Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Distributed
Computing

NFC Glove

Semesterthesis

Abiraam Varathan

abiraamv@ethz.ch

Distributed Computing Group
Computer Engineering and Networks Laboratory
ETH Zürich

Supervisors:
Klaus-Tycho Förster, Jochen Seidel
Prof. Dr. Roger Wattenhofer

January 7, 2013

Abstract

The near field communication (NFC) technology intends to improve user comfort by enabling new scopes of applications for the smartphone. But this comfort can not be experienced by everyone since not all smartphone models support NFC. Another disadvantage is that the smartphone needs to be in hand to use this technology. In this project a glove is developed that overcomes those disadvantages by providing NFC technology to a smartphone through a wireless channel. Referred to as the "NFC glove". To accomplish this target, the circuit to be embedded in a glove is developed in a first step, using already available hardware components, and in a next step the software for the glove and for the smartphone is designed. The outcome of this project serves as a platform on which innovative applications can be developed to take the user comfort to a next level.

Contents

Chapter 1

Introduction

1.1 Motivation

Near field communication (NFC), which is based on radio-frequency identification (RFID), is a set of standards for contactless data transaction. It is used in a wide field of applications such as contactless payment, electronic ticket, or user identification. This technology is now slowly arising in the smartphone market, providing them with new capabilities such as that of an electronic wallet or electronic ticket, to name a few. Pairing with a NFC tag, which can be programmed by NFC applications, allows to automate tasks such as changing the phone profile, creating and sending a text, launching an application, or any other arbitrary task. These possibilities allow the user to perform amusing, but also handy tasks with his smartphone.

The downside of this possibility is that the smartphone has to be taken out of the pocket. This is the point where the NFC glove steps into the picture. A glove that reads NFC tags and forwards the read data to the smartphone through a wireless channel, so that all the tasks can be performed while the smartphone is still in the pocket. Such a glove also allows to perform the tasks with smartphones that do not support the NFC technology.

1.2 Problem Statement

The goal of this project is to develop a NFC glove that is able to read and write NFC tags and send it to a mobile device. This has to be done by the following steps:

1. **Assembly**: Testing of the hardware components and wiring them together to form the NFC glove.
2. **NFC glove software**: Develop the software for the NFC glove.

3. **Mobile application**: Develop a basic mobile application that is able to communicate with the NFC glove, hence read and write NFC tags. This application should be easily modifiable and extendable.

After completing these steps, a simple but yet innovative use case has to be developed to showcase a utilization of the NFC glove.

Time Schedule

The schedule for this project is defined such that equal timeframes is allocated for the NFC glove part (assembly and NFC glove software) and for the mobile application part, namely six weeks each. After these twelve weeks, the NFC glove, the basic mobile application, and preferably the simple use case should be viable. The exact schedule can be found in Appendix C.

Chapter 2

Design Considerations

2.1 Goal

The system design consists of two parts: the glove itself and the application for a mobile device. The glove is connected through a wireless channel to the mobile device. The glove should execute the following tasks: receive and transmit data from and to the mobile application, as well as read and write NFC tags. The mobile application should be able to send and receive data to and from the NFC glove. Furthermore, it should provide the ability to trigger a certain event if a specific tag is read. Last but not least, the hardware components used for the glove should be small enough to actually fit in a real glove.

2.2 System Setup

Figure 2.1 gives a graphical overview of the general setup of the system. The processing unit and the NFC reader form the NFC glove.

The mobile application provides various functions to the user based on the basic commands of reading and writing a tag. The functions depend on the use case of the mobile application.

The processing unit forms the interface between the mobile application and the NFC reader. It has a wireless connection to the mobile device and a wired connection to the reader. The processing unit has two main tasks. The first task is to decode the incoming command from the mobile application and to perform the correct NFC instructions in order to complete the command. The second task is to forward the incoming data from the NFC reader to the mobile application.

The NFC reader simply performs the command received from the processing unit.

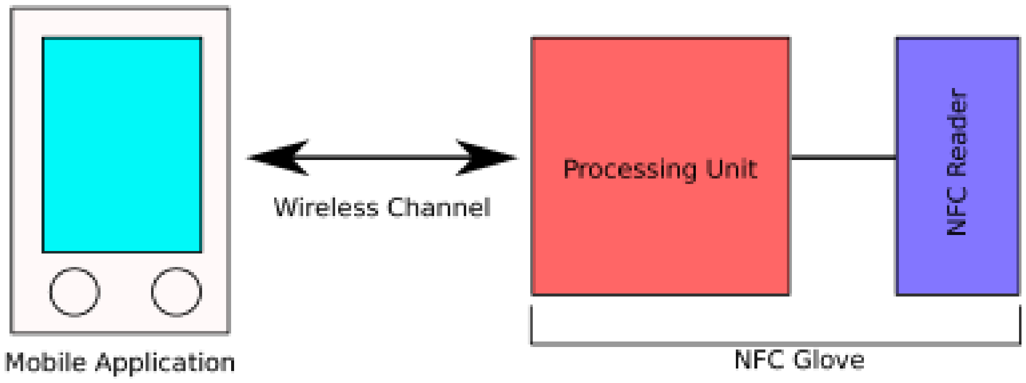

Figure 2.1: Coarse overview of the system to be developed

2.3 Software Architecture

Processing Unit

The software for the processing unit is provided with a wireless library and a NFC library. The wireless library offers basic functions for listening to the wireless channel, reading and sending data. Optionally, it can be enhanced with functions to manage the wireless connection, depending on the wireless technology used to interconnect the mobile device and the NFC glove. The NFC library offers all the necessary commands for the NFC reader and, in addition, offers functions to manage the connection to the reader.

Mobile Application

The mobile application is provided with a wireless library, which manages the wireless connection to the NFC glove, i.e. establishing the connection, maintaining the connection, reading the incoming data stream and sending data. This library should mainly operate in the background of the mobile application.

Chapter 3

NFC Glove Implementation

The NFC glove consists of three main components. As mentioned in Chapter 2, there has to be a processing unit, a NFC reader, and a wireless node to communicate with the mobile device. As the processing unit a simple Arduino board is chosen, a RFID reader serves as the reader, and a Bluetooth node is chosen to communicate with the mobile device. Testing of the NFC glove is done with a Mifare Classic 1k NFC tag that offers 1024 Bytes of data storage. Excluding the manufacturer data and the encryption keys, the tag offers 720 Bytes of net storage.

The time schedule allocated six weeks for this part of the project, three to four weeks thereof were used to become acquainted with the components and to test the compatibility of the RFID reader, the Bluetooth module, and the Arduino with small self-written programs. The rest of the time was used to develop the final Arduino application with the corresponding RFID library.

3.1 Hardware Components

The components are chosen primarily with the constraint that each of them should be as small as possible, but also the cost is a constraint to a certain extent. Appendix A summarizes the hardware components that are used for the NFC glove. The wiring of the system is shown in Figure 3.1. Note that the serial port of the Arduino, that is used to connect the RFID module is emulated in software.

3.1.1 Hardware Issues

When developing the NFC glove various hardware issues emerged. The crucial ones are summarized in the following list. The list also describes the steps that have to be taken to overcome these issues.

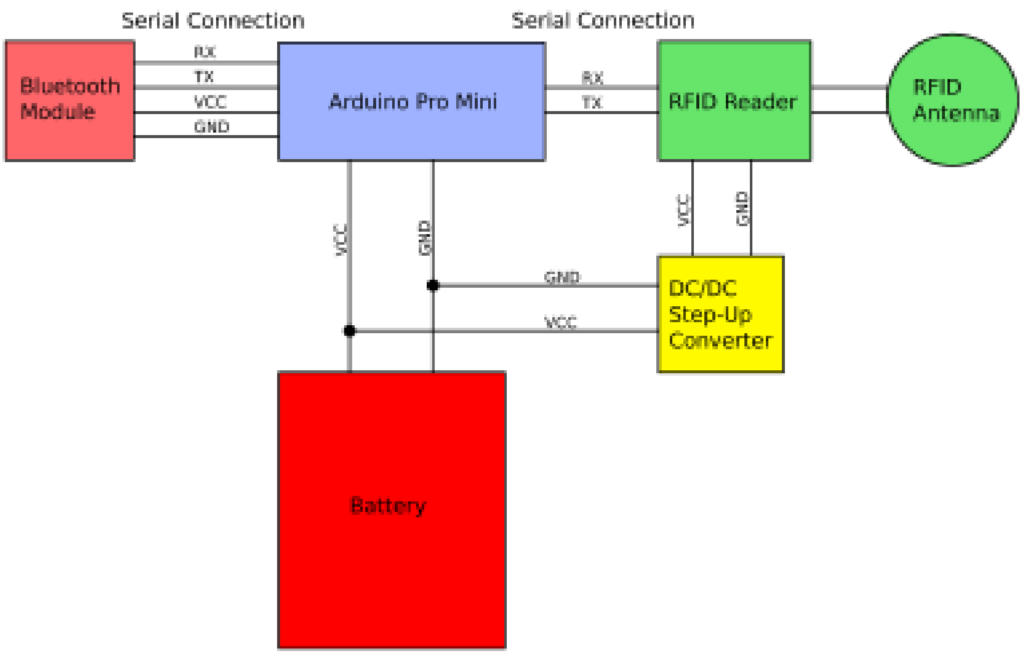

Figure 3.1: Wiring of the NFC glove

- The serial port of the Arduino has troubles to communicate with the Bluetooth module's default baud rate of 115200 Bd despite the listing in the specifications. Therefore, the baud rate of the Bluetooth module has now been lowered to 9600 Bd.
- The battery supplies a voltage of 3.7 V. The Arduino and the Bluetooth module are specified for a supply voltage of 3.3 V, the RFID module is specified for a supply voltage of 5 V. The Arduino and the Bluetooth module operate fine, but for the RFID module, the supplied voltage is too low to work properly. Therefore a 5 V DC-DC step-up converter is used between the battery and the RFID module.
- Since the Arduino and the RFID module have different supply voltages, the signal level for "HIGH" are not matching. To solve this problem, the reset pin of the RFID module has to be connected to ground.

3.2 Arduino Application

The Arduino application serves as an interface between the mobile application and the RFID reader, as it is responsible for the correct data flow. Therefore it has to be provided with a Bluetooth library and a RFID library. The Bluetooth library should handle the data transfer to and from the mobile application, and the RFID library should handle the data transfer to and from the RFID reader. The Arduino platform comes with a batch of libraries, including the Serial library for the serial ports of the Arduino, and the SoftwareSerial library for emulating serial ports on the digital ports of the Arduino. Since the bluetooth connection is only used to send and receive data, the available Serial library satisfied the requirements defined in Chapter 2. For the RFID library, the SoftwareSerial library has been extended to meet the requirements.

The Arduino application, which is written in C++, is basically a loop function that listens to both serial ports consecutively. If data is available on one of the ports, it calls the necessary functions from the corresponding library.

3.2.1 RFID Library

Reading and writing data from and to the RFID module, respectively, can only be done in blocks of 16 Bytes, and furthermore, blockwise authentication and response checking is needed. Implementing all this functions in the Arduino application would make the application overloaded and unclear. Therefore the SoftwareSerial library has been extended and the RFID library was created, which takes care of all these cumbersome processes. This library consists of all the commands that are needed to read and write a tag. The basic functions in this library are the *send_command* and the *read_response* functions. These two functions are responsible for sending commands, blockwise writing data, and blockwise reading data. This library provides the Arduino application with a search and read-, read- and write-function that read or write a whole tag. Furthermore it provides two variables, the *read_status* and *write_status* variable, to check the outcome of the reading and writing process.

Since the testing of the NFC glove is done with a Mifare Classic 1k tag, the library is optimized for this kind of tags currently. But the library is programmed in a way that it can easily be extended for other tag types.

RFID Data Packet

Figure 3.2 shows the packet that is transferred between the Arduino and the RFID module. The *Reserved-byte* is currently not used by the RFID reader but it nevertheless has to be sent. Important for a successful communication with the RFID reader are the *Length-byte* and the *Checksum-byte*. The *Length-byte* indicates the length of the payload, including the *Command-byte*. The *Checksum-byte* is calculated individually for every packet transfer by adding all Bytes except the *Header-byte*.

Header	Reserved	Length	Command	Data	Checksum
1 Byte	1 Byte	1 Byte	1 Byte	Up to 16 Bytes	1 Byte

Figure 3.2: The packet sent between the Arduino and the RFID reader

3.2.2 Software Issues

Two major software issues were experienced during the development of the NFC glove. The following list summarizes them and also describes the steps taken to solve them.

- At first the library was written in a way that the *read_response* function was called immediately after the *send_command* function. But this proved to be unreliable since the RFID reader needed time to process a command. At times, the reader was not given enough time to complete its task. Therefore a processing delay of 5 ms was inserted after each packet transfer to secure the reader enough processing time.
- The buffer size of both serial ports is limited to 64 Bytes. For the serial port to the RFID reader this did not turn out to be a problem since data transfer on this port always happens in packets of 18 Bytes. But for the serial port to the Bluetooth module this was a problem, because when a user wants to write a tag with his mobile application, more than 720 Bytes of data may arrive through this port. Increasing the buffer size manually led to a deadlock of the processor on the Arduino. Hence the data had to be divided and transferred in smaller packets. This issue was considered in the designing process of the basic application.

Chapter 4

Android Implementation

An Android phone is chosen as the mobile device. Because of its open source policy, Android provides good tutorials and descriptions of their libraries on their website[1], but also plenty of tutorials and libraries can be found on other websites. This simplifies the developing process of an application tremendously. The first step was to create a basic library that can be modified and extended very easily to match any case of application.

Six weeks were scheduled for this part of the work. The first two were used to familiarize with Android development and the remaining time was used to develop the basic application and the simple use case application.

4.1 Basic Library

The basic library is an Android application from which a NFC tag can be read and written. Additionally, the antenna of the RFID module can be turned off by the application in order save power. Reading or writing turns the antenna back on. The Bluetooth connection is managed by a separate library, the *BluetoothSerial* library, which mainly operates in the background. This library is called on startup of the application to search for the NFC glove among the paired devices and connects to it. Once the connection is established, it creates a thread that continuously listens to the Bluetooth port.

Figure 4.1 shows the GUI of the application. The first line displays the current status of the Bluetooth connection. The following lines display the read message from a NFC tag or failure messages. The text field is used to enter the message to be written on a tag, and is limited to 720 characters, i.e. 720 Bytes. The three buttons are used for the respective functions, search and read, write, and turn RFID antenna off.

[1]http://developer.android.com

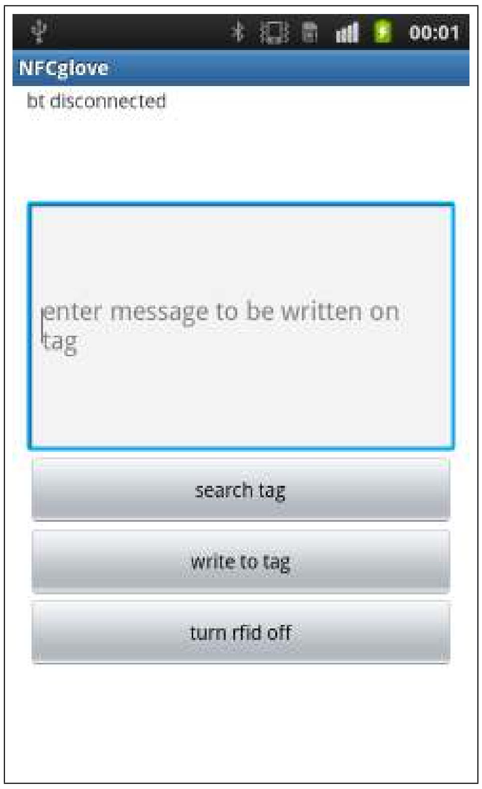

Figure 4.1: The GUI of the basic Android application

Bluetooth Data Packet

Figure 4.2 shows the Bluetooth packet sent from the mobile device to the NFC glove. The *command* byte indicates what command the user wants to execute. The *Length-Bytes* and the *data-Bytes* are only used when data is written to a tag. The issue of partitioning the data, mentioned in Chapter 3, was solved by sending the *Data-Bytes* in packets of 16 Bytes. And since the Arduino application is reading the buffer at the same time, no buffer overflow can occur.

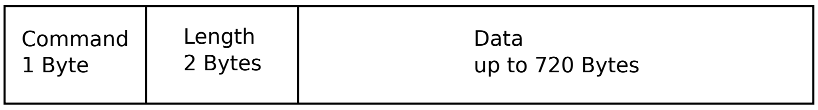

Figure 4.2: The packet sent between the Bluetooth module and the mobile device

4.2 Use Case: NFCmusic

After the first step was done by creating the basic library, the next goal was to create a new application by modifying and extending the basic library that demonstrates the functionality of the NFC glove in an interesting manner. Therefore the NFCmusic application was created. The idea behind this application is to associate a certain byte with a certain tone and to output it on phone speakers. For this reason the basic library is extended with an additional library, the *ToneGenerator* library, which generates sinusoidal pulses of a given frequency and then converts it to a 16 bit pulse-code modulated sound array to play it.

Figure 4.3 shows the GUI of the NFCmusic application. This application provides two functions. The first one, called by the first button, reads a tag and uses the integer representation of all 720 characters to map each of them into the audio frequency range and plays a melody. The second one reads a tag but only uses the first character which is mapped onto one of six adjacent piano key frequencies, allowing the user to virtually play a piano. Note that for the second function, the tags have to be written with predefined characters for the first byte.

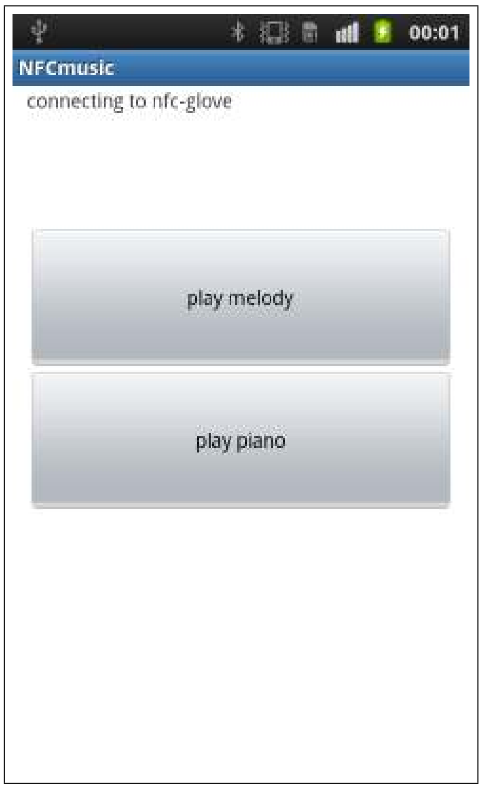

Figure 4.3: The GUI of the basic Android application

Chapter 5

Conclusion

This thesis dealt with the development of the NFC glove and the corresponding Android application. The outcome of this project has proven the feasibility of such a system, even though there are still some issues to be solved. This project now serves as a platform on which new interesting and innovative applications can be developed.

5.1 Future Work

This project is still in its pilot stage since it should be improved in some areas to reach its final stage. The major parts that have been identified to be improved in a next step are the following:

1. **Antenna**: The antenna used now is rigid and is therefore unqualified to be built in a glove. A smaller and flexible antenna would suit better.
2. **Glove**: The circuit has not been built in a glove yet and the wiring has to be optimized. This should be done in a next step after solving the antenna problem.
3. **RFID library**: As mentioned before, the RFID library only supports Mifare Classic 1k tags at the moment and therefore should be extended to support other tag types.

Other smaller improvements would be to incorporate all available commands of the RFID module into the RFID library and to add further failure checks such that a data transfer can be aborted early enough. Also the Arduino application allows improvements.

Of course, the Android applications offer prospects for improvements too. For example the GUI of the basic and the NFCmusic application could be improved to look more appealing. Furthermore, the *ToneGenerator* library of the NFC-music application could be improved and potentially extended to play other instruments than solely a piano.

Since this project serves as a platform, it provides the possibility to even come up with different amazing and innovative ideas that range from a simple visualization of the tag data up to a big game similar to Lasertag.

Bibliography

[1] : Roving Networks. RN-42/RN-42-N Class 2 Bluetooth Module Data Sheet. http://dlnmh9ip6v2uc.cloudfront.net/datasheets/Wireless/Bluetooth/Bluetooth-RN-42-DS.pdf

[2] : Roving Networks. Bluetooth Product User Manual. http://www.sparkfun.com/datasheets/Wireless/Bluetooth/rn-bluetooth-um.pdf

[3] : Sparkfun.com. Bluetooth Mate Silver Tutorial. http://www.sparkfun.com/tutorials/264

[4] : Sonmicro. 13.56 MHz RFID Mifare Read/Write Module Datasheet. http://www.sparkfun.com/datasheets/Sensors/ID/SM130.pdf

[5] : Arduino.cc. Arduino Pro Mini. http://arduino.cc/en/Main/ArduinoBoardProMini

[6] : Sparkfun.com. Arduino Pro Mini 3.3V Quickstart Guide. http://www.sparkfun.com/tutorials/244

[7] : Wikipedia.org. MIFARE. http://en.wikipedia.org/wiki/MIFARE

[8] : Arduino.cc. How to write libraries for the Arduino? http://playground.arduino.cc/Code/Library

[9] : Marc Boon. RFIDuino. https://github.com/marcboon/RFIDuino/tree/master/SM130

[10] : AvdM.nl. The communication between an Arduino and a SM130 RFID reader through UART. http://www.avdm.nl/articles/tech/communicationsm130arduino

[11] : Matt Bell. Android and Arduino Bluetooth communication. http://bellcode.wordpress.com/2012/01/02/android-and-arduino-bluetooth-communication/

[12] : Steve Pomeroy. Playing an arbitrary tone with Android. http://stackoverflow.com/questions/2413426/playing-an-arbitrary-tone-with-android

Appendix A

Hardware Components

The following list summarizes the hardware components used for the NFC glove and their usage.

- **Arduino Pro Mini 328-3.3V/8MHz:** Processing unit and data transfer interface.
- **SM130 Mifare 13.56MHz RFID Module:** RFID reader unit used to read NFC tags.
- **RFID Evaluation Shield-13.56MHz:** Antenna for the RFID module.
- **Bluetooth Mate Silver (RN-42):** Bluetooth adapter to connect with the mobile device.
- **NCP1402-5V Step-Up Breakout:** Voltage converter for the RFID module.
- **Polymer Lithium Ion Battery-3.7V**

Appendix B

Using RFID Library

In order to use the RFID library in a Arduino project, the folder *rfid_library* with the files *RFIDSerial.h* and *RFIDSerial.cpp* has to be copied to the Arduino's libraries folder first. On a linux system the path is usually

/usr/share/arduino/libraries/

Then in the project itself, the *RFIDSerial.h* has to be included and the *RFIDSerial* class has to be instantiated. The constructor looks the following:

RFIDSerial rfid_name(receive_pin, transmit_pin)

Appendix C

Time Schedule

Timetable: NFC Glove

	Week	Milestones
	1. 23.09 – 30.09	Read documentation
NFC Glove	2. 01.10 – 07.10	Communication between Arduino and BT module
	3. 08.10 – 14.10	Sending data via Arduino and BT module
	4. 15.10 – 21.10	
	5. 22.10 – 28.10	Communication between Arduino and NFC module
	6. 29.10 – 04.11	Reading/writing tags, send read tags via BT Finish RFID library
	7. 05.11 – 11.11	Finish BT library and Arduino application
Android Application	8. 12.11 – 18.11	
	9. 19.11 – 25.11	Basic Android application that communicates with BT module
	10. 26.11 – 02.12	
	11. 03.12 – 09.12	Send/read data to/from NFC tag
	12. 10.12 – 16.12	
	13. 17.12 – 23.12	Finish practical part
	14. 24.12 – 30.12	
	15. 31.12 – 06.01	Finish report