

Univerzitet Singidunum

Tehnički fakultet

**IoT – Evidencija prisustva putem WiFi-ja**

Diplomski rad

Mentor: Prof.Dr. Tanasković Marko

Student: Sekulović Milorad 201905

Beograd, 2017

Sadržaj

[Apstrakt 3](#_Toc485333411)

[Problem 3](#_Toc485333412)

[IoT 3](#_Toc485333413)

[Motivacija za projektom 4](#_Toc485333414)

[Druga rešenja 4](#_Toc485333415)

[Idejno rešenje 6](#_Toc485333416)

[Hardverska platforma 7](#_Toc485333417)

[Raspberry Pi 3 7](#_Toc485333418)

[Operativni sistemi 8](#_Toc485333419)

[SoC 11](#_Toc485333420)

[RAM 15](#_Toc485333421)

[I/O and LAN 15](#_Toc485333422)

[WiFi/Bluetooth 16](#_Toc485333423)

[Napajanje 16](#_Toc485333424)

[GPIO 16](#_Toc485333425)

[EKRAN 16](#_Toc485333426)

[Softverska implementacija 16](#_Toc485333427)

[Verifikacija 16](#_Toc485333428)

[Zakljucak 16](#_Toc485333429)

[Literatura 16](#_Toc485333430)

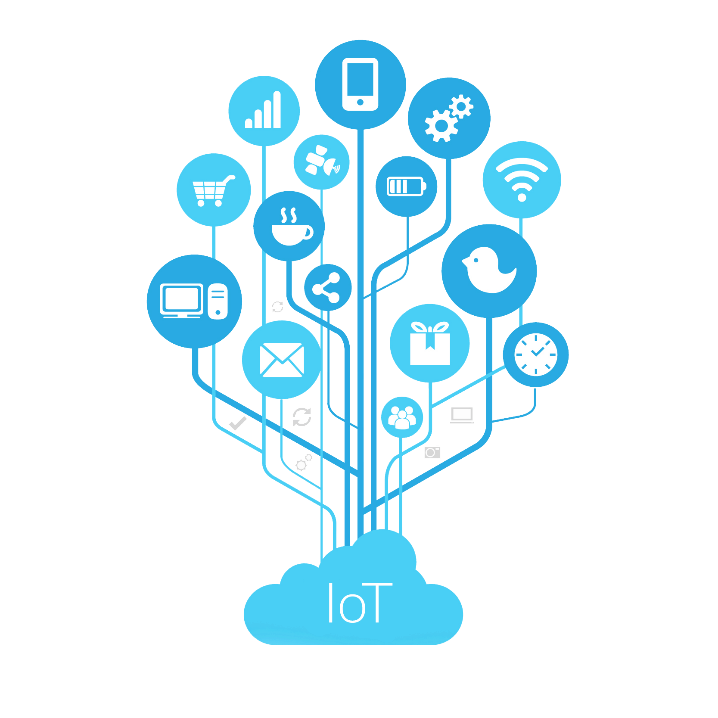
# Apstrakt

## Problem

U ovom radu se bavi problemom praćenja prisustva u raznim primjenama. Pod praćenjem prisustva podrazumijevaju se evidencije dolaska i odlaska, logovi prisutnosti i slično. Na ovaj način se automatizuje proces, jer je u mnogim primjenama nemoguće bez pomoći sistema evidencije zapravo vršiti evidenciju, što je uslovljeno velikom brojnošću ljudi koje treba evidentirati, kao i greškama namjernim ili nenamjernim za prijavom na evidenciju. Trenutno postoji veliki broj sistema za evidenciju i zasnovanim na raznim tehnologijama, ali o njima će biti riječi kasnije. Kada se govori o sistemu evidencije, postoje dvije strane, a to su strana koja evidentira prisustvo i strana koja je evidentirana. Evidencija se čuva na određenom medijumu. Ono što može biti dio sistema je i verifikacija u toku trajanja, da je subjekat evidencije zapravo prisutan. Konkretno problem evidencije se može javiti u kompanijama, na fakultetima, u saobraćaju, a sa razvojem IoT[[1]](#footnote-1)-a sve više i kod kuće, u privatne svrhe.

### IoT

To je mreža fizičkih stvari, ugrađene elektronike, senzora, aktuatora i softvera, koji mogu da razmjenjuju podatke među sobom i sa spoljnim uređajima . IoT-GSI[[2]](#footnote-2) je definisala IoT kao “Globalnu infrastrukturu informatičkog društva koja omogućava napredne usluge (fizičkim i virtualnim) umrežavanjem stvari, pritom se zasnivajući na postojećim i interoperabilnim informacionim i komunikacionim tehnologijama u razvoju. U tu svrhu termin – stvar predstavlja predmet fizičkog svijeta informatija ili riječ, koji je moguće identifikovati i koji može biti integrisan u komunikacionim mrežama”  
Sami termin IoT je predložen od strane Kevina Eštona 1999. Godine. Sva istraživanja pokazuju trend rasta samih uređaja I smatra se da će do 2020. Biti oko 26 milijardi uređaja. Ovakav tempo razvoja je I jedan od razloga za postanak IP protokola verzije šest, s obzirom da postojeći način adresiranja putem IP protokola verzije četiri ima mogućnost da adresira 4,3 miliona uređaja.



Ilustracija 1: IoT

## Motivacija za projektom

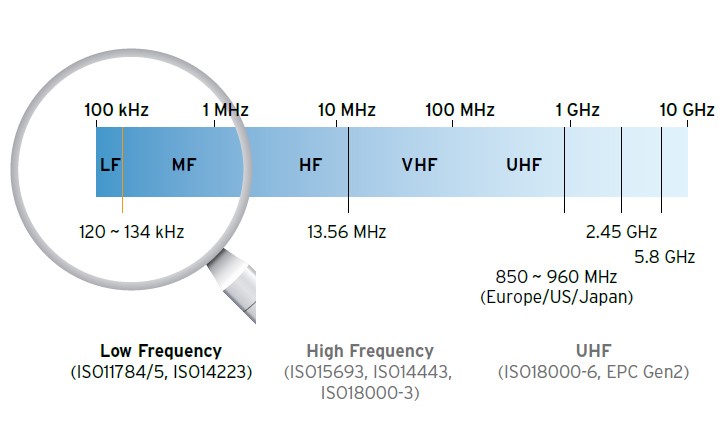
Sama motivacija za radom na ovakvom projektu se javila prilikom saradnje sa profesorima sa fakulteta I konstantnim radom na IoT projektima u okviru aktivnosti na fakultetu. Uz razvoj drugih projekata, javila se I ideja evidencije prisutnosti studenata na predavanjima, usljed potreba profesora da smanje vrijeme potrebno za evidenciju prisutnih studenata, kako na predavanjima, tako i na polaganjima ispita I kolokvijuma. Ovo je dalo osnovnu motivaciju za rešenjem konkretnog problema, koji bi omogućio olakšanje rada I uštedu vremena.

## Druga rešenja

Kao rešenja koja se već nalaze na tržištu i koriste, to su kontrola pristupa i evidencije putem beskontaktnih kartica, otiska prsta, sifrom, infra crvenim senzorima, laserskim senzorima, kamerama isl.

Zavisno od toga da li je bitno konkretno da se zna ko je i kada bio u odredjenoj prostoriji ili dijelu na koji je instaliran sistem, što zavisi i od projektovanja samog sistema, kao najpouzdaniji vid je kontrola pristupa i prisustva otiskom prsta. U praksi se međutim koristi najviše za kontrolu pristupa, ali ne i za praćenje vremena zbog potrebe da se postavi i na ulazu i na izlazu, što zahtijeva dodatne i bespotrebne troškove, takođe ukoliko postoji veliki broj ljudi koji treba da uđe u prostoriju za kratak period, to predstavlja problem.

Verifikacija putem šifre, u zavisnosti od broja korisnika zahtijeva veliki broj šifara koje bi bile dodijeljene svakom korisniku, a postoji i mogućnost da jedna osoba ukuca više puta istu šifru i prijavi se za ostale korisnike. Pristup putem kartica i tagova različite frekvencije u radijskom spektru je najčešće korišćen vid kontrole pristupa, kao i praćenja prisutnosti.

Kartice i tagovi se u pogledu frekvencije dijele na Nisko Frekventne, Visoko Frekventne i Ultra Visoko Frekventne, i oni se kreću u rangu oko 100kHz, 10MHz i 1GHz respektivno. Kartice mogu biti aktivne i pasivne, u zavisnosti od toga da li koriste neki izvor energije za napajanje ili ne. Svaki frekventni opseg ima svoje prednosti i mane, a postoje i hibridne kartice koje spajaju više prednosti ukoliko je potrebno. Razlike se ogledaju u količini podataka koje mogu da sačuvaju, blizini na kojoj mogu biti očitane, cijeni proizvodnje.

Ilustracija 2: Frekventni opsezi kartica koje rade na radio talasima

Mane ovog sistema za konkretan problem je potreba za konstantom izradom kartica za studente, kao i mogućnost da se jedna osoba otkuca različitim karticama.

Sistemi koji uključuju video nadzor se zasnivaju na prepoznavanju lica I zapravo su upraksi se pokazali kao mnogo komplikovaniji za implementaciju I pouzdano korišćenje. Za konkretan slučaj je rađeno testiranje s obzirom da je ovaj sistem bio isplativ sa korišćenjem postojećeg sistema video nadzora kao i Open-Source[[3]](#footnote-3) biblioteka za datekciju i prepoznavanje lica. U praksi je moguće porediti lice sa slikama u postojećoj bazi, ali bi takođe bilo potrebno napraviti mnogo više od jedne slike da bi sistem pouzdano radio, a slike bi morale imati razne izraze na licu, što nije uvijek moguće realizovati, s obzirom da bi i pristanak svih osoba bio potreban.

Ostali sistemi uglavnom se koriste za praćenje prisustva bez osvrta na identitet, a to su sistemi lasera i infra crvenih senzora koji samo imaju uvid o stanju da li i koliko osoba ima u dijelu na koji su senzori ugrađeni.

## Idejno rešenje

Kao idejno rešenje u ovom slučaju podrazumijeva se da svaka osoba ima mobilni uređaj koji može uspostaviti WiFi konekciju, a u suprotnom ručno se evidentira putem softvera koji bi se nalazio na uređaju koji je prenosiv. Sami uređaj bi imao ekran osjetljiv na dodir, mogućnost priključivanja na internet putem Ethernet-a, kao i mogućnost dijeljenja veze putem WiFi-ja. Interfejs na uređaju bi nakon njegovog uključivanja imao podešavanja praćenja prisustva kao i dugme za pokretanje, nakon čega bi mobilni uređaji se povezivali na WiFi. Omogućiti korisnicima korišćenje interneta je opciono i prilikom pokretanja se podešava. Prilikom prve konekcije bilo bi potrebno pristupiti web stranici koja se nalazi na samom uređaju i registrovati ime i broj indeksa, nakon čega bi uređaj čuvao podatke i registrovao tog korisnika sa svakom konekcijom. Konkretno to znači da korisnik koji se registruje na tom uređaju, ukoliko je čekirana opcija automatske konekcije na WiFi, ne mora da brine o tome da li je njegovo prisustvo evidentirano, dok korisnik koji prati prisustva ima generisanu tabelu sa brojevima indeksa, imenima, kao i vremenu koje je korisnik proveo. Dalje moguće je čuvati ove podatke na Cloud[[4]](#footnote-4)-u, kao i proširiti funkcionalnosti da uređaj komunicira sa drugim sklopovima i dijeli podatke, sačinjava razne statistike. Prednost ovakvih aplikacija i jeste na prvom mjestu mogućnost daljeg proširenja i prilagođavanja. Kao platforma za potrebe ovoj projekta odabran je Raspberry Pi, kompjuter koji sadrži sve komponente potrebne za funkcionisanje na jednoj ploči. Sam kompjuter će biti proširen sa ekranom osjetljivim na dodir, dijagonale 3,5 inča i imati zaštitnu kutiju od akrila, providne termo-otporne plastike. Ideja je da uređaj bude prenosiv i da uključivanjem u struju pokrene sistem sa gore pomenutim interfejsom za kontrolu i podešavanja sistema za praćenje prisustva.

# Hardverska platforma

Raspberry Pi platforma je dozivjela veliki uspjeh na trzistu i posjeduje dosta dokumentacije pa je iz tog razloga veoma lako zapoceti rad sa platformom. Sama platforma je SBC[[5]](#footnote-5) koja sadrži ploču na kojoj je mikroprocesor, memorija, I/O portovi. Sama platforma ima nekoliko verzija, s obzirom da nije proširiva sem u pogledu memorijskog prostora putem SD Kartice ili USB memorijskog diska. Konkretno u ovom slučaju je korišćen Raspberry Pi 3, Model B.

## Raspberry Pi 3

Hardverske specifikacije

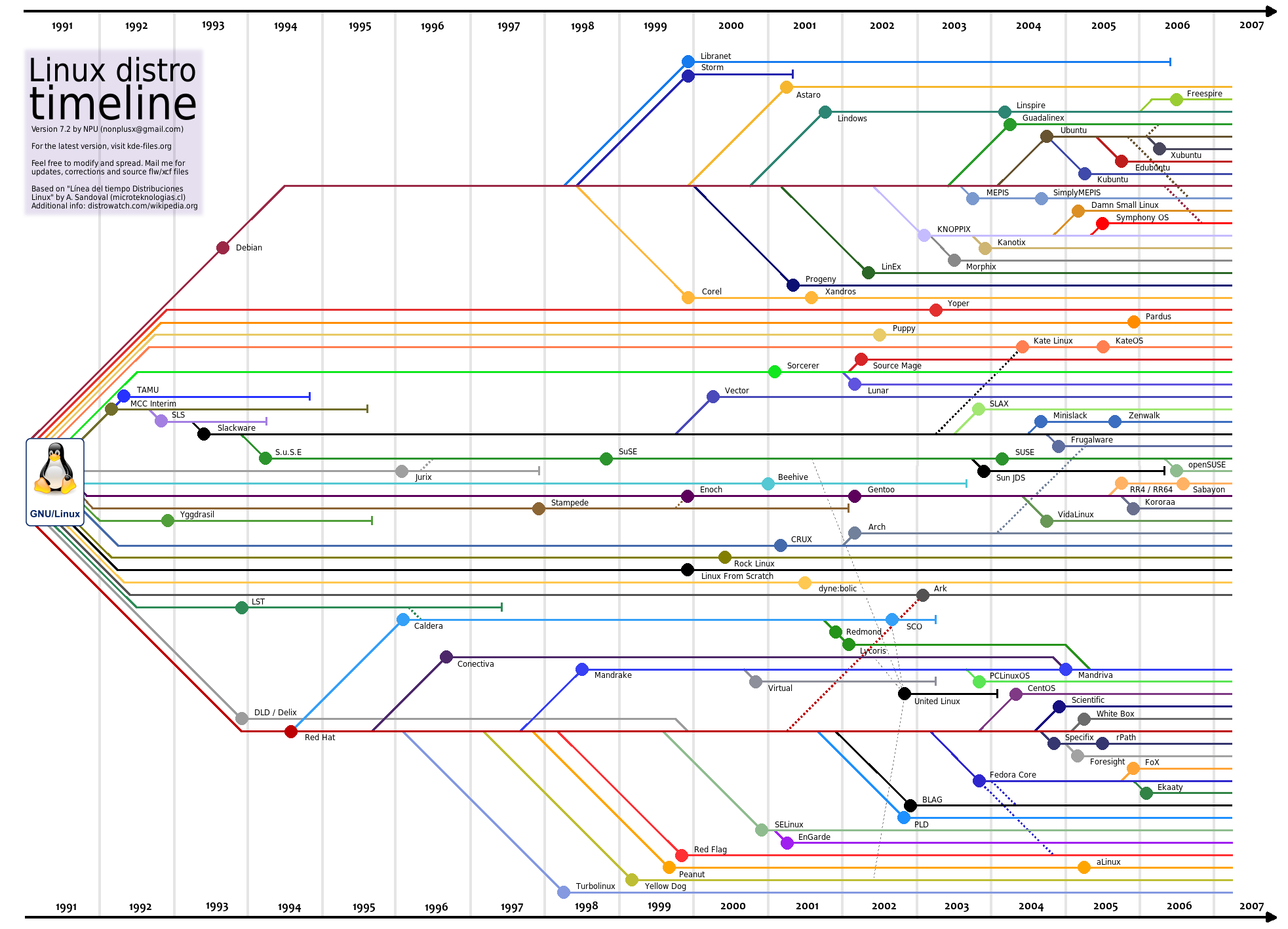
|  |  |
| --- | --- |
| SoC | BCM2837 |
| CPU | Quad Core Cortex A53 – 1.2GHz |
| Instrukcijski Set | ARMv8-A |
| GPU | 400MHz VideoCore IV |
| RAM | 1GB SDRAM |
| Storage | Micro-SD |
| Ethernet | 10/100 Mbps |
| Wireless | 802.11n / Bluetooth 4.1 LE |
| Video Output | HDMI / Composite |
| Audio Output | HDMI / 3.5mm audio |
| GPIO | 40 |

### Operativni sistemi

Ova platforma moze da pokrene vise raznih operativnih Sistema, I to instalacijom live verzije na micro sd kartici. Tu su Windows 10 IoT Core, Android Things, razne linux distribucije, pored njih je I RISC OS koji je prvenstveno pisan za ARM platformu. Standardno se preporučuje Linux distribucija Raspbian, s obzirom da ona sadrži sve potrebne drajvere I prilagođen je baš Raspberry Pi platformi, U ovoj distribuciji se nalazi mnostvo softvera za sam početak razvoja na ovoj platformi, kao što su: Wolfram, Mathematica, Python, BlueJ, Node-Red, Scratch. U suštini se za kontrolu GPIO pinova koristi Python, ali je moguće koristiti I druge programse jezike. Ako se koristi neka od linux distribucija, bitno je to da linux nije Real Time Operating System pa se ne može koristiti za razvoj projekata kod kojih je vrijeme izvršavanja najbitnija stvar. Kako se kod linux operativnih Sistema sva interakcija sa hardverom dešava na kernelskom nivou, dok se aplikacije pokreću u korisničkom prostoru I svaka je aplikacija u suštini process koji dobija procesorsko vrijeme, to znači da procesor određuje kada će I koliko prioriteta dodijeliti određenim priključenim uređajima. Kod projekata koji zahtijevaju velike vremenske tačnosti moguće je napisati prekidne rutine koje u određenim momentima vrše drugu logiku koja u tom trenutku ima veći prioritet, a kod linux kernel to nije moguće jer kernel odlučuje o dodjeli procesorskog vremena. U praksi postoje operativni sistemi koji su usmjereni ka tome da linux bude RTOS, kao što su RTLinux i QNX. U slučaju projekta na kom će se raditi vrijeme nije krucijalan faktor, pa je odabrana platforma sasvim dovoljna za ovu primjenu. Kašnjenja koja su uzrokovana ovom vrstom nedostatka operativnog Sistema, nisu od velike važnosti s obzirom da će samo ispitivanje korisnika da se vrši u vremenskom razmaku od deset I više sekundi.

#### Raspbian (Jessie)

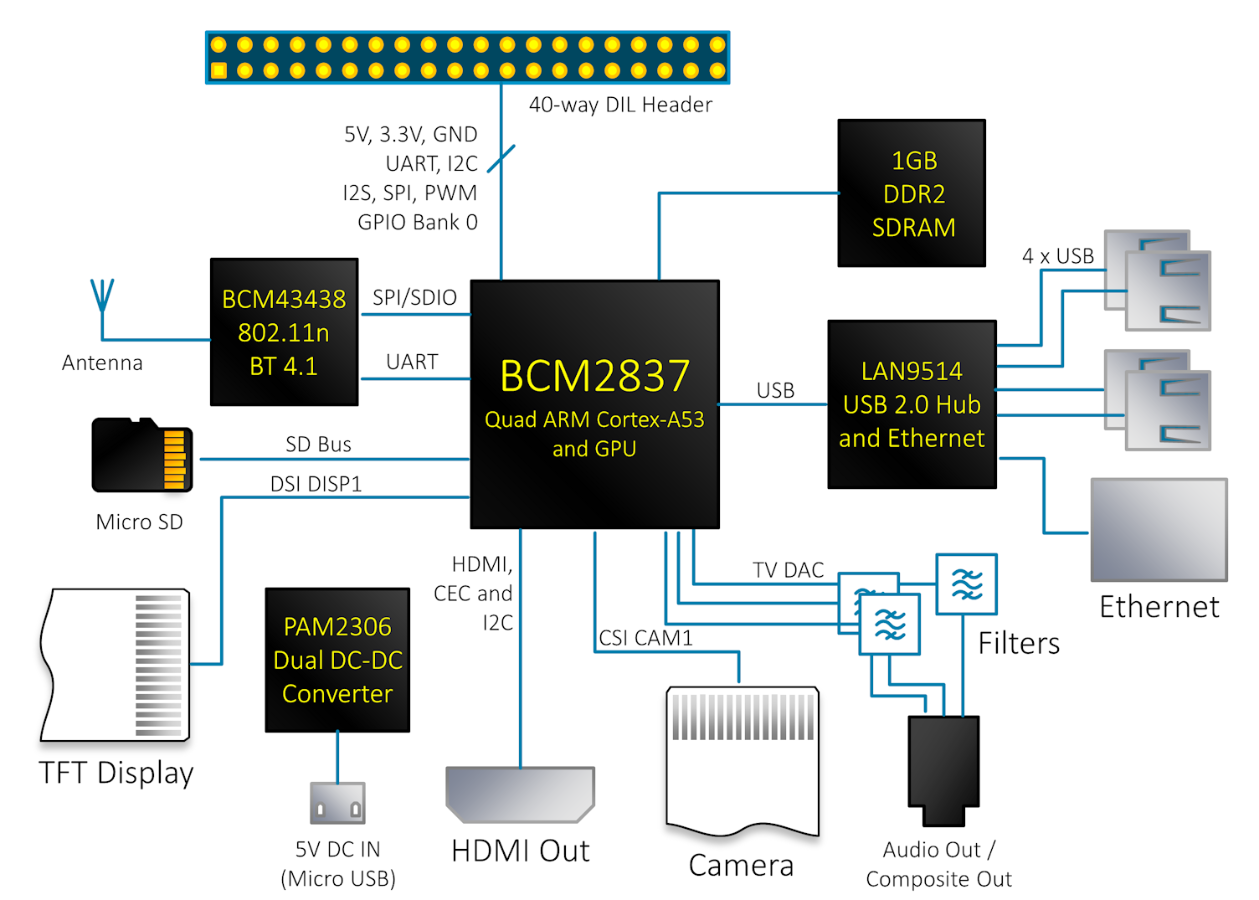
U ternutku pisanja aktuelna verzija operativnog sistema je Jessie. Raspbian je baziran na Debian operativnom sistemu, koji je najrasprostranjeniji u Linux svijetu. Jedna od odlika jeste da je dobro podržan i redovno ima ispravke i nadogradnje. Debian je volonterska organizacija koja se zasniva na: Debian Social Contract, Debian Free Software Guidelines i Debian Constitution, prema kojima developeri rade na softveru. Debian je podržan donacijama, dok se sama struktura odlučivanja u organizaciji poznata i lider projekta se bira jednom godišnje. Prilikom razvoja prate se standardi za kvalitet pisanja koda, a u cilju razvoja novih funkcionalnosti ili ispravki postojećih bug-ova. Nakon verifikacije i isporuke novog paketa, on se instalira u takozvani “pool” koji je kasnije distribuiran korisnicima širom svijeta. Po pitanju sigurnosti politika Debian-a je javno objavljivanje i rad na poboljšanjima i zakrpama. Raspbian kao package manager koristi DPKG [[6]](#footnote-6)i APT[[7]](#footnote-7) kao njegov front end. Grafičko okruženje u Raspbianu je nazvano.PIXEL (Pi Improved Xwindow Environment, Lightweight) koji je napravljen da približi korišćenje linux operativnog sistema ljudima koji se sa njim nisu do sada susretali.



Ilustracija 3: Vremenski prikaz razvoja Linux distribucija

#### Način instalacije

Instalacija operativnog sistema je procedura koja podrazumijeva odabir operativnog sistema i preuzimanje, a zatim pravljenje live sd kartice putem nekog od softverskih paketa kao što su Rufus ili Etcher, mada postoji veliki broj drugih. Nakon toga odabir .img ili .iso u softveru kao i odabir kartice na koju je potrebno instalirati OS.



Ilustracija 4: Raspberry 3 blok dijagram

### SoC

System on Chip je integrisano kolo koje može da sadrži procesorsku jedinicu, memoriju, grafičku jedinicu, WiFi, Bluetooth i slično. BCM2837 sadrži u sebi processor sa 512 KB dijeljene L2 keš memorije, grafičku jedinicu (VideoCore IV).

#### Keš memorija

Po memorijskoj hijerarhiji keš memorija je prva nakon procesorskih registara po brzini memorije, ali takođe i najmanjeg kapaciteta. Njenja efikasnost se zasniva na principu ne tako čestog prenošenja podataka u bržu mrežu i višestrukog pristupa istim podacima prije zamjene sa novim. Ovaj princip je moguć zbog lokalnosti referenci, koje mogu biti vremenske i prostorne. Prostorna lokalnost znači da je velika vjerovatnoća ako se pristupilo jednoj memorijskoj lokaciji, da će se pristupiti i nekoj okolnoj lokaciji. Vremenska lokalnost znači da ako se pristupilo nekoj memorijskoj lokaciji, postojivjerovatnoća da će se opet pristupati toj lokaciji, što je slučaj kod petlji. Procesor pri zahtjevu za nekom memorijskom lokacijom generiše njegovu adresu i ona se može nalaziti u Keš memoriji, ukoliko to nije slučaj podatak može biti u operativnoj memoriji, a u krajnjem slučaju može se nalaziti na sekundarnoj memoriji (npr. Hard disk). Kao integrisano kolo ili kao dio procesora, jedinica za upravljanje memorijom ili MMU-Memory Management Unit bavi se mapiranjem adresa. Ono što MMU radi jeste da vodi evidenciju o tome koji su djelovi operativne memorije trenutno smješteni u keš memoriji, o tome koji blok je potrebno izbaciti iz pune keš memorije, kao i ažuriranje. sadržaja operativne memorije kada je bilo upisa na lokacijama u keš memoriji, s obzirom da se u tom trenutku sadržaji na istoj memorijskoj lokaciji razlikuju. Tehnike preslikavanja:

* Direktno preslikavanje

Ovo je najjednostavnija tehnika preslikavanja koja smješta dolazeći blok iz operativne memorije na fiksnu lokaciju u keš memoriji. U ovom slučaju važi fiksna relacija gdje je i broj dolazećeg bloka iz operativne memorije, j broj bloka u keš memoriji,a NCB ukupan broj blokova. Keš memorija se sastoji od tag memorije i data memorije. Ako operativna memorija sadrži 4k blokova, keš memorija 128 blokova i veličina bloka je 16 memorijskih riječi onda važi da MMU upisuje memoriju tako što je dijeli na tri polja: polje taga(T), polje keš bloka(CB) i polje riječi (W)

*; ; ;*

Pa tako dobijamo da je T=5 bitova, CB = 7 bitova I W=4 bita, dok je A=16 bitova

|  |  |  |
| --- | --- | --- |
| T 5 | CB 7 | W 4 |
| A 16 bitna adresa u operativnoj memoriji | | |

Pomoću CB polja se određuje blok u keš memoriji, a nakon toga se poređenjem polja sa TM sa T poljem zaključuje da li je element u keš memoriji ili nije, ukoliko jeste traži se unutar bloka putem W polja, u suprotnom se prenosi iz operativne memorije u keš memoriju.

Tehnike preslikavanja:

* Asocijativno preslikavanje

Ovaj način preslikavanja je fleksibilniji jer se blok iz operativne memorije može smjestiti u bilo koji raspoloživi blok keš memorije. Adresa koja se generiše ima 2 polja za razliku od direktnog preslikavanja.

*; ;*

Ako koristimo iste podatke kao I za direktno preslikavanje onda imamo da je:

|  |  |
| --- | --- |
| T 12 | W 4 |
| A 16 bitna adresa u operativnoj memoriji | |

Pristupanje element se vrši tako što se vrijednost polja T nađe među postojećim tagovima u TM, ako T postoji, element je u odgovarajućem bloku pa se element unutar bloka traži pomoću W, u suprotno prenosi se blok iz op. memorije.

* Set-Asocijativno preslikavanje

Kod ove vrste preslikavanja memorije je podijeljena u skupove pri čemu svaki skup ima određen broj blokova. Blok se iz operativne memorije mapira u jedan set keš memorije prema formuli: gdje je i broj blokova operatiovne memorije, NS broj setova u keš memoriji I s – broj seta u keš memoriji koji se mapira. Adresa koju ova metoda generiše ima tri polja, a računaju se po sledećim formulama:

*; ; ;*

Gdje je BS broj blokova u setu. Koristeći podatke iz prethodnih primjera dobijamo

|  |  |  |
| --- | --- | --- |
| T 7 | S 5 | W 4 |
| A 16 bitna adresa u operativnoj memoriji | | |

Na osnovu polja S se određuje set u koji se mapira i ako se T nalazi među postojećim tagovima u TM za dati set onda se pomoću W polja pronalazi blok, a u suprotnom se prenosi iz operativne memorije.

Tehnike zamjene:

* Random Selection

Koristi se generator slučajnih brojeva koji generiše brojeve od 0 do N-1 gdje je N broj blokova keš memorije, tako da se odabrani nasumični blok u trenutku promjene briše i dovodi blok iz operativne memorije.

* FiFo: First In First Out

Ovom tehnikom se iz keš memorije izbacuje blok koji je najviše vremena proveo u keš memoriji.

* LRU: Last Recently Used

Ova tehnika se zasniva na izbacivanju bloka koji je najranije poslednji put korišten, ali zahtijeva i upotrebu keš kontrolera koji čuva istoriju referenci na sve blokove.

Tehnike ažuriranja:

* Tehnika upisa ako blok postoji u keš memoriji
  + Write trough

Svakom operacijom upisa se vrši upis u operativnu i keš memoriju

* + Write back

Upis se radi samo u keš memoriju, dok se upis u operativnu memoriju odlaže sve do trenutka kada blok treba da bude zamijenjen drugim blokom. Svaki blok u keš memoriji ima dirty bit koji je indikator na to da li je došlo do promjena na bloku u keš memoriji.

* Tehnika upisa ako blok ne postoji u keš memoriji
  + Write allocate

Blok se prebacuje iz operativne u keš memoriju, a onda se postupa kao u prethodnoj tehnici

* + Write no allocate

Blok se ne prebacuje, već se direktno vrši upis u operativnu memoriju.

* Tehnika čitanja ako blok postoji u keš memoriji – pročita se vrijednost
* Tehnika čitanja ako blok ne postoji u keš memoriji
  + Direktno prosleđivanje

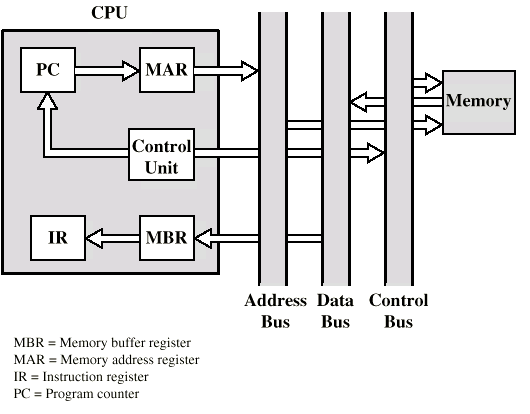
Blok se prebacuje iz operativne u keš memoriju a odmah po pristizanju se prosleđuje procesoru

* + Prosleđivanje sa zadrškom

Blok se prebacuje iz operativne u keš memoriju i tek kad se u potpunosti prebaci podatak se prosleđuje procesoru.

#### Procesor

Kao centralni dio svakog racunara, dio koji izvrsava sve kalkulacije u kompjuteru je procesor. Struktura procesora sadrži centralne komponente i registarske komponente. Pod centralnim komponentama podrazumijeva se Aritmeticko-logicka jedinica i ona služi da izvršava instrukcije programa. Registarske komponente omogućavaju rad ALU i to su: programski brojač koji sadrži adresu sledeće instrukcije(Program Counter), adresni registar memorije(Memory Address Register) koji sadrži adresu memorijske lokacije kojoj treba pristupiti, privatni registar podataka (Memory Data Register, takođe i Memory Buffer Register) koji sadrži pročitan podatak sa memorijske lokacije ili podatak koji treba da bude upisan, privatni registar instrukcije (Instruction Register) koji sadrži instrukciju, privatni registar izvorišnih operanada koji su vezani na ulazne linije ALU, privatni registar rezultata koji je vezan na izlaz iz ALU.

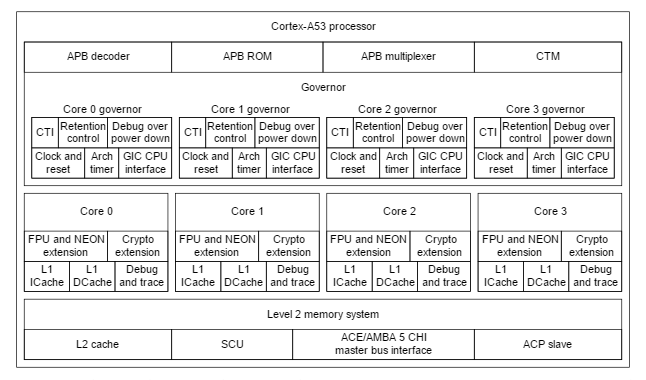


Ilustracija 5: Tok jednog ciklusa u procesoru, tokom preuzimanja podataka

Kako procesorski registri su mnogo brži od sledeće najbrže memorije, a to je Cache memorija koja može postojati u više nivoa, tj brzina i kapaciteta, procesor koristi algoritme za popunjavanje ove memorije. Ti algoritmi se baziraju na pogađanju instrukcija i podataka koji će biti potrebni procesoru u trenutku opsluživanja određenog toka. Prema nekim podacima današnji procesori pogađaju oko 80% potrebnih stvari, pa prema tome nema potrebe da se obraćaju sporijim memorijama u trenutku kada se izvršava neka lista instrukcija.

##### ARM Cortex-A53

U slučaju Raspberrz Pi platforme u pitanju je Cortex A53 procesor koji ima četiri jezgra, koji ima cache memoriju podijeljenu u 2 nivoa. Na prvom nivou se nalazi 32kB memorije raspoređene na 4 jezgra kao instrukcijski keš I keš podataka. Na drugom nivou se nalazi 512kB koji je dijeljen između svih jezgara. Procesor je 64 bitni, što znači da je širina instrukcija I memorijskih lokacija sa kojima processor može da radi 264 pa time doprinosi većem broju instrukcija(kada se pređe na 64bitne instrukcije, s obzirom da ARM I dalje koristi instrukcije dužine 32 bita), a I samoj optimizaciji rada procesora, što se oslikava u bržem radu procesora, kao I adresiranju većeg broja RAM memorijskih lokacija.



Ilustracija 6: ARM Cortex-A53 dijagram

#### Instrukcijski setovi

Instrukcijski set je skup instrukcija kojim se specificiraju operacije koje processor može da izvrši. Tipovi instrukcija su: aritmetičke, logičke, pomjeračke, instrukcije prenosa I instrukcije skoka. Instrukcijski setovi se mogu razlikovati I po broju operanada koji učestvuju u instrukcijama kao parametri. Pa tako na primer mašine koje rade sa 0 adresa (stek mašine), sa 1 adresom (akumulatorska mašina), 2 I 3 adrese (CISC I RISC mašine). Cortex A53 je Reduced Instruction Set Computer, iako je u poslednjoj verziji ARMv8-A arhitekture uveden veliki broj kriptografskih I atomse funkcije za citanje I pisanje. Primjer broja operanada kod instrukcija u instrukcijsom setu je:

3-adresni: prvi I drugi parametar su izvorišne adrese dok je treći odredišna adresa

2-adresni: prvi I drugi parametar su izvorišne adrese, a ujedno I jedan od njih je odredišna adresa

1-adresni: akumulator izvor I odredište

0-adresni: sa steka se čita I u stek se upisuje rezultat.

Kao aritmetičke operacije podrazumijevaju se sabiranje, množenje, dijeljenje, oduzimanje. Pod logičkim instrukcijama spadaju I, ILI, Ekskluzivno ILI, NE, kao i pomjeračke instrukcije koje pomjeraju binarne riječi za jedno mjesto ulijevo ili udesno. Instrukcije prenosa služe za prenos podataka sa jednog mjesta na drugo, i to: u memorijskim lokacijama, procesorskim registrima, u instrukciji kao neposredna veličina, u registrima kontrolera periferija, u akumulatoru, na stek. Takođe postoje instrukcije skoka, koji mogu biti uslovni i bezuslovni i omogućavaju uslovna izvršavanja instrukcija.

### RAM

SoC na poleđini sadrži ELPIDA B8132B4PB LPDDR2 SDRAM memoriju, ovakav način montiranja memorije na SoC se naziva i Package On Package.

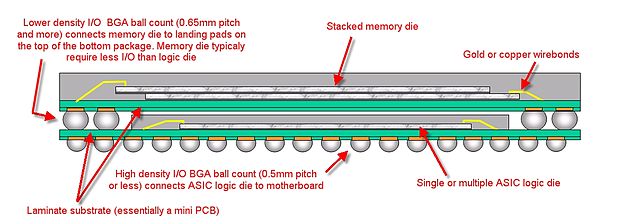
#### Tipovi RAM Memorije

* SRAM: Static Random Access
* DRAM: Dynamic Random Access
* FPM DRAM: Fast Page Mode DRAM
* EDO DRAM: Extended data-out DRAM
* SDRAM: Synchronous Dynamic Random Access
* DDR SDRAM: Double data rate synchronous dynamic RAM
* RDRAM: Rambus dynamic random access memory
* CMOS RAM: Complementary metal–oxide–semiconductor RAM
* VRAM: Video RAM ili MPDRAM - multiport dynamic random access memory

U ovom slučaju se radi o DDR SDRAM memoriji, tj. o Low Power verziji memorije koja radi na 400Mhz, što je jako bitno zbog potrošnje energije kao i zbog manjeg rasipanja eenergiju u vidu toplote.

#### Package on package

Ovo je metodata pakovanja integrisanih kola da bi se kombinovala logička i memorijska kola. Koristi se matrica kontaktnih tačkica (prethodno pinovi), tako da se montiranjem dobija stalna veza izmeđzu štampane ploče i kola. Postoje dvije konfiguracije i to: samo memorijsko vezivanje i momorijsko logičko vezivanje, tj. procesor i memorija, što je ovdje slučaj.



Ilustracija 7: Logičko-memorijsko slaganje kola

Ovim se vrši ušteda mjesta na štampanoj ploči, pa time je i izvodljivo da se dobije ovakav uređaj na tako malim dimenzijama.

### I/O and LAN

Na ploči se takođe nalaze SMSC LAN9514 kontroler koji je zadužen za kontrolu Etherneta I USB Hub-a

--- o hubovima

--- o lan kontroleru

--- o usb standardu

### WiFi/Bluetooth

.WiFi I Bluetooth se nalazi Broadcom modul za Wifi I Bluetooth.

--- o 2.4ghz wifiju

--- o bt 4.0 I 4.1

### Napajanje

O pam dc-dc

### GPIO

## EKRAN

# Softverska implementacija

A

# Verifikacija

A

# Zakljucak

A

# Literatura

a

1. Internet Of Things [↑](#footnote-ref-1)
2. Global Standards Initiative on Internet of Things [↑](#footnote-ref-2)
3. Open-Source: odnosi se na softver koji ima svoj izvorni kod dostupan pod “open-source” licencama, i njihov se kod može mijenjati, prilagođavati I poboljšavati. U zavisnosti od licence moguće je I izmenjeni kod komercijalno distribuirati. [↑](#footnote-ref-3)
4. Cloud: Ideja cloud tehnologije zasniva se na tome da svi podaci koji su neophodni korisniku (bile to aplikacije, dokumenti, hardver, ili nešto drugo) budu dostupni u svakom trenutku, naravno uz preduslov da je prethodno uspostavljena internet veza. Dakle, cloud na neki način predstavlja uslugu dostavljanja servisa umesto samog proizvoda. [↑](#footnote-ref-4)
5. SBC: Single Board Computer [↑](#footnote-ref-5)
6. DPKG: Debian Package Manager [↑](#footnote-ref-6)
7. APT: Advanced Package Manager [↑](#footnote-ref-7)