Primjena RFID čitača za evidenciju nazočnosti studenata u nastavi

Marjanović, Stjepan

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Sveučilišni studij

PRIMJENA RFID ČITAČA ZA EVIDENCIJU NAZOČNOSTI STUDENATA U NASTAVI

Završni rad

Stjepan Marjanović

Osijek, 2023.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA
I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

Obrazac Z1P - Obrazac za ocjenu završnog rada na preddiplomskom sveučilišnom studiju

Osijek, 16.09.2023.

Odboru za završne i diplomske ispite

Prijedlog ocjene završnog rada na preddiplomskom sveučilišnom studiju

Ime i prezime Pristupnika:	Stjepan Marjanović
Studij, smjer:	Sveučilišni prijediplomski studij Računarstvo
Mat. br. Pristupnika, godina upisa:	R4094, 28.07.2017.
OIB Pristupnika:	47876720961
Mentor:	izv. prof. dr. sc. Ivan Aleksi
Sumentor:	Josip Zidar, mag. ing. comp.
Sumentor iz tvrtke:	
Naslov završnog rada:	Primjena RFID čitača za evidenciju nazočnosti studenata u nastavi
Znanstvena grana rada:	**Arhitektura računalnih sustava (zn. polje računarstvo)**
Zadatak završnog rad:	Napraviti uređaj za evidenciju studenata u nastavi. Primijeniti RFID čitač iksica.
Prijedlog ocjene završnog rada:	Izvrstan (5)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 3 bod/boda Postignuti rezultati u odnosu na složenost zadatka: 3 bod/boda Jasnoća pismenog izražavanja: 3 bod/boda Razina samostalnosti: 3 razina
Datum prijedloga ocjene od strane mentora:	16.09.2023.
Datum potvrde ocjene od strane Odbora:	24.09.2023.

Potvrda mentora o predaji konačne verzije rada:	Mentor elektronički potpisao predaju konačne verzije.
	Datum:

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA
I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

IZJAVA O ORIGINALNOSTI RADA

Osijek, 25.09.2023.

Ime i prezime studenta:	Stjepan Marjanović
Studij:	Sveučilišni prijediplomski studij Računarstvo
Mat. br. studenta, godina upisa:	R4094, 28.07.2017.
Turnitin podudaranje [%]:	3

Ovom izjavom izjavljujem da je rad pod nazivom**: Primjena RFID čitača za evidenciju nazočnosti studenata u nastavi**

izrađen pod vodstvom mentora izv. prof. dr. sc. Ivan Aleksi

i sumentora Josip Zidar, mag. ing. comp.

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis studenta:

SADRŽAJ

1. UVOD 1

1.1. Zadatak završnog rada 1

2. PREGLED PODRUČJA 2

2.1. Uređaji slične funkcionalnosti dostupni na tržištu 2

2.1.1. ACS ACR1255U-J1 2

2.1.2. Anviz W1C Pro 2

2.2. Pregled korištenih tehnologija i metodologija 3

2.2.1. RFID 3

2.2.2. RFID čitač 4

2.2.3. RFID transponder 4

2.2.4. NFC tehnologija 5

2.2.5. ACR122U USB NFC Reader 6

2.2.6. Studentska iskaznica 7

2.2.7. MIFARE Classic 7

2.2.8. RASPBERRY PI 9

2.2.9. Raspberry Pi Zero W 11

2.2.10. Linux 11

2.2.11. Systemd 12

2.2.12. Libnfc 13

2.2.13. Yocto Project 13

2.2.14. Poky 14

2.2.15. BitBake 15

2.2.16. Slojevi i recepti 15

2.2.17. Bluetooth 15

2.2.18. Android 16

2.2.19. Kotlin 16

3. REALIZACIJA UREĐAJA 17

3.1. Hardverska izvedba 17

3.2. Programska podrška Raspberry Pi 17

3.3. Mobilna Aplikacija 25

4. ZAKLJUČAK 28

LITERATURA **29**

SAŽETAK **32**

ABSTRACT **33**

ŽIVOTOPIS **34**

1. UVOD

Vođenje brige o prisustvu studenata na predavanjima je repetitivan proces koji oduzima vrijedno vrijeme od svake vježbe i predavanja. Bilo da se studenti samostalno potpisuju ili ih predavač pojedinačno proziva proces dugo traje te bi bilo jednostavnije kada bi studenti mogli samo očitati svoju studentsku iskaznicu kada dođu na predavanje te time potvrditi svoje prisustvo predavanju. Kako bi sustav bio što jednostavniji za korištenje popis studenata se sprema na mobilni uređaj koji je zadužen za povezanost na internet i daljnje spremanje popisa, a sami čitač kartica koristi baterije te s mobilnim uređajem komunicira bežično kako bi postavljanje sustava prije predavanja bilo što brže.

1.1. Zadatak završnog rada

U radu je potrebno dizajnirati sustav za očitavanje studentskih iskaznica te pohranjivanje zapisa o očitanim karticama. Sustav bi trebao biti jednostavan za korištenje, prijenosan te biti brz za uspostavljanje na novoj lokaciji.

2. PREGLED PODRUČJA

2.1. Uređaji slične funkcionalnosti dostupni na tržištu

2.1.1. ACS ACR1255U-J1

Od istog proizvođača kao i čitač korišten u radu, ACR1255U-J1 je NFC čitač koji uz mogućnost spajanja putem USB veze ima i mogućnost spajanja putem Bluetooth 4.0 i LE tehnologije, prikazan na slici 2.1. Ima ugrađenu litij ionsku bateriju od koja 320 mAh mu pruža do 14 dana. Kompatibilan je sa svim popularnim standardima: ISO 14443 Part 4 Tip A i B, MIFARE, FeliCa, i sva 4 tipa NFC (ISO/IEC 18092) oznaka[28].

Sl. 2.1. ACS ACR1255U-J1

2.1.2. Anviz W1C Pro

Anviz W1C Pro terminal, prikazan na slici 2.2., ima mogućnost identificiranja korisnika pomoću beskontaktne kartice MIFARE standarda ili pomoću pina unesenog na terminalu. Terminal također može raditi na bateriju i bežično se povezati na mrežu, no problem je što terminal nije u potpunosti nezavisan, tj. zahtjeva ili dodatni server na mreži koji će pratiti prisustvo ili pretplatu na Go2Clock cloud servis[29].

Sl. 2.2. Anviz W1C Pro

2.2. Pregled korištenih tehnologija i metodologija

Svaki student pri početku studija dobiva studentsku iskaznicu. Studentske kartice imaju mogućnost identifikacije putem čipa, magnetske trake i beskontaktnog očitavanja. Za ovaj rad odlučeno je da se koristiti tehnologija beskontaktne identifikacije. Za beskontaktno čitanje UID broja koje je potreban za unikatnu identifikaciju studenta, odabran je čitač ACR122U konzistentnosti i brzine pri čitanju kartica. ACR122U čitač koristi USB konekciju za komunikaciju, što znači da za njegovo korištenje treba računalo, nije dovoljan samo mikro kontroler. Za tu ulogu odabrano je Raspberry Pi Zero W računalo radi njegove male veličine i potrošnje energije. No, s obzirom da su njegove performanse niske, javno dostupni operacijski sustavi iako mogu obavljati sve potrebno za rad sustava, cijeli sustav čine sporim i nekonzistentnim. Radi toga je odlučeno napraviti minimalnu potrebnu distribuciju pomoću Yocto projekta kako bi sustav imao prihvatljive performanse. Čitač i računalo se napajaju preko baterije, tj. prijenosnog punjača koji je najpraktičnije rješenje za ovaj problem. Kako računalo nema ekran niti bilo kakav drugi način interakcije s korisnikom, iako Raspberry Pi ima mogućnost povezivanja s internetom, odlučeno je napraviti aplikaciju za Android mobilni uređaj koja će pratiti sve očitane kartice. Za komunikaciju između računala i mobilnog uređaja je odlučeno koristiti Bluetooth tehnologiju bežične komunikacije. Dalje u poglavlju su sve korištene tehnologije pobliže objašnjene.

2.2.1. RFID

Radiofrekvencijska identifikacija (engl. *Radio-Frequency Identification,* RFID) je, kao što samo ime kaže, tehnologija radio frekvencijske identifikacije, tj. standard kojim se prenose podatci između čitača i transpondera. Primjene RIFD tehnologije su broje, no neke od češćih primjena su u skladištima za praćenje inventara, životinja, vozila, proizvodnje, dostava, beskontaktno plaćanje i, naravno, identifikaciju[2]. Sustavi koji koriste RFID tehnologiju mogu se razlikovati po tipu transpodnera koji koriste, po tipu čitača, po frekvenciji koju koriste te po udaljenosti na kojima obavljaju svoju svrhu. Domet na kojemu je moguća komunikacija je proporcionalan s frekvencijom signala, tj. signali više frekvencije imaju duži domet od signala manje frekvencije. Uobičajene frekvencije za RFID komunikaciju su:[1]

1. Niska frekvencija(LF): 30 – 300kHz
2. Visoka frekvencija ili radio frekvencija(HR/RF): 3-30MHz
3. Ultra visoka frekvencija(UHF): 300 MHz–3 GHz
4. Mikrovalovi: >3 GHz.

2.2.2. RFID čitač

Osnovna uloga RFID čitača je očitavanje i komunikacija s RFID transponderom. Sastoji se od antene koja pomoću elektromagnetskih valova i upravljačkog uređaja koji interpretira signale te šalje podatke dalje računalu na koje je čitač spojen. Jednostavni čitači imaju mogućnost komunikacije putem samo jednog protokola i samo na jednoj frekvenciji, dok složeniji imaju više mogućnosti[1]. Na slici 2.3. je prikazan primjer RFID čitača.

Sl. 2.3. Primjer RFID čitača

2.2.3. RFID transponder

Korijen naziva RFID transpondera se nalazi u terminu "transmitter / responder", koji točno opisuje njegovu funkciju, jer on odgovara na odaslani signal. Sastoji se od mikročipa koji sadrži određenu količinu memorije i mogućnost procesiranja signala, i od antene prilagođene frekvenciji za koju je namijenjen. Postoje dvije glavne kategorije RFID sustava: aktivni i pasivni. U aktivnim sustavima transponder dobiva energiju potrebnu za rad iz vanjskih izvora, kao što su npr. električna mreža, baterije ili neki drugi lokalni izvor energije. Aktivni transponderi se koriste kada je potreban veliki domet (i do 100m u ekstremnim slučajevima) ili kada je zbog okoline potrebna veća snaga elektromagnetskih valova, kao npr. u okolinama s puno vode ili metala. U pasivnim sustavima energija potrebna za rad transpondera dobiva se od same energije signala koji dolazi do transpondera. Maksimalni domet pasivnih transpondera je do svega nekoliko metara. Postoje tri različita načina rada pasivnih transpondera[1]:

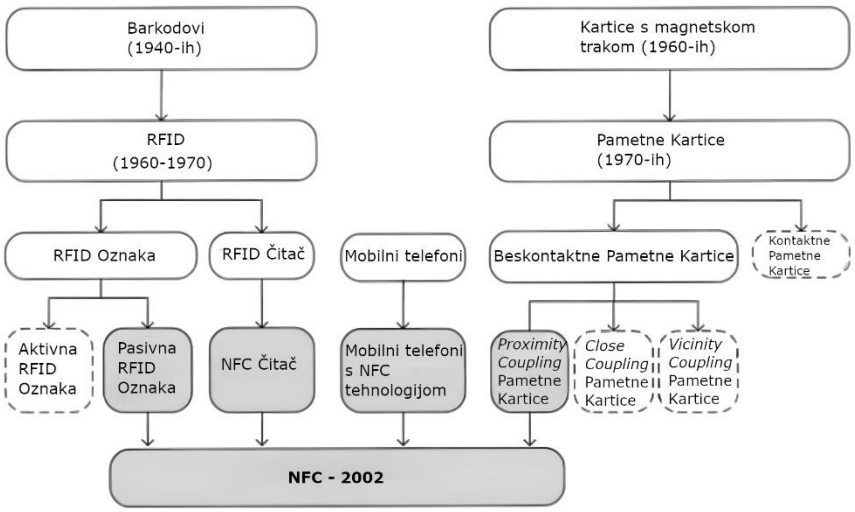
1. Elektromagnetska indukcija: i antena čitača i antena transpondera su napravljene u obliku zavojnice, te tako kada dođu u blizinu jedna drugoj čine transformator. Kroz antenu čitača prolazi struja koja stvara magnetsko polje radi kojeg se inducira struja u zavojnici transpondera. Moduliranjem faze, amplitude i frekvencije vala nosioca čitač komunicira s

trasponderom, a transponder odgovara variranjem opterećenja, a time i napona na svojoj zavojnici.

2. Reflektiranjem elektromagnetskog vala. Reflektivna karakteristika antene može se mijenjati mijenjanjem otpornika spojenog u paralelu, čime se postiže modulacija signala. Ova metoda se koristi uglavnom pri većim udaljenostima[2].
3. Elektrostatičkim uparivanjem. U ovoj metodi čitač i transponder se ponašaju kao dvije nabijene ploče. Mijenjanjem napona na jednoj utječemo na napon na drugoj. Ova metoda je efektivna samo za kratke udaljenosti i uglavnom se koristi za pametne kartice[8].

2.2.4. NFC tehnologija

Razvijena od strane Philipsa i Sonya 2002 godine, a 2003. godine prihvaćena kao ISO/IEC 18092 standard, NFC (engl. *Near Field Communication*) je podskup RIFD tehnologije namijenjen kratkometnoj komunikaciji, od svega nekoliko centimetara udaljenosti. Fizička implementacija NFC standarda je isključivo na 13,56 MHz[13]. Na slici 2.4. je prikazana povijest nastanka NFC standarda.



Sl. 2.4. Povijest nastanka NFC standarda[13]

Postoje 3 definirane vrste NFC komunikacije:[13]

1. Čitanje/Pisanje – uobičajen način rada NFC uređaja, u kojoj jedan NFC uređaj zahtjeva podatke od drugog uređaja, ili pokreće određenju radnju na njemu.
2. *Peer-to-peer* – standard dvosmjerne komunikacije između dva uređaja za razmjenu podataka.

3. Emulacija kartice – NFC uređaj se ponaša kao beskontaktna kartica. U ovom načinu rada jedan uređaj se može ponašati kao više različitih kartica u različito vrijeme. Ovaj način rada danas koriste mobilni uređaji za beskontaktno plaćanje. Emulacija kartice nije bila predviđena originalnim standardom.

2.2.5. ACR122U USB NFC Čitač

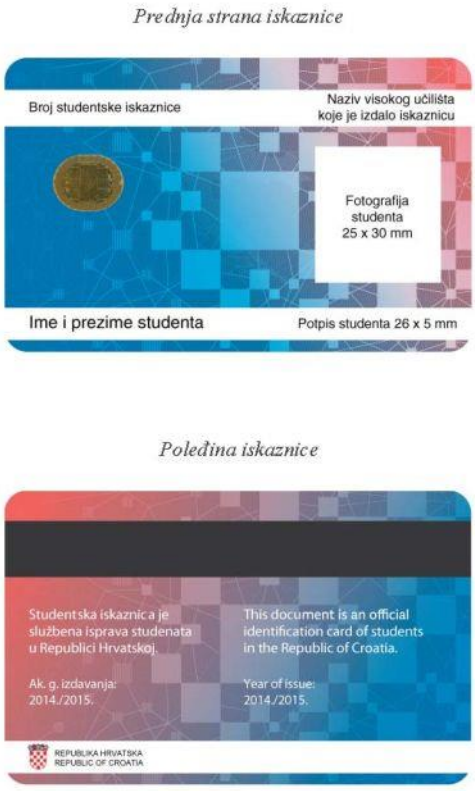
ACR122U je čitač kartica proizvođača Advanced Card Sytems iz Hong Konga, prikazan na slici 2.5. Podržava kartice frekvencije 13,65 MHz, s kojima može komunicirati maksimalnom brzinom do 424 kbit/s[4]. Na računalo se pomoću USB 1.1 priključka preko kojeg može postići komunikaciju do 12 Mbit/s[3]. Čitač podržava PC/SC standard, za koji driveri dolaze standardno s Windows operacijskim sustavom, a za distribucije Linuxa mogu se preuzeti kao dio *pcsc-tools* paketa. Proizvođač preporuča čitač za korištenje u svrhe državnih e-usluga, bankarstva, prijevoza, programa lojalnosti i za ograničavanje pristupa. Prema specifikaciji čitač je kompatibilan sa sljedećim standardima pametnih kartica:

1. ISO 14443 Part 4, Tip A i B
2. MIFARE Classic
3. FeliCa
4. ISO/IEC 18092, sva 4 tipa oznaka.

Sl. 2.5. ACR122U čitač korišten za rad

2.2.6. Studentska iskaznica

Svakom studentu u Republici Hrvatskoj je po početku studija dodijeljena studentska iskaznica od strane sveučilišta kojem student pripada. Na svakoj studentskoj iskaznici se nalazi ime i prezime studenta, njegova slika i naziv visokog učilišta koje je izdalo studentsku iskaznicu. Svaka kartica je jedinstvena za svakog studenta i nije prenosiva[9]. Identifikacijski broj studenta se nalazi na više mjesta na studentskoj iskaznici: na magnetskoj traci koja se nalazi na poleđini kartice, na čipu koji je vidljiv s prednje strane kartice te u transponderu koji omogućava bežično očitavanje kartice. Za ovaj rad odabrano je da će se podatci očitavati bežičnim putem. Sve studentske iskaznice su tipa MIFARE Classic 4k, sa UID brojem od 8 znakova, tj. 4 bajta. Od akademske godine 2023./2024. u upotrebu ulaze nove studentske iskaznice tipa MIFARE Desfire EV3[7], no za vremena pisanja ovog rada one nisu još u uporabi, ali prema specifikaciji čitača kartice bi trebale raditi. Na slici 2.6. je prikazan primjer studentske iskaznice.



Sl. 2.6. Primjer studentske iskaznice[9]

2.2.7. MIFARE Classic

MIFARE (engl. *Mikron FARE Collection System*) je tip beskontaktnih kartica originalnog podrijetla od Nizozemske tvrtke NXP. Proizvedene su prema standardu ISO/IEC 14443 Type-A.

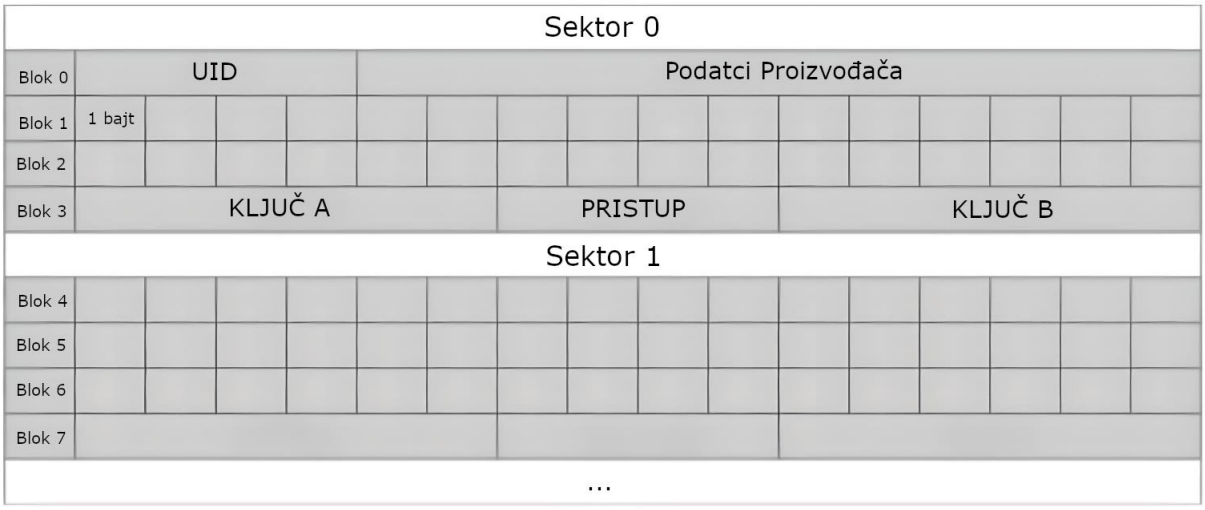
Prema proizvođačevoj specifikaciji očitanje ovih kartica je moguće sa udaljenosti do 10cm, sa brzinom prijenosa do 106 kbit/s. Komunikacija s karticom se vrši signalom frekvencije 13,56 MHz[10]. Zahvaljujući inteligentnom sustavu sprječavanja kolizija tijekom prijenosa podataka moguće je komunicirati s karticom i kada je više kartica prisutno u blizini. Kartice u sebi sadržavaju EEPROM memoriju veličine od 1 do 4 kB. U slučaju studentske kartice memorija je veličine 4 kB. Kartice su prema proizvođaču dizajnirane za vijek trajanja od 10 godina[10]. Na slikama 2.7. i 2.8. prikazani su primjeri implementacije MIFARE Classic standarda.

Sl. 2.7. Primjer MIFARE Classic kartica

Sl. 2.8. Primjer MIFARE Classic oznaka

Jedan blok memorije je veličine 16 bajta, a zatim se blokovi grupiraju u sektore. Prva 32 sektora se sastoje od 4 bloka, a ostali se sastoje od 16 blokova. UID, tj. jedinstveni identifikacijski broj kartice se nalazi u prvom bloku prvog sektora. On se može samo čitati i ne može se mijenjati. Razlika od standardu ISO/IEC 14443 Type-A prema kojemu su napravljene MIFARE kartice je u

tome što je prijenos podataka enkriptiran. Crypto-1 algoritam enkripcije je razvio NXP, algoritam koristi simetrične ključeve veličine 48 bita, ključevi A i B. Na slici 2.9. je prikazana shema memorije. Ključ A se ne može pročitati iz memorije, a čitanje ključa B, ovisno o implementaciji, može biti omogućeno ili onemogućeno. No enkripcija Crypto 1 enkripcija je probijena još 2008. godine te se danas MIFARE Classic kartice ne preporučuju za bilo kakvu uporabu gdje je sigurnost bitna[11]. Od tada je NXP izašao s novijim standardima u kojima je sigurnost poboljšana, među kojima je i MIFARE Desfire EV3 koji će koristiti novije studentske iskaznice.



Sl. 2.9. Shema memorije[12]

2.2.8. RASPBERRY PI

Raspberry Pi je jeftino minijaturno računalo, prikazano na slici 2.10., koje ima sve mogućnosti kao bilo koje drugo stolno ili prijenosno računalo. Prvi primjerak Raspberry Pi računala u prodaju dolazi 2012 godine. Razvila ga je Raspberry Pi Foundation, dobrotvorna organizacija koja razvija nove modele još i danas. Svrha prvog modela je bila promocija računalnih znanosti te promoviranje učenja programiranja. Od tog dana prodano je više od 40 milijuna primjeraka, a tržište Raspberry Pi računala i dodataka za njih danas vrijedi više od 1milijarde dolara[14]. Razvoj prvog modela Raspberry Pi računala je započeo 2006. godine na sveučilištu Cambridge u Engleskoj jer su profesori primijetili smanjen broj studenata zainteresiranih za računarstvo. Kako bi zainteresirali mlade odlučili su napraviti jeftino računalo koje će im omogućiti da isprobaju svoje ideje. Rezultat njihovog rada je prvi model Raspberry Pi Model B, koji je unatoč svojim

ograničenim specifikacijama započeo revoluciju hobi računarstva[14]. Specifikacije Modela B su:[15]

- SOC: Broadcom BCM2835
- CPU: 700 MHz ARM1176JZFS
- RAM: 512MB SDRAM
- Portovi: 2x USB, HDMI, 3,5mm audio
- GPIO: 26 pinova.

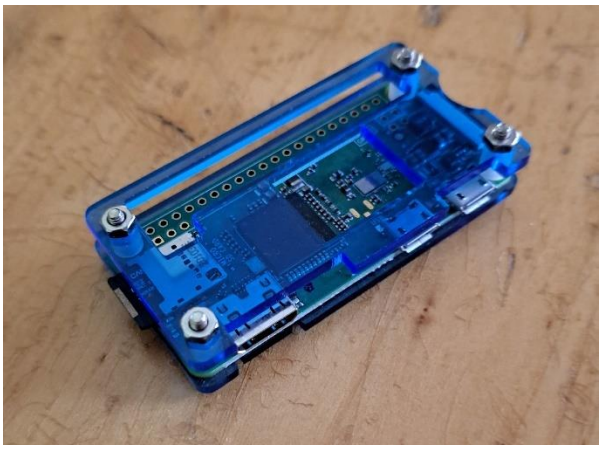
Sl. 2.10. Raspberry Pi Model B

Danas je u prodaji dostupan Raspberry Pi 4, prikazan na slici 2.11., koji može imati do 8GB RAM-a, procesor sa četiri Cortex A72 jezgre koje rade taktom od 1.5 GHz te kao takav jači od prvog modela.

Sl. 2.11. Raspberry Pi 4

2.2.9. Raspberry Pi Zero W

Za izradu ovog rada korišten je Raspberry Pi Zero W. Pušten je u prodaju 2017. godine, a specifikacijama je identičan originalnom modelu. 5 godina dodatnog razvoja je rezultiralo računalom veličine ESP32 mikro kontrolera. Također, dizajniran je da se uklapa u USB 2.0 specifikaciju, što znači da nikada neće trošiti više od 500mA struje. Radi toga je idealan za korištenje na bateriju te se može pokrenuti s bilo koje prijenosnog punjača za mobitel. Na slici 2.12. je prikazan uređaj korišten u ovom radu.



Sl. 2.12. Raspberry Pi Zero W korišten za rad

2.2.10. Linux

GNU/Linux je kolokvijalno ime za operacijske sustave temeljene na Linux kernelu. Linus Torvalds je započeo rad na Linux kernelu kao student 1991. godine, a njegov projekt je danas prerastao u sustav koji pokreće velik dio računalnog svijeta danas. Za ugradbene sustave operacijski sustavi temeljeni na Linux kernelu su danas najpopularniji izbor, za što ima više razloga:[16]

1. Za Linux je već razvijen dobar *scheduler, networking stack*, podrška za USB, Wi-Fi, Bluetooth, mnoge medije za pohranu i slično.
2. Linux je kompatibilan sa svim najpopularnijim arhitekturama računala danas, kao što su: x86, Arm, MIPS, PowerPC, RISC-V itd.
3. Linux je *open source* te imamo mogućnost modificirati bilo koji dio želimo. Možemo dodati protokole i mogućnosti koje nam trebaju, ali također maknuti one koje nam ne trebaju.

4. Zajednica vezana u Linux je iznimno aktivna. Svakih 8 do 10 tjedana izlazi nova verzija kernela, a na njemu radi više od 1000 developera. Radi toga velika je šansa da će sav novi hardware biti podržan te da će biti podržani svi novi protokoli i standardi.

Radi svega toga Linux je idealan izbor za kompleksne sustave, no to ne znači da nema razloga zašto ne biste odabrali neki drugi način za razvoj svojeg projekta:[16]

1. Velika kompleksnost je također i mana, jer je teško pratit sve promjene istovremeno dok do njih dolazi te nitko ne može biti stručan za svaki dio Linux sustava.
2. U slučaju slabog ili ograničenog hardware-a, *real-time* operacijski sustav možda može biti bolji izbor, ako npr. VxWorks ili QNX.

Linux distribucije danas omogućuju jednostavno instaliranje Linux kernela u kombinaciji sa svim ostalim slobodnim software-om potrebnim za korištenje računala danas te kao takve iako često vrlo slične jedna drugoj, predstavljaju zasebne operacijske sustave.

Jedne od najpopularnijih distribucija za osobna računala su:[17]

1. MX Linux
2. Mint
3. EndeavourOS
4. Debian
5. Manjaro
6. Ubuntu
7. Pop! OS
8. Fedora.

No, za potrebe ovog rada sve gotove distribucije su daleko pre kompleksne, a radi toga i spore, npr. Raspberry Pi Zero koji je korišten za upravljanje čitačem kartica treba više od dvije minute da pokrene Raspbian, službenu distribuciju za Raspberry sustave, koja je već napravljena s ciljem da radi na sporom hardware-u. Taj problem je najlakše riješiti tako što ćemo napravit jednostavnu distribuciju pomoću Yocto projekta[18].

2.2.11. Systemd

Systemd je menadžer servisa za Linux operacijske sustave, u toj ulozi je naslijedio init.d sustav. Prilikom pokretanja sustava pokreće se s PID 1 te je zaslužan za inicijalizaciju sustava i pokretanje svih potrebnih servisa. Osigurava da su svi servisi pokrenuti pravim redoslijedom kako je definirano u .service datotekama te nadgledava rad servisa i pokušava ih oporaviti ili ponovno pokrenuti u slučaju greške[20].

2.2.12. Libnfc

Libnfc je *library* koji omogućava jednostavno korištenje NFC uređaja iz *userspace* sloja. Kompatibilan je sa svim popularnim NFC čitačima poput ACR122, SCL3711, SCL3711m, FEITIAN bR500 i R502[6].

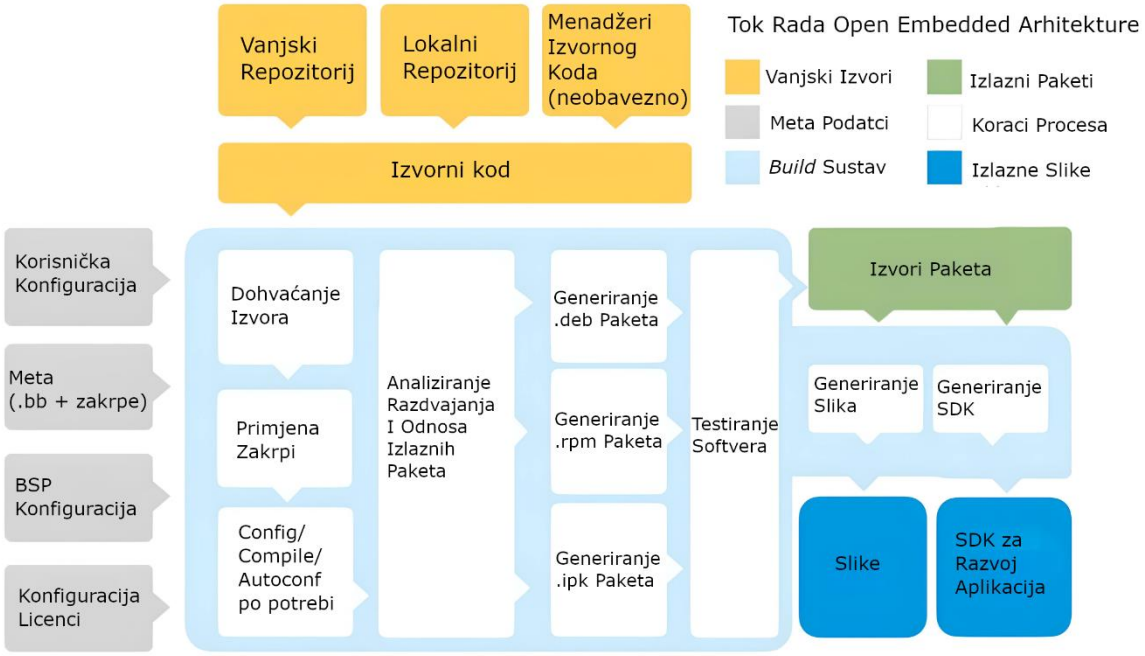
Također dolazi s nfc-list alatom, prikazanim na slici 2.13., koji omogućava jednostavno dohvaćanje UID broja kartice:

```
stjepan@stjepan-HP:~/poky/build$ nfc-list
nfc-list uses libnfc 1.8.0
NFC device: ACS / ACR122U PICC Interface opened
1 ISO14443A passive target(s) found:
ISO/IEC 14443A (106 kbps) target:
    ATQA (SENS_RES): 00  02
       UID (NFCID1): af  f6  4a  0d
      SAK (SEL_RES): 38
                ATS: 78  f7  b1  02  4a  43  4f  50  76  32  34  31
```

Sl. 2.13. nfc-list

2.2.13. Yocto Project

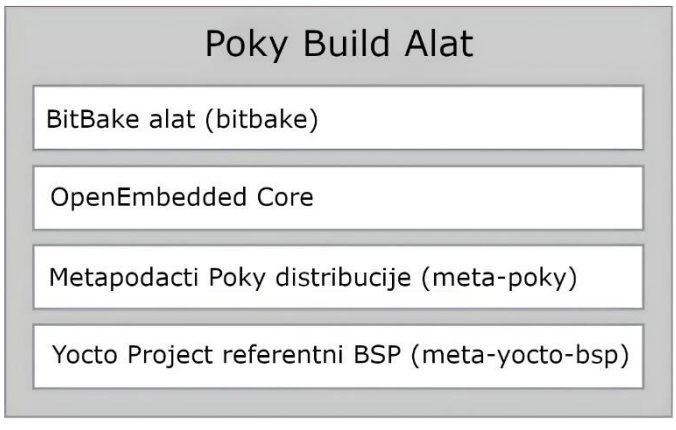
Yocto Project je *open-source* kolaborativni projekt koji omogućava developerima da kreiraju sisteme zasnovane na Linux kernelu za ugradbene projekte bez obzira na hardversku arhitekturu proizvoda[5]. Može generirat posebne građene distribucije zasnovane na glibc i musl C standardu, a zahvaljujući Zephyr projektu podržava i *tool-chain* za *real-time* operacijske sustave namijenjene *bare-metal* developmentu[19]. Konačni proizvod su dijelovi potpunog operacijskog sustava: Linux *kernel*, *bootloader*, i *root* datotečni sustav(*rootfs*), i to sve organizirano tako da radi zajedno. Glavni cilj projekta je smanjivanje ovisnosti o *host* Linux distribuciji tako što on automatski priprema okruženje potrebno za uspješnu kompilaciju distribucije. Velika prednost je ponovno iskorištavanje već odrađenog posla tako što od između dva *build*-a kompajlira samo ono što se promijenilo i preuzimaju samo oni repozitoriji koji su se promijenili, sve ostalo se ponovo iskorištava. Time se postiže to da iako prvi *build,* u ovisnosti o snazi *host* računala, može trajati i više od 2 sata, ako su razlike male u odnosu na prethodni build, svaki sljedeći može trajati i manje od minute. Način rada je prikazan na slici 2.14.



Sl. 2.14. Tok rada Open Embedded arhitekture[5]

2.2.14. Poky

Poky je referentna distribucija Yocto projekta koja koristi *build* sistem napravljen od strane OpenEmbedded zajednice. Nije ovisna o platformi i vrši unakrsno kompajliranje pomoću BitBake alata. Također pruža mehanizam za jednostavnu integraciju i kombinaciju više tisuća *open-source* projekata u jednu koherentnu Linux distribuciju[19]. Glavne komponente su prikazane na slici 2.15.



Sl. 2.15. Glavne komponente Poky distribucije[19]

Poky u osnovnoj konfiguraciji podržava samo sljedeće arhitekture:[18]

- ARM (qemuarm, qemuarm64)
- x86 (qemux86)
- x86-64 (qemux86-64)
- PowerPC (qemuppc)
- MIPS (qemumips, qemumips64)
- Texas Instruments BeagleBone (beaglebone)
- Freescale MPC8315E-RDB (mpc8315e-rdb)
- Intel x86(genericx86 i genericx86-64)
- Ubiquiti Networks EdgeRouter Lite (edgerouter).

Za sve dodatne arhitekture potrebno je koristiti dodatne slojeve.

2.2.15. BitBake

BitBake je *task scheduler* i sustav za izvršavanje koji analizira Python i Shell skripte. Ovisno o analiziranim skriptama generira zadatke koji su poredani ovisno o ovisnostima u kodu. Također prati sve zadatke koji se izvode kako bi osigurao predvidljivo izvođenje programa te maksimizirao iskorištenje resursa[19].

2.2.16. Slojevi i recepti

Slojevi u Yocto projektu služe za organiziranje strukture projekta. U osnovi, slojevi su kolekcije različitih recepata, konfiguracijskih datoteka i bilo kojih drugih datoteka potrebnih za izgradnju distribucije. Slojevi se slažu hijerarhijski tako da jedan sloj može nadograđivati ili izmjenjivati drugi sloj.

Recepti su upute koje govore sustavu kako da dohvati kod iz repozitorija, kako da ga konfigurira , kompajlira i u konačnici zapakira. Recepti imaju nastavak .bb(BitBake) ili .bbappend ako se nadovezuju na drugi recept.

2.2.17. Bluetooth

Bluetooth je tehnologija za bežično povezivanje uređaja i dijeljenje podataka među njima. Standardizirana je kao IEEE 802.15.1 protokol te predstavlja PAN mrežu(engl. *Personal Area Network*). Trenutno Bluetooth tehnologijom upravlja organizacija Bluetooth SIG (engl. *Bluetooth Special Interest Group*) osnovana 1998. godine i trenutno broji više od 20,000 članova. Grupu su osnovale tvrtke Toshiba, Intel, Ericsson i Nokia[22].

Postoje tri definirane Bluetooth topologije:[23]

1. jedan-na-jedan
2. jedan-na-više
3. više-na-više.

Trenutno najnoviji standard je Bluetooth 5.4 koji omogućava brzinu prijenosa do 2 Mb/s[23].

Za komunikaciju između Raspberry Pi Zero W i Android smartphonea na kojem se bilježi prisustvo studenta se koristi Bluetooth 4.1, jer je to najnoviji standard koji Raspberry podržava.

2.2.18. Android

Android je operacijski sustav za mobilne uređaje kojega razvija kompanija istog imena Andoid Inc. koju nakon početnog uspjeha kupuje tadašnji Google, tj. danas Alphabet. Google osniva OHA (engl. *Open Handset Alliance*) 2007. godine s ciljem daljnjeg razvoja operacijskog sustava i novih standarda. Android je danas instaliran na više od 70% svih mobilnih uređaja, a uz Applov iOs kontroliraju zajedno skoro cijelo tržište.

2.2.19. Kotlin

Moderan programski jezik dizajniran da zamjeni Javu za svrhu razvoja Android aplikacija, s time da je u isto vrijeme uputnosti kompatibilan sa starim Java kodom. Dizajnira ga JetBrains te na tržište izlazi 2011, a odobrenje za korištenje u Android aplikacijama dobiva od Googla 2017. godine.

Kotlin je objektno orijentiran programski jezik sa statičkim tipovima podataka namijenjen za višeplatformsko programiranje[26].

3. REALIZACIJA UREĐAJA

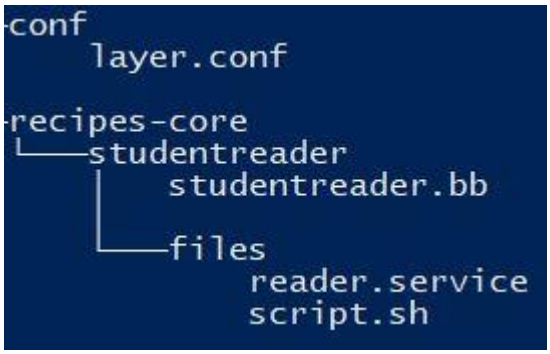
3.1. Hardverska izvedba

Sami hardverski sklop je relativno jednostavan. Sastoji se od Raspberry Pi Zero W ugradbenog računalnog sustava na kojeg je povezan ACR122U NFC čitač kartica. Za napajanje računala i čitača kartica koristi se prijenosni punjač Xiaomi Power Bank, no sama baterija nije bitna te bi radi iznimno niske potrošnje struje Raspberry Pi Zero W računala, od manje od 500mA, bilo koji prijenosni punjač trebao raditi. Sami čitač i računalo su pričvršćeni za prijenosni punjač tako da čine jednu cjelinu. Cijela izvedba je prikazana na slici 3.1.

Sl. 3.1. Čitač spojen na Raspberry Pi Zero W

3.2. Programska podrška Raspberry Pi

Osnova programske podrške implementirana je kao jedan Yocto sloj. Svaki sloj se sastoji od najmanje dva direktorija: conf te jedan ili više direktorija s receptima. U mojem slučaju to je samo jedan direktorij s receptima koji se zove recepies-core. Struktura sloja je prikazana na slici 3.2.



Sl. 3.2. Struktura sloja

Unutar conf direktorija se treba nalaziti layer.conf datoteka unutar koje su definirane sve datoteke koje se nalaze unutar sloja kao i same ovisnosti sloja o drugim slojevima i sam naziv sloja. Sadržaj layer.conf datoteke je prikazan na slici 3.3.

```
BBPATH .= ":${LAYERDIR}"

BBFILES += "${LAYERDIR}/recipes-*/*/*.bb \
            ${LAYERDIR}/recipes-*/*/*.bbappend"

BBFILE_COLLECTIONS += "meta-reader"
BBFILE_PATTERN_meta-reader = "^${LAYERDIR}/"
BBFILE_PRIORITY_meta-reader = "6"

LAYERDEPENDS_meta-reader = "core"
LAYERSERIES_COMPAT_meta-reader = "kirkstone"
```

Sl. 3.3. Sadržaj layer.conf datoteke

„Kirksotne” predstavlja granu Yocto projekta sa kojom je sloj u toj verziji kompatibilan. Ta grana je odabrana zato što je u vrijeme pisanja najnovija LTS grana dostupna.

Unutar direktorija s receptima se nalazi moj recept “studentreader”. Svaki recept mora sadržavati .bb datoteku unutar koje je opisan. A imamo datoteke potrebne za rad recepta, a da ih ne dohvaćamo iz nekog repozitorija, onda se te datoteke trebaju nalazit unutar files direktorija. Sadržaj studentreader.bb datoteke je prikazan na slici 3.4.

```
SUMMARY = "Student Reader application"
LICENSE = "MIT"
LIC_FILES_CHKSUM = "file://${COMMON_LICENSE_DIR}/MIT;md5=0835ade698e0bcf8506ecda2f7b4f302"

SRC_URI = "file://script.sh"
SRC_URI += " file://reader.service"

inherit systemd

SYSTEMD_SERVICE_${PN} = "reader.service"


S = "${WORKDIR}"

do_install() {
        install -d ${D}/etc/reader
        install -m 0777 script.sh ${D}/etc/reader/
        install -d ${D}/${systemd_system_unitdir}/
        install -m 0644 ${WORKDIR}/reader.service ${D}/${systemd_system_unitdir}/

}

FILES:${PN} += " /etc/reader/script.sh"
FILES:${PN} += " ${systemd_system_unitdir}/reader.service"

REQUIRED_DISTRO_FEATURES= "systemd"
```

Sl. 3.4. Sadržaj studentreader.bb datoteke

S obzirom da je funkcionalnost čitača implementirana kao Systemd servis, unutar recepta je potrebno navesti “inherit systemd”, kao i da je Systemd neophodan za funkciju.

Unutar do_install bloka se navode sve operacije koje se trebaju izvesti da bi recept bio uspješan. U ovom slučaju to znači da treba kreirati direktoriji /etc/reader, unutar njega dodati skiptu script.sh te reader.service datoteku dodati kao Sytemd servis.

U FILES varijablu treba dodati sve datoteke koje trebaju nastati kako bi BitBake znao da su nakon izvršavanja recepta nastale sve potrebne datoteke te da nisu nastale datoteke viška.

Nakon što je konfiguriran sloj, potrebno je još napraviti izmjene u bblayers.conf i local.conf datotekama unutar poky/build/conf direktorija. Sadržaji tih datoteka prikazani su na slikama 3.5. i 3.6.

```
# POKY_BBLAYERS_CONF_VERSION is increased each time build/conf/bblayers.conf
# changes incompatibly
POKY_BBLAYERS_CONF_VERSION = "2"

BBPATH = "${TOPDIR}"
BBFILES ?= ""

BBLAYERS ?= " \
  /home/stjepan/poky/meta \
  /home/stjepan/poky/meta-poky \
  /home/stjepan/poky/meta-yocto-bsp \
  /home/stjepan/poky/meta-raspberrypi \
  /home/stjepan/poky/meta-openembedded/meta-oe \
  /home/stjepan/poky/meta-reader \
  "
```

Sl. 3.5. Sadržaj bblayes.conf datoteke

Unutar bblayers.conf datoteke je potrebno samo dodati putanje svih slojeva koje koristimo. Sloj meta-raspberrypi je potreban ako želimo da slika bude kompatibilna sa Raspberry Pi računalom, a sloj meta-openembedded/meta-oe služi za libnfc, jer u njemu ima već definiran recept.

Unutar local.conf potrebno je izmijeniti MACHINE varijablu u MACHINE ??= "raspberrypi0-wifi" te na kraj datoteke dodati:

```
DISTRO_FEATURES:append = " systemd"
VIRTUAL-RUNTIME_init_manager = "systemd"
IMAGE_INSTALL:append = " vim tmux libnfc studentreader"
IMAGE_FEATURES:remove = "splash"
```

Sl. 3.6. Dodano u local.conf datoteku

Uklanjanje featurea „splash“ je iz razloga što uzrokuje grešake u logu, a s obzirom da je služi samo za iscrtavanje Raspberry loga tijekom dizanja sustava, možemo ga ukloniti.

U “IMAGE_INSTALL” varijablu se dodaju svi recepti koje želimo dodati, vim i tmux nisu neophodni, ali olakšavaju korištenje uređaja u slučaju bilo kakvih kasnijih promjena.

Sada se može pokrenuti komanda “source oe-init-build-env” kako bi se okruženje pripremilo za izgradnju slike, što je prikazano na slici 3,7. Nakon toga je potrebno pokernuti “bitbake core-image-base” za izgradnju slike, što je prikazano na slici 3.8. Nakon što sve završi slika se na SD karticu prebacuje pomoću bmaptool alata, što je prikazano na slici 3.9.

```
stjepan@stjepan-HP:~/poky$ source oe-init-build-env

### Shell environment set up for builds. ###

You can now run 'bitbake <target>'

Common targets are:
    core-image-minimal
    core-image-full-cmdline
    core-image-sato
    core-image-weston
    meta-toolchain
    meta-ide-support

You can also run generated qemu images with a command like 'runqemu qemux86'

Other commonly useful commands are:
 - 'devtool' and 'recipetool' handle common recipe tasks
 - 'bitbake-layers' handles common layer tasks
 - 'oe-pkgdata-util' handles common target package tasks
```

Sl. 3.7. source oe-init-build-env

```
stjepan@stjepan-HP:~/poky/build$ bitbake core-image-base
Loading cache: 100% |################################################################################| Time: 0:00:00
Loaded 2825 entries from dependency cache.
NOTE: Resolving any missing task queue dependencies

Build Configuration:
BB_VERSION           = "2.0.0"
BUILD_SYS            = "x86_64-linux"
NATIVELSBSTRING      = "universal"
TARGET_SYS           = "arm-poky-linux-gnueabi"
MACHINE              = "raspberrypi0-wifi"
DISTRO               = "poky"
DISTRO_VERSION       = "4.0.12"
TUNE_FEATURES        = "arm thumb vfp arm1176jzfs callconvention-hard"
TARGET_FPU           = "hard"
meta
meta-poky
meta-yocto-bsp       = "kirkstone:e42cc7d900fd2f1b6a12184cb3e4c81d5bda5206"
meta-raspberrypi     = "kirkstone:80a12f7fddfeae28c43242765374e7ade3a2a59e"
meta-oe              = "kirkstone:529620141e773080a6a7be4615fb7993204af883"
meta-reader          = "kirkstone:e42cc7d900fd2f1b6a12184cb3e4c81d5bda5206"

Initialising tasks: 100% |###########################################################################| Time: 0:00:02
Sstate summary: Wanted 0 Local 0 Mirrors 0 Missed 0 Current 1955 (0% match, 100% complete)
NOTE: Executing Tasks
NOTE: Tasks Summary: Attempted 4919 tasks of which 4919 didn't need to be rerun and all succeeded.
```

Sl. 3.8. bitbake core-image-base

```
stjepan@stjepan-HP:~/poky/build$ sudo bmaptool copy tmp/deploy/images/raspberrypi0-wifi/core-image-base-raspbe
rrypi0-wifi.wic.bz2 /dev/sdb
bmaptool: info: discovered bmap file 'tmp/deploy/images/raspberrypi0-wifi/core-image-base-raspberrypi0-wifi.wi
c.bmap'
bmaptool: info: block map format version 2.0
bmaptool: info: 99533 blocks of size 4096 (388.8 MiB), mapped 53756 blocks (210.0 MiB or 54.0%)
bmaptool: info: copying image 'core-image-base-raspberrypi0-wifi.wic.bz2' to block device '/dev/sdb' using bma
p file 'core-image-base-raspberrypi0-wifi.wic.bmap'
bmaptool: info: 100% copied
bmaptool: info: synchronizing '/dev/sdb'
bmaptool: info: copying time: 13.9s, copying speed 15.2 MiB/sec
```

Sl. 3.9. bmaptool

Unutar Yocto recepta dodana je datoteka reader.service, čiji je sadržaj prikazan na slici 3.10. Ta datoteka definira Systemd servis, a ključni argument je „ExecStart“ koji definira skriptu koja se pokreće dio servisa. Skripta je prikazana na slici 3.11. Skripta pokreće u pozadini Bluetooth kanal pomoću rfcomm watch naredbe, koja za razliku od rfcomm listen u slučaju da kanal bude prekinut sama ponovno kreće čekati idućeg klijenta[21]. Nakon toga skripta kontinuirano poziva nfc-list te kada kartica bude očitana UID kartice šalje na /dev/rfcomm0, koji predstavlja Bluetooth kanal.

```
[Unit]
Description=Student Reader Service
After=bluetooth.service

[Service]
Type=simple
Restart=always
RestartSec=1
StartLimitIntervalSec=0
ExecStart=/bin/sh /etc/reader/script.sh

[Install]
WantedBy=multi-user.target
```

Sl. 3.10. Sadržaj reader.service datoteke

```
#!/bin/sh

rfcomm watch 0 &

while [[ 1 == 1 ]]
do
sleep 0.2
OUTPUT=`nfc-list`
NUM=`echo "$OUTPUT" | wc -l`
if [ "$NUM" -gt 3 ]; then
ID=`echo "$OUTPUT" | sed -n 6p | cut -c 22-`
echo "$ID" > /dev/rfcomm0
sleep 2
fi
done
```

Sl. 3.11. Sadržaj script.sh datoteke

Pri prvom pokretanju uređaja treba osigurati da se reader servis automatski pokreće s sustavom, što se postiže komandama prikazanim na slici 3.12.

```
systemctl start reader
systemctl enable reader
```

Sl. 3.12. Potrebne komande

Nakon toga ispis status komande treba biti isti kao na slici 3.13.

```
root@raspberrypi0-wifi:~# systemctl status reader
● reader.service - Student Reader Service
     Loaded: loaded (/lib/systemd/system/reader.service; enabled; vendor preset: disabled)
     Active: active (running) since Thu 2022-04-28 18:37:19 UTC; 53s ago
   Main PID: 167 (sh)
      Tasks: 3 (limit: 527)
     CGroup: /system.slice/reader.service
             ├─ 167 /bin/sh /etc/reader/script.sh
             ├─ 170 rfcomm watch 0
             └─1191 sleep 0.2

Apr 28 18:37:19 raspberrypi0-wifi systemd[1]: Started Student Reader Service.
root@raspberrypi0-wifi:~# _
```

Sl. 3.13. systemctl status reader

Prije nego što se može uspostaviti veza između Android mobilnog uređaja i Raspberry Pi uređaja potrebno je konfigurirati Bluetooth. Prvo moramo promijeniti dvije linije u konfiguraciji Bluetooth

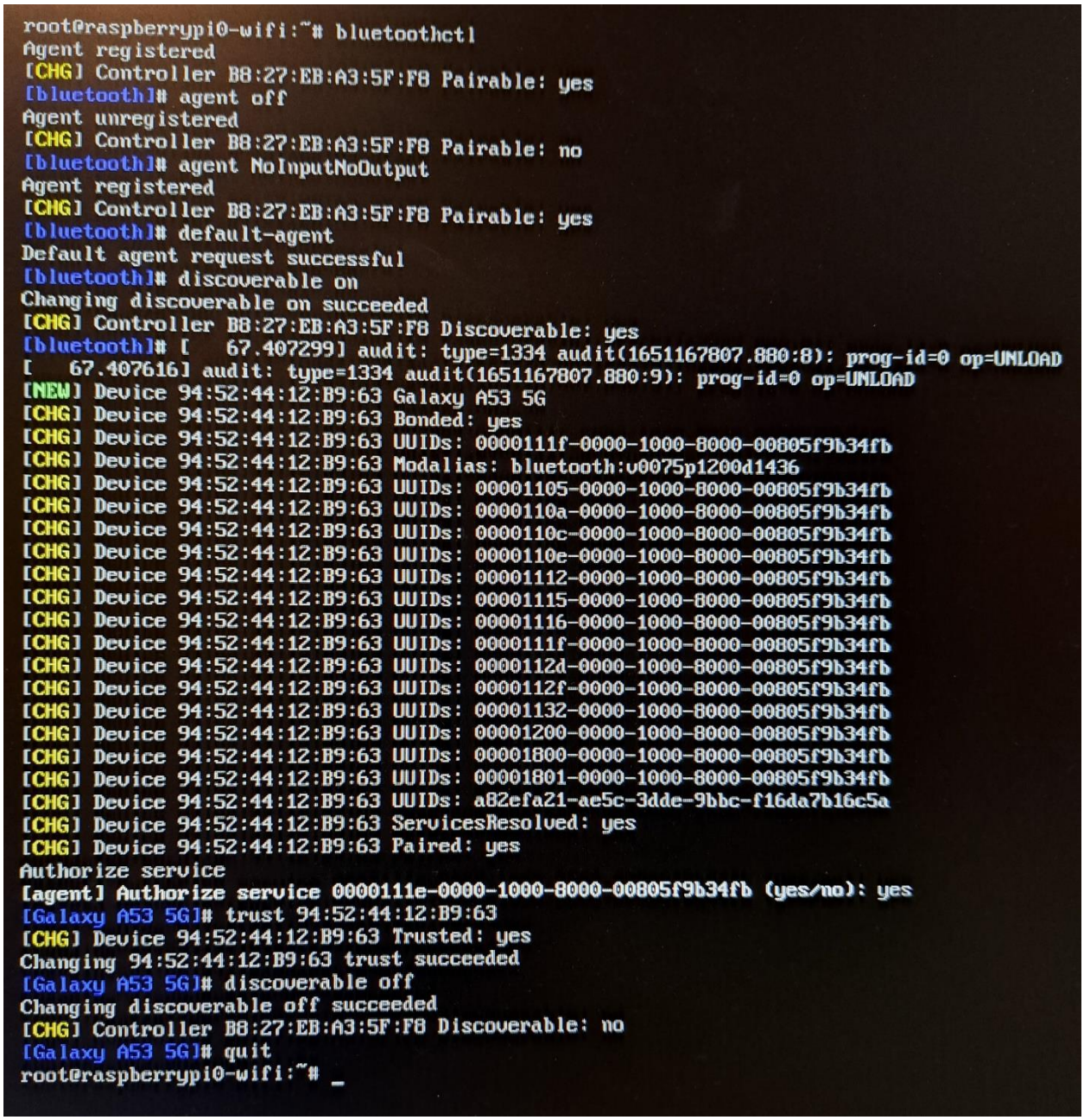
servisa, tj. u datoteci /ets/systemd/system/dbus-org.bluez.service. Potrebne promjene su prikazane na slici 3.14.

```
ExecStart=/usr/lib/bluetooth/bluetoothd --compat
ExecStartPost=/usr/bin/sdptool add SP
```

Sl. 3.14. Konfiguracija Bluetooth servisa

Bluetooth se mora pokrenuti u compatibility modu jer je od bluez verzije 5.44 rfcomm alat označen kao “deprecated”, a slične jednostavne zamjene još nema. Komanda “sdptool add SP” pokreće serijski port preko kojega se vrši komunikacija[24].

Nakon toga potrebno je postaviti Bluetooth agent koji se bavi uparivanjem uređaja. Odabran je agent “NoInputNoOutput” koji omogućava brzo uparivanje bez ikakve interakcije s uređajem, no zahtjeva da se prvo uparivanje obavi preko konzole. No ovo ima i prednosti jer onemogućava bilo kome osim vlasniku uređaja spajanje, a time su spriječena i ometanja u radu. Cijeli postupak je prikazan na slici 3.15.



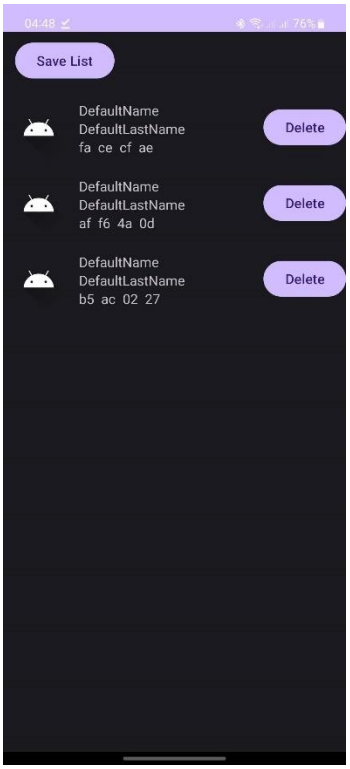
Sl. 3.15. Konfiguracija Bluetooth agenta i uparivanje uređaja

Sada kada je uparen uređaj je konačno spreman za uporabu.

3.3. Mobilna Aplikacija

Aplikacija je jednostavna lista svih studentskih iskaznica koje su skenirane, koje se kasnije mogu spremiti na mobilni uređaj u obliku tekstualne datoteke u koju je zapisana cijela lista studenata. Primjer aplikacije s 3 registrirana studenta je prikazan na slici 3.16., a primjer ta ista 3 studenta zapisana u datoteku je prikazan na slici 3.17. Za implementaciju serijske Bluetooth veze iskorištena je biblioteka android-bluetooth-serial koja pruža jednostavnu implementaciju serijske komunikacije putem funkcije koje se poziva pri primljenoj poruci. Ta funkcija zatim preko

StudentAdapter objekta dodaje novi element u recyclerView. U planu je bilo još putem REST API preuzeti podatke za studenta s ISSP servera preko UID broja očitanog s studentske iskaznice, no nažalost nisam dobio pravo pristupa bazi podataka. No, UID brojevi se spremaju u tekstualnu datoteku te je moguće kasnije povući podatke o studentu vezanog za taj UID. Na slikama 3.18. – 3.22. su prikazani ključni dijelovi koda potrebni za rad aplikacije.



Sl. 3.16. Aplikacija s tri očitane kartice

```
fa  ce  cf  ae  , DefaultName, DefaultLastName
af  f6  4a  0d  , DefaultName, DefaultLastName
b5  ac  02  27  , DefaultName, DefaultLastName
```

Sl. 3.17. Primjer tekstualnog dokumenta s očitanim karticama

```
btSave.setOnClickListener { it: View!
    val currentDateTime = LocalDateTime.now().format(dateTimeFormatter)
    val folderDir = Environment.getExternalStoragePublicDirectory(Environment.DIRECTORY_DOWNLOADS)
    val fileName = "Students_" + currentDateTime.toString() + ".txt"
    Toast.makeText( context: this, fileName, Toast.LENGTH_LONG)
        .show()
    val f = File(folderDir, fileName)
    for (student in students) {
        var tmpString = student.id + ", " + student.name + ", " + student.lastName + "\n"
        f.appendText(tmpString)
    }
```

Sl. 3.18. Pohranjivanje podataka u datoteku

```
private fun onMessageReceived(message : String) {
    val student = StudentData( name: "DefaultName", lastName: "DefaultLastName", message, url: "")
    studentAdapter.addItem(student)
    studentRecyclerView.scrollToPosition( position: students.size-1)

}
```

Sl. 3.19. Handler pristigle poruke

```
    bluetoothManager.openSerialDevice(mac).subscribeOn(Schedulers.io()).observeOn(AndroidSchedulers.mainThread()).subscribe(this::onConnected, this::onError)


}

private fun onConnected(connectedDevice: BluetoothSerialDevice){
    deviceInterface = connectedDevice.toSimpleDeviceInterface();
    deviceInterface.setListeners(this::onMessageReceived, this::onMessageSent, this::onError);
}
```

Sl. 3.20. Postavljanje Bluetooth veze

```
<uses-permission android:name="android.permission.BLUETOOTH" />
<uses-permission android:name="android.permission.BLUETOOTH_CONNECT" />
<uses-permission android:name="android.permission.BLUETOOTH_SCAN" />
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
```

Sl. 3.21. Potrebna dopuštenja za aplikaciju

```
dependencyResolutionManagement { this: DependencyResolutionManagement
    repositoriesMode.set(RepositoriesMode.FAIL_ON_PROJECT_REPOS)
    repositories { this: RepositoryHandler
        google()
        mavenCentral()
        maven { url  = uri( path: "https://jitpack.io") }
    }
}
```

Sl. 3.22. Dodavanje novog repozitorija potrebnog za android-bluetooth-serial

4. ZAKLJUČAK

Kako bi sustav za evidenciju studenata radio, potrebno je više tehnologija koje moraju raditi zajedno kao jedan jedinstveni sustav. Istražen je način rada studentskih iskaznica i metoda pristupa podatcima pohranjenim na istim iskaznicama. Čitači kartica omogućavaju bežični pristup podatcima pohranjenim na iskaznicama te su istraženi razni standardi pristupa karticama i njihove implementacije. Dobiven je uvid u mogućnosti računala Raspberry Pi Zero W te načini na koje se iz slabih računalnih sustava može izvući što više funkcionalnosti pomoću Yocto projekta kojim se može razviti Linux distribucija sa svime što je nužno potrebno i bez viškova. Istražene su mogućnosti bežične komunikacije putem Bluetooth tehnologije i mogućnost serijske komunikacije putem iste. Razvijeni sustav omogućava brzo i jednostavno bežično povezivanje čitača beskontaktnih kartica s mobilnim uređajem na siguran način. Nažalost dohvaćanje podataka o studentima nije uspješno izvedeno radi nemogućnosti pristupa bazi podataka studenata, za koju radi velike količine osobnih podataka velikog broja studenata se ne može lako dobiti pristup.

LITERATURA

[1] Klaus Finkenzeller, RFID Handbook: Fundamentals and Applications in Contactless Smart Cards and Identification, Second Edition, John Wiley & Sons, 2003

[2] RFID (radio frequency identification) https://www.techtarget.com/iotagenda/definition/RFID-radio-frequency-identification (pristup: 3.9.2023.)

[3] ACR122U Application Programming Interface V2.04 https://www.acs.com.hk/download-manual/419/API-ACR122U-2.04.pdf (pristup: 3.9.2023.)

[4] ACR122U Technical Specifications V3.06 http://www.acs.com.hk/download-manual/418/TSP-ACR122U-3.06.pdf (pristup: 3.9.2023.)

[5] Yocto Project Documentation https://docs.yoctoproject.org/ (pristup: 3.9.2023.)

[6] Lib NFC https://github.com/nfc-tools/libnfc (pristup: 3.9.2023.)

[7] Srce dokumentacija https://wiki.srce.hr/pages/viewpage.action?pageId=126715078 (pristup: 3.9.2023.)

[8] RFID coupling techniques - backscatter, capacitive, inductive https://www.electronics-notes.com/articles/connectivity/rfid-radio-frequency-identification/coupling-techniques-capacitive-inductive-backscatter.php (pristup: 5.9.2023.)

[9] Pravilnik o studentskoj ispravi https://narodne-novine.nn.hr/clanci/sluzbeni/2014_07_90_1830.html (pristup: 5.9.2023.)

[10] NXP MIFARE Classic https://www.nxp.com/products/rfid-nfc/mifare-hf/mifare-classic/mifare-classic-ev1-1k-4k:MF1S50YYX_V1 (pristup: 5.9.2023.)

[11] Cloning Cryptographic RFID Cards for 25$?, Timo Kasper, Ingo von Maurich, David Oswald, Christof Paar, Horst Görtz Institute for IT Security, Ruhr-University Bochum, Germany http://www.proxmark.org/files/Documents/13.56%20MHz%20-%20MIFARE%20DESFire/Cloning_Cryptographic_RFID_Cards_for_25USD-WISSEC_2010.pdf (pristup: 5.9.2023.)

[12] Infosec - NFC Mifare https://securityguill.com/nfc_mifare.html (pristup: 5.9.2023.)

[13] Vedat Coskun, Kerem Ok, Busra Ozdenizci, Near Field Communication – From Theory to Practice, Wiley, 2012

[14] The life of Pi: Ten years of Raspberry Pi https://www.cam.ac.uk/stories/raspberrypi (pristup: 6.9.2023.)

[15] Raspberry Pi Model B https://docs.rs-online.com/6403/0900766b8127da4b.pdf (pristup: 6.9.2023.)

[16] Frank Vasquez Chris Simmonds, Mastering Embedded Linux Programming, Packt Publishing, 2021

[17] Distorwatch https://distrowatch.com/ (pristup: 6.9.2023.)

[18] Alex González, Embedded Linux Development Using Yocto Project Cookbook, Packt Publishing, 2018

[19] Otavio Salvador, Daiane Angolini, Embedded Linux Development Using Yocto Project, Packt Publishing, 2023

[20] systemd man stranica https://man7.org/linux/man-pages/man1/systemd.1.html (pristup: 8.9.2023.)

[21] rfcomm man stranica https://linux.die.net/man/1/rfcomm (pristup: 8.9.2023.)

[22] Bluetooth SIG https://www.bluetooth.com/develop-with-bluetooth/join/member-directory/ (pristup: 8.9.2023.)

[23] Bluetooth 5.4
https://www.bluetooth.com/wp-content/uploads/2023/02/2301_5.4_Tech_Overview_FINAL.pdf (pristup: 8.9.2023.)

[24] sdptool man stranica
https://linux.die.net/man/1/sdptool (pristup: 8.9.2023.)

[25] Statistika tržišta mobilnih uređaja
https://www.statista.com/statistics/272698/global-market-share-held-by-mobile-operating-systems-since-2009/ (pristup: 8.9.2023.)

[26] Kotlin docs
https://kotlinlang.org/docs/home.html (pristup: 11.9.2023.)

[27] android-bluetooth-serial
https://github.com/harryjph/android-bluetooth-serial (pristup: 11.9.2023.)

[28] ACR1255U-J1 ACS Secure Bluetooth® NFC Reader Reference Manual V1.13
https://www.acs.com.hk/download-manual/7664/REF-ACR1255U-J1-1.13.pdf (pristup: 15.9.2023.)

[29] Anviz W1C Pro Cloud Based Wireless Terminals
https://www.anviz.com/file/goview/9403.html (pristup: 15.9.2023.)

SAŽETAK

Razmatra se sustav za evidenciju studenata skeniranjem studentskih iskaznica. Sustav se sastoji od ACR122U čitača kartica, Raspberry Pi Zero W računala za upravljanje čitačem i slanje podataka Bluetooth vezom na mobilni uređaj. Kako bi Raspberry Pi Zero W što efikasnije obavljao svoj zadatak napravljena je jednostavna Linux distribucija samo za tu namjenu. Na mobilnom uređaju se lista studenata trajno pohranjuje.

Ključne riječi: RFID, Linux, Yocto, Android, MIFARE

ABSTRACT

Title: Application of RFID readers for recording student attendance in classes

A system for keeping student records by scanning student ID cards is being considered. The system consists of an ACR122U card reader, a Raspberry Pi Zero W computer for controlling the reader and sending data via Bluetooth connection to a mobile device. In order for the Raspberry Pi Zero W to perform its task as efficiently as possible, a simple Linux distribution was created just for that purpose. The list of students is permanently stored on the mobile device.

Keywords: RFID, Linux, Yocto, Android, MIFARE

ŽIVOTOPIS

Stjepan Marjanović rođen je u Vinkovcima, 4. srpnja 1998. godine. Pohađao je osnovnu školu Fra Bernardina Tome Leakovića u Bošnjacima. Srednju školu upisuje u Županji 2013. godine, Prirodoslovno matematički smjer Gimnazije Županja. 2017. godine upisuje preddiplomski sveučilišni studij Računarstva na Fakultetu elektrotehnike, računarstva i informacijskih tehnologija Osijek.

_____

Potpis autora