Muhammad Nadeem

NFC DATA ACQUISITION PLATFORM DESIGN AND DEVELOPMENT TO COLLECT DATA FROM IOT SENSOR NODE

Thesis

CENTRIA UNIVERSITY OF APPLIED SCIENCES

Bachelor of Engineering, Information Technology

November 2023

ABSTRACT

Centria University of Applied Sciences	**Date** 06.11.2023	**Author** Muhammad Nadeem
Degree programme Bachelor of Engineering, Information Technology		
Name of thesis NFC DATA ACQUISITION PLATFORM DESIGN AND DEVELOPMENT TO COLLECT DATA FROM IOT SENSOR NODE		
Centria supervisor Jari Isohanni		**Pages** 33+10
Instructor representing commissioning institution or company. Kari Halonen, Aalto University		

The use of Near Field Communication (NFC) technology has a growing trend in its application for wireless transactions, including activities such as money transfers, loyalty coupons, ticketing, and secure data transfers within limited proximity. In recent years, smartphone manufacturers have included NFC modules in their mobile devices. There is a rising trend toward incorporating it into wearable medical devices to facilitate power and data transmission wirelessly over short distances. This thesis assesses the implementation of NFC communication to gather data from IoT sensor nodes and organize it for further processing. An Arduino-based hardware platform using NFC technology is designed and developed by employing commercially available off-the-shelf modules and components.

The Arduino-based microcontroller board is configured with an NFC controller to receive data from the IoT sensor node. The data acquisition hardware platform's functionality is validated with a custom antenna design for enhanced reading range. The environmental data from the IoT sensor nodes is collected, organized, and stored in the local memory as text files. The study also discusses the potential use of the acquisition platform for various internet-of-things (IoT) applications. The project work is conducted inside the laboratories and workshops of the electrical engineering department at Aalto University under the supervision of Prof. Kari Halonen's research group.

Keywords
Data organization, hardware design and development, internet-of-things (IoT), **n**ear field communication (NFC), sensor nodes, wireless data acquisition.

CONCEPT DEFINITIONS

ADC	Analog to Digital Converter
ALU	Arithmetic Logic Unit
BT	Bluetooth
CPU	Central Processing Memory
DAC	Digital to Analog Converter
DBS	Direct Broadcast Satellite
EMV	Euro pay, Mastercard, and Visa
I^{2}C	Inter-integrated Circuit
I/O	Input and Output
IP	Interface and Protocol
IoT	Internet of Things
ISO	International Organization of Standards
IT	Information Technology
LCD	Liquid Crystal Display
MISO	Master In Slave Out
MOSI	Master Out Slave In
NFC	Near Field Communication
NFC-F	Near Field Communication Felica
NXP	Next Experience
NTAG	Nanotechnology Technical Advisory Group
P2P	Peer-to-peer
PIN	Personal Identification Number
RF	Radio Frequency
RFID	Radio Frequency Identification
SCL	Serial Clock Line
SDL	Serial Data Line
SPI	Serial Peripheral Interface
T1T	Type 1 Tag
T2T	Type 2 Tag
T4AT	Type 4A Tag
UART	Universal Asynchronous Receiver Transmitter
USB	Universal Serial Bus

Wi-Fi	Wireless Fidelity
µCs	Microcontrollers

CONTENTS

ABSTRACT **2**

CONCEPT DEFINITIONS **3**

CONTENTS **5**

1 INTRODUCTION **1**

2 TECHNICAL BACKGROUND **2**
- **2.1 Radio frequency identification (RFID)** **2**
- **2.2 RFID tags** **3**
- **2.3 Optical barcodes** **3**
- **2.4 Comparison between RFID-tags and optical barcodes** **4**
- **2.5 Near field communication (NFC)** **4**
  - 2.5.1 NFC standards 5
  - 2.5.2 NFC modes 5
  - 2.5.3 NFC equipment 8
  - 2.5.4 NFC tags 8
- **2.6 Microcontroller** **9**
  - 2.6.1 Microcontroller construction 10
  - 2.6.2 Microprocessor 11
  - 2.6.3 Input and output units 11
  - 2.6.4 Program and Data Memory 12
  - 2.6.5 Data memory and other components 12
  - 2.6.6 Microcontroller programming 12
  - 2.6.7 Arduino microcontroller platform 13
  - 2.6.8 The Net Duino microcontroller platform 14
- **2.7 Communication Protocols** **14**
  - 2.7.1 Serial Peripheral Interface (SPI) 15
  - 2.7.2 Inter-integrated Circuit (I^2C) 16

3 MATERIALS AND METHODS **18**
- **3.1 Hardware design** **19**
  - 3.1.1 Microcontroller 19
  - 3.1.2 Voltage Regulator 20
  - 3.1.3 NFC controller 21
  - 3.1.4 Antenna design 22
  - 3.1.5 SPI interface 23
  - 3.1.6 UART to USB interface 24
- **3.2 Software design** **25**
  - 3.2.1 Algorithm of firmware 25
  - 3.2.2 Flowchart design 26
  - 3.2.3 Firmware code 26

4 MEASUREMENTS AND DISCUSSION **29**

5 CONCLUSION AND FUTURE PROSPECTS 32

REFERENCES

APPENDICES
APPENDIX 1
APPENDIX 2
APPENDIX 3

FIGURES

FIGURE 1. NFC frequency over RFID range on electromagnetic spectrum. 3
FIGURE 2. NFC devices operating modes..........6
FIGURE 3. NFC tag with read-write mode operation..........6
FIGURE 4. The NFC peer-to-peer communication mode with initiator and receiver. 7
FIGURE 5. The NFC-enabled device uses card emulation mode for payment transactions...........8
FIGURE 6. The NFC passive tag with an antenna. 9
FIGURE 7. Microcontroller Harvard architecture..........10
FIGURE 8. Microcontroller architecture...........11
FIGURE 9. Hierarchy of programming languages 13
FIGURE 10. The SPI communication protocol between master and slave devices...........15
FIGURE 11. The I²C bus connection between the master and several peripheral. 17
FIGURE 12. Block diagram of NFC data acquisition platform. 18
FIGURE 13. The front and back side of the NFC data acquisition board...........19
FIGURE 14. The circuit diagram of microcontroller ATmega328P for NFC data acquisition board. ..20
FIGURE 15. The circuit diagram of the LDO voltage regulator...........21
FIGURE 16. The schematic of the NFC controller circuit and matching network for the antenna. 22
FIGURE 17. The designed antenna for the NFC data acquisition platform...........23
FIGURE 18. USB to UART converter to establish communication interface 24
FIGURE 19. The flowchart represents the algorithm of the NFC data acquisition platform. 26
FIGURE 20. The measurement setup of NFC data acquisition platform with an IoT sensor node. 29
FIGURE 21. Relationship of antenna reading range with the conductor area of antenna...........31
FIGURE 22. The received power by NFC antenna vs the distance from transmitting antenna. 31

TABLES

TABLE 1. Design parameters of the NFC antennas...........23
TABLE 2. The measurements of the designed NFC antennas. 30

1 INTRODUCTION

Wireless communication allows the data exchange between two or more points that are not connected by any network cable. It allows the user to receive and transmit information in ambulatory conditions. This system is also considered the ‘world without borders’ in information technology. The wireless communication systems used for data transmission are radio frequencies (RF), Bluetooth devices (BT), infrared (IR), Wireless fidelity (Wi-Fi), and near field communication (NFC). Wireless technology has mostly been used in radio, television, and direct broadcast satellite (DBS). The NFC uses radio frequency identification (RFID) technology in peer-to-peer communication mode to establish two-way communication with other devices over short distances of a few centimeters for a brief time span at 13.56 MHz.

The NFC technology facilitates effective interaction with both active and passive in the immediate vicinity. The use of NFC technology has the potential to eliminate errors to avoid adverse consequences for corporate operations. NFC technology is widely integrated into many contemporary products used by individuals in professional settings. The development of NFC technology is a result of the dedicated efforts of several firms and people. NFC has a competitive advantage over competitors due to its enhance security measures. Commercial NFC communication devices have a limited range of up to 4cm however, in this thesis, an NFC acquisition platform is designed and developed with an enhanced reading range. The NFC receiver antenna is redesigned using the online antenna design tool of STMicroelectronics to improve the reading range.

The structural component of the thesis has many key elements, including the technical background of NFC technology and the hardware components, the description of materials and methods used in the study, the presentation of results, and the conclusion of the study with future prospects.

2 TECHNICAL BACKGROUND

This chapter discusses the technical background of Radio Frequency Identification (RFID) and Near Field Communication (NFC) and the evolutionary paths and technological importance of these technologies. A brief introduction to the historical background of RFID and radar technology and its further journey to commercialization. The initial development and its transformative impact in inventory tracking and supply chain management since the introduction of this technology. It further explores the emergence of NFC, its standardization and rapid integration into mobile devices. The details of passive and active tags of NFC are discussed in detail along with their standard protocols. Furthermore, the chapter provides insight into microcontroller architectures, programming, and communication protocols like SPI and I²C, offering a comprehensive understanding of these embedded systems. The aim of this comprehensive is to overview of the historical context, technological advancements, and varied applications of RFID, NFC, microcontrollers, and communication standards.

2.1 Radio frequency identification (RFID)

In the 1940s, when radar technology first showed evidence of practical usage, the foundation of RFID was laid. German aircraft used a specific pattern of manoeuvrings during World War II to establish a secret channel of communication between the pilot and the radar operator at the base. The technique uses backscattered signals in a manner like that of widespread RFID technology to expedite the identification procedure. Mario Cardullo invented the first transponder in the 1970s, while Charles Walton introduced the first resonant frequency identification devices around the same time. The first passive RFID readers with a range of several meters appeared in the early 1970s. The RFID technology has initially seen a slow development phase because of expensive hardware for tags and scanners. Walmart, a major retailer, initially started adopting RFID technology for commercial purposes in 2003. (Zheng, and Kaiser 2016, 25)

By 2005, the top 100 suppliers had used RFID technology to identify their shipments. After that people adopted the use of RFID technology to track commodities and to enhance supply chain management. The most common applications of RFID include supply chain management, security measures, and the tracking of important assets and individuals. Improvements in antenna and microchip technology have led to the development of efficient and affordable RFID systems. However, the widespread use of RFID technology has been prevented by the persistent use of optical barcodes. The most recent advancements in RFID technology include smart antennas, ultra-wideband radios, advanced signal processing methods, and anti-collision algorithms (Zheng, and Kaiser 2016, 25). FIGURE 1 depicts the ISO protocols of RFID over the electromagnetic spectrum where ISO 14443 and ISO 15693 represent NFC communication at a high frequency of 13.56 MHz.

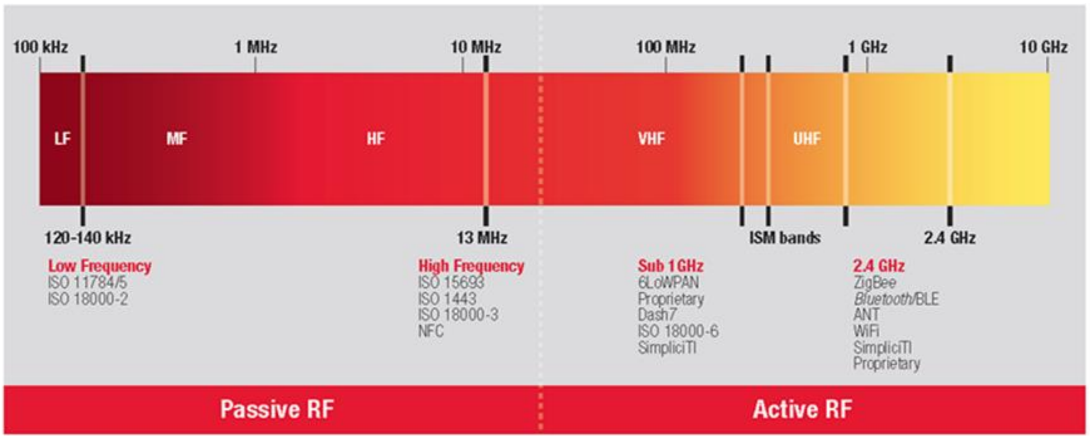


FIGURE 1. NFC frequency over RFID range on electromagnetic spectrum (C. Factory, 2023).

2.2 RFID tags

An RFID system is composed of two main components: a tag, which can also be referred to as a label or chip, and a reader. The process begins with the reader sending a query to the tag, prompting it to transmit information in response. Depending on the specific application, this action can yield various outcomes. For instance, it might result in a numerical readout on a portable device, the transmission of data to a remote backend payment system hundreds of kilometres away, or the automatic update of a point-of-sale system or inventory database. In the domain of RFID technology, the terms used to describe transponders, like tags, chips, or labels, are often interchangeable. However, it is worth noting that the term ‘chip’ typically implies a smaller unit, while ‘tag’ often denotes a larger device. Regardless of the nomenclature, the key function of these RFID devices is to facilitate efficient data communication and capture. Most labels equipped with RFID technology utilize the designator ‘label’, emphasizing their role as data carriers within the system. (Thornton, Lanthem & Thornton 2006, 13)

2.3 Optical barcodes

Barcodes have made a significant impact on industry and commerce since their creation in 1949. These are used widely across the global supply chain, from retail product procurement to warehouse inventory management, invoice accounting, advertising, healthcare patient and medication identification.

Barcodes offer a simple and adaptable solution, cost efficiency, and high accuracy. Their widespread use has had a profound impact on various domains, with diverse and far-reaching consequences. The size and resolution of barcode images are critical factors for reliable scanning, especially for image-based scanners. Ensuring a minimum-width resolution for the narrowest section is essential for successful scans. (Jurado-Verdu, Guerra, Rabadan & Perez-Jimenez 2022, 1)

2.4 Comparison between RFID-tags and optical barcodes

Barcodes and RFID represent two distinct technological paradigms. Barcodes are two dimensional visual representations of data that rely on optical devices, like barcode scanners or smartphone cameras, for scanning and reading. They require physical contact or a very short scanning range. Barcodes store limited data, primarily numeric or alphanumeric characters, and are widely employed in supply chain management, inventory control, and retail. Their accessibility is broad, as they can be scanned with various devices. However, their open visibility makes them less secure. On the other hand, RFID is a wireless communication technology operating within a very short range, typically a few centimetres, making it ideal for secure, close-proximity interactions. RFID has a broader data capacity, accommodating text, URLs, and more extensive information. It finds applications in contactless payments, access control, and data sharing between smartphones, but it requires specialized RFID chips for communication, offering a higher level of security and is particularly suited for secure transactions and access control within a limited range. The choice between barcodes and RFID depends on the specific needs and security considerations of the application at hand. (Thornton, Lanthem & Thornton 2006, 13; Jurado-Verdu, Guerra, Rabadan & Perez-Jimenez 2022, 1)

2.5 Near field communication (NFC)

Even though the NFC project was launched in 2002, the international organization for standardization (ISO) did not approve it as a standard until 2003. The first standardized technology for NFC tags was introduced in 2006. NFC tags are small and sticker-like devices that are used to carry information for NFC-enabled devices to read and write. It is possible to read the contents of the tag by presenting it to the antenna of an NFC-enabled device. The NFC tags adhere to specific protocols for NFC-enabled devices to establish communication. One of the salient features of NFC at that time was its use in smart posters which apart it from other technologies. It was possible to print digital content onto a real-world poster using NFC tags. Viewers can gain access to the digital information outfitted with NFC over portions of the poster by swiping a smartphone or other NFC device. Because it was no longer necessary to spend money printing takeaways owing to Smart Posters, the idea of their possible application in public venues like trade exhibits instantly comes to mind. (Sabella 2016, 16-17).

The Nokia 6131, unveiled in February 2006, stood out as a trailblazer in the world of smartphones by being the first to incorporate NFC technology. However, it is worth noting that not all variants of the Nokia 6131 included NFC capabilities, as this technology was still in its early stages and not universally adopted. One of the groundbreaking features of this device was its utilization of NFC tags. These tags allowed users to seamlessly transfer ownership of their phones and share data simply by tapping their devices. This innovation laid the foundation for many of the NFC applications we see today, from contactless payments to smart home automation. By 2010, when the Samsung Nexus S hit the market, NFC technology had become more widespread. The Samsung Nexus S, being NFC-enabled, showcased how far this technology had come since the release of the Nokia 6131 in 2006, with a broader range of applications and a more robust user experience. (Sabella 2016, 16-17)

2.5.1 NFC standards

The first NFC standard is NFC-A and the technology behind is NFCIP 1 and NFCIP 2 as standardized by ISO 18092. The contactless chip card having type A, parts 2, and part 3 under standard ISO 14443 was a major contribution towards the development of these standards. The NFC Forum has referred NFC-A for tags T1T, T2T, T4AT, and P2P. The second NFC standard is NFC-B which has notable resemblance to the technology used in ISO 14443 type B. Throughout history, the use of this technology has been exclusively confined to the realm of ISO NFC IP 2. The NFC-B protocol is being used for tags designated as T4BT and P2P within the NFC Forum. The NFC-F standard of NFC has resemblances to the well-recognized tags NFC-IP1 and NFC-IP2 under ISO 18092. The influential Japanese standard, JIS X6319-4, had a significant influence on the evolution of the internationally recognized ISO standards. The widespread use of NFC-F on Felica is often referred to as a peer-to-peer communication protocol. (Paret 2016, 12)

2.5.2 NFC modes

NFC Technology has various modes of operation. FIGURE 2 shows the most common modes of reader/ writer, peer-to-peer, and card emulation for NFC communication. These modes are important to establish the communication between the NFC reader and passive tags or to establish communication between two active devices. The card emulation mode helps an active device to emulate the passive NFC card. These modes are further discussed below. (Coskun, Ok &Ozdenizci 2012, 42)

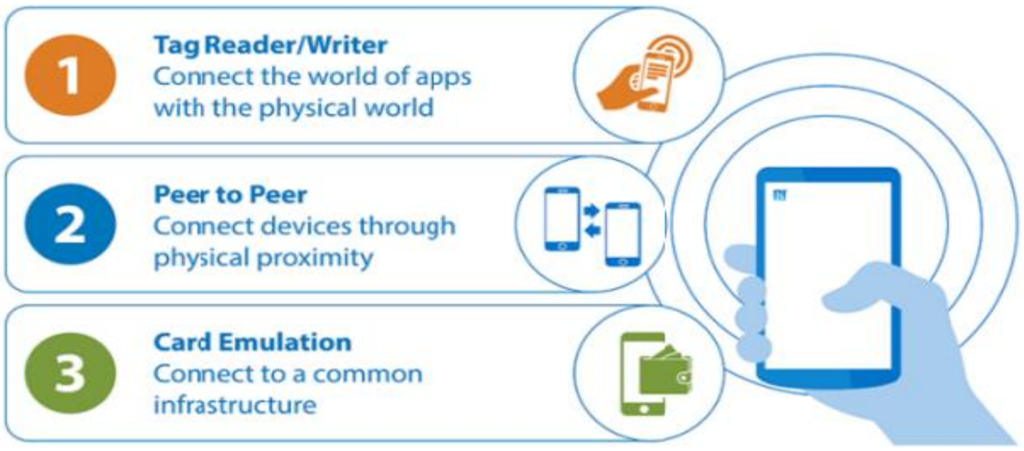

1
Tag Reader/Writer
Connect the world of apps with the physical world
2
Peer to Peer
Connect devices through physical proximity
3
Card Emulation
Connect to a common infrastructure

FIGURE 2. NFC devices operating modes (Lesas & Miranda 2016, 21).

The NFC reader/writer mode of operation is used to establish a communication channel between an NFC-enabled device and an NFC tag. An NFC device has the capability to retrieve, and store encoded data on a tag. This information has the potential to be attached to many items. The reader and writer designations represent two potential operation modes. In the context of NFC, a mobile device has the capability to either retrieve data from an NFC tag when operating in reader mode or transmit data to an NFC tag when operating in writer mode (Coskun, Ok &Ozdenizci 2012, 42). FIGURE 3 shows that the data included inside the NFC tag must adhere to the communication standard under protocols outlined by the NFC forum. It can alternatively, be represented in a format specific to a particular proprietary system.

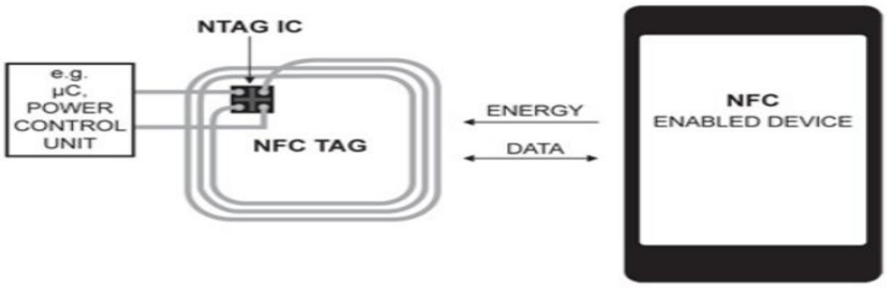


FIGURE 3. NFC tag with read-write mode operation (Lesas & Miranda 2017, 22).

When the NFC devices are configured in peer-to-peer mode, one device acts as the initiator, and the other device acts as the target. In this operation mode, two NFC-enabled devices can establish communication and exchange data. This is facilitated by a dedicated power supply integrated into each device. In this operational mode, while one device is engaged in broadcasting, the second device is obligated to receiving mode. After the completion of the broadcasting process by the first device, the second device may then start its transmission. This mode of operation designates one device as the initiator while the other device is configured as the target. The initiator device is responsible for initiating the request and the communication, while the target device receives the request from the initiator to receive the data and replies accordingly. (Coskun, Ok & Ozdenizci 2012, 42).

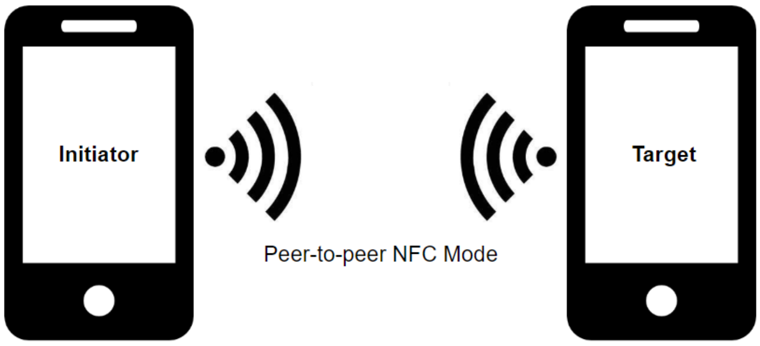


FIGURE 4. The NFC peer-to-peer communication mode with initiator and receiver.

In card emulation mode, an NFC-enabled mobile device can serve as a substitute for a traditional credit card (NFC tag). Smart cards are designed based on existing card models, including credit cards, debit cards, and loyalty cards for contactless transactions. Tens of thousands of mobile applications are available for download on NFC-enabled smartphones to execute the card emulation mode. The existing smart card infrastructure has the capability to provide card emulation for payment transactions. When an NFC device operates in card emulation mode, it assumes the role of an active NFC target. When it comes into proximity with an NFC reader, it detects the reader and emulates the card reading mode. To guarantee the integrity of data and safeguard sensitive information, NFC-enabled devices use a secure element for the purpose of storing and overseeing the data during the card emulation mode (Coskun, Ok & Ozdenizci 2012, 42). FIGURE 5 shows an NFC-enabled mobile device emulating a credit card for the teller machine for payments.

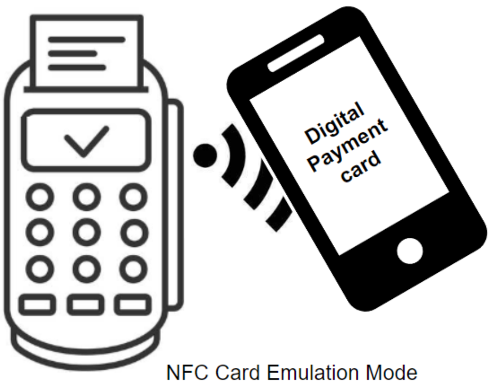


FIGURE 5. The NFC-enabled device uses card emulation mode for payment transactions.

2.5.3 NFC equipment

The memory capacity of a chip in an NFC tag is usually limited to one kilobyte in the peripheral devices. An NFC controller facilitates the interaction between the NFC chip and the operating system of the host device. A fundamental component of an NFC device comprises a of minimum an antenna that is connected to a modulator-demodulator. This modulator-demodulator is responsible for the conversion of electromagnetic signals into digital data, as well as the reverse process of converting digital data into electromagnetic signals. (Lesas & Miranda 2017, 7)

2.5.4 NFC tags

The NFC tag may have many different types of physical appearances like stickers, key chains, and bracelets. These entities share similarities with QR codes, but they differ significantly in their usability. Unlike QR codes, for which initiation of the required software is required before scanning, these entities only require a simple ‘tap’ to be scanned. NFC tags serve as a bridge between the digital and physical domains, possessing the ability to activate or deactivate various functionalities, including alarms, devices, and digital processes. With a smartphone, a mere tap on the screen effortlessly triggers any of these functions (Lesas & Miranda 2017, 7).

NFC tags have an immutable universal unique identity supplied to them during production. In this way, a connected object may be readily differentiated based on the universal unique identity of the NFC tag to which it is attached. FIGURE 6 shows an NFC tag that is a transparent NXP-made sticker that is part of the NTAG family. Depending on the application's use case, a terminal application that is coupled with an NFC reader may get the universal unique identity of an item. Examples of these include electronic passports, identification cards, and tourist information displays. Furthermore, the application can perform actions like making a reservation, kicking off events in the virtual or physical world, and recording those events by appending pertinent contextual data for reference or tracking purposes. (Lesas & Miranda, 2017)

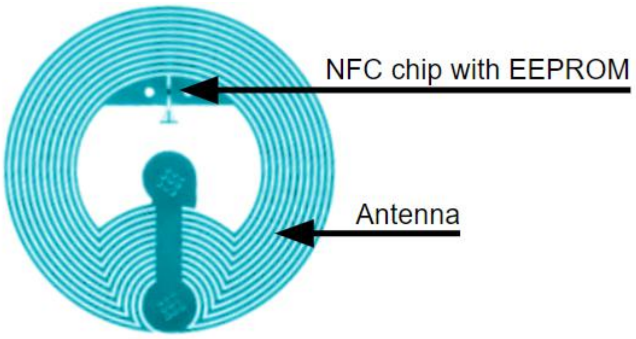


FIGURE 6. The NFC passive tag with an antenna (Lesas & Miranda, 2017).

2.6 Microcontroller

Microcontrollers are autonomous computing devices that are integrated into larger systems. The term 'embedded' refers to the idea of integrating a computer or device directly into specific equipment, which means it's not easily accessible for external purposes. Everyday household appliances like microwaves and television remotes have tiny computers inside, called microcontrollers. These microcontrollers follow instructions based on the inputs they receive and produce outputs tailored to the specific job of the appliance. While microcontrollers can be considered computer systems, they have limited memory because they are meant to do very specific tasks. Unlike regular computers, which follow the Von Neumann architecture with separate data buses for instructions and data storage, microcontrollers use the Harvard architecture. This design helps them perform their specialized functions efficiently (Westcott & Westcott 2023, 216). FIGURE 7 shows the importance of observing that Harvard architecture is characterized by the presence of two distinct memory regions, namely one dedicated to storing instructions and another specifically designated for data storage.

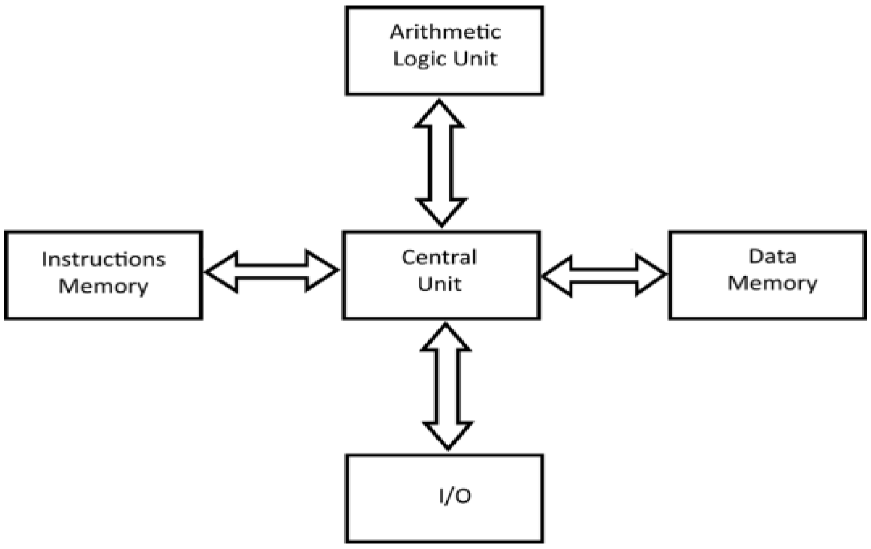


FIGURE 7. Microcontroller Harvard architecture (Westcott & Westcott 2023, 216).

Programmers may strategically use distinct memory regions by implementing address schemes for both the data memory and the instruction memory. For example, they may use a 16-bit approach for one and an 8-bit method for the other. In regular computer systems, it is important to stick to the same method for both instruction and data memory. However, having two different methods gives microcontroller designers some flexibility. They can put super-fast machine-level code in one memory area for the core function of the microcontroller. Meanwhile, they can use more user-friendly tools in other areas for giving instructions. (Westcott & Westcott 2023, 216)

2.6.1 Microcontroller construction

Microcontroller architecture refers to the fundamental design and structure of a microcontroller, which includes the organization of its central processing unit (CPU), memory, input/output ports, and other essential components that determine its functionality and capabilities. FIGURE 8 depicts how various microcontroller parts are combined onto one processor chip, and this is becoming increasingly common because of ongoing technological progress. This setup allows the microcontroller to work on its own as a self-contained unit. However, sometimes external modules are connected to meet the requirements of a specific application. For instance, an extra memory can be added externally. The external and internal memories are usually discussed separately. (Zurawka, Schaeuffele & Carey 2016, 53)

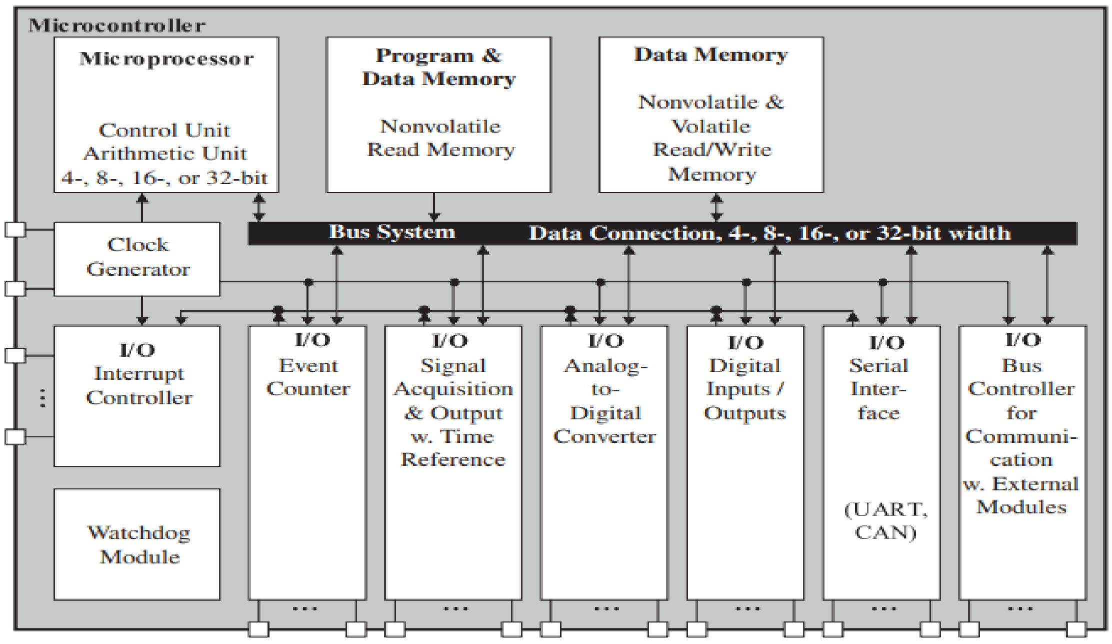

Microcontroller
Microprocessor
Control Unit
Arithmetic Unit
4-, 8-, 16-, or 32-bit
Program & Data Memory
Nonvolatile Read Memory
Data Memory
Nonvolatile & Volatile Read/Write Memory
Bus System
Data Connection, 4-, 8-, 16-, or 32-bit width
Clock Generator
I/O Interrupt Controller
Watchdog Module
I/O Event Counter
I/O Signal Acquisition & Output w. Time Reference
I/O Analog-to-Digital Converter
I/O Digital Inputs / Outputs
I/O Serial Inter-face
(UART, CAN)
I/O Bus Controller for Communi-cation w. External Modules

FIGURE 8. Microcontroller architecture (Zurawka, Schaeuffele & Carey 2016, 53).

2.6.2 Microprocessor

The microprocessor serves as the central processing unit (CPU) within a computing system. It comprises two integral components: the control unit and the arithmetic logic unit (ALU). The ALU is responsible for executing arithmetic and logical operations, whereas the control unit ensures the accurate execution of instructions retrieved from program memory. This functional segmentation streamlines adaptability to diverse real-world applications through the implementation of context-appropriate programming techniques. (Zurawka, Schaeuffele & Carey 2016, 52)

2.6.3 Input and output units

I/O modules assume the responsibility of overseeing the data exchange between CPU and peripheral devices. These elements encompass various constituents, including I/O device circuits tasked with handling program interruptions, as well as bus systems that play a pivotal role in establishing connections with other control devices, such as the Controller Area Network (CAN). (Zurawka, Schaeuffele & Carey 2016, 52).

2.6.4 Program and Data Memory

The microcontroller comprises two crucial components program memory and data memory. Program memory, often referred to as instruction memory which is responsible for storing the instructions executed by the computer's processor. These instructions are typically stored in non-volatile memory, which possesses the characteristic of data retention even in the absence of power. This non-volatile memory is well-suited for preserving control methods, including open-loop and closed-loop control algorithms, as well as constant parameter sets for both types of control. The organization of the program and its associated parameters often follows a specific pattern, with sets of parameters stored in distinct memory sections. Consequently, the terms ‘program’ and ‘data’ and their utilization in memory management are important in this context. (Zurawka, Schaeuffele & Carey 2016, 52)

2.6.5 Data memory and other components

Data memory is the designated area where all data modified during program execution is stored. Often referred to as RAM (Random Access Memory), this memory section is chosen for its unique qualities, particularly its capability to both read and write data. Memory technologies that support both read and write operations are most suitable for this purpose. It's important to note that such memory can be either volatile or non-volatile, with the choice contingent on the application's requirements. The bus system, serving as a vital communication network, interconnects various microprocessor components within the microcontroller. Additionally, the microcontroller includes a clock generator, also known as an oscillator, responsible for ensuring that all internal actions within the microcontroller adhere to a synchronized clock frequency. Furthermore, a watchdog module, which encompasses a set of monitoring functions, is employed to vigilantly observe and maintain situational awareness within the system. (Zurawka, Schaeuffele & Carey 2016, 52)

2.6.6 Microcontroller programming

To ensure the smooth operation of microcontrollers, it is imperative to provide them with instructions that the CPU can understand. Figure 9 shows the hierarchy of programming languages. Machine code, also known as machine language which consists of binary digits, specifically zeros and ones and plays a fundamental role in issuing commands in computing systems. Given the inherent intricacy of machine language, assembly language acts as an intermediary tool for translating elementary instructions into machine code. Assembly language is widely recognized as a low-level computer language, operating closer to the hardware. Low-level languages, like assembly language, offer a broader applicability across various computer environments, but they involve a somewhat higher level of programming complexity. Typically, programmers first work with high-level programming languages and then employ a compiler to convert the source code from the high-level language into assembly language, which the microcontroller can process. (Westcott & Westcott 2023, 219)

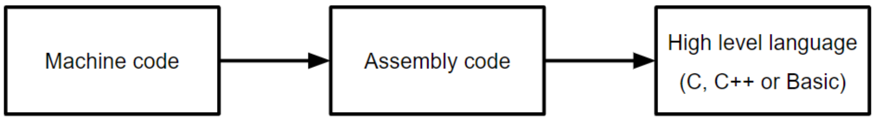


FIGURE 9. Hierarchy of programming languages (Westcott & Westcott 2023, 218).

FIGURE 9 explains that when software developers engage in discussions about code, a set of instructions is specifically addressed within a program, rather than focusing entirely on machine code. Coding is the process of creating these instructions. Source code refers to a collection of instructions expressed in a particular programming language. It is important to note that source code is not written in a language easily understood by end users. Many microcontrollers can be programmed using computer languages designed to be user-friendly and operate at a high level. Examples of such languages include the C programming language family, which encompasses C, C++, and C#, as well as their various dialects, with Visual Basic language being particularly prominent. Dialects in computer languages can be likened to dialects in natural languages, as they encompass variations within the language across different regions and social groups. The characteristics of the programming environment and the intended use of a programming language are reflected as dialects. (Westcott & Westcott 2023, 219)

2.6.7 Arduino microcontroller platform

Massimo Banzi and David Cuartielles, from Italy, have jointly developed the Arduino platform. This innovative microcontroller was designed with cost-efficiency and user-friendliness in mind, enabling hobbyists and students to create and experiment with prototypes for embedded devices. Users can build and test devices they conceive or embark on projects inspired by the designs of others. The name ‘Arduino’ was bestowed upon the platform in homage to a popular pub in Ivrea, Italy. Its CPU is powered by an Atmel microcontroller, and its programming language is based on C++. Notably, Arduino is equipped with a bootloader, simplifying the process of uploading data to its flash memory. Additionally, the bootloader is optimized to seamlessly interact with various shields, also known as daughterboards, which can be easily attached directly to the CPU, enhancing its versatility and functionality. (Westcott & Westcott 2023, 221)

Peripheral circuit boards can connect directly to the CPU motherboard with the help of various communication protocols. Some peripherals link to the primary circuit board without requiring cables or computer buses because it is an integrated part of the board as compared to other components. The Arduino platform is famous for its cross-platform compatibility which makes it versatile and suitable for a variety of operating systems such as Windows, Linux, and Mac. The Arduino microcontroller has various development hardware kits available in the market these days. The Arduino is an open-source platform and is available for research community to create innovative projects. (Westcott & Westcott 2023, 221)

2.6.8 The Net Duino microcontroller platform

Following the principles of open hardware, the Net Duino microcontroller was developed. While its primary focus was to help the hobbyists and beginners, it is worth noting that Net Duino can also be effectively employed for prototyping business projects. This showcases its versatility in accommodating a wide range of applications. The Net Duino operates within the NET Micro Framework, which is essentially a version of Microsoft's .NET framework tailored for low-memory embedded devices. This framework serves as the foundational operating environment for the Net Duino system, providing the necessary tools and infrastructure for its operation. (Westcott & Westcott 2023, 221)

2.7 Communication Protocols

The communication standards are a crucial part of the embedded systems. These standards ensure the communication interfaces of embedded systems with the peripherals and other microcontrollers. The programmable input output pins are an important part of the microcontroller which allows them to engage in communication with various peripherals using specific protocols. Serial communication protocols such as universal asynchronous receiver and transmitter (UART), the serial peripheral interface (SPI) bus, the inter-integrated communication (I^2C) bus, and universal serial bus (USB) are prominent communication interfaces used for microcontrollers. These interfaces ensure efficient communication within embedded systems. (Daniele 2018, 16)

2.7.1 Serial Peripheral Interface (SPI)

The SPI technology was introduced in the late 1980s firstly with the aim of asynchronous serial communication with peripherals. It introduced a range of improvements to the communication process. A main component of SPI is the use of a serial clock line (SCL), which plays a crucial role in synchronizing the communication between endpoints. The SPI protocol operates on a master-slave system, where one device acts as the master, retaining control over the communication. In typical configurations, a shared three-wire bus facilitates one-to-many communication. The primary device, often referred to as the microcontroller, shares this communication bus with one or more subordinate devices, known as slaves. To initiate communication with a particular slave device, a distinct slave select signal is employed to designate and address that specific device connected to the bus. The bus employs a pair of independent signals for data transfer, each signal being dedicated to a specific direction. (Daniele 2018, 17.)

Furthermore, SPI utilizes a single clock line to synchronize the two endpoints of the connection. This clock line, generated by the master, significantly enhances the reliability of data transmission, allowing for higher bit rates compared to traditional UART communication. One of the key reasons behind SPI's continued popularity across multiple generations of microcontrollers is its simplicity in designing slave devices. The design of slave devices can be as straightforward as implementing a single shift register, making it a versatile and efficient choice. SPI finds application in a wide range of electronic devices, including sensor devices, LCD displays, flash memory controllers, and network interfaces, underscoring its versatility and utility in diverse technologies (Daniele 2018, 17). FIGURE 10 shows the SPI communication interface developed between master and slave devices. Master sets its clock frequency to a value lower than or equal to the maximum frequency supported by the slave device before sending data.

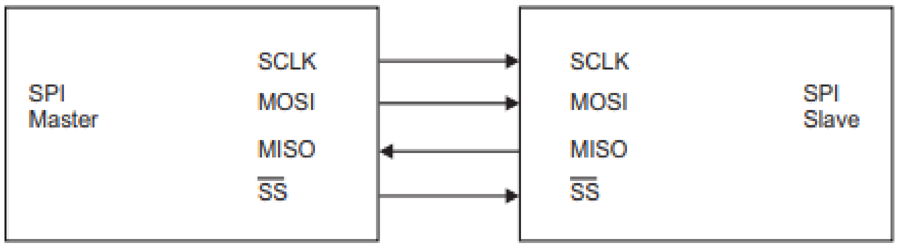


FIGURE 10. The SPI communication protocol between master and slave devices (Kothari 2013, 143).

Commonly used clock ranges are around 170 MHz, and the master selects the specific chip to communicate with by enabling the corresponding slave. The master must temporarily halt the issuance of clock cycles during the transition from analog to digital (ADC) processing. Each SPI clock cycle enables one-way data transfer in full duplex mode. The master out slave in (MOSI) line carries data from the master to the slave, while the master in slave out (MISO) line allows the master to read data transmitted by the slave. Notably, a transmission can occur in fewer than four steps. During these transmissions, two shift registers are utilized, each with the same word size, typically eight bits – one belonging to the master and the other to the slave. The least significant bit is the first to leave the register, followed by the most significant bit. When a register is released, the values in the master and slave registers are swapped and then stored in the device's memory. As further information is sent, the shift registers are updated, and the process continues. (Kothari 2013, 143-144.)

There can be some variability in the transmission clock cycles in the SPI communication. When there has been no contact for a while, the master typically ceases its clock toggling, effectively deactivating the slave. If needed, a master may initiate multiple 8-bit data phrase transmissions. Some devices, like the Texas Instruments TSC2101, employ 16-bit words in their digital-to-analog (DAC) and ADC. Slaves on the bus that lack a dedicated slave line are not expected to respond to the MOSI, MISO, or clock signals. In SPI, a master can communicate with only one slave at a time. If the clock pulses in an SPI are too fast, certain slave devices may disregard them. For instance, when accessing a digital integrated circuit (IC) scan chain through SPI, a 32-bit command word is sent, and a 153-bit response is received, with one bit corresponding to each pin. SPI devices also have the capability to interrupt a host CPU through a separate signal line. (Kothari 2013, 143-144.)

2.7.2 Inter-integrated Circuit (I^{2}C)

The I^{2}C protocol is a serial communication bus designed for short-distance communication, supporting multiple masters operating at low bandwidth. Its applications have expanded beyond individual circuit boards and are now used to connect low-speed peripheral devices in various systems, including motherboards, embedded systems, and cellular devices. The newer iterations of the I^{2}C protocol offer enhanced functionalities and data transfer rates, making it attractive for consumer and automotive electronics. The fundamental architecture of I^{2}C involves a 7-bit address space, with 16 addresses reserved, limiting the total number of nodes that can communicate on a single bus. (Kothari 2013, 139.)

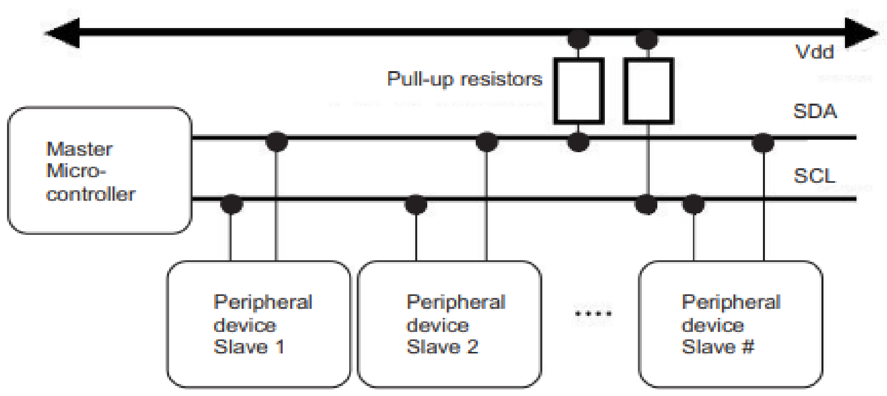

Vdd
Pull-up resistors
SDA
SCL
Master Micro-controller
Peripheral device Slave 1
Peripheral device Slave 2
....
Peripheral device Slave #

FIGURE 11. The I²C bus connection between the master and several peripherals (Kothari 2013, 139).

Each I²C device has a unique 7-bit address, the maximum number of nodes is determined by the address space and the total bus capacitance, which is limited to 400pF. The bidirectional communication between devices on the bus is facilitated by two lines SDA, responsible for data transmission and SCL, which carries the clock signal for synchronization. Typically, voltage levels of +5 V or +3.3 V are used. FIGURE 11 illustrates the master-slave communication protocol used in the Controller Area Network, Local Interconnect Network, and the I²C protocol. However, the I²C bus supports multiple masters, allowing more than one integrated circuit or device to initiate data transfers. The device initiating the connection is known as the 'Master', while the one receiving instructions from the 'Master' is referred to as the 'Slave'. Each master device generates its clock signals during data transfers, as clock signal generation is always performed by the master. (Kothari 2013, 139.)

3 MATERIALS AND METHODS

This chapter discusses a comprehensive examination of the resources used to develop the hardware of the NFC acquisition platform. It comprises the methodologies used to attain the desired goals for both hardware and firmware development. This chapter is divided into hardware and software sections. The hardware section contains the use of off-the-shelf components including an ATmega328P microcontroller, NXP PN532 chip as the NFC controller, and corresponding antenna circuit. Additionally, an SPI interface, a UART to USB converter circuit, and a power management circuit are also included. The software component comprises the bootloader, the algorithm of the code, the flowchart of the code, the NFC peer-to-peer configuration code for the target, and the necessary libraries.

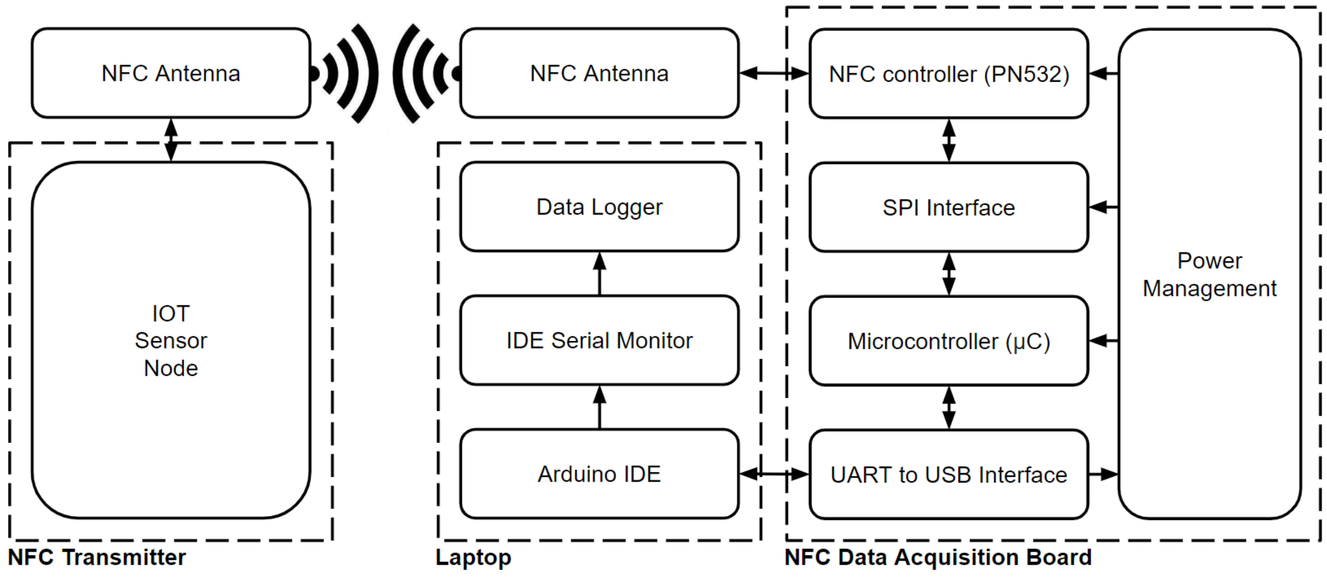


FIGURE 12. Block diagram of NFC data acquisition platform.

The block diagram presented in FIGURE 12 offers an insightful view of the NFC data acquisition platform. It showcases the intricate interplay between the NFC hardware, the laptop interface, and the IoT sensor node. At the core of this platform, the NFC data acquisition board is equipped with a robust power management circuit, ensuring a steady supply of regulated voltages to all its components. The integration of decoupling capacitors is a crucial element in maintaining the stability of voltage levels across the board. A standardized voltage level of 3.3 V is adopted to energize all the hardware components effectively, optimizing their performance. For a comprehensive understanding of the NFC data acquisition board, it is important to examine the complex details of both its hardware and software, which will be explored further in the forthcoming sections. This platform facilitates the seamless communication between various elements of the NFC ecosystem, enabling efficient data acquisition and transmission.

3.1 Hardware design

The hardware design of the NFC-based data acquisition platform includes the selection of various off-the-shelf hardware components suitable for the operation. The design and development of the required circuitry to ensure the operations of each sub-block and the development of the communication interfaces to establish the control and data signal exchange. FIGURE 13 shows the front and back sides of the NFC data acquisition board with all its components and interfaces. The hardware includes the microcontroller, NFC controller, NFC antenna design, power management, programming interface, debugging light-emitting diodes (LEDs), and the required communication interfaces. The schematics and the printed circuit board (PCB) are designed in the Altium PCB designer software under the lab license of Aalto University. The PCB boards are fabricated by the commercial manufacturer. The assembly, soldering, and testing are performed in the circuit development labs of Aalto University. The details of all these components are further discussed in the subsections.

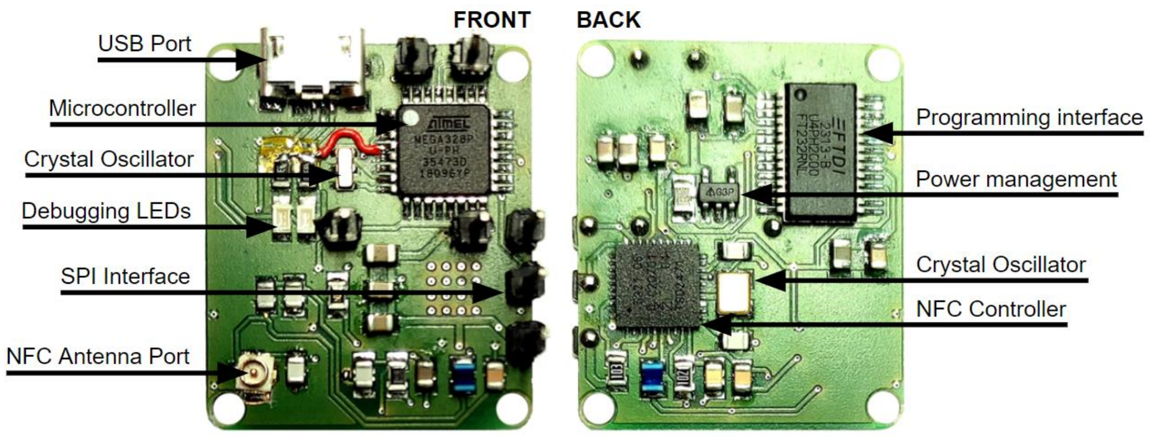


FIGURE 13. The front and back side of the NFC data acquisition board.

3.1.1 Microcontroller

The ATmega328P by Microchip is selected as a fundamental block for the NFC data acquisition board because it is commonly used for Arduino-based hardware platforms and embedded system applications. It has 32 KB flash memory and 1024B electrically erasable programmable read-only memory (EEPROM), SPI, UART, and I2C interface, and a wide variety of analog and digital input-output (I/O) pins. A thin quad flat pack (TQFP) is selected for this work because of its ease of handling and soldering on area-optimized PCBs. FIGURE 14 illustrates the circuit diagram of the microcontroller denoted as the ATmega328P. An external crystal with a frequency of 8 MHz is used to provide a reliable clock to the microcontroller for operations on a 3.3 V level. PINs 4, 6, and 18 facilitate the input supply for the system. Decoupling capacitors are used to stabilize the supply voltages. The SPI interface of the ATmega328P is used to establish the communication interface for the NXP PN532 NFC controller chip. The UART interface is converted to USB for the laptop interface to program the microcontroller and monitor and record the received data on the serial port. The digital I/O ports are used to establish the control for SPI and burn the bootloader to the microcontroller.

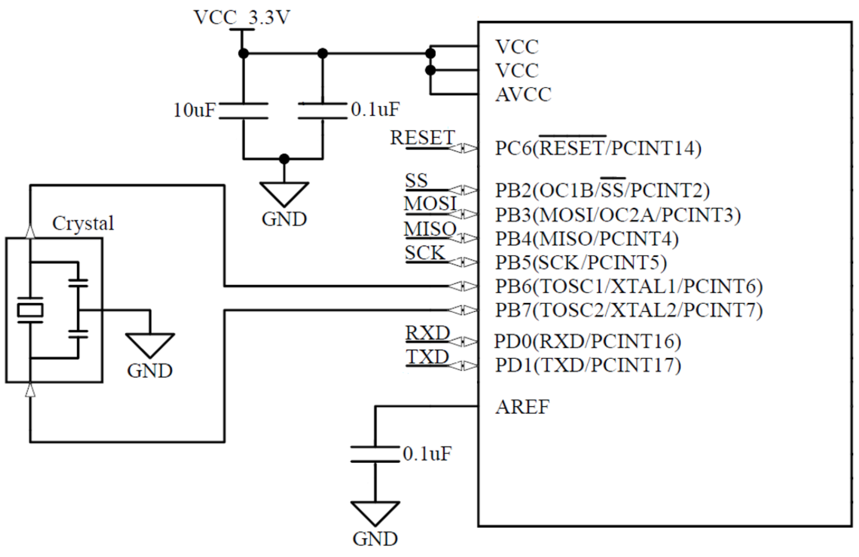


FIGURE 14. The circuit diagram of microcontroller ATmega328P for NFC data acquisition board.

3.1.2 Voltage Regulator

The power management circuit is designed to power all the components of the NFC data acquisition board. The voltage level of 3.3 V has been adapted to ensure the operation of digital data transactions among various communication protocols. The wire power of 5 V is taken from the USB port of the laptop, and it is regulated to 3.3 V to power the microcontroller and NFC controller. A low drop-out (LDO) voltage regulator AP2112 by Diodes Inc. is employed to regulate the input supply voltage to a fixed output of 3.3 V. The LDO regulator can provide 600 mA with dropout voltages of only 250 mV. Because of its capacity to produce a consistent and controlled voltage supply, it is the most suitable choice to power the NFC data acquisition board. Because the voltage dropout is minimal, Quiescent current which is also known as ‘standby current’, is low, which is useful for applications that need to save energy since it allows for greater efficiency. Since it has a maximum output current of around 600 mA, it is suitable for a wide variety of applications that need either low or medium levels of power. The output voltage is accurately regulated, and there is only a small degree of change in it. The system is equipped with a safety function that will switch it off automatically if it becomes extremely hot, protecting the user from any potential injuries. FIGURE 15 illustrates the circuit diagram of the voltage regulator circuit.

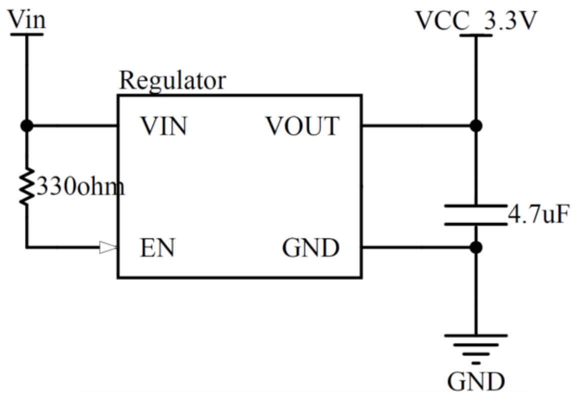


FIGURE 15. The circuit diagram of the LDO voltage regulator.

3.1.3 NFC controller

A versatile NFC controller integrated circuit chip PN532 by NXP can support a wide range of NFC forum protocols. These protocols are necessary to establish communication with various NFC tag types. The NFC controller PN532 is chosen for the NFC data acquisition board because of its precise performance across a range of interfaces, including SPI, I2C, and UART, among others. It ensures that two devices can communicate securely manner with each other. It is capable of peer-to-peer connectivity, a card emulation mode, and a read-write mode. These communication modes are used to test and debug the firmware during the development process of the NFC-based data acquisition platform. The read range of commercially available NFC antennas is from 4 cm to 6 cm with a PN532 NFC controller. FIGURE 16 illustrates the schematic diagram of the PN532 chip with its support circuitry and communication interface. The operational functionality of the system requires the use of an external crystal oscillator to generate a frequency of 27.12 MHz.The decoupling capacitors have been used to provide a steady and uninterrupted supply voltage. The functioning of this device is configured on a 3.3 V input supply voltage to have the same digital level as the microcontroller. The SPI communication interface is employed to establish communication between the NFC controller and the microcontroller.

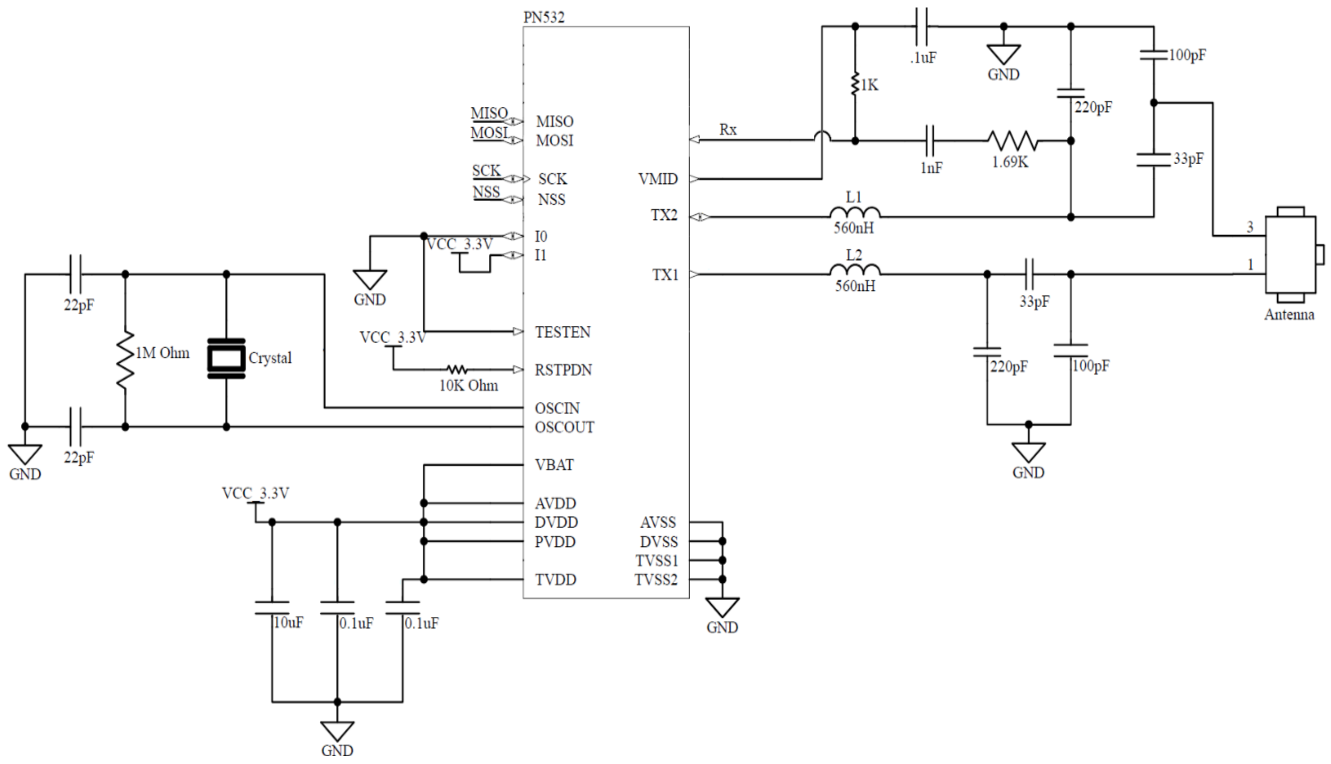
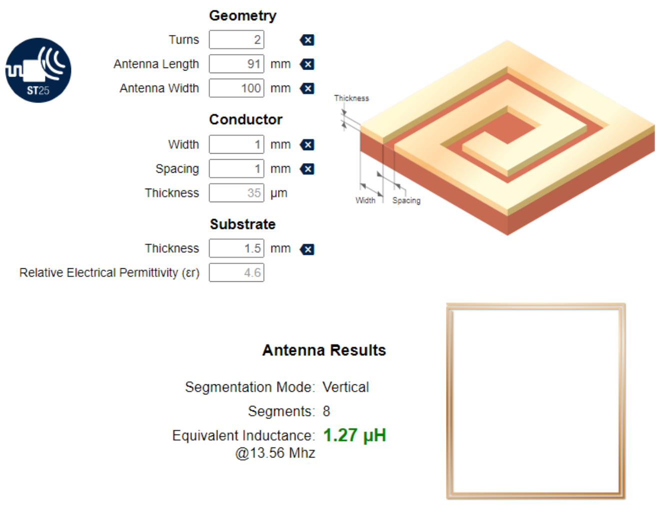


FIGURE 16. The schematic of the NFC controller circuit and matching network for the antenna.

An antenna matching circuit is designed to achieve impedance matching for optimal power transfer from the PN532 chip to the NFC antenna. The matching circuit comprises a combination of inductors, capacitors, and resistors, specifically designed based on the specifications of the antenna. These specifications are dependent on the intended purpose of the antenna. Successful impedance matching through this circuit contributes to the establishment of a stable communication system and enables enhanced distance for peer-to-peer communication of the NFC data acquisition platform. The Hirose uFL connectors are used in matching networks for the antenna connections.

3.1.4 Antenna design

The antenna design tool of STMicroelectronics (NFC-Antenna, 2023) is used to design the custom antennas for the NFC data acquisition platform. FIGURE 17 shows the antenna design parameters such as geometry parameters, conductor parameters and substrate parameters inside the design tool of STMicroelectronics. TABLE 1 shows the design parameters of five different loop antennas designed at the frequency of 13.56 MHz which has been adopted according to the matching network by (S. Hameed, 2023). The antenna geometry, conductor width, spacing is changed and the conductor thickness, substrate thickness and relative permittivity is kept same for all antennas. The substrate thickness of the FR04 PCB board is 1.5 mm and the relative permittivity is 4.6.



Geometry

Turns 2
Antenna Length 91 mm
Antenna Width 100 mm

Conductor

Width 1 mm
Spacing 1 mm
Thickness 35 µm

Substrate

Thickness 1.5 mm
Relative Electrical Permittivity (εr) 4.6

Antenna Results

Segmentation Mode: Vertical
Segments: 8
Equivalent Inductance: 1.27 µH
@13.56 Mhz

FIGURE 17. The designed antenna for the NFC data acquisition platform (NFC Inductance, 2023).

Advanced Design System (ADS) is a versatile electronic design tool, commonly used in engineering to design and simulate high-frequency circuits and systems. The ADS is employed to create an NFC antenna for the NFC data acquisition platform. The design parameters obtained from the online NFC antenna design tool of STMicroelectronics are used in ADS to visualize the physical layout of the NFC antenna, simulate its performance, and fine-tune its characteristics for optimal functionality. The NFC antennas are designed and fabricated inside Aalto University labs using a PCB milling machine (S63, LPKF ProtoMat). This integrated design process ensures that the antenna meets the specific requirements of the NFC communication protocols, enabling effective wireless communication between devices without data loss.

TABLE 1. Design parameters of the NFC antennas.

Antenna Name	**No. of Turns**	**Length x Width (cm)**	**Conductor Width (mm)**	**Spacing (mm)**	**Inductance (µH)**
Ant_2x2	8	2 x 2	0.20	0.49	1.47
Ant_3x3	6	3 x 3	0.50	0.60	1.51
Ant_4x4	7	4 x 4	1.12	1.00	1.55
Ant_8x4	4	8 x 4	1.60	1.00	1.50
Ant_12x8	4	12 x 8	6.00	2.0	1.50

3.1.5 SPI interface

The SPI interface is employed to establish the communication between the microcontroller and the NFC controller of the NFC data acquisition board. The SPI interface is also used to burn the Arduino bootloader using an in-circuit serial programmer (ISP) with Arduino's integrated development environment (IDE). The SPI interface is chosen because of its primary benefit, which is high data transmission with minimal latency. SPI is widely used for exchanging data between devices over a small distance, where the data transfer must take place promptly and efficiently. This interface makes use of the pins designated as serial clock (SCK), master out slave in (MOSI), master in slave out (MISO), and chip select (CS). The SPI interface bus is pulled up to a 3.3 V level to establish digital communication on the same voltage level.

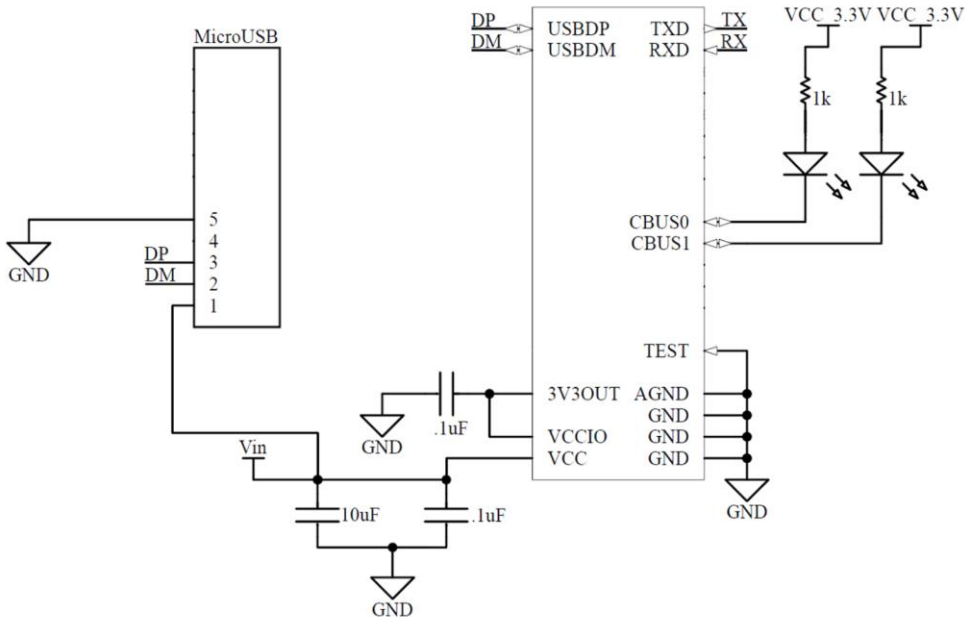


FIGURE 18. USB to UART converter to establish communication interface between NFC data acquisition platform and laptop.

3.1.6 UART to USB interface

The NFC data acquisition board is equipped with ATmega328P which supports the UART communication interface. The UART interface is employed to establish the USB interface with a laptop. The USB to UART converter FT232RL by FTDI chip is used to establish the UART to USB interface between the laptop and microcontroller of the NFC data acquisition platform. It makes it easy to integrate USB connections into a range of apps, hence facilitating communication as well as programming and debugging. The FT232RL has an integrated level converter which is used to convert the data lever of USB from 5 V to 3.3 V for the UART interface of the microcontroller. FIGURE 19 shows the circuit diagram of the USB to UART converter protocol used to program the microcontroller and acquire data while using a laptop. The decoupling capacitors are used to ensure the stability of the power supply whereas LEDs provide a visual indication of data transfer.

3.2 Software design

Software plays an important role in the development of embedded systems. Therefore Arduino-based firmware is developed for the NFC data acquisition platform. This software serves as the bridge that enables the hardware platform to function efficiently for data collection from the IoT sensor node. The primary objective of the software is to program the NFC data acquisition platform, configure its communication ports, conduct rigorous testing, and facilitate code debugging. To achieve this, a systematic approach is taken, starting with the creation of an algorithm and flowchart that outlines the operational logic of the software.

The key function of the developed software is to facilitate the seamless reception and processing of data from the IoT sensor node. This communication is achieved through a peer-to-peer mode that establishes a direct link between the IoT sensor node and the NFC data acquisition platform. The exchange of data, between the NFC controller and microcontroller, is facilitated over the SPI communication interface by ensuring fast and reliable data transfer. Received data, accompanied by timestamp and sensor information, is subsequently reorganized within the software, optimizing its utility. The data is then restructured by the microcontroller and sent to the serial port, where it is stored in a text file for further analysis and processing. In essence, the software plays an important role in the data acquisition process, enhancing the platform's functionality and contributing to the efficiency of data collection and analysis. A Windows-based application, CoolTerm (Meier, 2023) is employed to collect and record data from the microcontroller of the NFC data acquisition board connected to the serial port. The data from these text files may be retrieved by analysis tools like MATLAB to further process and plot the results.

3.2.1 Algorithm of firmware

Algorithms are step-by-step sets of instructions or procedures designed to perform a particular task. In the development of firmware code for microcontrollers, algorithms are vital. They outline the precise sequence of actions for microcontrollers, required to achieve the desired functionality. The algorithm ensures that the code is organized, efficient, and capable of handling complex tasks. Without well-defined algorithms, firmware development would be prone to errors. They are the building blocks of microcontroller code to interact with hardware components, process data, and respond to various inputs efficiently. The algorithm steps are attached in appendix 3 for the microcontroller which provide a structured framework for the microcontroller to execute the required tasks, ensuring that data from IoT sensor nodes is efficiently acquired, organized, and transmitted for further analysis.

3.2.2 Flowchart design

Flowcharts are valuable tools in software development as they provide a visual representation of complex algorithms and processes, making it easier to develop the code, understand the process, analyze the system, and convey the workflow. Similarly, in the context of the NFC data acquisition platform, the flowchart plays a critical role in translating the algorithm into a visual guide. FIGURE 20 represents the flowchart for the NFC data acquisition platform which outlines a structured sequence of actions, beginning with initialization and library inclusion. It proceeds further through various checks and configurations and finally forms a loop for continuous operation. Each symbol represents a specific task or decision, enhancing the clarity of the operation to facilitate debugging and maintenance. It also ensures the reliable functionality of the NFC data acquisition system.

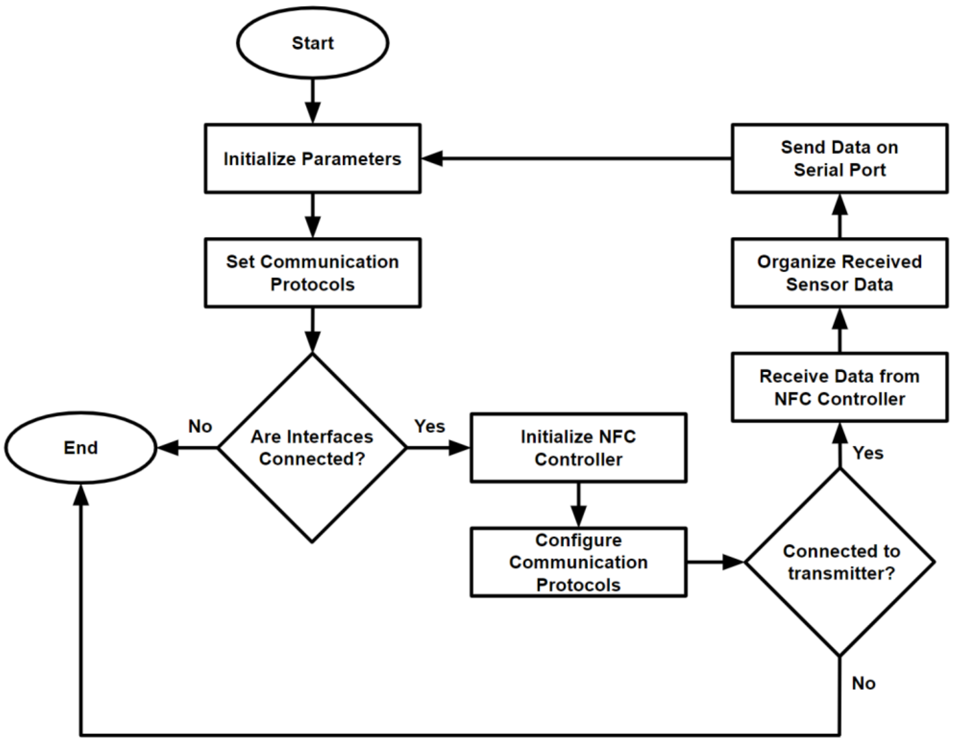

Start
Initialize Parameters
Send Data on Serial Port
Set Communication Protocols
Organize Received Sensor Data
Receive Data from NFC Controller
End
No
Are Interfaces Connected?
Yes
Initialize NFC Controller
Yes
Configure Communication Protocols
Connected to transmitter?
No

FIGURE 19. The flowchart represents the algorithm of the NFC data acquisition platform.

3.2.3 Firmware code

The firmware code for the NFC data acquisition platform is a critical component that enables the platform to interface with the IoT sensor node to collect data. In this code, the use of libraries plays a pivotal role in simplifying the development process. The two primary libraries employed, `PN532.h` and `SPI.h`, provide essential functionalities for NFC communication and SPI communication. The `PN532.h` library (Seeeds, 2023) is crucial as it offers a high-level interface for PN532 NFC modules. This simplifies the complex low-level communication protocols required for NFC devices, making it accessible to developers.

```
#include <PN532.h>
#include <SPI.h>
#define PN532_CS 10
PN532 nfc(PN532_CS);
#define  NFC_DEMO_DEBUG 1
```

Firmware Code 1. The definition of firmware libraries and classes.

The `#define` directive is used to specify the Chip Select (CS) pin for the PN532 module, typically set to digital I/O (pin number 10) of the microcontroller. The `PN532` class is instantiated with the specified CS pin. The `#define NFC_DEMO_DEBUG 1` line enables or disables debugging features, making it easier to troubleshoot. These libraries and the associated code facilitate efficient communication with NFC devices, providing a strong foundation for the development of a data acquisition platform. They abstract low-level hardware intricacies and empower developers to focus on implementing the core functionality of the thesis work, ensuring the reliable and secure acquisition of NFC data.

```
void setup(void) {
#ifdef NFC_DEMO_DEBUG
  Serial.begin(9600);
  Serial.println("Hello!");
#endif
  nfc.begin();
  uint32_t versiondata = nfc.getFirmwareVersion();
  if (! versiondata) {
#ifdef NFC_DEMO_DEBUG
  Serial.print("Didn't find PN53x board");
#endif
  while (1);}
#ifdef NFC_DEMO_DEBUG
  Serial.print("Found chip PN5");
  Serial.println((versiondata>>24) & 0xFF, HEX);
  Serial.print("Firmware ver. ");
  Serial.print((versiondata>>16) & 0xFF, DEC);
  Serial.print('.');
  Serial.println((versiondata>>8) & 0xFF, DEC);
  Serial.print("Supports ");
  Serial.println(versiondata & 0xFF, HEX);
#endif
  nfc.SAMConfig();}
char DataIn[16];
```

Firmware Code 2. The setup loop code for the microcontroller of the NFC data acquisition platform.

The setup function is a crucial part of the firmware that runs once at the beginning. Its primary role is to initialize and configure hardware components, set up communication interfaces, and establish the initial state of variables, preparing the microcontroller for its main operation defined in the loop function. The serial connection with a baud rate of 9600 is established for debugging purposes, allowing for the printing of debugging messages to the serial monitor. The NFC library is then initialized, and the firmware version of the NFC controller is obtained and printed to the serial monitor. In case the module is not found, the program enters an infinite loop to halt execution. This code also configures the Secure Access Module (SAM) on the NFC module, which is crucial for secure NFC communication. Lastly, a character array called `DataIn` is declared to store incoming data from NFC communication, enabling further data processing within the program. This function prepares the NFC data acquisition platform for operation and provides valuable debugging information for monitoring its status and functionality.

```
void loop(void) {
 if(nfc.configurePeerAsTarget())
 {
   if(nfc.targetTxRx(DataOut,DataIn))
   {
    #ifdef NFC_DEMO_DEBUG
      char* timestamp = strtok(DataIn, ",");
      char* data = strtok(NULL, ",");
      Serial.print(timestamp);
      Serial.print(",");
      Serial.println(DataIn);
    #endif
    }
  }
}
```

Firmware Code 3. Then loop function of the NFC data acquisition platform repeats the set processes.

4 MEASUREMENTS AND DISCUSSION

The validation measurements are performed with the designed NFC data acquisition board using Arduino-integrated development environment. The SPI interface is used with master Arduino running as ISP to upload the bootloader firmware to the microcontroller of the NFC data acquisition board. After burning the Arduino bootloader, the UART to USB interface is used to program, debug the microcontroller and to transfer data from NFC acquisition platform to the laptop on serial port. Arduino IDE serial monitor is used to print the various error messages to debug the firmware code during the development phase. The basic programs are burned initially to verify the functionality of the designed hardware. The commercial NFC controller shield (VMA211) by Velleman Group for the Arduino UNO development kit is used as an NFC transmitter to establish and test the peer-to-peer communication mode for the designed circuitry before interfacing it with the IoT sensor node developed by (Hameed, 2023).

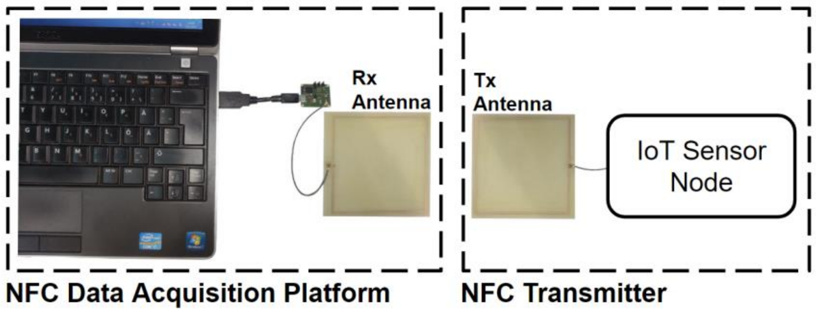


FIGURE 20. The measurement setup of NFC data acquisition platform with an IoT sensor node.

Three measurement scenarios are organized in this thesis work to evaluate the functionality of the NFC data acquisition platform. In the first measurement scenario, the developed NFC data acquisition platform is deployed in the measurement setup with a custom designed IoT sensor node to collect and record the measurement data. The IoT sensor node was developed by a fellow bachelor’s thesis student to monitor the environmental parameters are configured in NFC peer-to-peer communication mode (Hameed, 2023). FIGURE 20 shows the measurement setup of the NFC data acquisition platform integrated with the IoT sensor node. The data sent from the IoT sensor node is organized with timestamps and then sent over the serial port to Arduino IDE to monitor. A Windows-based application, CoolTerm (Meier, 2023) is employed to collect and record data on serial port from the NFC data acquisition platform. The data is organized in comma-separated format and saved in text files. The data from these text files is available to be retrieved by analysis tools like MATLAB to further process and plot the results.

TABLE 2. The measurements of the designed NFC antennas.

Antenna Name	No. of Turns	Length x Width (cm)	Conductor Width (mm)	Spacing (mm)	Impedance (Ω)	Antenna Area (cm^2)	Reading Distance (cm)
Ant_2x2	8	2 x 2	0.20	0.49	2.92+128j	0,96	4
Ant_3x3	6	3 x 3	0.50	0.60	1.55+135j	2,88	4,5
Ant_4x4	7	4 x 4	1.12	1.00	1.19+145j	8,20	5,5
Ant_8x4	4	8 x 4	1.60	1.00	1.03+139j	12,95	7,6
Ant_12x8	4	12 x 8	6.00	2.0	0.64+142j	67,20	19,5

In the second measurement scenario, the specifications of designed antenna as described in subsection 3.1.4 are measured using vector network analyzer (VNA ZNB8, Rohde & Schwarz) in radio frequency (RF) lab at Aalto University. The real and reactive part of the NFC antenna impedance are measured at center frequency of NFC communication (13.56 MHz). The pair of same antennas are deployed on IoT sensor node and NFC data acquisition platform to measure the maximum reading distances on centimeter scale. The measurements are recorded in TABLE 2 along with other designs and measured parameters of NFC antennas. The induced voltage e_{ind} in the inductive loop of NFC antenna can be evaluated according to Faraday's law as described in Eq. (1) where B is magnetic flux density A is the area of conductor and N is the number of loops. If center frequency of 13.56 MHz is same for all the antennas used in the study, then induced voltage in the reader antenna will be directly proportional to the sum of loop areas of the designed antenna as shown in Eq. (2). The Area of conductor for each loop is calculated and sums are tabulated in TABLE 2 for all designed antennas.

$$e_{ind} = -\frac{\delta}{\delta t} B \iint A \, dS \tag{1}$$

$$e_{ind} \propto \sum_{n=1}^{N} A_n \tag{2}$$

The maximum reading range of the NFC antennas is correlated with the area occupied by conductor in all loops of the NFC antenna. FIGURE 21 depicts that the reading range is almost linearly proportional to the area of the conductor in the designed antennas when the inductance of all the antennas is kept constant to use the same matching network designed for the NFC controller PN532 by NXP electronics. The induced voltages are directly proportional to the conductor area hence indirectly corelating with the effect on the maximum reading range of the designed NFC antennas.

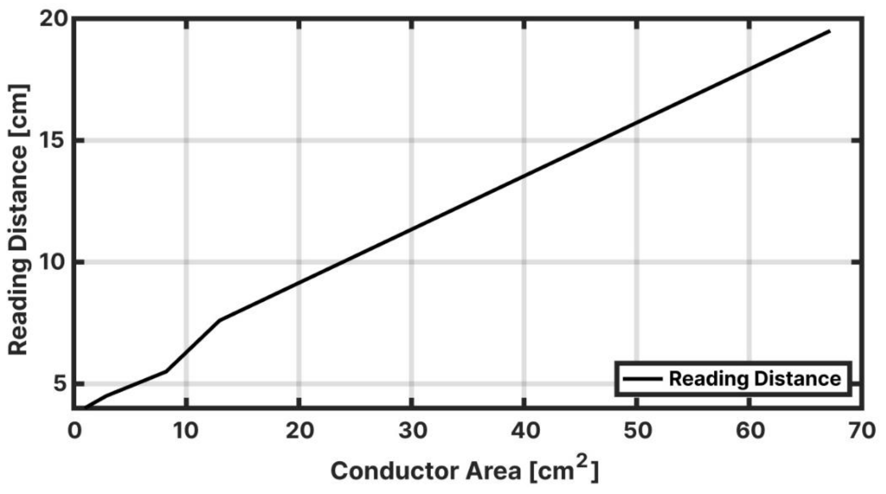


FIGURE 21. The relationship of maximum reading range of the NFC antennas with the conductor area.

In third measurement scenario, an NFC antenna of size 10 x 9 cm with 2 turns having conductor width of 1 mm and spacing of 1mm is used to analyse the transmitted power. The measurement setup is established to measure the received power at 13.56 MHz with real-time signal analyser RSA5126B by Tektronix in IC design lab of Aalto University. The distance between the transmitter and receiver antennas is varied and the received power is recorded. The plot of received power in dBm over the reading distance is shown in FIGURE 22. It is observed that the received power drops linearly when the reading distance is increased. The maximum power of 11 dBm was recorded at 1cm distance and the minimum power of -17dBm at 19cm distance from the transmitter antenna.

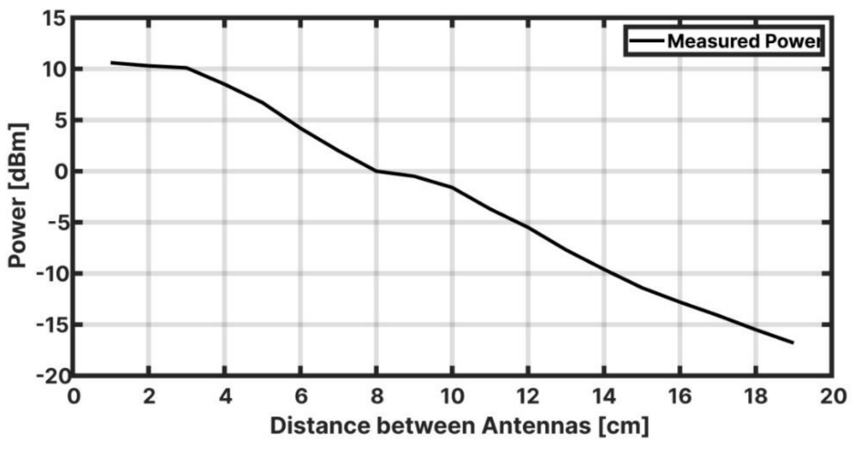


FIGURE 22. The received power by NFC antenna drops linearly when the distance from transmitting antenna is increased.

5 CONCLUSION AND FUTURE PROSPECTS

This work has explored the utilization of NFC technology in the IoT applications for wireless data acquisition. The project has involved the design and development of an NFC data acquisition platform to collect data from an IoT sensor node. The functionality of the system has been effectively demonstrated by conducting through validation measurements. This work on NFC data acquisition platform not only facilitates peer-to-peer communication with IoT sensor nodes for data collection but also provides valuable insights into antenna design and optimization. The findings of this study emphasize the direct correlation between conductor area on designed antenna and reading range, shedding light on the importance of these parameters to enhance the reading distances. Furthermore, the analysis of transmitted power and its relationship with reading distance contributes to a deeper understanding of NFC system performance which will be helpful for the energy harvesting based applications.

The development of an enhanced NFC data acquisition platform, which boasts an extended reading range, demonstrates the practical feasibility of this technology. These advancements offer a valuable contribution to the field of IoT, with implications for various industries, including healthcare, agriculture, and environmental monitoring. By understanding the interplay between antenna design, reading range, and transmitted power, the study opens the door to further exploration and innovation in NFC technology, providing a foundation for future research and development towards the energy harvesting based applications in this dynamic and rapidly evolving field.

REFERENCES

Coskun, V. et al. (2012) Near Field Communication (NFC) - from theory to practice. John Wiley & Sons Incorporated.

CXJRFID Factory (2019) Which RFID frequency fits your needs, CXJRFIDFactory. Available at: https://www.cxjrfidfactory.com/which-rfid-frequency-fits-your-needs/ (Accessed: 07 May 2023).

Hameed, S. (2023). IoT Sensor Node Design with NFC Data Communication using off-the-shelf components. Centria University of Applied Sciences.

Jurado-Verdu, C. et al. (2022) 'Barcolits: Barcodes using LED tags and Optical Camera Communications', 2022 IEEE 18th International Conference on Factory Communication Systems (WFCS) [Preprint]. doi:10.1109/wfcs53837.2022.9779172.

Kothari, D.P. (2012) Embedded systems. New Delhi: New Age International.

Lacamera, D. (2020) Embedded Systems Architecture: Explore Architectural Concepts, pragmatic design patterns, and best practices to produce robust systems. Birmingham: Packt Publishing.

Lesas, A.-M., et al. (2017) The art and science of NFC programming. London, UK: ISTE, Ltd.

NFC inductance: RF design: Edesignsuite: STMicroelectronics (no date) NFC Inductance | RF Design | eDesignSuite | STMicroelectronics. Available at: https://eds.st.com/antenna (Accessed: 15 June 2023).

Paret, D. (2016) Antenna designs for NFC devices. London: Wiley-ISTE.

Roger Meier's freeware (no date) Roger Meier's Freeware. Available at: https://freeware.the-meiers.org/ (Accessed: 18 June 2023).

Sabella, R. R. (2016). NFC For Dummies. Standards Information Network.

Schauffele, J., et al. (2019). Automotive software engineering: Principles, processes, methods, and tools (2nd ed.). SAE International.

Seeeds. PN532: NFC library using PN532 to read/write card and communicate with android. Retrieved June 18, 2023, from https://github.com/Seeed-Studio/PN532

Thornton, F. et al. (2006) RFID security. Rockland, MA: Syngress Publishing.

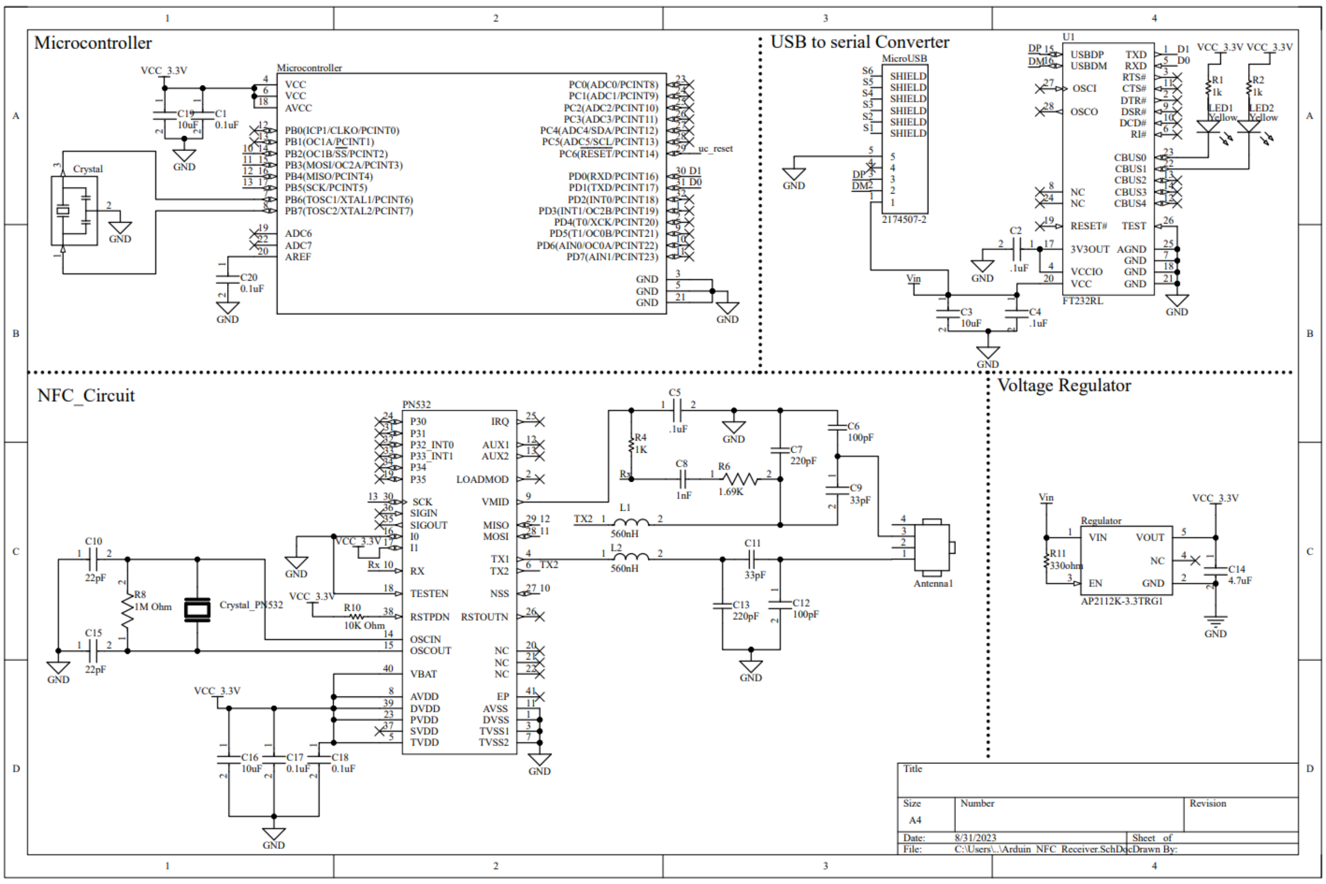
Westcott, S., et al. (2023). Basic electronics: Theory and practice (4th ed.). Mercury Learning & Information.

Zheng, F., et al. (2016). Digital Signal Processing for RFID (1st ed.). John Wiley & Sons.

APPENDICES

APPENDIX 1

The schematic diagram of NFC data acquisition board.



APPENDIX 2

The firmware code of NFC data acquisition board.

```
//This example reads a NFC/RFID memory block. It is tested with a new
NFC/RFID 1K cards. Uses default keys.
//Contributed by Seeed Technology Inc (www.seeedstudio.com)
//Modified by Muhammad Nadeem for bachelor’s Thesis to use as firmware
code for NFC data acquisition board.

//Liberaries inclusion
#include <PN532.h>
#include <SPI.h>

/*Chip select pin can be connected to D10 or D9 which is hardware
optional*/
/*if you the version of NFC Shield from SeeedStudio is v2.0.*/
#define PN532_CS 10
PN532 nfc(PN532_CS);

#define  NFC_DEMO_DEBUG 1

void setup(void) {
#ifdef NFC_DEMO_DEBUG
  Serial.begin(9600);
  Serial.println("Hello!");
#endif
  nfc.begin();

  uint32_t versiondata = nfc.getFirmwareVersion();
  if (! versiondata) {
#ifdef NFC_DEMO_DEBUG
    Serial.print("Didn't find PN53x board");
#endif
    while (1); // halt
  }
#ifdef NFC_DEMO_DEBUG
  // Got ok data, print it out!
  Serial.print("Found chip PN5");
  Serial.println((versiondata>>24) & 0xFF, HEX);
  Serial.print("Firmware ver. ");
  Serial.print((versiondata>>16) & 0xFF, DEC);
  Serial.print('.');
  Serial.println((versiondata>>8) & 0xFF, DEC);
  Serial.print("Supports ");
  Serial.println(versiondata & 0xFF, HEX);
#endif
  // configure board to read RFID tags and cards
  nfc.SAMConfig();
}

char DataOut[]="HELLO INITIATOR"; //16bytes
char DataIn[16];//Should be 16bytes

void loop(void) {
  // Configure PN532 as Peer to Peer Target
 if(nfc.configurePeerAsTarget()) //if connection is error-free
 {
    //trans-receive data
   if(nfc.targetTxRx(DataOut,DataIn))
   {
    #ifdef NFC_DEMO_DEBUG
      char* timestamp = strtok(DataIn, ",");
      char* data = strtok(NULL, ",");
      Serial.print(timestamp);
      Serial.print(",");
      Serial.println(DataIn);
    #endif
    }
  }
}
```

APPENDIX 3

Algorithm

1. Include required code libraries and initialize parameters.
   Begin by including the necessary code libraries to access functions and data structures. Initialize parameters such as communication ports, data buffers, and timestamps.
2. Set communication protocols.
   Define the communication protocols required for the NFC data acquisition platform, ensuring compatibility with IoT sensor nodes.
3. Check the connectivity of all communication interfaces.
   Verify the availability and functionality of communication interfaces, such as SPI or UART, to guarantee seamless data exchange.
4. Check and initialize the NFC controller.
   Ensure the NFC controller is operational by verifying its status and initializing it if necessary.
5. Configure NFC working protocol.
   Specify the NFC working protocol that is compatible with the IoT sensor nodes.
6. Check the connectivity with the IoT sensor node.
   Establish and validate a connection with the IoT sensor node in peer-to-peer mode.
7. Receive the data from the NFC controller.
   Continuously monitor and receive data transmitted by the NFC controller.
8. Organize received sensor data with proper time stamps.
   Process the incoming sensor data, incorporating relevant timestamp information for data organization.
9. Send data on the serial port at a set baud rate.
   Transmit the organized data via the serial port, configuring the baud rate for efficient data transfer.
10. Repeat the function loop.
    Continue looping through the specified functions to ensure ongoing data acquisition, processing, and communication, thereby maintaining the platform's functionality.