VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

BAKALÁŘSKÁ PRÁCE

Brno, 2020 Lukáš Kyzlík

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY
A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY
DEPARTMENT OF CONTROL AND INSTRUMENTATION

SYSTÉM LOGOVÁNÍ PŘÍCHODŮ POMOCÍ NFC
NFC READER WITH SUPERVISORY COMMUNICATION

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE AUTHOR	Lukáš Kyzlík
VEDOUCÍ PRÁCE SUPERVISOR	Ing. Václav Kaczmarczyk, Ph.D.

BRNO 2020

BRNO UNIVERSITY OF TECHNOLOGY FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

Bachelor's Thesis

Bachelor's study program **Automation and Measurement**

Department of Control and Instrumentation

Student:	Lukáš Kyzlík	***ID:***	195377
Year of study:	3	***Academic year:***	2019/20

TITLE OF THESIS:

NFC reader with supervisory communication

INSTRUCTION:

Make a HW and SW design of NFC card readers. When the card is placed next to the reader, the authorization request is generated and sent to the supervisory system through either WiFi or metallic Ethernet. The user should be informed whether the authorization was successful or not.

1) Make a research of existing solutions.
2) According to existing standards create own hardware and software solution.
3) Perform necessary tests and make appropriate documentation.
4) Integrate your solution with existing system.

RECOMMENDED LITERATURE:

Norma ISO/IEC 14443

Date of project specification:	3.2.2020	***Deadline for submission:***	8.6.2020

Supervisor: Ing. Václav Kaczmarczyk, Ph.D.

doc. Ing. Václav Jirsík, CSc.
Chair of study program board

WARNING:
The author of the Bachelor's Thesis claims that by creating this thesis he/she did not infringe the rights of third persons and the personal and/or property rights of third persons were not subjected to derogatory treatment. The author is fully aware of the legal consequences of an infringement of provisions as per Section 11 and following of Act No 121/2000 Coll. on copyright and rights related to copyright and on amendments to some other laws (the Copyright Act) in the wording of subsequent directives including the possible criminal consequences as resulting from provisions of Part 2, Chapter VI, Article 4 of Criminal Code 40/2009 Coll.

ABSTRACT

The Bachelor thesis focuses on the research of NFC readers with supervisory communication and subsequent design of hardware and software modules for a prototype of such reader. The final solution consists of ESP32 microcontroller, PN532 NFC module and custom PCB which ensures the connection of components and distribution of power from battery or network. The reader communicates with the existing backend system using Wi-Fi. The communication is secured using HTTPS and reader periodically broadcast messages about its correct functioning. Simple API specification was created for the communication. The software was developed in C language using ESP-IDF framework. The system is compared to existing solutions on the market in the conclusion.

KEYWORDS

NFC, RFID, access control, authentication, ESP32, ESP-IDF, PN532, HTTPS

ABSTRAKT

Bakalářská práce se zabývá průzkumem NFC čteček pro přístupové systémy a následným návrhem harwaru a softwaru prototypu takovéto čtečky. Výsledný systém se setává z mikrokontroléru ESP32, NFC modulu PN532 a speciálně navržené DPS, která zajišťuje propojení komponent a napájení z baterie nebo sítě. Čtečka komunikuje pomocí Wi-Fi s existujícím systém kontroly příchodů. Komunikace je zabezpečena pomocí HTTPS a čtečka posílá pravidelné zprávy o svém správném fungování. Pro komunikaci byla vytvořena jednoduchá API specifikace. Software byl vytvořen v jazyce C pomocí ESP-IDF frameworku. V závěru práce je systém porovnán s některými existujícími řešeními na trhu.

KLÍČOVÁ SLOVA

NFC, RFID, přístupový systém, autentizace, ESP32, ESP-IDF, PN532, HTTPS

KYZLÍK, Lukáš. *Systém logování příchodů pomocí NFC*. Brno, 2020. Dostupné také z: `https://www.vutbr.cz/studenti/zav-prace/detail/126922`. Bakalářská práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky. Vedoucí práce Václav Kaczmarczyk.

ROZŠÍŘENÝ ABSTRAKT

Cílem práce je prozkoumat existující řešení hardwarových a softwarových modulů přístupových systémů a navrhnout vlastní systém založený na čtečce karet využívající technologii Near Field Communication (NFC).

V první kapitole, jsem provedl rešerši informací o vlastnostech a standardech NFC technologie, které jsou podstatné pro návrh čtečky karet. Mezi nejdůležitější vlastnosti NFC zařízení patří komunikační módy a signalizační módy. Komunikační módy (čtení/zápis, peer-to-peer, emulace karet, bezdrátové nabíjení) určují, jaké režimy práce zařízení podporuje, a signalizační módy (NFC-A, NFC-B, NFC-F), jaký komunikační protokol a norma jsou využívány. NFC karty (tagy) existují v pěti různých typech, podle toho, jaké módy podporují a jaké normy splňují. Většina zařízení se řídí některou ze tří hlavních technických norem: ISO/IEC 14443, ISO/IEC 15693 nebo JIS X 6319.

Kapitola 2 obsahuje průzkum existujících řešení dostupných na trhu. V rámci něho jsem blíže prozkoumal 2 celosystémová řešení kontroly přístupu a 3 samostatné NFC čtečky. Prozkoumaná systémová řešení byla IMAporter od IMA a Kisi od stejnojmenné společnosti a prozkoumané samostatné čtečky ACR1252U od Advanced Card Systems, RC-S380/S od Sony a SimpleLink CC3200 od Texas Instruments a DLP Design.

Porovnal jsem hlavní parametry jako např. podporované módy a protokoly, podporované typy karet, vnější rozhraní, dosah NFC, cena a rozměry a došel k závěrům užitečným pro návrh vlastního řešení: Samostatné čtečky se pohybují v cenovém rozmezí od 30 do 50 USD. Systémová řešení jsou o hodně dražší a nejsou příliš flexibilní z vývojářského hlediska. Maximální rozměry prozkoumané čtečky byly 147 × 50 × 20 mm a můj návrh by se do nich měl také vejít. Obvyklé čtečky podporují všechny tři signalizační módy NFC-A, NFC-B a NFC-F. Dosahy NFC se různí od 5 do 70 mm. Obvyklá vnější rozhraní jsou USB a Wi-Fi.

Na základě rešerše o NFC technologii a průzkumu trhu jsem vytvořil vlastní návrh hardwaru a softwaru NFC čtečky pro přístupový systém. V posledních dvou kapitolách se práce věnuje popisu těchto návrhů a jejich zhodnocení.

Hardware NFC čtečky se obecně skládá z následujících komponent: antény, integrovaného obvodu pro NFC, mikrokontroléru a backendového systému (serveru). Mezi anténou a NFC obvodem je *RF rozhraní*, mezi NFC obvodem a mikrokotolérem *hostitelské rozhraní* a mezi mikrokontrolérem a backendem *vnější rozhraní*. Pro navrhovanou čtečku jsem zvolil následující hardware:

- *PN532 modul* jako NFC obvod

- *ESP32 DevKit v1* jako desku s mikrokontolerem
- *Microsoft IIS server* jako existující backendový systém

Jako hostitelské rozhraní bylo využito SPI a jako vnější rozhraní Wi-Fi. Dále jsem navrhnul vlastní DPS, která propojuje komponenty a zajišťuje jejich napájení. Čtečka může být napájena z Li-Pol baterie nebo z 5 V stejnosměrného zdroje. DPC byla navržena pomocí open-source programu KiCad.

Čtečka podporuje čtení karet odpovídající signalizačnímu módu NFC-A a mezinárodní normě ISO/IEC 14443A.

Dále jsem srovnal navržený hardware s existujícími řešeními. Navržený prototyp čtečky má rozměry 128 × 44 × 26 mm, to přesahuje maximální rozměry z průzkumu trhu pouze ve výšce o 6 mm. Odhadovaná cena výroby prototypu je cca 56 USD, což přesahuje maximální cenu z průzkumu trhu o 7 USD. Můžeme tedy usuzovat, že prototyp v těchto směrech není tak dobrý jako produkty na trhu a přibližuje se jim. Rozměry a cenu výroby by bylo možné vylepšit další optimalizací rozložení komponent a výrobou ve větším množství v tomto pořadí. Prototyp čtečky také nepodporuje u produktů na trhu běžné NFC-B a NFC-F módy, ty ale nejsou k základní funkci přístupového systému nutné.

V neposlední řadě jsem pro čtečku vytvořil software. Je napsaný v jazyce C a zkompilovaný pomocí Espressif IoT Development Framework (ESP-IDF). Aplikace využívá komponenty, které jsou součástí frameworku a dostupné pod licencí Apache License 2.0 a komponent pn532 pro komunikaci s NFC modulem, který je dostupný na GitHub pod licencí Unlicense.

Aplikace má následující funkcionalitu: Čtečka čeká na detekci karty odpovídající normě ISO/IEC 14443A. Když je karta nalezena, přečte její ID a 32 bytů dat z její paměti a pošle tato data přes Wi-Fi na backendový server. Server zkontroluje s databází identitu držitele karty a pošle zpět informaci, zda má držitel přístup. Čtečka poté indikuje uživateli, zda má přístup, pomocí červeného nebo zeleného světla dvoubarevné LED. Pokud je čtečka odpojena od zdroje a dochází jí baterie, indikační LED se rozsvítí oranžově. Čtečka navíc v pravidelných intervalech 10 s posílá serveru informaci o tom, že pracuje správně.

Program čtečky je tvořen čtyřmi softwarovými komponenty, které mají následující funkce:

- *NFC komponent* – komunikace s PN523 moudlem
- *Wi-Fi komponent* – komunikace s backendovým serverem
- *GPIO komponent* – kontrola LED a stavu napájení

- *Main komponent* – hlavní program realizující výše popsanou funkcionalitu pomocí metod z ostatních komponentů

Software obsahuje řadu funkcí, které zajišťují bezpečnost přístupového systému: Spojení se serverem je šifrováno pomocí Hypertext Transfer Protocol Secure (HTTPS) a identita serveru je ověřována pomocí SSL certifikátu. Identita čtečky je ověřována pomocí hashe vytvořeného funkcí HMAC SHA-256 z ID čtečky a při kompilaci náhodně generovaného textového souboru. Bezpečná aktualizace firmwaru včetně certifikátu a hashe může být teoreticky realizována pomocí over-the-air update přes HTTPS, který platforma podporuje, to ale není pro snížení složitosti projektu implementováno.

Softwarová část projektu nebyla porovnána s existujícími řešeními na trhu, protože jejich software není většinou volně dostupný, odpovídá ale standardům a normám pro NFC a Wi-Fi.

Hlavním přínosem práce, je vytvoření a zveřejnění nového návrhu NFC čtečky na bázi platformy ESP32 a jeho porovnání s existujícími čtečkami. Kód aplikace a návrh DPS bude zveřejněn na GitHub.

DECLARATION

I declare that I have written the semestral project titled "NFC Reader with Supervisory Communication" independently, under the guidance of the advisor and using exclusively the technical references and other sources of information cited in the project and listed in the comprehensive bibliography at the end of the project.

As the author I furthermore declare that, with respect to the creation of this semestral project, I have not infringed any copyright or violated anyone's personal and/or ownership rights. In this context, I am fully aware of the consequences of breaking Regulation § 11 of the Copyright Act No. 121/2000 Coll. of the Czech Republic, as amended, and of any breach of rights related to intellectual property or introduced within amendments to relevant Acts such as the Intellectual Property Act or the Criminal Code, Act No. 40/2009 Coll., Section 2, Head VI, Part 4.

Brno 06/06/2020 ....

author's signature

ACKNOWLEDGEMENT

I would like to thank to my supervisor Ing. Václav Kaczmarczyk, Ph.D. for his professional guidance, support and improvement suggestions for the work. Furthermore, I would like to thank to Robert Rössler and Robert Jonathan Šimon for valuable advice and to my parents for emotional support.

Brno 06/06/2020 . . . .

author's signature

Contents

Introduction **15**

1 NFC Technology **16**

1.1 NFC vs RFID . . . 16

1.2 NFC Communication Modes . . . 16

1.2.1 Reading/Writing Mode . . . 16

1.2.2 Peer-to-peer Mode . . . 17

1.2.3 Card Emulation Mode . . . 17

1.2.4 Wireless Charging Mode . . . 17

1.3 NFC Signaling Modes . . . 17

1.4 NFC Tag Types . . . 18

1.4.1 Type 1 . . . 18

1.4.2 Type 2 . . . 18

1.4.3 Type 3 . . . 18

1.4.4 Type 4 . . . 18

1.4.5 Type 5 . . . 19

1.5 NFC Standards and Norms . . . 19

1.5.1 ISO/IEC 14443 . . . 19

1.5.2 JIS X 6319 . . . 20

1.5.3 ISO/IEC 15693 . . . 20

2 Market Research **21**

2.1 NFC Access Control System Solutions . . . 21

2.1.1 IMAporter . . . 21

2.1.2 Kisi . . . 22

2.2 Standalone NFC Readers . . . 22

2.2.1 Advanced Card Systems ACR1252U . . . 23

2.2.2 Sony RC-S380/S . . . 23

2.2.3 SimpleLink CC3200 NFC Card Reader . . . 25

2.3 Market Research Conclusions . . . 25

2.3.1 Price and Customization . . . 25

2.3.2 Dimensions . . . 26

2.3.3 Signalling Modes . . . 26

2.3.4 Antenna . . . 26

2.3.5 External Interfaces . . . 26

3 Hardware Design **28**

3.1 Architecture 28
3.1.1 General PCD Architecture 28
3.1.2 PCD Prototype Architecture 28
3.2 PN532 Transceiver Module 29
3.2.1 Communication and Signalling Modes 29
3.2.2 Interfaces 29
3.2.3 Security 30
3.3 ESP32 DevKit v1 Board 30
3.3.1 ESP32-WROOM-32 Module 31
3.3.2 ESP32 Microcontrller 31
3.3.3 Security 32
3.4 Main PCB 32
3.4.1 KiCad 33
3.4.2 Power Distribution 33
3.4.3 Power Management 34
3.5 MIFARE Classic PICC 34
3.5.1 Architecture 34
3.5.2 Data Storage and Authentication 35
3.6 Dimensions 36
3.7 Comparison to Market Research Conclusions 36

4 Software Design **39**
4.1 ESP-IDF 39
4.1.1 ESP-IDF Project Structure 40
4.2 Functionality 40
4.3 Architecture 41
4.4 NFC Component 41
4.4.1 NFC Configuration 41
4.4.2 Communication between the PN532 and the PICC 42
4.4.3 Communication between the ESP32 and PN532 42
4.4.4 Card Data Storage on the ESP32 43
4.5 Wi-Fi Component 44
4.5.1 Wi-Fi Configuration 44
4.5.2 Communication with the Remote Server 44
4.6 GPIO Component 45
4.6.1 GPIO Configuration 45
4.6.2 Reading Battery Voltage 46
4.6.3 Controlling LEDs 46
4.7 Main Component 46

4.7.1 Tasks 47
4.7.2 Resource Management 47
4.8 Communication Security 48
4.8.1 Server Authentication 48
4.8.2 Reader Authentication 49
4.8.3 Secure Firmware Update 49

Conclusion **51**

Bibliography **53**

List of appendices **57**

A Content of Attached CD **58**

B Scheme of the Reader PCB Circuits **59**

List of Figures

2.1 IMAporter reader 22
2.2 Kisi system hardware 23
2.3 ACR1252U USB NFC Reader 24
2.4 Sony RC-S380/S reader 24
2.5 SimpleLink CC3200 reader with a tag 25
3.1 Architecture of the NFC reader [36] 29
3.2 PN532 module 30
3.3 ESP32 DevKit v1 Board 31
3.4 3D render of the PCB crated with KiCad 33
3.5 NFC card 35
3.6 EEPROM organization on ISO/IEC 14443A card [21] 36
4.1 Workflow with ESP-IDF framework [5] 39
4.2 Data transfer between host controller, PN532 module and target PICC [21] 42

List of Tables

1.1 NFC signalling modes [40] [11] 17
1.2 NFC tag types [19] [2] 19
2.1 Comparison of devices and systems on market 27
3.1 Most important parameters of the PN532 module 30
3.2 Prices of the components and PCB manufacturing for single prototype 37
4.1 InDataExchange command structure [21] 43
4.2 Content of the custom `log_data_t` structure 43
4.3 Content of the custom `http_response_t` structure 45

Listings

4.1 Minimal ESP-IDF example project file tree 40
4.2 Definition of the card data logging function 43
4.3 All possible response strings 45
4.4 Semaphore API usage example 47

Introduction

The goal of the thesis is to research access control systems with supervisory communication based on identification cards and design software and hardware modules for a reader prototype for such a system. Systems which can log the presence of the users based on identification cards are recently finding many applications. Office buildings, schools, hospitals, factories and other buildings use electronic systems to control access of people to specific areas, to check attendance etc. Automated hotels and hostels are appearing which don't need any front desk thanks to this technology and many new applications for this technology will probably be developed in the near future.

In the thesis, I am focusing on the design of modules of an access control system reader using Near Field Communication (NFC) technology. NFC is a very popular identification technology for its convenience, security, power efficiency and backwards compatibility with Radio Frequency Identification (RFID).

The first chapter summarizes important knowledge about NFC technology essential for an NFC reader design. It contains information about NFC features and its standardization.

In the second chapter, I completed market research which was focused on access control system solutions as well as standalone NFC readers which can be incorporated into larger solutions. I derived some conclusions about how should my prototype be designed from the research at the end of the chapter.

The design of the NFC reader prototype hardware and software modules is described in the last two chapters. The hardware of the system is based on an ESP32 system-on-chip, PN532 NFC module and custom PCB to connect them and distribute power. I compare my final hardware design to the conclusions from the market research. The software was written in C language using ESP-IDF platform and its components. Security of the prototype is ensured thanks to HTTPS connection and other security features.

1 NFC Technology

Near Field Communication (NFC) is a technology enabling communication between two electronic devices on a short distance up to 10 cm [36]. It is relatively low-power, energy-efficient and low-speed communication technology. NFC technology is used for contactless payments, transfer of files between devices, tags on objects providing logistics info and more. It has been developed from Radio Frequency Identification (RFID) technology [33].

1.1 NFC vs RFID

NFC has more capabilities than RFID technology but is based on RFID protocols. Both technologies operate at frequency 13.56 MHz.

RFID is designed mainly for identification and only low amounts of data can be exchanged with it. There is a reader device and tag with identity information. *Passive RFID* powers the tag with energy from the reader. *Active RFID* uses tags with a power source to extend the communication range.

NFC reader is capable of reading RFID tags but has more modes. It can exchange data both ways with another NFC device or act as a tag.

This section was sourced from [35], [39].

1.2 NFC Communication Modes

There are four modes of operation of NFC devices specified by the NFC Forum. Information about the first three modes was sourced from [15], [30], [29].

1.2.1 Reading/Writing Mode

In this mode, the active NFC device can read or write data from passive tags. In the writing mode, the initiator device writes data on the tag. Data are transmitted by amplitude modulation of the 13.56 MHz signal in the speed of 106 Kbit/s. Previously present data on the tag are overwritten. In reading mode, initiator device powers tag and reads the data which are transferred by load modulation. These modes are used to get information from the tag (e.g. identity of the owner) or to initiate some action (e.g. device paring).

1.2.2 Peer-to-peer Mode

Two active NFC devices can exchange data (e.g. photos, contacts) in peer-to-peer mode. Both devices generate electromagnetic field and data is transferred by amplitude modulation. It requires less maximum power because both devices have their power supply. While one device transmits, other is listening and then the roles are switched. Data rates can be 106, 212 or 424 Kbit/s.

1.2.3 Card Emulation Mode

This mode allows the device to appear to a reader as though it was an RFID tag. In this mode, the device doesn't use any energy. The mode can be used in existing systems originally designed for passive tags (e.g. access control, electronic tickets, payment card).

1.2.4 Wireless Charging Mode

In this mode, the NFC device can charge small IoT devices. Single NFC antenna can manage both communication and charging. Standardized power transfer classes are 250, 500, 750 and 1000 mW. [25]

1.3 NFC Signaling Modes

Signalling modes define which contactless communication protocols are used.

NFC Forum Standard	**NFC-A**		**NFC-B**		**NFC-F**	
International Standard	**ISO/IEC 14443A**		**ISO/IEC 14443B**		**FeliCA JIS X6319-4**	
	polling	**listening**	**polling**	**listening**	**polling**	**listening**
Coding	Modified Miller	Manchaster	NRZ-L	NRZ-L	Manchaster	Manchaster
Modulation	ASK 100%	ASK	ASK 100%	BPSK	ASK 10%	ASK
Data Rate	106 kbit/s	106 kbit/s	106 kbit/s	106 kbit/s	212 / 424 kbit/s	212 / 424 kbit/s
Frequency	13.56 MHz	13.56 MHz ± 848 kHz	13.56 MHz	13.56 MHz ± 848 kHz	13.56 MHz	13.56 MHz

Tab. 1.1: NFC signalling modes [40] [11]

1.4 NFC Tag Types

Five standardized types of NFC tags exist in the present. Each has different parameters, functionality and price. This means each is suitable for different applications. This section was sourced from [2].

1.4.1 Type 1

The NFC Type 1 tags use the simplest and slowest chips from all tag types. On the other hand, they can have more memory and are cheaper than Type 2 because of their simplicity. They are used for read-only applications e.g. business cards, Bluetooth pairing.

1.4.2 Type 2

The NFC Type 2 tags are faster and have more functionality. They have enough functionality to be suitable for many applications while maintaining a low price point. This makes Type 2 tags the most widespread NFC tag type. Typical applications are low-value monetary transactions, one-use tickets for transport and events, URL redirects etc.

1.4.3 Type 3

The NFC Type 3 tags have a wide range of functionality but are also quite expensive. They were developed by Sony in Japan and are widely used in Asia. This tag type is used for multiple-use tickets, electronic ID cards, medical devices, home electronics etc.

1.4.4 Type 4

The NFC Type 4 tags have the most memory and functionality of all types. As only tag type they allow to perform authentication and support ISO/IEC 7816 which requires the ability to self-modify. Therefore, they are mainly used for security applications e.g. transit tickets.

1.4.5 Type 5

The NFC Type 5 is standard recently adopted by the NFC Forum. It is used for instance for ski passes and pacing of medication and other products.

	Type 1	**Type 2**	**Type 3**	**Type 4**	**Type 5**
ISO/IEC Standard	14443A	14443A	JIS 6319-4	14443A/B	15693
Signalling Mode	NFC-A	NFC-A	NFC-F	NFC-A, NFC-B	NFC-V
Memory	96 B to 2 KB	48 B to 2 KB	2 KB	32 KB	64 KB
Data Access	read/write	read/write, read-only	read/write, read-only	read/write, read-only	read/write
Data Rate	106 kbit/s	106 kbit/s	212 / 424 kbit/s	106 / 212 / 424 kbit/s	26.48 kbit/s
Anti-collision	no	yes	yes	yes	yes
Price	low	low	high	medium / high	low / high

Tab. 1.2: NFC tag types [19] [2]

1.5 NFC Standards and Norms

1.5.1 ISO/IEC 14443

ISO/IEC 14443 is standard issued by the International Organization for Standardization and International Electrotechnical Commission that defines requirements for proximity identification cards.

Standard uses specific terminology for the devices. The reader of the cards is called *Proximity Coupling Device (PCD)* and the card *Proximity Integrated Circuit Card (PICC)*.

Norm has four parts:

- *ISO/IEC 14443-1* defines physical characteristics of PICCs.
- *ISO/IEC 14443-2* defines characteristics of fields provided by devices and rules for communication between a PCD and PICC.
- *ISO/IEC 14443-3* defines how to initialize communications between a PICC and PCD, specifies request and answer to request commands, and rules to avoid collisions in communication.
- *ISO/IEC 14443-4* defines a half-duplex block transmission protocol and commands used for communication between PCD and PICC after the initialization.

This section was sourced from [24].

1.5.2 JIS X 6319

JIS X 6319 is Japanese Industrial Standard issued by the Japanese Industrial Standards Committee with title *Specification of implementation for integrated circuit cards.*

Standard consists of four parts with the following titles:

- *JIS X 6319-1* – Integrated circuit cards with contacts
- *JIS X 6319-2* – Proximity cards
- *JIS X 6319-3* – Common commands for interchange
- *JIS X 6319-4* – High speed proximity cards

Mainly *JIS X 6319-4* is relevant for this work. This standard defines physical characteristics, transmission protocols, file structure and more parameters for PICCs. It is used in FeliCa systems developed by Sony and corresponds to NFC-F mode defined by the NFC Forum.

This section was sourced from [18].

1.5.3 ISO/IEC 15693

ISO/IEC 15693 is standard by the International Organization for Standardization and International Electrotechnical Commission. Its section *ISO/IEC 15693-2* specifies field characteristics of *Vacinty Coupling Device (VCD)* and *Vacinity Integrated Circuit Card (VICC).* Signaling mode of this standard is labeled as NFC-V by the NFC Forum [26].

It has longer operating distance and slower data rates (see table 1.2) compared to *ISO/IEC 14443* because it was originally designed only for access control applications [41].

2 Market Research

The goal of this chapter is to examine NFC devices freely available on the market and compare their parameters. Findings of this research and analysis are used in the following chapters about NFC device design.

2.1 NFC Access Control System Solutions

There are several existing commercial system solutions for the identification of users and access logging. System solutions provide not only the reader but also PICCs, controller and other hardware as well as support software such as mobile applications and cloud software. Alternatively, they provide some ways of integration with other existing solutions.

2.1.1 IMAporter

IMAporter is NFC and Bluetooth reader with supporting software which can be integrated with existing security systems. It is developed by IMA company in the Czech Republic.

The reader uses a combination of contactless identification technologies: RFID, NFC and *Bluetooth Low Energy (BLE)*. It uses RFID or NFC technology with the NFC-A protocol up to 5 cm. It is compatible with MIFARE Classic 1K (Type 2), DESFire EV1 (Type 4), Ultralight, NTAG20x, NTAG21x (Type 2), PayPass, etc. tag formats. For the distances from 5 cm to 10 m it uses BLE technology. The system can automatically select the most suitable technology each time. The reader uses wired communication trough Wiegand or RS232 to connect with an existing security system.

IMAporter solution can be used with Android 4.4+ and iOS 7+ devices. Virtual keys can be assigned to devices remotely with the help of a dedicated mobile app. During the normal usage of the system, no app needs to be run.

This section was sourced from [8].

Fig. 2.1: IMAporter reader

2.1.2 Kisi

Kisi is physical security system consisting of multiple products developed by the company of the same name with offices in New York City and Stockholm.

In the basic version a system consists of:

- wall reader which uses NFC and BLE technologies
- controller that can electrically control doors, elevators, turnstiles etc.
- PICC or NFC/BLE enabled mobile phone
- control software running in cloud

Kisi Reader uses NFC up to 2 cm range and BLE up to 23 cm. It can be connected to a backend system by Wi-Fi or Ethernet and is programmable through API. It is powered through *Power over Ethernet (PoE)* interface.

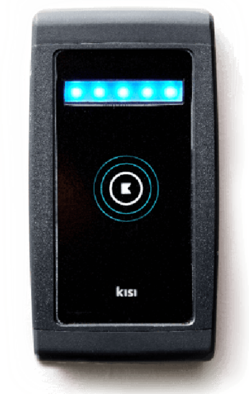
Company develops compatible *Kisi Pass* PICC with MiFare DesFire EV1 (Type 4) format. Cards use 128 bit AES encryption.

Part of the access control system is the backend cloud which allows remote control of access rights trough mobile app for Android and iOS.

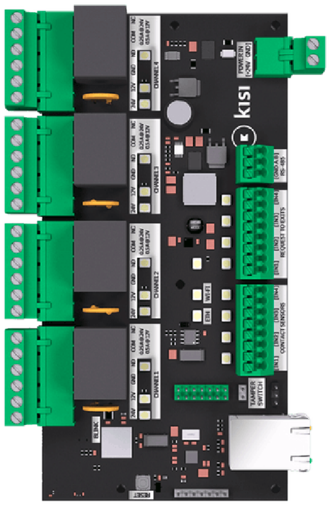
This section was sourced from [10].

2.2 Standalone NFC Readers

Apart from the system solutions, standalone NFC readers are available on the market. These readers can usually be connected to a personal computer trough external



(a) Kisi reader



(b) Kisi cotroller (without case)

Fig. 2.2: Kisi system hardware

interface and controlled trough API.

2.2.1 Advanced Card Systems ACR1252U

ACR1252U is a standalone NFC reader developed by Advanced Card Systems Ltd.

It supports NFC-A, NFC-B and NFC-F protocols and thus tag Types 1 to 4 as well as other NFC devices. Supported communication modes are reading/writing, peer-to-peer and card emulation. The reader is certified by NFC Forum for these modes. The device is capable of 424 Kbit/s data rate and its maximum operating distance is 5 cm. As a security feature, it has *Secure Access Module (SAM)* class A card slot which can be used for authentication.

It can be connected to a personal computer by USB and controlled by *Personal Computer/Smart Card (PC/SC)* API. PC/SC is a specification of interface between smart cads and personal computers developed for Windows and Unix systems by PC/SC Working Group.

This section was sourced from [1].

2.2.2 Sony RC-S380/S

RC-S380/S is an NFC reader produced by Sony as FeliCa standard device.

Fig. 2.3: ACR1252U USB NFC Reader

It supports NFC-A, NFC-B and NFC-F protocols and thus tag Types 1 to 4. Supported communication speeds are 106, 212 and 424 Kbit/s. Maximum operating distance is only 5 mm. The reader is certified by the NFC Forum.

The external interface is USB and PC/SC API can be used to control the device. Furthermore, Sony provides a Software Development Kit (SDK) for the development of software operating the reader.

This section was sourced from [12].

Fig. 2.4: Sony RC-S380/S reader

2.2.3 SimpleLink CC3200 NFC Card Reader

This device consists of *CC3200* microcontroller developed by Texas Instruments and *TRF7970* NFC chip by DLP Design.

TRF7970 booster pack is add-on board with *TRF7970A* NFC integrated circuit. It is designed to work with microcontrollers made by Texas Instruments. It supports NFC-A, NFC-B, NFC-F and NFC-V protocol and thus all five tag types. CC3200 is a microcontroller with *Cortex-M4* processor by ARM. It is connected by the SPI interface to the NFC integrated circuit which it controls.

The external interface is Wi-Fi. The device can be programmed by flashing it with Uniflash software and SDK is also provided.

This section was sourced from [17].

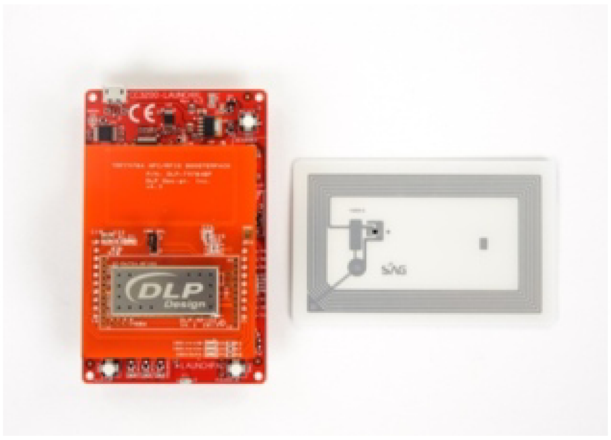


Fig. 2.5: SimpleLink CC3200 reader with a tag

2.3 Market Research Conclusions

Through analysis of the researched solutions, I have arrived at some general conclusions which should help me during the design of hardware and software modules of my solution.

2.3.1 Price and Customization

Whole system solutions are usually expensive and have limited customization options, while standalone readers can be purchased as both full products and development kits. In both cases, the price of standalone readers is usually in the range of

30 to 50 USD.

2.3.2 Dimensions

Maximal dimensions of the researched PCD were 147 × 50 × 20 mm. My design should approximately fit inside these dimensions.

2.3.3 Signalling Modes

Most commercially available standalone readers support NFC-A, NFC-B and NFC-F signalling modes. However, for simple reader support of NFC-A will be sufficient.

2.3.4 Antenna

Range of the reader antennas is quite variable, from 5 to 70 mm. Smaller antennas provide higher maximum filed but it drops faster with distance [37].

2.3.5 External Interfaces

Common external interfaces for the standalone readers available on the market are USB and Wi-Fi.

There are three main levels of integration of the NFC circuit and microcontroller used in the devices on the market:

- *bare-metal/RTOS integration* – Registers of the NFC circuit are directly exposed to microcontroller trough host interface. The microcontroller is usually *bare-metal* (without OS) or runs a *Real-Time Operating System (RTOS).*
- *embedded system* – NFC circuit is embedded system with firmware which can communicate with the microcontroller running OS (Linux, Android etc.).
- *high integration* – NFC circuit and microcontroller are combined into one unit with firmware [36].

Bare-metal/RTOS is the most suitable bor a simple NFC reader.

	IMAporter	**Kisi**	**ACR1252U**	**Sony RC-S380/S**	**SimpleLink CC320**
System	yes	yes	no	no	no
Technolgies	NFC, BLE	NFC, BLE	NFC	NFC	NFC
NFC Protocols	NFC-A	NFC-A/B	NFC-A/B/F	NFC-A/B/F	NFC-A/B/F/V
NFC Tag Types	2, 4	4	1, 2, 3, 4	1, 2, 3, 4	1, 2, 3, 4, 5
External Intrafaces	Wiegand, RS232	Ethernet, PoE, Wi-Fi	USB	USB	Wi-Fi
NFC Range [mm]	70	19	50	5	unknown
Reader Price	unknown	599 USD	31 USD	49 USD	30 USD
Reader Dimensions [mm]	117 x 50 x 20	147 x 75 x 42	98 x 65 x 12.8	60 x 11 x 100	unknown

Tab. 2.1: Comparison of devices and systems on market

3 Hardware Design

As the hardware part of the project, I designed the NFC reader (PCD) prototype which can communicate with existing backend server. Components and architecture of the prototype ware selected based on findings from market research and availability on the market.

3.1 Architecture

3.1.1 General PCD Architecture

Basic NFC reader consists of following components:

- antenna
- NFC integrated circuit (NFC IC)
- embedded microcontroller (MCU)
- backed system

Antenna is connected with NFC IC by *RF interface* and NFC IC to MCU by *host interface*. Finally, MCU is connected to backed system by *external interface*.

The component which is not an essential part of the NFC reader but is frequently used to improve authentication security is *Secure Access Module (SAM)*.

This section was sourced from [36], [37].

3.1.2 PCD Prototype Architecture

I selected following components for the prototype design:

- *PN532 transceiver module* – PCB module with highly integrated NFC IC and antenna
- *ESP32 DevKit v1* – break-out board containing *ESP32* MCU
- *Microsoft IIS server* – backend system
- main PCB with power management circuits and module connections
- Li-Pol battery

The components are connected with following interfaces:

- *SPI* as a host interface
- *Wi-Fi* as an external interface

MIFARE Classic 1K PICC was selected to be used with the prototype.

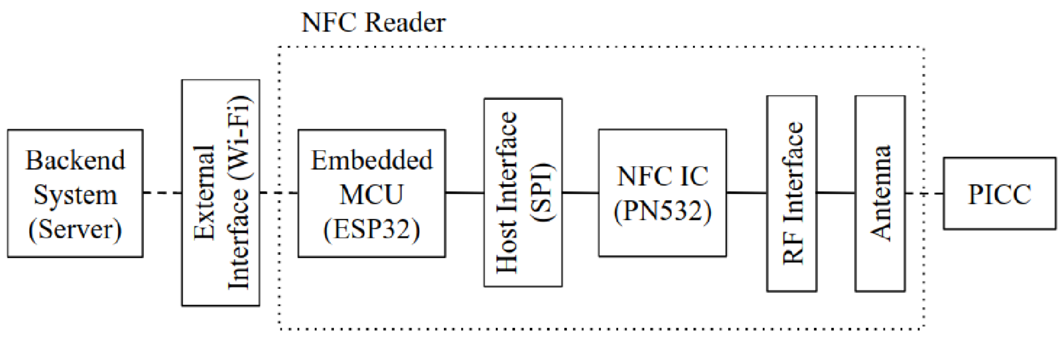


Fig. 3.1: Architecture of the NFC reader [36]

3.2 PN532 Transceiver Module

PN532 is highly integrated transceiver module based on the *80C51* microcontrller designed by NXP Semiconductors. This section was sourced from [21], [20].

3.2.1 Communication and Signalling Modes

It supports reading/writing communication mode for NFC-A, NFC-B (ISO/IEC 14443A/B) and NFC-F (FeliCa) signalling modes, card emulation mode for NFC-A and NFC-F, and peer-to-peer mode for ISO/IEC 18092. I am using NFC-A protocol in the reading/writing mode, compatible with MIFARE Classic 1K and 4K PICCs, for the functionality of the prototype.

3.2.2 Interfaces

RF interface is integrated into the module and supports speeds 212 and 424 kbit/s. The PN532 module is compatible with three host interfaces: *Serial Peripheral Interface (SPI)*, *Inter-Iterated Circuit (I^2C)* and *High Speed UART (HSU)*. Host interface can be selected using hardware switches on the module. I chose SPI connection with the MCU for the prototype. Communication protocol with the MCU is described in more detail in 4.4.3.

3.2.3 Security

The PN532 module doesn't have an integrated SAM. However, SAM can be connected through dedicated *NFC-WI/S^2C* interface and this way full secure PICC functionality can be ensured. To reduce the complexity of the project the prototype doesn't implement this but it can be added in the future.

Parametr	**Value**
NFC Protocols	NFC-A, NFC-B, NFC-F
NFC Communication Modes	reading/writing, card emulation, peer-to-peer
External Interfaces	SPI, I^2C, HSU, NFC-WI/S^2C (for SAM)
NFC Range [mm]	50
Dimensions [mm]	$43 \times 40 \times 4$
Price [USD]	4 to 6

Tab. 3.1: Most important parameters of the PN532 module

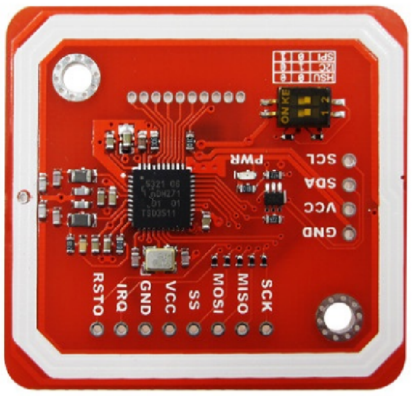


Fig. 3.2: PN532 module

3.3 ESP32 DevKit v1 Board

The ESP32 DevKit v1 is development break-out board crated by DOIT. It is based on *ESP32-WROOM-32* module which is based on the *ESP32* MCU. The board provides 25 digital input/output pins, 6 analogue input pins and 2 analogue output pins. It can be powered directly trough VIN pin or via the USB micro B port. Recommended input voltage is 7 to 12 V. Board operating voltage is 3.3 V. [14]

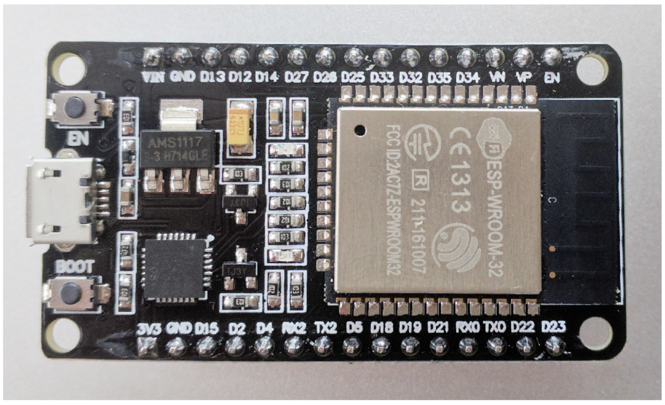

VIN GND D13 D12 D14 D27 D26 D25 D33 D32 D35 D34 VN VP EN
EN
BOOT
AMS1117
ESP-WROOM-32
3V3 GND D15 D2 D4 RX2 TX2 D5 D18 D19 D21 RX0 TX0 D22 D23

Fig. 3.3: ESP32 DevKit v1 Board

3.3.1 ESP32-WROOM-32 Module

The ESP32-WROOM-32 is a surface mount IC containing *ESP32* MCU and providing some additional hardware and functionality. It uses FreeRTOS, popular RTOS distributed under MIT open-source license. [4] [6]

3.3.2 ESP32 Microcontrller

The ESP32 is MCU with integrated Wi-Fi and Bluetooth designed for IoT and wearable electronics applications. Its main features are following:

- *Wi-Fi IEEE 802.11 b/g/n* with Infrastructure Station, Soft-AP, and Promiscuous modes
- *Bluetooth v4.2 BR/EDR* and *Bluetooth Low Energy*
- low-power management
- cryptographic hardware acceleration
- wide range of peripheral interfaces

The peripheral interfaces of ESP32 interesting from the point of view of PCD prototype design are following:

- 34 × General Purpose Input/Output (GPIO)
- 12-bit Analog-to-digital Converter (ADC)
- 4 × SPI
- 2 × I^2C [1]

[1]Can be used as an alternative to SPI to connect to PN532 module.

This microcontroller was selected for the prototype design based on its compatibility with the PN532 module, integrated Wi-Fi, easy development using DevKit boards, and low price.

This section was sourced from [27].

3.3.3 Security

This section was sourced from [3].

Remote Access

Remote communication on ESP32 MCU can be secured using CA certificates to validate the identity of the remote server (see 4.8.1). The application that I developed for the prototype implements this.

Physical Access

ESP32 has flash encryption and secure boot features. When the flash memory is encrypted, data are decrypted at-runtime by the MCU and can't be read by any other hardware. Secure boot is ensuring that the firmware has not been tampered with. The check is being done on every boot. To reduce the complexity of the project the prototype doesn't implement these features but they can be added in the future.

3.4 Main PCB

I designed a custom PCB for the prototype. The PCB contains slots for the PN532 and ESP32 modules, circuits which connect the modules, circuits for the distribution and management of the electrical power, and red-green indicator LED. Schemes of the PCB circuits can be found in appendix B. There are 4 screw holes with 3 mm diameter in the corners of the PCB which allow it to be mounted on a surface. The PCB was designed using *KiCad* software.

3.4.1 KiCad

KiCad is a bundle of open-source software for creating circuit schemes and PCB layouts with Gerber format output. It is available for Windows, Linux, and macOS. I selected it for the project because it is freely available and easy to use while having all features necessary for the project. [9]

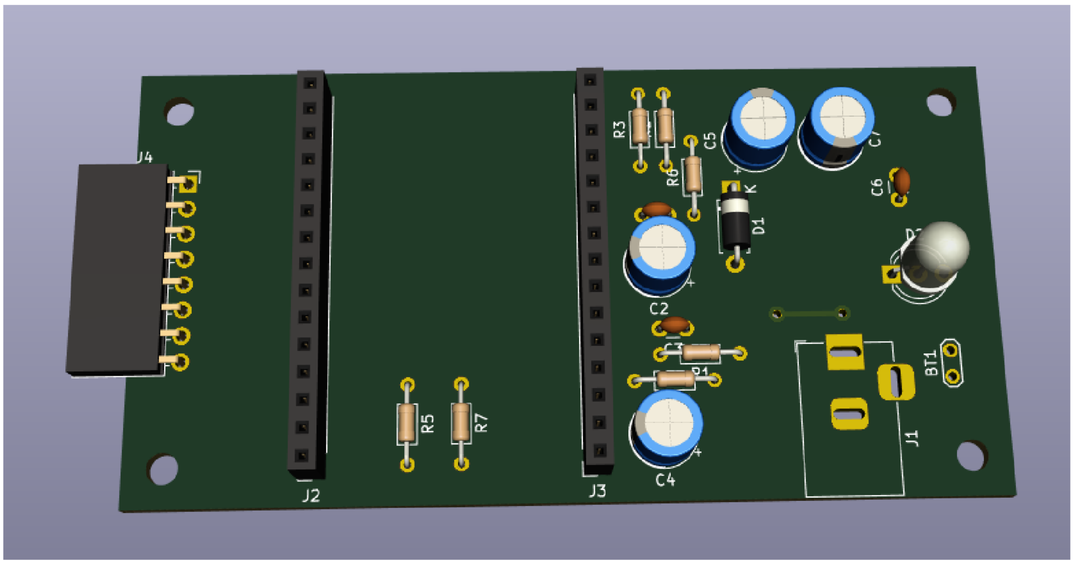


Fig. 3.4: 3D render of the PCB crated with KiCad

3.4.2 Power Distribution

The prototype can be powered by either 5 V DC source which connects to the PCB via barrel jack connector or by 5 V Li-Pol rechargeable battery. When the prototype is powered from the DC source, the battery is charging and when it is unplugged, the battery takes over. Battery charging is managed using the *MCP73832* battery management IC. A combination of MOSFET transistor and Shottky diode ensures switching between battery and source power.

Because battery and DC source operate at 5 V and ESP32 board at 3.3 V, they are connected with a voltage step-down circuit. It consists of *LF33* voltage regulator and three capacitors which serve to crate low-pass filters to limit the fluctuations in voltage which could compromise the operation of the MCU.

3.4.3 Power Management

Parts of the circuit are connected directly to ESP32's GPIO and ADC pins to allow for the detection of current power status by the software (see 4.6).

Firstly, the software can detect whether the reader is powered from the battery or the source. The positive voltage node of the source is connected to GPIO input through a voltage divider, which halves the voltage. If the DC source is connected, GPIO pin voltage is 2.5 V which reads as *high* and is safely under 3.3 V limit. In case the source is disconnected, pin voltage is close to 0 V and reads as *low*.

Secondly, the voltage of the battery and thus an approximate level of charge can be read. The positive voltage of the battery connector is connected through a voltage divider to an ADC pin where it can be read. ADC1, 12-bit analogue-to-digital converter of the ESP32 chip, is used [27].

3.5 MIFARE Classic PICC

I selected *MIFARE Classic 1K* PICC for the use with the PCD prototype because it supports all necessary modes and has a low price. It is Type 2 tag compliant with ISO/IEC 14443A (NFC-A signalling mode) with 1 kB *Electrically Erasable Programmable Read-only Memory (EEPROM)* and 6-byte key memory access protection with read/write or write-only modes. [23]

3.5.1 Architecture

PICC consists of following parts:

- antenna
- microcontroller
- EEPROM memory

Tag is powered by electromagnetic induction trough antenna when in the proximity of PCD and the same electromagnetic field is used to transfer the information. Operation of the PICC is controlled by microcontroller and data to be broadcasted (e.g. ID number of the tag) is stored in the EEPROM. [23]

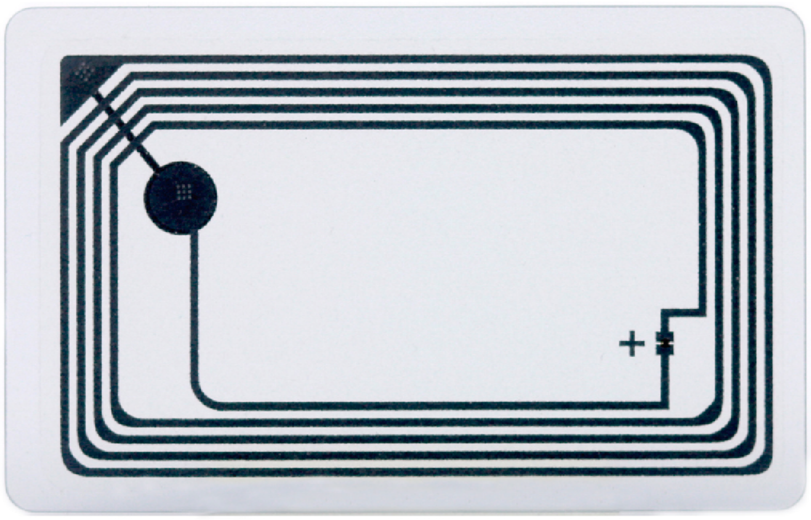


Fig. 3.5: NFC card

3.5.2 Data Storage and Authentication

MIFARE Classic card has EEPROM which is organized in 16 4-block sectors. Each sector consists of 3 *data blocks* and *sector trailer* with the exception of the sector 0. Sector 0 is made up from *manufacturer block* (block 0), 2 *data blocks* (blocks 1 to 2) and *sector trailer* (block 3). Each block consists of 16 bytes.

Manufacturer block contains 4 or 7-byte card ID and the rest of it is filled with manufacturer data. This block can't be modified.

Data block can be used to store 16 bytes of any data depending on the needs of the programmer.

Sector trailer holds authentication keys A (mandatory) and B (optional) and access bits which determine access rights for reading and writing to all blocks of the particular sector. Access bits and keys can be read or modified only with the knowledge of one or both keys, depending on the current setting of access bit for the sector trailer. Key A can never be read, only modified with the knowledge of the current one.

This section was sourced from [21].

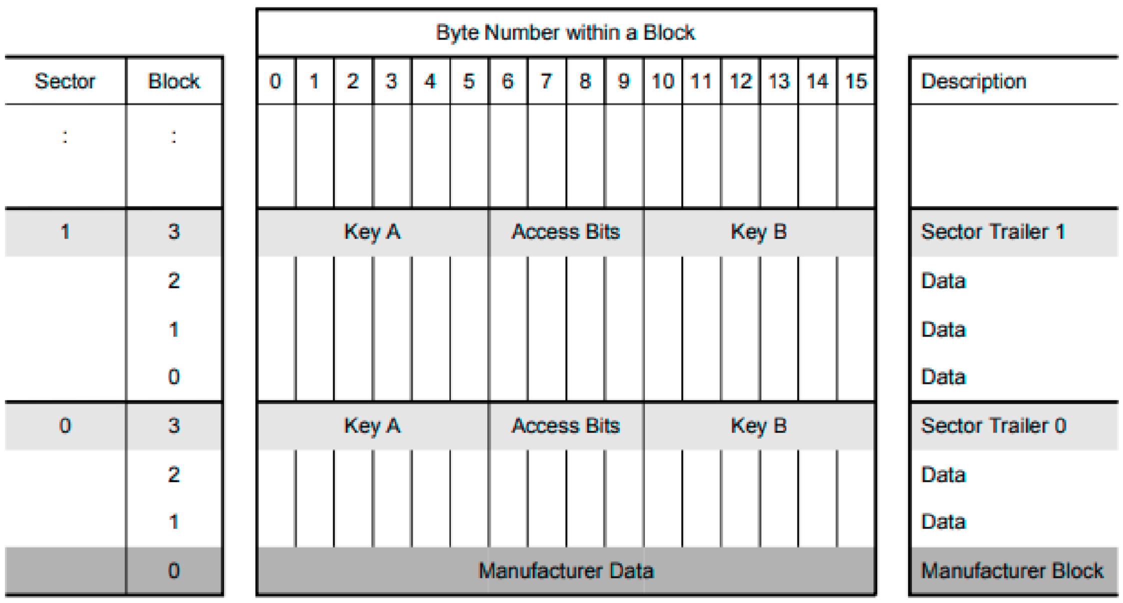


Fig. 3.6: EEPROM organization on ISO/IEC 14443A card [21]

3.6 Dimensions

The width and the depth of the PCB are 83.82 and 43.18 mm respectively. The PN532 socket is on the side and when mounted the module extends the prototype width for another 40 mm to 127.82 mm total. The height of the PCB is 26 mm and is determined by the height of the battery on the bottom side, the height of the PCB, and the tallest component on the top side which is ESP32 board mounted on the pin header. Thus dimensions of the prototype are approximately 129 × 44 × 26 mm. This is excluding a plastic cover box which could add up to 3 mm in each direction.

3.7 Comparison to Market Research Conclusions

In this section, I am comparing the most important properties of the designed prototype to the information gathered during the market research.

The dimensions of the designed hardware prototype are 128 × 44 × 26 mm, which fits within the maximal dimensions from the market research summarized in table 2.1 (147 × 50 × 20 mm) except for height. This is a good result considering it is a prototype design. It could be optimized by replacing more THT components with SMD variants and by better battery placement.

I calculated the price of the components and PCB manufacturing for making a single prototype in the table 3.2. The prices are based on e-shop GM Electronic (`gme.cz`) and PCB manufacturer Plošné spoje Korunní (`pskorunni.cz`). I converted them to USD to allow for a comparison with the data from the market research [2]. Total price for manufacturing a single prototype is approximately 56 USD which is slightly over the maximal price of a standalone PCD from the market research (49 USD). However, this price could be significantly reduced by mass-producing the PCD, e.g. with 100 pieces, the price would only be approximately 21 USD per device (including PCB and components). Therefore, the design could at least theoretically be competitive on the market in terms of a price.

ID	**Component**	**Am**	**Price per Piece CZK**	**Total Price CZK**	**Total Price USD**
BT1	LiPol battery 1200mAh	1	199	199.0	8.56
C1, C3, C6	Capacitor 100nF THT	3	2.8	8.4	0.36
C2, C4	Capacitor 10μF THT	2	1.1	2.2	0.09
C5, C7	Capacitor 100μF THT	2	1.2	2.4	0.10
D1	Shottky diode THT	1	2.8	2.8	0.12
D2	Red-green LED THT 5mm	1	6.7	6.7	0.29
J1	Barrel Jack	1	7.5	7.5	0.32
J2, J3	Pin socket 1x15 (for ESP32)	2	7.8	15.6	0.67
J4	Pin socket 1x8 (for PN532)	1	6.8	6.8	0.29
Q1	PMOS transistor SMD	1	5.6	5.6	0.24
R1	Resistor 2kΩ THT	1	2.9	2.9	0.12
R2,R3	Resistor 330Ω THT	2	2.9	5.8	0.25
R4, R6, R7	Resistor 4.7kΩ THT	3	2.9	8.7	0.37
R5	Resistor 1kΩ THT	1	2.9	2.9	0.12
U1	MCP73832 IC SMD	1	15.31	15.3	0.66
U2	LF33 IC SMD	1	25	25.0	1.08
	DC source 5V	1	135	135.0	5.81
	PCB with manufacturing	1	848	848.0	36.46
	TOTAL			**1300.6**	**55.93**

Tab. 3.2: Prices of the components and PCB manufacturing for single prototype

The prototype uses Wi-Fi as an external interface which is common in devices on the market.

[2]I used the official exchange rate from 5th of June 2020: 0.043 USD = 1 CZK

It only supports NFC-A signalling mode unlike most devices on the market which also support NFC-B and NFC-F. However, NFC-F is supported by the PN532 module and could be added by upgrading the software.

4 Software Design

As the software part of the project, I designed software for ESP32 system-on-chip controlling the reader prototype. The software was written using C language and *Espressif IoT Development Framework (ESP-IDF)*.

4.1 ESP-IDF

ESP-IDF is an official development framework for ESP32 system-on-chip developed by Espressif. To create applications for ESP32, following software is needed:

- toolchain
- bulid tools
- ESP-IDF
- text editor or IDE

The toolchain is software that compiles C code for ESP32. Build tools are software tools needed to create a full application. In this case, build tools used are *CMake* and *Ninja*. ESP-IDF itself is a bundle of libraries and other source code which serves as API to operate the toolchain. It contains software components that allow a user to easily use Bluetooth, networking, other communication peripherals and protocols, storage, RTOS system etc. on the ESP32.

It is possible to build an application and flash it to ESP32 hardware using terminal commands. The framework also contains a monitor to view debug messages from ESP32. [5]

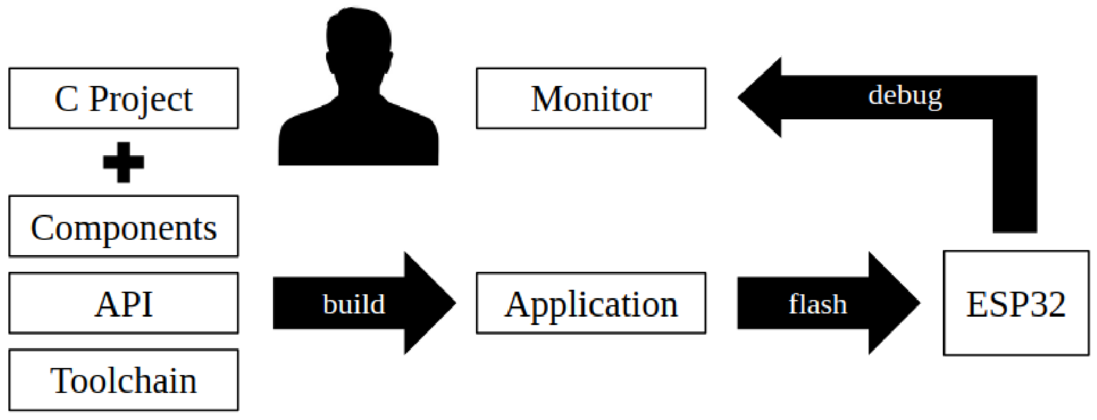


Fig. 4.1: Workflow with ESP-IDF framework [5]

4.1.1 ESP-IDF Project Structure

Source code of the project is divided into components. Each project has a special main component in which the program starts. Apart from the main component project can contain other optional components. Optional components work similarly to libraries and usually contain code that can be used in multiple projects. They can also be used to give more structure to a bigger project. Project folder, as well as a folder of each component, contains its own `CMakeLists.txt` file. This file contains instructions for CMake on how to build the project such as project name, component dependencies etc. `sdkconfig` file stores configuration parameters of the project. [5]

```
project_name/
    CMakeLists.txt
    sdkconfig
    components/
        comp1/
            CMakeLists.txt
            src1.c
        comp2/
            CMakeLists.txt
            src1.c
    main/
        CMakeLists.txt
        src1.c
        src2.c
    build/
```

Listing 4.1: Minimal ESP-IDF example project file tree

4.2 Functionality

The reader waits for detection of ISO/IEC 14443A card. When the card is detected, it reads the card's ID and another 32 bytes from its EEPROM memory and sends this data over Wi-Fi to a backend server. The server checks card data against a database and sends back information whether the card owner has access rights. Upon processing the response, the prototype reader signals it to a user with a flash of its indicator LED. Red light for "access denied" or green for "access granted". If the reader is unplugged and the battery charge level is critical, the indicator LED lights up orange and other indications are disabled until the reader is plugged in.

Furthermore, the reader sends in regular intervals of 10 s information about its status to the server. This way, the server continuously verifies that the reader is functional.

All of the communication with the backend server is realized over a secure encrypted HTTPS connection.

4.3 Architecture

Software project consists of following components with following purposes:

- *NFC component* (`card_reader_nfc`) – communication with a PN532 module
- *Wi-Fi component* (`card_reader_wifi`) – communication with a backend server
- *GPIO component* (`card_reader_gpio`) – control of LEDs and reading power status
- *Main component* (`main`) – main program

Apart from these components designed by me, project contains PN532 component (`pn532`) available on [7] under Unlicense [1]. This component is a library of functions to control PN532 hardware module created by GitHub user *binh8994*. Also, functions from components and examples which are part of ESP-IDF environment are used. The ESP-IDF components are avalible under Apache License 2.0 [2].

4.4 NFC Component

NFC component (`card_reader_nfc`) provides functionality to authenticate ISO/IEC 14443A card and read data from it using a PN532 module and store them in a structured way in the reader memory.

4.4.1 NFC Configuration

The communication between ESP32 and PN532 module over SPI is configured and configuration parameters are stored in dedicated `pn532_t` structure on startup. This is done using methods from the PN532 component warped inside my `nfc_setup()` method.

[1] `https://unlicense.org/`

[2] `https://www.apache.org/licenses/LICENSE-2.0`

4.4.2 Communication between the PN532 and the PICC

When the PICC is detected by PN532 module, the software reads card ID which can be accessed freely and blocks 4 and 5 (first 2 blocks of sector 1) which can be accessed only using authentication key A. 6-byte key A is set on all compatible cards and stored on the reader during the program upload. This way no one can create a fake card without the knowledge of the key A. Both the card ID and 32 bytes of data from blocks 4 and 5 are used to securely identify the user (card holder). This communication is handled by PN532 module firmware based on commands from the ESP32 host controller.

4.4.3 Communication between the ESP32 and PN532

To issue authentication, and other commands, ESP32 host controller communicates with the PN532 module over SPI using a special communication protocol. The protocol is based on exchanging information frames. Sending and reading the frames is managed by the functions from the PN532 component.

Reader software operates with following frames:

- *InDataExchnage command frame* – authentication and reading data
- *ACK frame* – successful transmission of data
- *NACK frame* – asking to repeat the response

This section was sourced from [21].

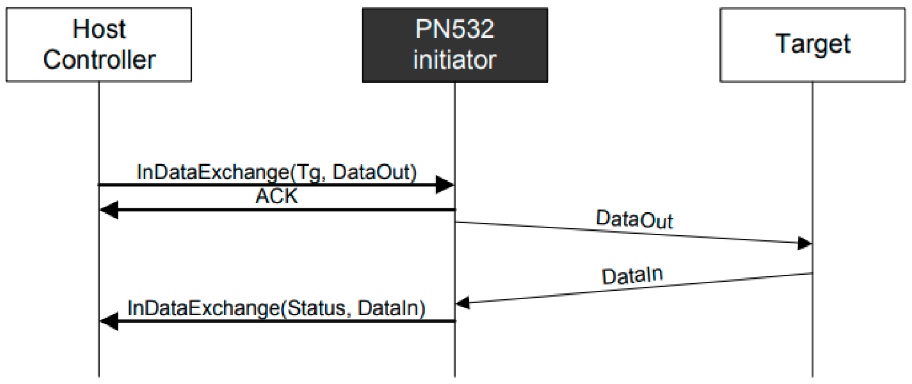


Fig. 4.2: Data transfer between host controller, PN532 module and target PICC [21]

Byte	Name	Description
0	Cmd	specifies command type (e.g. authentication, reading, writing)
1	Addr	address (block number) associated with the command
2..17	Data	data to be read/written

Tab. 4.1: InDataExchange command structure [21]

4.4.4 Card Data Storage on the ESP32

Software stores data from one card detection in specially defined custom structure `log_data_t`. The structure holds information read from the PICC and ID of the reader.

Data is read from card and stored into the custom structure in following steps:

1. initialize `log_data_t` structure with default values
2. store reader ID in the structure
3. read card ID and store it in the structure
4. perform authentication, read 2 data blocks and store them in the structure

Finally, NFC component provides function `nfc_arrayToApiString()` to convert data into REST API style string that is used by Wi-Fi component to send the data to backend server (see 4.5.2).

Identifier	Data Type	Size [B]	Description
cidLen	`uint8_t`	1	length of the card ID in bytes
rid	`uint8_t[8]`	8	reader ID
cid	`uint8_t[8]`	8	card ID
data	`uint8_t[32]`	32	2 16-byte blocks of card data

Tab. 4.2: Content of the custom `log_data_t` structure

```c
uint8_t nfc_logCard(pn532_t *obj, log_data_t *logData, uint8_t *rid
    , uint8_t *keyA) {
  nfc_initLogData(logData);
  nfc_setReaderID(logData, rid);
  if(nfc_readCardID(obj, logData) == -1) {
    ESP_LOGE(TAG_NFC, "Reading Card ID failed");
    return 1;
  }
  if(nfc_authReadData(obj, logData, keyA)) {
    ESP_LOGE(TAG_NFC, "Reading Card Data failed");
    return 2;
  }
  return 0;
}
```

Listing 4.2: Definition of the card data logging function

4.5 Wi-Fi Component

Wi-Fi component (`card_reader_wifi`) provides functionality to configure Wi-Fi, connect to the network and backend server, send card and reader data using secure HTTPS connection and read replies of the server.

4.5.1 Wi-Fi Configuration

ESP32 chip provides several Wi-Fi modes, the most important from which are Station Mode (STA), in which it connects to an access point and Access Point Mode (Soft-AP), in which other devices can connect to it [5]. I used STA mode in the application because the reader operates like a client which connects to the backend server via the Internet. The application supports WPA, WPA2 and WEP security modes. Configuration is handled by `wifi_setup()` method.

4.5.2 Communication with the Remote Server

ESP32 chip is a client which communicates with a backend server using HTTP. I selected this solution because it is simple, can be well secured by using HTTPS variant of the protocol (see 4.8.1) and is suitable for the amounts of data exchanged. I defined a special simple API for this exchange.

Sending Data to the Server

Data are sent with a GET request in a *query string* part of the URL. They are encoded in the typical format as sequence of key-value pairs separated by `&` delimiter (e.g. `key1=value1&key2=value2`). Values are encoded in a hexadecimal format. Apart from the values in a query string, the reader key hash is sent as a cookie in the HTTP header. This hash serves to authenticate the reader as described in 4.8.2.

Two kinds of messages are being sent: *Card log message* is sent after card detection and contains all data from the `log_data_t` structure. *Alive message* is sent in regular intervals and contains only reader ID.

Receiving Data from the Server

Data are received in a response message to the GET request as a string in the message body. The string has the following format: three-digit response code, space, and response message. Code starting by 1 signifies response to a card log message and 2 response to an alive message. If the following two digits are 00, it signifies OK response and other codes specify errors.

```
100 OK
101 Unregistered reader
102 Access denied (unknown user)
200 OK
201 Unregistered reader
```

Listing 4.3: All possible response strings

Upon receiving, the response is sored in custom `http_response_t` structure on ESP32 and can be evaluated.

Identifier	**Data Type**	**Size [B]**	**Description**
apiCode	`uint32_t`	4	response code
apiMessage	`char[2045]`	2045	response message

Tab. 4.3: Content of the custom `http_response_t` structure

4.6 GPIO Component

GPIO component (`card_reader_gpio`) provides functionality to control onboard and indicator LEDs, and read current power and battery status.

4.6.1 GPIO Configuration

After the start of the program, the component configures digital GPIO pins for the control of the onboard an indicator LEDs as output pins and digital pin for the power management (see 3.4.3) as an input pin. It also configures ADC1 to 12-bit width and one of its pins as an input pin. This pin is used to read the battery voltage. This is handled by `gpio_setup()` method.

4.6.2 Reading Battery Voltage

The level of battery charge can be approximated from the voltage on the analogue pin connected to the battery. `gpio_getBatteryVolatge()` method returns current battery voltage in mV which is calculated using formula 4.1. Critical battery voltage can be configured and a warning is indicated as described in 4.2.

$$V \ = \ V_{raw}\frac{V_{ref}}{2^L}A \ \doteq \ V_{raw}\frac{1100}{2^{12}} \cdot 6 \ \doteq \ 1.6\,V_{raw} \tag{4.1}$$

V	battery voltage [mV]
V_{raw}	raw reading from the pin (values 0 to 4095)
V_{ref}	ADC1 reference voltage [mV], $V_{ref} = 1100\,mV$
L	width of the ADC [bit], $L = 12\,bit$
A	amplification constant to compensate the voltage divider [-], $A = 6$

4.6.3 Controlling LEDs

Indicator and onboard LEDs can be set to a desired state using dedicated methods `gpio_setIndicatorLed()` and `gpio_setOnboardLed()`. Indicator LED has four possible states `LED_RED`, `LED_GREEN`, `LED_ORANGE`, and `LED_OFF`. I decided to use onboard LED for debug testing purposes only and it has `LED_OFF` and `LED_ON` states.

4.7 Main Component

Main component (`main`) contains main program that controls the reader and realizes functionality described in 4.2. It is also an entry point of the program.

On startup, setup functions for NFC, Wi-Fi and GPIO setup are called. Than reader key hash is generated and threads for the main functionalities are started.

4.7.1 Tasks

The component uses feature of FreeRTOS which allows to split a program to parallel threads (tasks). Following tasks with following purposes are implemented:

- *Card Read Task* (`cardReadTask()`) – reading card data, sending it to a back-end server, processing response and indicating it to a user
- *Alive Task* (`aliveTask()`) – sending regular messages about the reader status to a backend server
- *Battery Warning Task* (`batteryWarningTask()`) – checking battery and power status and indicating to a user when the level of charge is critical

4.7.2 Resource Management

Tasks use some of the same resources and because they run in parallel, access to those resources must be restricted to one task at the time to prevent conflicts and race conditions. For this, the Semaphore API in the FreeRTOS is used. Semaphore variables and functions can restrict access to bits of code as shown in code snippet 4.4.

```c
SemaphoreHandle_t semaphore1 = NULL;

// A task that creates a semaphore
void task1(void * pvParameters)
{
    // Create the semaphore to guard a shared resource
    vSemaphoreCreateBinary(semaphore1);
}

// A task that uses the semaphore
void task2(void * pvParameters)
{
    // See if the resource is free and if not wait here until
    timeout passes
    // parameter 1:  semaphore variable
    // parameter 2:  timeout in ticks (portMAX_DELAY = never)
    if (xSemaphoreTake(semaphore1, portMAX_DELAY) == pdTRUE) {
    {
        // We can use the resource here ...
        xSemaphoreGive(semaphore1); // Release resource
    }
    else
    {
        // Execute if semaphore timeout passes (in this case never)
    }
}
```

Listing 4.4: Semaphore API usage example

Two semaphores are used in my application:

- `httpSemaphore` to protect access to the HTTPS communication
- `indLedSemaphore` to project access to the red-green indicator LED

Information about FreeRTOS in this section was sourced from [5].

4.8 Communication Security

The following must be ensured during a secure communication between the reader prototype and the server:

- *confidentiality* – Data can't be read by someone else.
- *integirty* – Data can't be changed by someone else.
- *authenticity* – Both parties entering the communication can identify each other.

This can be achieved by using cryptographic algorithms and cryptographic keys. Algorithms can be symmetric or asymmetric. Symmetric algorithms use the same secret key on both sides. Asymmetric algorithms use public and private key pair.

This section was sourced from [38].

4.8.1 Server Authentication

The backend server is authenticated using TSL certificate and HPPTS.

HTTPS

Hypertext Transfer Protocol Secure (HTTPS) is a version of HTTP with the TSL (Transport Security Layer). It allows for encrypted client-server communication and authentication. The protocol is based on asymmetric cryptography.

The severer has a certificate. The client first verifies if the server's certificate was signed by a trusted Certificate Authority (CA). If the authentication is successfully completed, they will exchange a new symmetric key and establish communication which ensures integrity and confidentiality of the data.

Certificates are signed by either publicly trusted CA (e.g. Google CA) or self-signed. Self-signed means that the developer created their own CA to sign the certificates.

This section was sourced from [34], [31].

Implementation

I have used `esp_http_client` component built in the ESP-IDF to implement HTTPS communication [5]. The client certificate is embedded as binary data directly in the application firmware. Certificate from trusted CA or self-signed certificate can be embedded based on the backend server configuration.

4.8.2 Reader Authentication

To be authenticated the reader has apart from its ID a unique hash – *reader key*. This hash is sent as a cookie in the header of every HTTP request of the prototype reader and the server has a database of registered reader IDs and corresponding reader keys which is used to verify the readers.

The reader key hash is 32 bytes long and generated using *Hash-based Message Authentication Code Secure Hashing Algorithm 256 (HMAC SHA-256)* which generates a hash from a key and a message [28]. The reader ID is used as the key and the message is a content of the `rkey_seed.txt` file embedded in the firmware. This file was randomly generated before uploading the firmware on the prototype and the hash added to the server database.

4.8.3 Secure Firmware Update

Because the certificate will eventually expire, there must be a way to update it. In my implementation the certificate is embedded in the firmware, so the firmware must be updated. This can be done in one of two ways: with physical access over USB or remotely using over-the-air (OTA) update. OTA update can be done over secured HTTPS connection each time before the certificate expires. OTA updates have not been implemented on the prototype to reduce the complexity of the project. [32]

The reader ID and/or `rkey_seed.txt` file and subsequently the reader key can be updated using OTA update if needed.

Conclusion

The goals of the thesis were achieved in the following steps:

I identified that the most important features of NFC to be understood during the hardware and software design are communication modes, signalling modes and tag types. Furthermore, I researched that the most relevant standards for this work are ISO/IEC 14443, JIS X 6319-4 and ISO/IEC 15693-2.

Market research, which I conducted, examined 2 access control system solutions and 3 standalone NFC readers which served as a reference for my design of the modules. Key parameters of all researched products are summed up in the table and general conclusions summarized at the end of the chapter 2. Maximal dimensions and price of researched standalone reader ware 49 USD and 147 × 50 × 20 mm respectively.

I designed a reader prototype following existing standards. I selected PN532 NFC transceiver module and ESP32 DevKit v1 board with Wi-Fi support as the main hardware components based on findings from the market research and their market availability. This components are connected and powered trough custom PCB designed using KiCad software. The dimensions of the final prototype are 128 × 44 × 26 mm which exceeds maximal dimensions from the market research only in height for 6 mm. The price of components and PCD manufacturing was approximately 56 USD which exceeds maximal price from the market research for 7 USD. Therefore, I conclude that the prototype is worse from the point of view of price and size than similar products on the market but only by a close margin. This difference could be eliminated by further optimization and production on a bigger scale. The reader is also lacking some of the common features of the products on the market such as NFC-B and NFC-F signalling mode support but those features are not essential for its access control functionality.

For the reader prototype, I designed software application using ESP-IDF. It is written in C programming language and utilizes built-in components of the ESP-IDF and PN532 component available under Unilicense. The software is capable of reading the information from the card and sending it to a backend server. To integrate the prototype with existing backend system (server) I designed simple API specification which defines communication messages between reader and server. Furthermore, the application contains additional security features such as: sending regular "alive" messages to the server to notify it about its correct functioning, communicating using encrypted HTTPS, verifying the reader identity with a unique hash, and displaying low battery warning. The reader software has not been compared to other software solutions due to their public unavailability but compiles with the NFC and Wi-Fi standards and norms.

Main contribution of the thesis is creating and publishing new design of a NFC reader using ESP32 chip and comparing it to the existing designs of NFC readers. The application code and PCB design will be also published on GitHub.

Bibliography

[1] Advaced Card Systems. ACR1252U: USB NFC Reader III (NFC Forum Certified Reader). [online], [cit 11.12.2019].
URL: `https://www.acs.com.hk/en/products/342/acr1252u-usb-nfc-reader-iii-nfc-forum-certified-reader/`

[2] Dummies. The 5 NFC Tag Types. [online], [cit 8.12.2019].
URL: `https://www.dummies.com/consumer-electronics/5-nfc-tag-types/`

[3] ESP-Jumpstart. Security Considerations. [online], [cit 14.5.2020].
URL: `https://docs.espressif.com/projects/esp-jumpstart/en/latest/security.html`

[4] ESP32-WROOM-32 Datasheet. [online], [cit 14.5.2020].
URL: `https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32_datasheet_en.pdf`

[5] Espressif. ESP-IDF Programming Guide. [online], [cit 5.5.2020].
URL: `https://docs.espressif.com/projects/esp-idf/en/latest/esp32/index.html`

[6] FreeRTOS. About FreeRTOS Kernel. [online], [cit 14.5.2020].
URL: `https://www.freertos.org/RTOS.html`

[7] GitHub. pn532-esp-idf. [online], [cit 7.5.2020].
URL: `https://github.com/binh8994/pn532-esp-idf`

[8] IMAporter. [online], [cit 10.12.2019].
URL: `https://www.imaporter.com/en/`

[9] KiCad. About KiCad. [online], [cit 31.5.2020].
URL: `https://kicad-pcb.org/about/kicad/`

[10] Kisi. Kisi Reader Pro Specs. [online], [cit 10.12.2019].
URL: `https://help.kisi.io/hc/en-us/articles/360000795394-Kisi-Reader-Pro-Specs`

[11] RF Wireless World. NFC A vs NFC B vs NFC F-Difference between NFC-A,NFC-B,NFC-F. [online], [cit 8.12.2019].
URL: `https://www.rfwireless-world.com/Terminology/NFC-A-vs-NFC-B-vs-NFC-F.html`

[12] Sony. USB NFC Reader RC-S380/S. [online], [cit 11.12.2019].
URL: https://www.sony.net/Products/felica/business/products/RC-S380.html

[13] Unlicense. Unlicense Yourself: Set Your Code Free. [online], [cit 7.5.2020].
URL: https://unlicense.org/

[14] Zerynth Docs. DOIT Esp32 DevKit v1. [online], [cit 14.5.2020].
URL: https://docs.zerynth.com/latest/official/board.zerynth.doit_esp32/docs/index.html

[15] NFC Forum. What Are The Operating Modes Of NFC Devices? [online], 2013, [cit 7.12.2019].
URL: https://nfc-forum.org/resources/what-are-the-operating-modes-of-nfc-devices

[16] NTAG213F/216F: NFC Forum Type 2 Tag compliant IC with 144/888 bytes user memory and field detection. [online], 2015, [cit 25.12.2019].
URL: https://www.nxp.com/docs/en/data-sheet/NTAG213F_216F.pdf

[17] TI Designs SimpleLink Wi-Fi Enabled NFC Card Reader. [online], 2015, [cit 11.12.2019].
URL: http://www.ti.com/lit/ug/tidua70/tidua70.pdf

[18] Engineering360. JIS X 6319-4: 2016. 2016.
URL: https://standards.globalspec.com/std/10070623/jis-x-6319-4

[19] TN1216 Technical note: ST25 NFC guide. [online], 2016, [cit 8.12.2019].
URL: https://www.st.com/content/ccc/resource/technical/document/technical_note/f9/a8/5a/0f/61/bf/42/29/DM00190233.pdf/files/DM00190233.pdf/jcr:content/translations/en.DM00190233.pdf

[20] PN532/C1: Near Field Communication (NFC) controller. [online], 2017, [cit 11.5.2020].
URL: https://www.nxp.com/docs/en/nxp/data-sheets/PN532_C1.pdf

[21] UM0701-02: PN532 User Manual Rev. 02. [online], 2017, [cit 8.5.2020].
URL: https://www.nxp.com/docs/en/user-guide/141520.pdf

[22] AN11740: PN5180 Antenna design guide. [online], 2018, [cit 24.12.2019].
URL: https://www.nxp.com/docs/en/application-note/AN11740.pdf

[23] MF1S50YYX V1: MIFARE Classic EV1 1K. [online], 2018, [cit 31.5.2020].
URL: https://www.nxp.com/docs/en/data-sheet/MF1S50YYX_V1.pdf

[24] Online Browsing Platform. ISO/IEC 14443. [online], 2018, [cit 7.12.2019]. URL: `https://www.iso.org/obp/ui/#iso:std:iso-iec:14443:-1:ed-4:v1:en`

[25] NFC Forum. NFC Forum Specification Allows Wireless Charging Of IoT Devices Using Shared Antenna. [online], 2019, [cit 25.12.2019]. URL: `https://nfc-forum.org/nfc-forum-specification-allows-wireless-charging-of-iot-devices-using-shared-antenna/`

[26] Online Browsing Platform. ISO/IEC 15693. [online], 2019, [cit 28.12.2019]. URL: `https://www.iso.org/obp/ui/#iso:std:iso-iec:15693:-2:ed-3:v1:en`

[27] ESP32 Series Datasheet. [online], 2020, [cit 14.5.2020]. URL: `https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf`

[28] HMAC In: Wikipedia: the free encyclopedia . San Francisco (CA): Wikimedia Foundation, 2001. [online], 2020, [cit 31.5.2020]. URL: `https://en.wikipedia.org/wiki/HMAC`

[29] COSKUN, V.; OK, K.; OZDENIZCI, B.: *Professional NFC Application Development for Android*. Wrox, 2013.

[30] DARDÉ, L.: NXP Blogs. Start a conversation with NFC: Three operating modes. [online], 2014, [cit 25.12.2019]. URL: `https://blog.nxp.com/communication-infrastructure/5g-wireless/start-a-conversation-with-nfc-three-operating-modes`

[31] DENTELLA, L.: ESP32 (21) – Mutual authentication. [online], 2017, [cit 24.5.2020]. URL: `http://www.lucadentella.it/en/2017/08/25/esp32-21-mutua-autenticazione/`

[32] DENTELLA, L.: ESP32 (37) – HTTPS OTA. [online], 2018, [cit 24.5.2020]. URL: `http://www.lucadentella.it/en/2018/10/27/esp32-37-ota-via-https/`

[33] FAULKNER, C.: What is NFC? Everything you need to know. [online], 2017, [cit 6.12.2019]. URL: `https://www.techradar.com/news/what-is-nfc`

[34] HEATON, R.: How does HTTPS actually work? [online], 2014, [cit 24.5.2020]. URL: `https://robertheaton.com/2014/03/27/how-does-https-actually-work/`

[35] IGOE, T.; COLEMAN, D.; JEPSON, B.: *Beginning NFC: Near Field Communication with Arduino, Android & Phonegap.* O'Reilly Media, Inc., 2014.

[36] JOFRE, J.: NFC reader design I – How to build your own reader. [online], 2016, [cit 6.12.2019].
URL: `https://www.nxp.com/video/nfc-reader-design-i-how-to-build/your-own-reader:BUILD-YOUR-NFC-READER`

[37] JOFRE, J.: NFC reader design II – Antenna design considerations. [online], 2016, [cit 24.12.2019].
URL: `https://www.nxp.com/video/nfc-reader-design-ii-antenna-design/considerations:BUILD-YOUR-NFC-READER-2`

[38] LEROUX, E.: Security in NFC Readers. [online], [cit 25.12.2019].
URL: `https://www.themobileknowledge.com/wp-content/uploads/2017/05/Security-in-NFC-Readers-Public.pdf`

[39] MAERTENS, A.: Ecom Blog. RFID vs. NFC – What is the difference? [online], 2017, [cit 7.12.2019].
URL: `https://www.ecom-ex.com/blog/post/rfid-vs-nfc-what-is-the-difference/`

[40] MINIHOLD, R.: Near Field Communication (NFC) Technology and Measurements. [online], 2011, [cit 8.12.2019].
URL: `https://cdn.rohde-schwarz.com/pws/dl_downloads/dl_application/application_notes/1ma182/1MA182_5E_NFC_WHITE_PAPER.pdf`

[41] ROBERTI, M.: RFID Journal. What Is the Difference Between ISO 15693 and ISO 14443A Tags? [online], 2013, [cit 28.12.2019].
URL: `https://www.rfidjournal.com/blogs/experts/entry?11979`

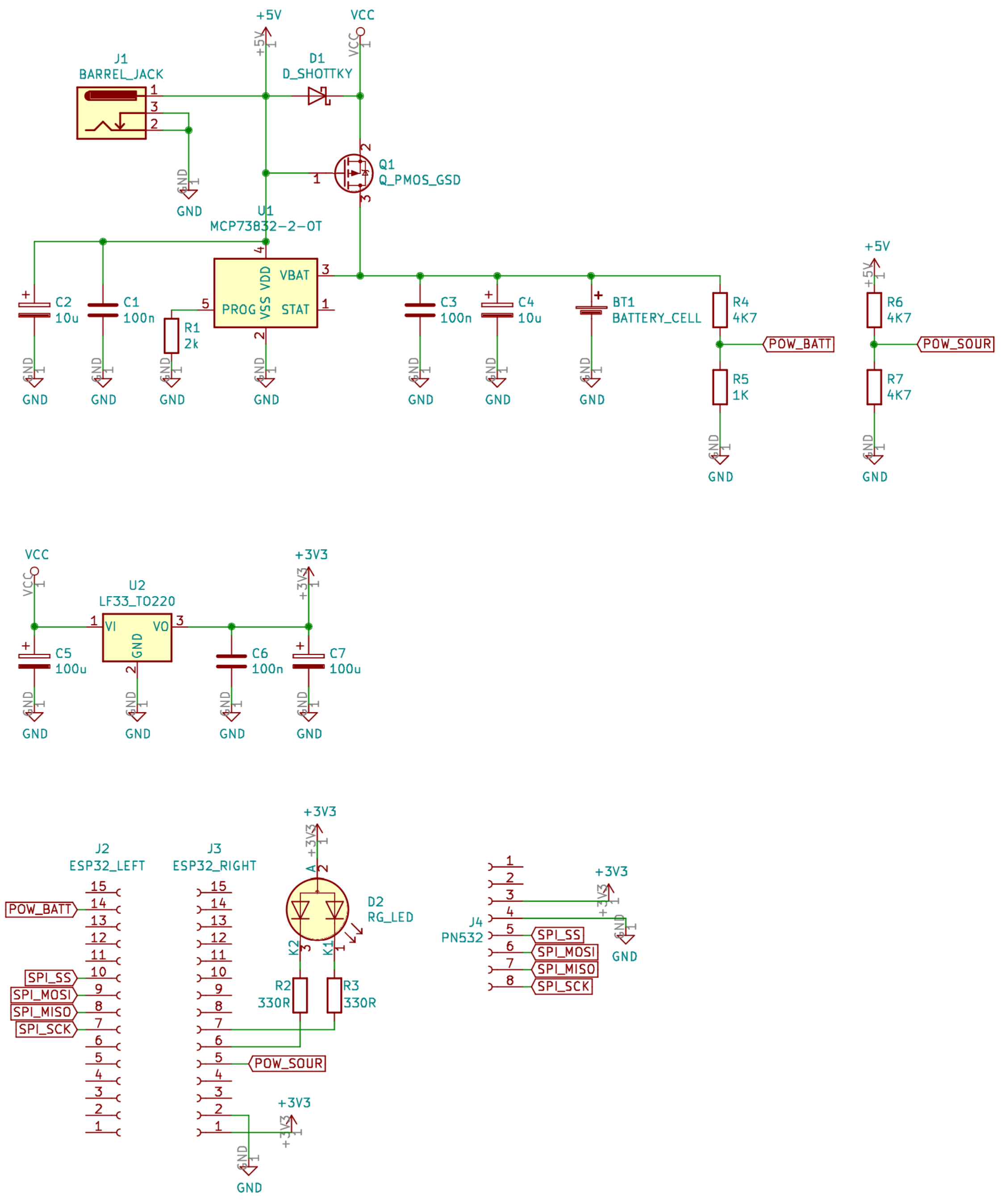
List of appendices

A Content of Attached CD **58**

B Scheme of the Reader PCB Circuits **59**

A Content of Attached CD

```
lukas_kyzlik_bachelor_thesis.pdf
lukas_kyzlik_attachments/
    hardware/
        nfc_reader
            nfc_reader.kicad_pcb
            nfc_reader.net
            nfc_reader.pro
            nfc_reader.sch
    software/
        nfc_reader_esp32_client/
            CMakeLists.txt
            components/
                card_reader_gpio/
                    CMakeLists.txt
                    card_reader_gpio.c
                    card_reader_gpio.h
                    component.mk
                card_reader_nfc/
                    CMakeLists.txt
                    card_reader_nfc.c
                    card_reader_nfc.h
                    component.mk
                card_reader_wifi/
                    CMakeLists.txt
                    card_reader_wifi.c
                    card_reader_wifi.h
                    component.mk
                    server_cert.pem
                pn532/
                    CMakeLists.txt
                    component.mk
                    pn532.c
                    pn532.h
            main/
                CMakeLists.txt
                component.mk
                main.c
                rkey_seed.txt
            sdkconfig
```



Sheet: /
File: nfcid_reader.sch

Title: NFC Reader with Supervisory Communication

Size: A4 Date: 2020-06-06 **Rev:**
KiCad E.D.A. kicad 5.1.5-52549c5~86~ubuntu16.04.1 Id: 1/1