od standardnih operacijskih sustava je to što je nastao kao kreacija ljudi diljem svijeta, te se i danas

ljudi bave njegovim nadograđivanjem, a izvorni kod mu je u potpunosti slobodan za izmjene. Linux operacijski sustav podržan je na skoro svim danas poznatim platformama, a osim što pokreće osobna računala, pokreće i servere te je i Andorid OS baziran na Linux jezgri. Rad u Linuxu većinu vremena odvija se na ograničenom korisničkom računuu, jer zbog svoje otvorenosti mogao bi biti laka meta virusima i zlonamjernim softverima. Stoga, rad u ograničenom korisničkom računuu ne znači ograničene mogućnosti, nego sigurnost od neželjenih posljedica. Raditi u Linuxu možemo na dva načina, slično kao kod Windows i OS X operacijskih sustava, kroz tekstualno sučelje poznatije kao terminal, ili kroz grafičko sučelje poznatije kao GUI¹³. Rad kroz GUI je nešto na što smo već navikli prilikom upotrebljavanje naših standardnih računala, ali rad u terminalu nam daje veće mogućnosti i jednostavnije je, razlog tomu je što tu samo moramo upisati naredbe u komandni reda koje želimo da se izvrše. Međutim, postoji i mali nedostatak pri radu u terminalom, a to je što trebamo naučiti osnovne naredbe kako bi mogli raditi u njemu.

Raspbian

Raspbian je operacijski sustav koji je pokrenut na mom Raspberry Pi koji sam koristio za izradu ovoga rada. Ime mu dolazi od riječi Raspberry Pi i Debian (vidi sliku 1.1). Raspbian je Linuxova distribucija bazirana na Debianu, besplatan je i pruža nam nešto više od običnog operacijskog sustava. Dolazi zajedno sa više od 35 000 paketa. Kreirala ga je mala skupina programera koji su obožavatelji Raspberry Pi hardvera i naravno Debian projekta. Prva verzija Raspbian bila je gotova 2012. godine, no on se još uvijek aktivno razvija, a od 2015. godine Raspbery Pi fondacije ga službeno promovira kao primarni operacijski sustav za uređaje iz Raspbery Pi obitelji. Raspbian zajedno sa Raspberry Pi-om je odličan i za edukaciju zato što uz njega dolazi i IDLE¹⁴, što je IDE¹⁵ upravo za programski jezik Python.



Slika 1.12: Logo Raspbiana

1.5 Python

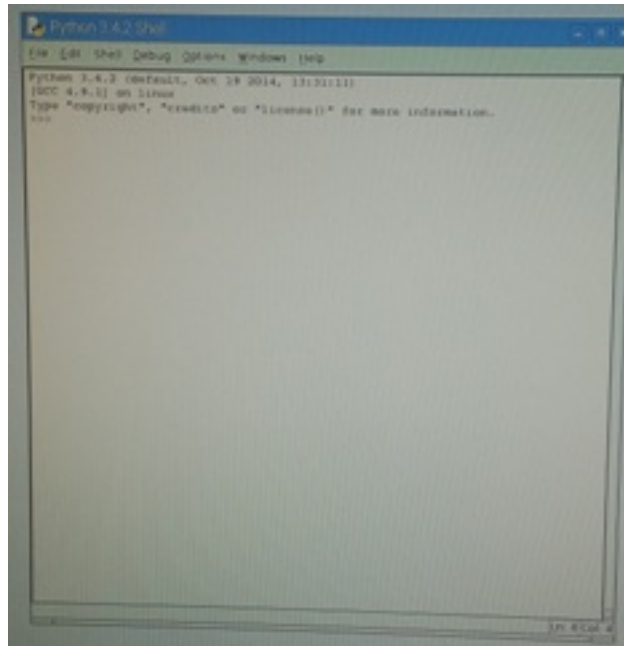
Razlog zašto smo izabrali upravo Python programerski jezik za izradu ovog rada i projekta opisanog u poglavlju 2 na stranici 11 je zato što je on najbolje podržan jezik za pogramiranje na Raspberry Pi-u. Dakako, mogli smo programirati i u nekim drugim jezicima ali Python najbolje kontrolira hardver i za njega postoji jako puno sučelja za programiranje aplikacija koja su prilagođena korisnicima i lagana za koristiti. Python je jezik visoke razine, dakle postoji puno detalja koji su sakriveni od programa i programer ih ne mora gledati. Na primjer, u Pythonu ne moramo definirati tip varijeble, on sam vodi brigu o tome kakav tip treba dodijeliti varijabli kada je instanciramo. Python je objektno orijentiran jezik, što znači da ima klase i još puno drugih značajki koji ga čine pogodim za korištenje. Python također

¹³eng. Graphics User Interface

¹⁴eng. Integrated Development and Learning Environment

¹⁵eng. Integrated Development Environment

ima i svojih nedostataka. Općenito, spor je u odnosu na kompajlerske jezike zato što proces interpretacije njegovih naredbi zahtjeva vrijeme. Dakle, Python je interpreterski jezik koji prevodi i pokreće linije koda jednu za drugom, što znači da dok ne završi jednu liniju koda neće prijeći na drugu, i to je razlog zašto Pythonu treba vremena, a koliko mu točno treba to je teško predvidjeti. Ali moj cilj nije sada govoriti o nedostacima Pythona, ja ću ga koristiti za pisanje manjeg koda koji nije previše zahtjevan.



Slika 1.13: Python 3.4.2 Shell

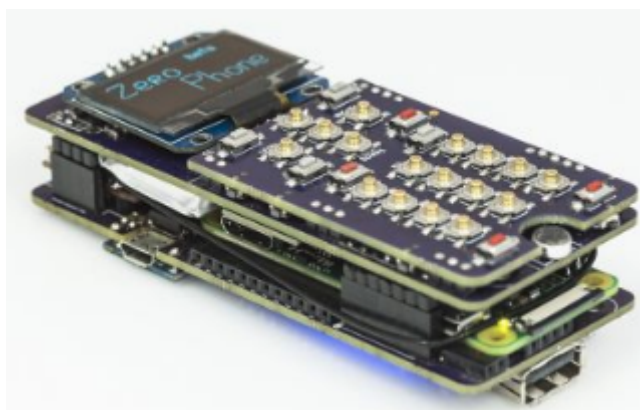
Ugrubo, postoje dvije verzije Pythona, Python 2 i Python 3, i obje verzije su podržane na Raspberry Pi-u. Međutim to je samo gruba podjela. Postoji još puno podverzija u svakoj od ove dvije. Iako je Python 3 novija verzija, razumno je da se samo ona koristi, Python 2 se još uvijek koristi zato što postoji mnoštvo starih kodova i softvera otvorenih za nadogradnju koji se još uvijek koriste. Za programiranje u Pythonu imamo također i dva moguća programska okruženja koja podržavaju Raspberry Pi. Jedno je IDE¹⁶, što je kod Pythona IDLE¹⁴, a drugo je korištenje tekstualnog editora i interpretera odvojeno. Drugo programsko okruženje radi na principu da Python program zapišemo u nekom od tekstualnih editora koji podržavaju naš Raspberry Pi i tako spremljen u datoteci pokrećemo u terminalu, odnosno Pythonovom shellu, kao skriptu. Za razliku od toga kada radimo u IDLE¹⁴ tekstualni editor i shell imamo na jednome mjestu. Kako ću u svom projektu motore upravljati pomoću Python programa, za kontrolu GPIO⁸ iglica na Raspberry Pi-u trebam dvije Pythonove datoteke, RPi.GPIO i pigpio. RPi.GPIO je set Pythonovih datoteka i kodova koji je uključen u Raspbian operacijski sustav i za njeno pokretanje dovoljno je upisati naredbu "import RPi.GPIO". Pigpio je Pythonova biblioteka za Raspberry Pi koja kaže pigpio daemonu da dopusti kontroliranje GPIO⁸ iglica i kod nju su sve GPIO⁸ iglice identificirane onako kako ih vidi Broadcomov procesor na Raspberry Pi-u.

¹⁶eng. Integrated development environment

Poglavlje 2

Projektni zadatak

Kako je Raspberry Pi i mikroracunalo i mikrokontroler to mu omogućuje širku uporabu, od toga da programeri početnici mogu učiti na njemu pa do raznih eksperimentiranja. Ono što će ja u ovom poglavlju raditi jest kontrola 3 vrste motora, DC motor, servo motor i step motor, pomoću Python programskog koda i njegovih već spomenutih biblioteka. No osim ovog mog postoji još mnoštvo praktičnih projekata koje možemo uraditi sa Raspberry Pi-om. Neki od njih su stvarno jako zanimljivi i zato su zavrijedili da ih spomenem u ovom poglavlju. Prvi na toj listi je izgradnja pristojnog stolnog računala, iako Raspberry Pi ima sve bitne značajke računala postoji par problema koji ga zadržavaju da to ne bude. Drugi je izgradnja pametnog telefona kojeg pokreće Raspberry Pi Zero



Slika 2.1: Pametni telfon kojeg pokreće Raspberry Pi

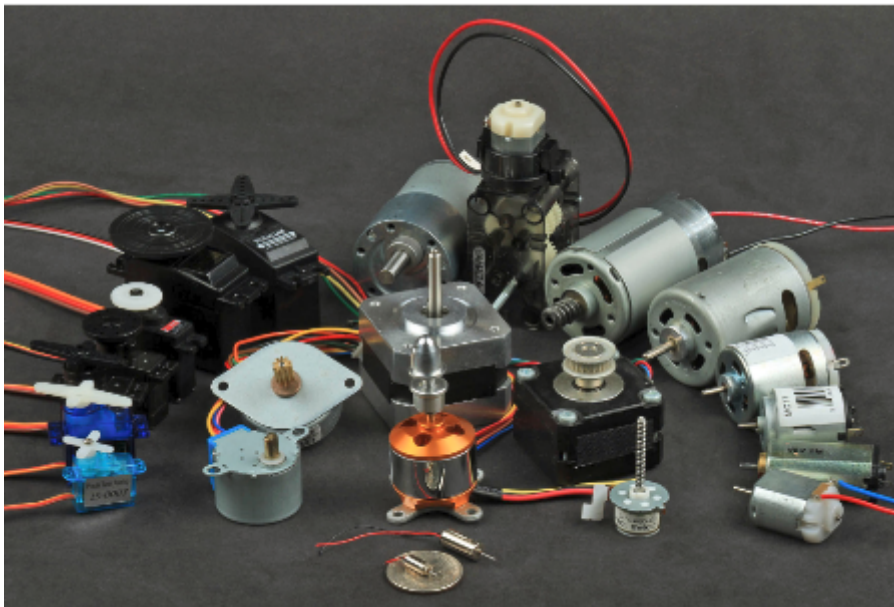
Idući projekt je izgradnja kreiranje vlastitog AI¹ asistenta. Nadalje možemo pomoću Raspberry Pi napraviti i električnu hranilicu za mačke ili pse, također zanimljiva stvar je kreiranje vlastitog retro arkadnog stroja. Navesti ću još jedan i jako ozbiljan projekt koji spada među IoT projekte a to je izgradnja printera povezanog na internet.

2.1 Vrste motora

Motori su nešto bez čega je danas gotovo nemoguće zamisliti svijet. Prva pojava motora u smislu u kome i danas postoje, a to je da neku vrstu energije pretvara u mehaničku energiju, datira još iz daleke 1712. i bio je to Newcomenov parni stroj koji je toplinsku energiju

¹eng. Artificial intelligence

pretvarao u mehaničku. Danas se koriste motori koje možemo grubo podijeliti u dvije vrste, motori s unutarnjim izgaranjem i elektromotori. U robotici se koriste elektromotori, koji električnu energiju, koja ih pokreće, pretvaraju u mehaničku energiju, koja obavlja neki rad. Postoje dvije vrste elektromotora, istosmjerni motori koji se napajaju iz izvora istosmjerne struje, te izmječnični motori koji se napajaju iz izvora izmjenične struje. Kako ne bi bilo zabune u daljnjem čitanju ovoga rada, radi lakšeg razumijevanja i zato što ćemo se samo elektromotorima i baviti umjesto riječi “elektromotor” koristit ću samo riječi “motor”. Prvi istosmjerni motor pojavio se 1833, napajao se istosmjernom strujom iz baterije sastavljene od galvanskih ćelija, a nastao je na temelju spoznaja o djelovanju sile magnetnog polja na vodič kroz koji teče električna struja. Pojava izmjeničnih motora vezana je za Nikolu Teslu i njegov rad pod nazivom “Novi sustav motora na izmjeničnu struju i transformatora”, a spominje se da je prvi put primjenjen 1888. godine. Motor se sastoji od rotora, ležajeva, statora, zračnog prostora, navoja i komutatora. Rotor je pokretni dio koji okreće osovinu motora kako bi ona mogla stvarati mehaničku energiju. Ležajevi potpomažu rotor i omogućuju mu da se okreće oko svoje osi. Stator je statični dio motora, a sastoji se od navoja i permanentnih magneta. Zračni prostor je rastojanje između statora i rotora, te ima jako važan učinak na rad motora zato što preveliki zračni prostor negativno utječe na rad motara, ali s druge strane i previše mali može uzrokovat oštećenja i buku. Navoji su žice u obliku spirale, obično su omotane oko magnetske jezgre kako bi se mogli stvoriti magnetski polovi kada se stave pod napon. Komutator se sastoji od međusobno izoliranih prstenastih segmenata i osovine elektromotora a služi kao mehanizam koji pretvara izmjeničnu struju i napon, inducirane u rotorskim vodičima elektromotora, u istosmjernu struju i napon, te može raditi i obratno.



Slika 2.2: Vrste motora

Motori koje ću ja koristiti u ovom projektnom zadatku su DC motor, servo motor i stepper motor, odnosno njihove manje inačice koje koriste istosmjernu struju, te su namjenjeni za edukaciju. Stoga, prilikom njihova korištenja treba strogo obratiti pozornost na dvije važne stvari, a to su naponski i strujni zahtjevi. Prva važna stvar kod rukovanja s motorima je shvatiti koju voltažu oni koriste. Neki mali hobi motori čija je svrha samo demonstriranje

kotrole motora pomoću Raspberry Pi pokreću se već pri struji od 1,5 volti. Ali ima i onih kojima je potrebno od 6 do 12 volti. Druga važna stvar jer shvatit koliku jakost struje treba naš motor. Upravljački čip za motore je dizajniran da proušta do 1,2A po motoru, ali ukupno najviše 3A. Motore koji zahtjevaju dodatan izvor električne energije ne mogu se pokrenuti sa baterijom od 9V, stoga se za njihovu uporabu preporuča korištanje akumilatora ili više paketa NiMH baterija.

2.1.1 DC motor

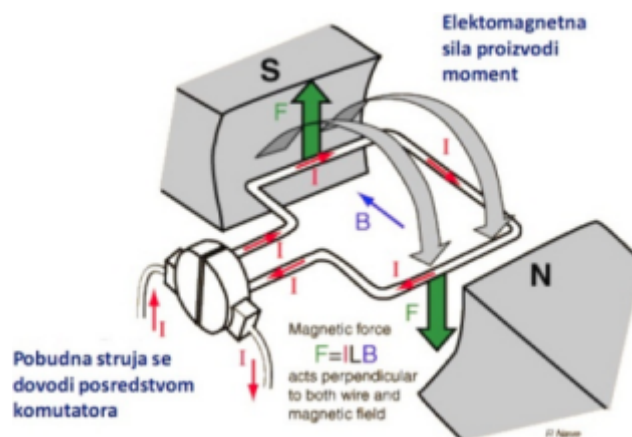


Slika 2.3: DC motor

Najčešće korišteni tip motora u robotici su DC motori, ima ih raznih oblika i veličina, ali najbitnija podjela je prema tome da li imaju četkice, ili ne. Ova podjela DC motora odnosi se na način na koji oni pretvaraju istosmjernu struju iz baterija u izmjeničnu struju potrebnu za pokretanje motora. Ako se ova pretvorba odvija na mehaničkom principu, što znači da segmenit komutatora, koji se nalaze na krajevima rotora, rotiraju oko statičnih četkica, koje se nalaze na unutarnjem rubu statora, onda se radi o vrsti DC motora sa četkicama. DC motori bez četkica za razliku od DC motora sa četkicama uz to što nemaju četkice, nemaju ni komutator, a zavojnice na statoru su direktno spojene sa mikroprocesorskim regulatorom, te uz pomoć senzora položaja izazivaju vrtnju rotora. Možemo reći da DC motor bez četkica radi na sličnom principu kao izmjenični motor. Tip DC motora sa četkicama je jeftiniji od motora bez četkica i češće se koristi, ali ima i svojih nedostataka poput životnog vijeka četkica, stvaranje taloga koji nastaje potrošnjom četkica, maksimalna brzina te buka. DC motora bez četkica su brži od onih sa četkicama, efikasniji, imaju smanjeno trenje a time i

duži vijek trajanja, finija kontrola motora, te stvaraju manju buku. Međutim, tip motora bez četkica ima potrebu za opsežnim upravljačkim sklopama što mu je također nedostatak.

Općenito rad DC motora možemo opisati na sljedeći način, kad struja prođe kroz zavojnicu stvara se magnetsko polje koje se suprotstavlja permanentnom magnet, te nastaje sila koja ide gore ili dole što se određuje pravilom desne ruke. Kada se motor okreće smjer struje će se promijenit, što povlači činjenicu da se permanentni magnet uvijek protivi zadržavanju polariteta elektromagnetske sile i zbog toga će se motor nastaviti okretati dokle god je priključen na izvor električne energije. Prilikom korištenja DC motora u robotici treba li bi uzeti u obzir neka njihova osnovna svojstva poput smjera vrtnje, brzine, napona, struje, snage i okretnog momenta. DC motori većinom imaju dvije kleme preko kojih se spajaju na napon čiji polaritet određuje smjer vrtnje motora a amplituda brzinu koja se mjeri u okretajima po minuti (rpm²). Svaki DC motor ima određeni napon koji označava nazivni ili primjenjeni napon pod kojim motor radi u normalnim uvjetima, u praski je važan nazivni napon zato što pokazuje maksimalni preporučeni napon. Prilikom rada motora na nazivnom naponu struja ovisi o opterećenju i povećava se s povećanjem opterećenja, zbog toga je važno ne dopustiti motoru da radi s prekomjernim opterećenjima koja ga mogu zaustaviti. Snaga motora je produkt napona i struje a okretni moment se definira kao produkt sile i udaljenosti od središta osovine motora.



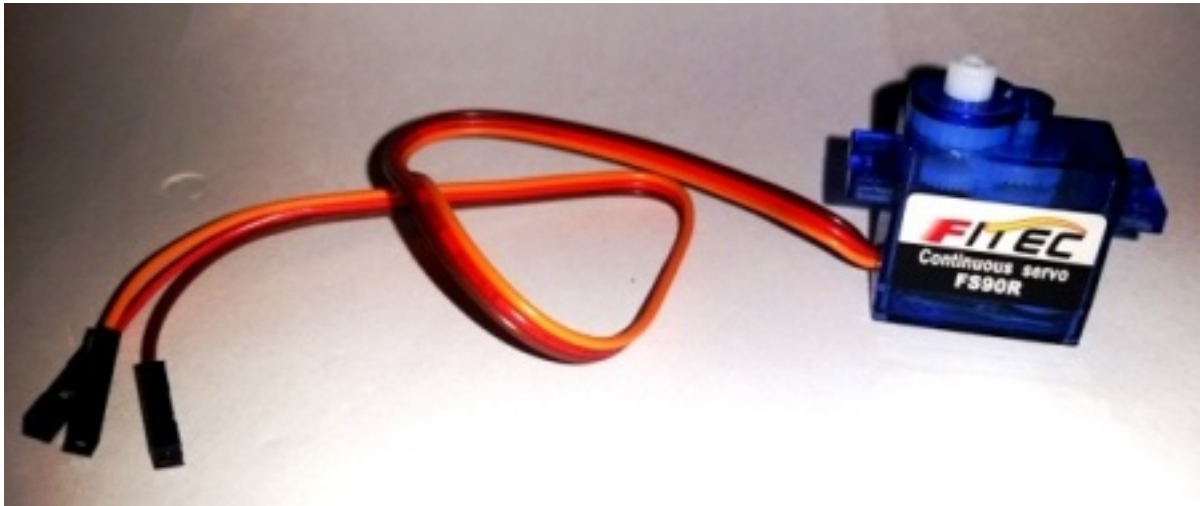
Slika 2.4: Princip rada DC motora

2.1.2 Servo motor

Servo motor je složeni tip motora čiji sustav čine klasičan DC motor, sustav zubčanika, potencijometra i sustava za upravljanje. Zajednička osobina koja krase sve vrste servo motora je mogućnost pružanja velike preciznosti pri pomjeranju glavne osovine, doslovno možemo odrediti točan kut za koji će se osovina zakrenuti. Prva primjena servo mehanizama bila je u vojne svrhe za kontrolu teškog naoružanja te u navigacijskoj opremi u pomorstvu. Danas se još koriste i u automatiziranim strojnim alatima, antenama za praćenje satelita, zrakoplovima na daljinsko upravljanje, automatskim navigacijskim sustavima na brodovima i zrakoplovima te u sustavima za kontrolu protuavionskih oružja. Ovako široka primjena servo motora zahtjeva i veću pokretačku snagu, pa se kod ovakvih primjera primjene za stvaranje mehaničke energije umjesto DC motora koriste izmjenični motori, hidraulika, pneumatika

²eng, Rotations per minute

ili magnetski principi. Standardni servo motori koji se koriste u robotici imaju tri žice, jednu za napon (4-6 V), druga je uzemljenje i treća je za kontrolu. RC servo motor je vrsta koja se najčešće koristi zbog svoje dostupnosti, pouzdanosti i jednostavnosti upravljanja sa mikroprocesorima. RC servo motori su motori male snage koji se mogu napajati s malim baterijama i drugim izvorima istosmjernje struje u rasponu od 100mA do 2A.

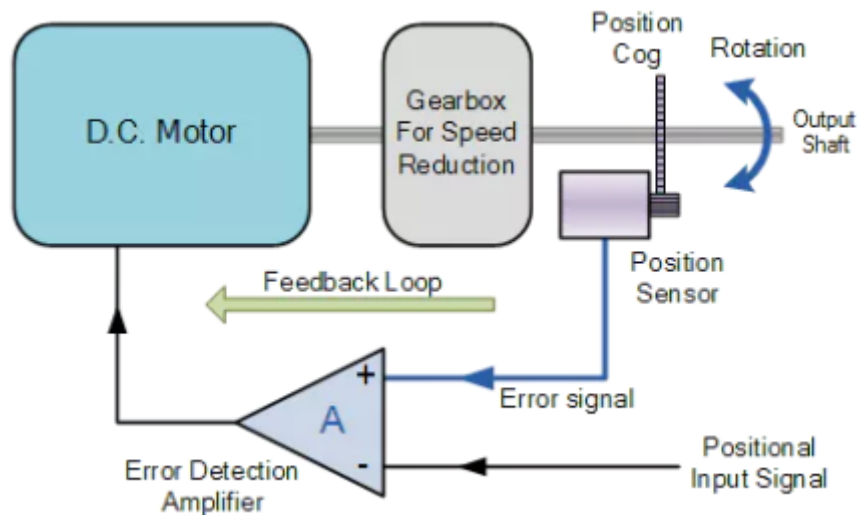


Slika 2.5: Servo motor

Servo motori općenito rade na sljedeći način, DC motor služi za pomicanje sustava zupčanika koji služe za smanjenje broja okretaja te povećanje okretnog momenta. Kada seпусти inicijalni električni impuls kroz sustav potencijometar pomoću senzora položaja detektira položaj glavne osovine i šalje povratni signal u regulatorsko pojačalo. Regulatorsko pojačalo prima dva električna impulsa, ovaj iz potencijometra i onaj inicijalni, te dokle god su oni različiti slati će poruku DC motoru da se vrti. Jednom kada glavna osovina servo motora dođe u položaj u kojemu će potencijometar vratiti impuls jednak inicijalnom tada će regulatorsko pojačalo reći DC motor da prestane s vrtnjom. Vidimo da kad pustimo jedan kodirani signal osovina dolazi do željenog položaja i mirovat će sve do trenutka dok se se kodirani signal ne promijeni. Kao i svaki motor tako i servo motor ima svoje vlastite karakteristike kao što su napon, struja, brzina rada, okretni moment, kontrolni impuls i rezolucija. Napon napajanja i jakost struje posebni su za svaku vrstu servo motora i ovise o primjeni, RC servo motori napajaju se iz izvora napona od 4 do 6 volti i jakosti od 100mA do 2A. Brzina rada servo motora definirana je kao vrijeme koje je potrebno osovini da dođe u određenu poziciju, općenito brzina rada im je u rasponu od 0.05s/60° do 0.2s/60°, a standardne vrijednosti okretnog momenta su u rasponu od 0.5 do 10 kg/cm. Kontrolni impuls se odnosi na vrstu impulsa koji se koristi za pozicioniranje osovine i dvije su glavne vrste kontrolni impulsa koji se koriste kod RC servo motora a to su središnji položaj u rasponu 1-2 ms i 1.25-1.75 ms. Rezolucija definira sa kojom preciznoću se osovina pozicionira kad primi signal vanjskom naredbom, općenito servo motori imaju rezoluciju u rasponu od 1° do 10°.

2.1.3 Stepper motor

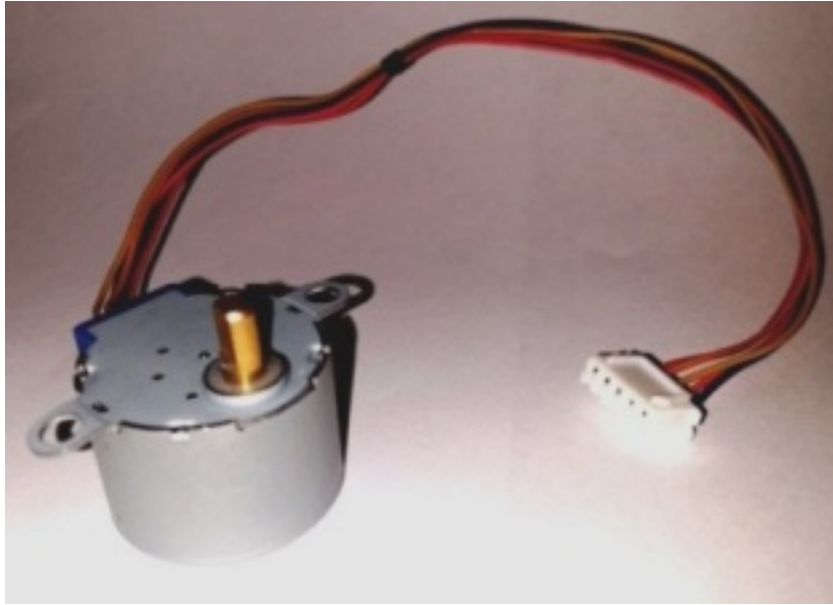
Stepper motor je istosmjerni električni motor bez četkica na čiju rotaciju direktno utječe nekoliko značajki električnog impulsa koji se pušta kroz njega. Ono što krasi ove motore je



Slika 2.6: Shematski prikaz principa rada servo motora

nevjerojatna preciznost postavljanja glavne osovine u određeni položaj što mu omogućuje nešto što se zove “korak”. Dije se u dvije skupine, bipolarne koji rade pomoću pozitivnog i negativnog napona (npr. bipolarni stepper motori namijenjeni za manje poslove rade rasponu od 2.5V do -2.5V) te unipolarne koji rade samo pomoću pozitivnog napona (npr. unipolarni stepper motori namijenjeni za manje poslove rade rasponu od 5V do 0V). Obje vrste imaju identičnu strukturu koju čine zavojnice namotane na elektromagnete, što predstavlja stator, te rotora na kome se nalaze magneti. Stepper motor radi na sličnom principu kao i DC motor bez četkica, dakle jedini dio motora koje se okreće je rotor. Kada pomoću mikrokontrolera pustimo električni impuls kroz zavojnice koje obavijaju elektromagnete stvara se magnetska sila koja utječe na magnet na rotoru i izaziva njegovu rotaciju. Ako uzmemo jedan elektromagnet za početni i prvo na njega djelujemo sa električnim impulsom to će njegove zube učiniti magnetski privlačnima za magnet na rotoru. Zatim ovaj prvi elektromagnet isključimo i impulsom djelujemo na sljedeći što će izazvati efekt odbojnosti magnet na rotoru i prvog elektromagneta, te efekt privlačnosti tih istih magnet na rotoru i drugog elektromagneta. Primjenjujući analogan postupak i na sljedeće elektromagnete izaziva vrtnju rotora. Upravo ovaj pomak kojega izazivaju efekti odbojnosti i privlačnosti naziva se “korak”. Dakle, stepper motor rotaciju od puno kruga može podijeliti u veliki broj “koraka” i time dobiti preciznost zakretanja glavne osovine za točno određenu veličinu kuta. Ono u čemu se razlikuju od ostalih motora je načinu okretanja rotora, umjesto mehanizma za povratne informacije o položaju osovine imaju stepper kontroler te imaju veliki okretni moment pri maloj brzini.

Stepper motori su dobar izbor za kontrolu položaja osovine, poluge ili drugog pokretnog dijela mehatroničkog uređaja, a koriste se u običnim i 3D printerima, CNC mašinama, klima uređajima, stalcima za fotoaparate, robotici itd. Budući da se navoji stepper motora moraju pravilo napajati kako bi se postigao pravilan rad trebamo osim električne specifikacije poznavati i mehaničke prilike rada sa njima. Neke važne specifikacije su voltiža, struja, redoslijed, kut koraka, brzinu pulseva i okretni moment. Većina stepper motora ima procjenu napona od 5,6 ili 12 volti, ne preporuča se preopterećivanje navoja zato što, za razliku od konvencionalnih istosmjernih motora, možete ih spaliti ako primjenjujete napon veći za 30% od vaše procjene. Procjena struje ovisi o projektu, običajene vrste stepper motora mogu podnijeti struju u rasponu od 50mA do 1A, što su veći napon i struja veći je okretni



Slika 2.7: Stepper motor

moment. Prilikom korištenja stepper motora moramo odrediti pravilan redoslijed pulseva koji određuju ispravan rad, a kada se puls pošalje prema motoru on napreduje jedan korak i osovina se pomiče za određeni broj stupnjeva. Brzina pulseva određuje brzinu motora, za određivanje broja okretaja po minuti (rpm^2) potrebno je znati veličinu kuta za koji se osovina zakrene u jednom koraku i broj pulseva u minuti. Okretni moment koji stvara stepper motor nije jako velik, standardno što može pružiti je samo par grama po centimetru, stoga ako trebamo veći okretni moment potreban nam je sustav zubčanike koji će igrati ulogu mjenjača brzine.

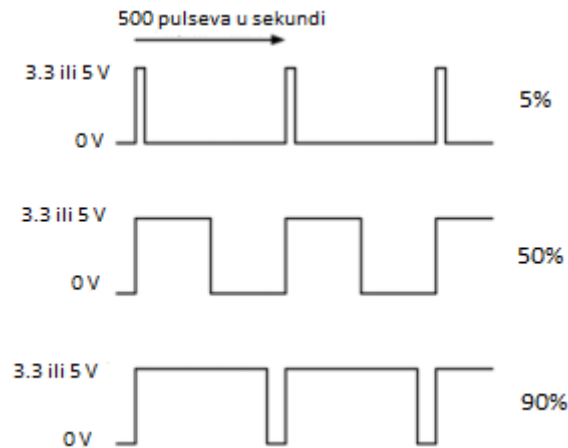
2.2 Upravljanje motorima

U ovoj sekciji ću uz pomoć slika i uz nešto malo teksta demonstrirati kako se može upravljati elektromotorima preko Raspberry Pi-a koristeći programski koda napisan u Pythonu, te uz pomoć Pythonovih biblioteka RPi.GPIO i pigpio. Vrlo važna stvar kod ovoga i sličnih projektnih zadataka je da budete vrlo pažljivi pri spajanju uređaja koje koristite, te da jako dobro proučite specifikacije sa svaki uređaj kako ne bi došlo do neželjenih posljedica, kao npr. pregaranja ploče Raspberry Pi-a ili elektromotora.

Komponente koje sam koristio za izradu ovog praktičnog projekta su:

- Raspberry Pi
- Zaslona
- Tipkovnica i miš
- HDMI kabal
- Adapter za napajanje za Raspberry Pi
- 2 DC motora

- Servo motor
- Stepper motor
- H-bridge
- Driver za stepper motor
- Vanjski napon za pokretanje stepper motora



Slika 2.8: PWM prikazan grafički

Za kotroliranje DC motora i servo motora, koji reagiraju samo na analogni signal, koristit ćemo PWM³ tehniku. Razlog tomu je što naš Raspberry Pi generira samo digitalni signal, a nema konverter koji bi konvertirao digitalni signal u analogni. Stoga, u par rečenica ćemo reći nešto o PWM³ tehnici. Ova tehnika nam omogućuje kontrolu nad uređajima koji reagiraju na analogni signal sa uređajima koji odašilju samo digitalni signal. Jednostavnije rečeno, pomoću ove tehnike obmanjujemo analogne uređaje, te ih tako možemo kontrolirati pomoću digitalnog signal koji oni vide kao analogni. PWM³ ćemo najlakše objasniti pomoću slike 2.8 na kojoj su prikazana 3 stepenasta grafa, zapravo svaki od tih grafova predstavlja PWM³. Sva tri grafa imaju isti vremenski period (1s) i isti broj odašiljanih pulseva (u našem slučaju 500), ono što se razlikuje je postotak vremena u kojemu se odašilju pulsevi koji predstavljaju visoki napon (u slučaju Raspberry Pi-ja visoki napon je 3.3V ili 5V), odnosno postotak vremena u kojemu se odašilje niski napon (u slučaju Raspberry Pi-ja niski napon je 0V). Upravo ovaj postotak vremena odašiljanja visokog napona zove se radni ciklus i time mijenjamo brzinu motora. Naš Raspberry Pi, što možemo vidjeti i na slici, 2.8 odašilje 500 pulseva u sekundi (tj. frekvencija signala je 500 Hz) na što motori ne mogu reagirati, ono što motori vide je radni ciklusa, tj. to im predstavlja analogni signal i raditi će pod naponom koji je ovisi o veličinim radnog ciklusa. Naprimjer, kod prvog grafa na slici 2.8 radni ciklus iznosi 5%, što znači da će motor vidjeti analogni signal čiji je napon 0.25V ili 0.165V (ovisno jeli motor priključen na 5V ili 3.3V iglicu). Dakle, ovih 500 pulseva u sekundi analogni signal i brzinu motora mijenjamo promjenom intenziteta tog signala, tj promjenom veličine radnog ciklusa. Intenzite se kreće u rasponu od 0 do 100 (0 = 0% napona, 100 = 100% napona). Kako

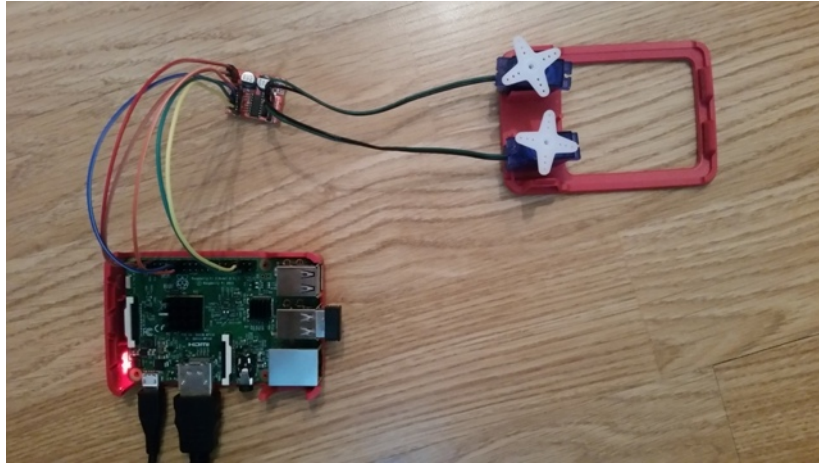
³eng. Pulse Width Modulation

bi mogli koristiti PWM tehniku moramo prvo instancirati objekt `RPi.GPIO.PWM(pin,freq)` koji ekapsulira metode pomoću kojih generiramo PWM signal, mijenjamo mu radni ciklus te zaustavljamo. Objekt prima 2 parametra, pin koji označava iglicu kroz koju će se signal odašiljati i freq je parametar koji određuje koliko želimo frekvenciju signala.

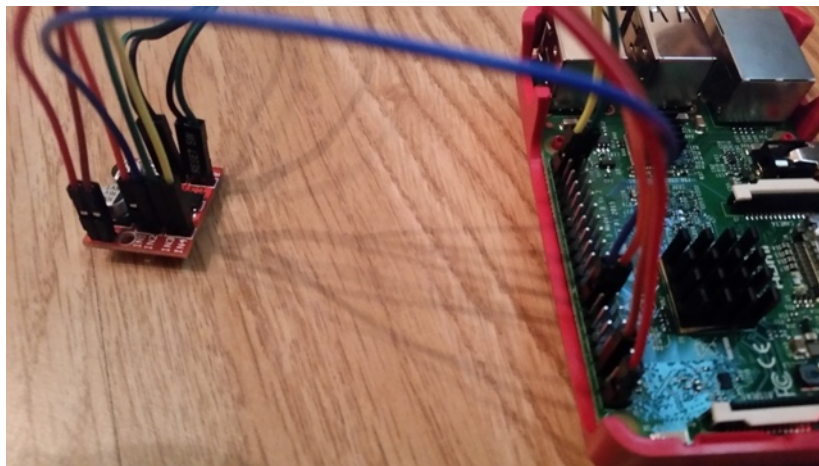
Upravljanje DC motorima

Za početak, demonstrirati ćemo upravljanje DC motorima koje ćemo kontrolirati pomoću već spomenutog PWM³ mehanizma, te Python `RPi.GPIO` biblioteka pomoću koje vršimo kontrolu nad GPIO iglicama. GPIO iglice ćemo dohvaćati pomoću fizičkog načina numeriranja. Kako bi koristili pinove potrebno je prvo naznačiti hoće li oni služiti za slanje ili primanje napona, tj za ulaz ili izlaz informacija. Da bi mogli pokrenuti 2 DC motora potrebno je enkapsulirati klasu `Motor` (vidi programski kod 2.1) a zatim klasu `ABMotors` (vidi programski kod 2.2) i na kraju napraviti glavni program (vidi programski kod 2.3) koji će ih pokrenuti. Također, za pokretanje oba DC motora, istovremeno, potreban nam je i H-bridge, posebna elektronička sklopka koja omogućuje upravljanje dvama motorima. Naponom, uzemljenjem i GPIO iglice na Raspberry Pi-u sapajamo na H-bridge, a onda motore spajamo na njega. Napon koji Raspberry Pi propusti prema njemu omogućuje pokretanje motora, kroz iglice IN1 i IN2 šaljemmo vrijednosti u vidu PWM-a kojima naznačujemo kretanje motora A, a kroz IN3 i IN4 vrijednosti kojima naznačujemo kretanje motora B. Iglice ćemo spajati na sljedeći način (vidi sliku 2.10):

- H-bridge + iglica na Pi-jevu iglicu 2 (5V)
- H-bridge - iglica na Pi-jevu iglicu 6 (GND)
- H-bridge IN1 iglica na Pi-jevu iglicu 15
- H-bridge IN2 iglica na Pi-jevu iglicu 16
- H-bridge IN3 iglica na Pi-jevu iglicu 37
- H-bridge IN4 iglica na Pi-jevu iglicu 38
- Motori na MOTOR-A i MOTOR-B iglice



Slika 2.9: Upravljanje DC motorima



Slika 2.10: Kako spojiti DC motore da bi mogli upravljati sa oba istovremeno

Ako motore postavimo tako da na njima vidimo otvor kroz koji prolaze žice, onda će se pokretanjem koda 2.3 oba motor, i A i B vrtjeti u pozitivnom smjeru. U slučaju da u istom programskom kodu na liniji 9 promijenimo prvi parametar “i” funkcije “ABmotors.set_speeds” u “-i”, tj. funkciji kao prvi parametar prosljeđujemo negativne vrijednosti, tada će se motor A vrtjeti u negativnom smjeru. Ako promijenimo drugi parametar na isti način kao i prvi, tada će se motor B vrtjeti u negativnom smjeru.

```

1 import RPi.GPIO as GPIO
2
3 GPIO.setmode(GPIO.BOARD)
4
5
6 class Motor:
7     __slots__ = '_pin_positive', '_pin_negative', '_pwm_positive', '_pwm_negative'
8
9     def __init__(self, pin_positive, pin_negative):
10

```



```

11     self._pin_positive = pin_positive
12     self._pin_negative = pin_negative
13     GPIO.setup(self._pin_positive, GPIO.OUT)
14     GPIO.setup(self._pin_negative, GPIO.OUT)
15     self._pwm_positive = GPIO.PWM(self._pin_positive, 500)
16     self._pwm_negative = GPIO.PWM(self._pin_negative, 500)
17     self._pwm_positive.start(0)
18     self._pwm_negative.start(0)
19
20     def set_speed_positive(self, speed):
21         if speed > 100:
22             speed = 100
23         if speed < 0:
24             speed = 0
25         self._pwm_negative.ChangeDutyCycle(0)
26         self._pwm_positive.ChangeDutyCycle(speed)
27
28     def set_speed_negative(self, speed):
29         if speed > 100:
30             speed = 100
31         if speed < 0:
32             speed = 0
33         self._pwm_positive.ChangeDutyCycle(0)
34         self._pwm_negative.ChangeDutyCycle(speed)
35
36     def stop(self):
37         self._pwm_positive.ChangeDutyCycle(0)
38         self._pwm_negative.ChangeDutyCycle(0)
39
40     def set_speed(self, speed):
41         if speed > 0:
42             self.set_speed_positive(speed)
43         elif speed == 0:
44             self.stop()
45         else:
46             self.set_speed_negative(-speed)

```

Programski kod 2.1: Enkapsulacija klase Motor

```

1 from motor import Motor
2
3 class ABMotors:
4     __slots__ = 'A_motor', 'B_motor'
5
6     def __init__(self, motor_A_positive=15, motor_A_negative=16,
7                 motor_B_positive=37, motor_B_negative=38):
8         self.A_motor = Motor(motor_A_positive, motor_A_negative)
9         self.B_motor = Motor(motor_B_positive, motor_B_negative)
10
11     def set_speeds(self, speed_A, speed_B):
12         self.A_motor.set_speed(speed_A)
13         self.B_motor.set_speed(speed_B)

```

Programski kod 2.2: Enkapsulacija klase ABMotors

```

1 from ABmotors import ABMotors
2 from motor import GPIO
3 import time
4
5 if __name__ == '__main__':
6     try:
7         ABmotors = ABMotors()
8         for i in range(101):
9             ABmotors.set_speeds(i, i)
10            time.sleep(1)
11    finally:
12        GPIO.cleanup()

```

Programski kod 2.3: Glavna skripta pomoću koje pokrećemo DC motore

Upravljanje servo motorom



Slika 2.11: Upravljanje servo motorom

Za upravljanje servo motorom koristimo pigpio biblioteku koja također služi za kontrolu nad GPIO iglicama i sadrži metodu `set_servo_pulsewidth` koja je nama potrebna, iglice ćemo u ovom slučaju dohvaćati pomoću njihovog GPIO načina numeriranja. Kako bi smo bili u mogućnosti koristiti ovu biblioteku u terminalu unutar naredbenog retka upisujemo naredbu “`sudo pigpiod`”, također trebamo paziti i da prije nego ugasimo Raspberry Pi potrebno je ugasiti i taj proces, odnosno unutar naredbenog retka upisati “`sudo killall pigpiod`”. Nadalje, unutar Python programa moramo instancirati objekt `pigpio.pi` koji enkapsulira mnogo metoda za rad s elektroničkim komponentama, pa tako i metodu `set_servo_pulsewidth` (vidi liniju 4 u programskom kodu 2.4). Metoda `set_servo_pulsewidth` ima dva parametra `pin` i `pulsewidth`. Parametar `pin` označava kroz koju iglicu će Raspberry Pi odašiljati signal prema motoru (treba voditi računa da je `pin` samo broj od prave oznake iglice, koja inače oblika `GPIOx`, gdje je `x` u rasponu od 2 do 27, a `x` je u našem slučaju broj `pin`), a `pulsewidth` parametar označava koliko pulseva šaljemo prema motoru, mogući raspon pulseva je od 500 do 2500. Program koji pokreće servo motor napisan je u samo jednoj skripti (vidi programski kod 2.4), razloga tomu je što smo imali samo jedan motor i cilj je bio samo testirati vrti li se. Motor spajamo na Raspberry Pi na sljedeći način (vidi sliku 2.11):

- Crvene žice motora s 2. iglicom (5V)
- Smeđe žice motora s 6. iglicom (GND)
- Narančastu žicu s 7. iglicom (GPIO4⁴ iglicom)

Ukoliko prema motoru šaljem 1500 pulseva, ako je dobro kalibriran (okretati potenciometar na motoru sve dok se motor ne prestane okretati), on bi treba mirovati. Ako motor postavimo tako da mu vidimo otvor kroz koji prolaze žice slanjem pulseva u rasponu od 1500 do 2500 (rastući niz) on bi trebao početi ubrzavati u pozitivnom smjeru, a slanjem pulseva u rasponu od 1500 do 500 (padajući niz) on trebao usporaviti i nakon toga početi ubrzavati u negativnom smjeru.

```

1 import pigpio
2 import time
3
4 pi = pigpio.pi();
5 pin = 4
6
7 if __name__ == '__main__':
8     pi.set_servo_pulsewidth(pin,1500)
9     print('tuning\n')
10    time.sleep(180)
11
12    print('first direction\n')
13    for j in range(1500,2501):
14        pi.set_servo_pulsewidth(pin , j)
15        print('forward ' + str(j))
16        time.sleep(0.01)
17
18    print('second direction\n')
19    for j in range(1500,499,-1):
20        pi.set_servo_pulsewidth(pin ,j)
21        print('backward ' + str(j))
22        time.sleep(0.01)
23
24    pi.set_servo_pulsewidth(pin ,0)

```

Programski kod 2.4: Skripta pomoću koje pokrećemo servo motor

Upravljanje stepper motorom

O pokretanju stepper motora reći ćemo samo da je također kao i kod DC motora korištena Pythonovu RPi.GPIO biblioteka, a pinove smo dohvaćali pomoću GPIO načina numeriranja, što se u našem kodu postiže naredbom “GPIO.setmode(GPIO.BCM)”. Još smo koristili i driver stepper motora kako bi smo motor mogli povezati sa Raspbrry Pi-om i vanjskim izvorom napona, koji se sastoji od 4 Maxell baterije veličine D od 1.5V, što je također nešto novo i što nismo koristili pri upravljanju DC motora i servo motora. Stepper motor sam pokrenuo samo pomoću jedne Python skripte, glavne skripte (vidi programski kod 2.5), a iglice sam spajao na sljedeći način (vidi sliku 2.12):

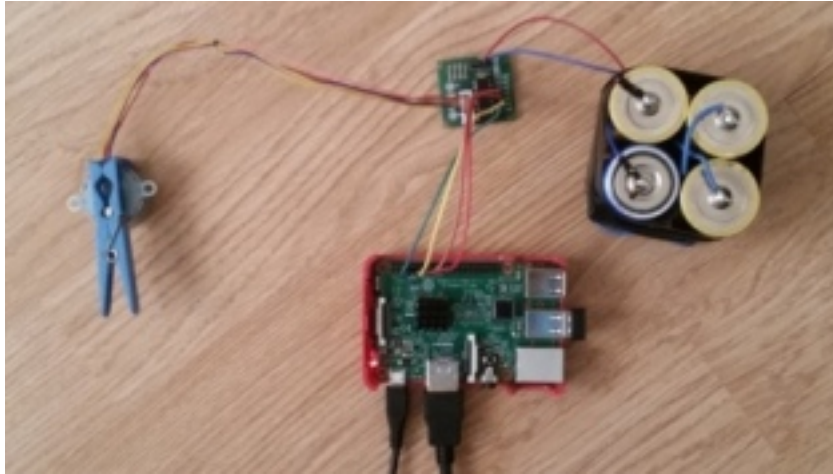
⁴GPIO način numeriranja

- Driver + iglicu s pozitivnim polom vanjskog napona (crvena žičica)
- Driver - iglicu s negativnim polom vanjskog napon (plava žičica)
- Driver IN1 iglica na Pi-jevu iglicu 7 (GPIO4⁴)
- Driver IN2 iglica na Pi-jevu iglicu 11 (GPIO17⁴)
- Driver IN3 iglica na Pi-jevu iglicu 13 (GPIO27⁴)
- Driver IN4 iglica na Pi-jevu iglicu 15 (GPIO22⁴)



Slika 2.12: Kako spojiti stepper motor

Ako motor postavimo tako da mu vidimo otvor kroz koji prolaze žice, pokretanjem programskog koda 2.5 osovina motora će se okretati u negativnom smjeru. Ukoliko želimo da se osovina okreće u pozitivnom smjeru to ćemo postići tako što ćemo u 29 liniju unijeti sljedeći kod: `“GPIO.output(StepPins[pin], Seq[-(halfstep+1)][pin])”`. Odnosno unosom ovoga koda promijenit ćemo redoslijed pobude elektromagneta u stepper motoru, a ovaj redoslijed predstavljen je listom `“Seq”`. Što znači, ako samo pokrenemo programski kod 2.5 elementi liste `“Seq”` uzimati će se redoslijedom odozgo prema prema dole. Naprimjer, uzmemo li prvi element liste `“Seq”`, što je opet lista čija je vrijednost `(1,0,0,0)`, pobuditi ćemo samo prvi elektromagnet, uzmemo li drugi element (čija je vrijednost `(1,1,0,0)`) te iste liste buditi ćemo i prvi i drugi elektromagnet, a uzmemo li pak treći element (čija je vrijednost `(0,1,0,0)`) pobuditi ćemo samo drugi elektromagnet. Očito je da vrijedi sljedeće, 1 označava pobudu, a pozicija na kojoj se 1 nalazi u elementu liste `“Seq”` označava koji se elektromagneti pobuđuju. Sada je lako zaključiti što će se događati daljnjim izvršavanjem programa, a što u slučaju kada kod iz linije 29 zamjenimo gore navedenim kodom. I za kraj ćemo pojasniti for petlju u liniji 29, jedinu preostalu nejasnoću. Općenito, stepper motorima kojima se osovina okreće za polukoračnom metodom dovoljno je 8 puta proći kroz elemente liste `“Seq”` (njih zovemo koraci) i glavna osovina bi se zarotirala za puni krug. Dakle, za jedan ciklus moramo proći kroz 8 koraka, a za jedna okret od punog kruga moramo proći 8 ciklusa. Kako naš stepper motor ima i sustav zubčanika koji izaziva smanjenje brzine u vrijednosti $1/64$, mi moramo načiniti $8 * 8 * 64 = 4096$ koraka. Dakle, kako lista `“Seq”` ima 8 elemenata, koji predstavljaju slijed pobuđivanja elektromagneta, glavna osovina da bi načinila rotaciju od punoga kruga moramo proći $4096/8 = 512$ puta kroz tu listu.



Slika 2.13: Upravljanje stepper motorom

```
1 import time
2 import RPi.GPIO as GPIO
3
4 GPIO.setmode(GPIO.BCM)
5
6 StepPins = [4,17,27,22]
7
8 for pin in StepPins:
9     print ('Setup pins')
10    GPIO.setup(pin , GPIO.OUT)
11    GPIO.output(pin ,0)
12
13
14 Seq = [[1,0,0,0],
15        [1,1,0,0],
16        [0,1,0,0],
17        [0,1,1,0],
18        [0,0,1,0],
19        [0,0,1,1],
20        [0,0,0,1],
21        [1,0,0,1]]
22
23 for i in range(512):
24     for halfstep in range(8):
25         for pin in range(4):
26             GPIO.output(StepPins[pin],Seq[halfstep][pin])
27             time.sleep(0.001)
28
29 GPIO.cleanup()
```

Programski kod 2.5: Skripta pomoću koje pokrećemo stepper motor

Literatura

- [1] E. Upton, G. Halfacree, Raspberry Pi User Guide, John Wiley & Sons Ltd., UK, 2012
- [2] https://en.wikipedia.org/wiki/Raspberry_Pi
- [3] <https://en.wikipedia.org/wiki/Raspbian>
- [4] https://en.wikipedia.org/wiki/Electric_motor
- [5] <https://hr.wikipedia.org/wiki/Elektromotor>
- [6] <https://opensource.com/resources/raspberry-pi>
- [7] <https://thepihut.com/blogs/raspberry-pi-tutorials/17817296-alternative-raspberry-pi-operating-systems>
- [8] <https://www.coursera.org/learn/raspberry-pi-platform/home/welcome>
- [9] <https://www.hackster.io/taifur/complete-motor-guide-for-robotics-05d998>
- [10] <https://www.raspberrypi.org/documentation/usage/gpio-plus-and-raspi2/>
- [11] <https://www.raspberrypi.org/magpi/raspberry-pi-3-specs-benchmarks/>
- [12] <https://www.raspbian.org/>
- [13] http://www.robotiksistem.com/dc_motor_properties.html
- [14] http://www.robotiksistem.com/servo_motor_types_properties.html
- [15] http://www.robotiksistem.com/stepper_motor_types_properties.html
- [16] M.Schmidt, J. Carter, Raspberry Pi: A Quick-Start Guide, 2nd Edition, The Pragmatic Bookshelf, Dallas, Texas & Raleigh, North Carolina USA, 2014
- [17] S. Shah, Learning Raspberry Pi, Packt Publishing Ltd., Birmingham, UK, 2015
- [18] W. Gay, Mastering the Raspberry Pi, Apress Berkely, CA, USA, 2014
- [19] razni izvori s Google images