MASTERTHESIS

Development of mobile application for personalized auricular vagus nerve stimulation

Submitted to:

Institute of Electrodynamics, Microwave and Circuit Engineering

at the TU Wien

Advisors:

Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Kaniusas Eugenijus

Projektass. Dipl.-Ing. Kampusch Stefan, BSc.

By:

Andreas Knor, BSc.

PersID: 1528219

Vienna, September 2019

Abstract

In recent years a new therapeutic method has gained attention: the electrical stimulation of the vagus nerve. As it gains more and more reputation as effective treatment, for instance, in neurological or mental disorders, the supporting technology behind this therapy, also must steadily improve. One method of performing vagus nerve stimulation (VNS) is the stimulation of the auricular vagus nerve with a wearable stimulation device. In this work, the process of monitoring and controlling the therapy, done with such wearable microelectronic device, will be advanced. To achieve this objective, an everyday object was introduced into the therapeutic cycle, the smartphone. As target-operating system for the application, the Android™ Platform has been chosen. With the usage of up-to-date development tools like android studio, MPLAB X, source tree and some test equipment like an oscilloscope the development was done. The connection between the device and the smartphone was established through near field communication (NFC) using the protocol standardized by ISO/IEC 15693. Also the C-code of the stimulator was altered to allow the NFC connection. The built application aims to help the medical professionals to personalize the therapy for each patient, by providing four adjustable parameters: amplitude, frequency, pulse count per burst and pulse width. Furthermore the user of the application is given a tool to monitor the battery and the impedance of the electrodes, which is gives information about the quality of the connection of the electrodes. This research has brought forth an application which satisfies the basic need for more information during VNS treatment. However since mobile phone applications are part of a high changing environment not only will the application need maintenance to sustain its quality and appearances, but also has still room for improvements by means of better calculations, faster connection, or smoother user interface to be validated in future usability tests. Also a user system which allows patient to use this application with a limited access (read only), could be implemented in the future.

Kurzfassung

In den letzten Jahren hat eine neue therapeutische Methode Beachtung gefunden: die elektrische Stimulation des Vagus Nervs. Da diese Behandlung, zum Beispiel bei neurologischen oder mentalen Störungen, immer mehr an Ansehen gewinnt, muss sich auch die unterstützende Technologie, die hinter dieser Therapie steht, stetig verbessern. Eine Methode zur Durchführung einer Vagus Nerv Stimulation (VNS) ist die Stimulation des Vagus Nervs am Ohr mit einem tragbaren Stimulationsgerät. In dieser Arbeit wird der Prozess der Überwachung und Steuerung der Therapie mit einem solchen tragbaren mikroelektronischen Gerät vorbessert. Um dieses Ziel zu erreichen, wurde ein Alltagsgegenstand in den Therapiezyklus eingeführt, das Smartphone. Als Zielbetriebssystem für die Anwendung wurde die Android ™ gewählt. Unter Verwendung aktueller Entwicklungstools wie Android Studio, MPLAB X, Source Tree und einiger Testgeräte, wie einem Oszilloskop wurde die App Entwicklung durchgeführt. Die Verbindung zwischen Stimulator und Smartphone wurde durch Nahfeldkommunikation (NFC) unter Verwendung des von ISO / IEC 15693 standardisierten Protokolls hergestellt. Außerdem wurde der C-Code des Stimulators geändert, um die NFC-Verbindung zu ermöglichen. Die App soll den Ärzten dabei helfen, die Therapie für jeden Patienten individuell anzupassen, indem vier einstellbare Parameter bereitgestellt werden: Amplitude, Frequenz, Pulszahl pro Burst und Pulsbreite. Darüber hinaus erhält der Anwender der App ein Tool zur Überwachung der Batterie und der Impedanz der Elektroden, welche Auskunft über die Qualität der Verbindung der Elektroden gibt. Diese Forschung hat eine App hervorgebracht, die das Grundbedürfnis nach mehr Information während der Behandlung der VNS befriedigt. Da jedoch Mobiltelefonanwendungen Teil einer sich stark verändernden Umgebung sind, muss die Anwendung nicht nur gewartet werden, um ihre Qualität und ihr Erscheinungsbild zu erhalten, sondern es besteht auch noch Raum für Verbesserungen durch bessere Berechnungen, schnellere Verbindungen oder eine glattere Benutzeroberfläche, welche validiert werden müssen durch zukünftige Usability-Tests. In Zukunft könnte auch ein Benutzersystem implementiert werden, das es dem Patienten ermöglicht, diese Anwendung mit eingeschränktem Zugriff (Lesezugriff) zu verwenden.

Acknowledgements

I want to thank the people, who made this thesis possible:

- **My advisor Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Kaniusas, Eugenijus**, for providing me with the opportunity of this fascinating topic and environment.

- **Projektass. Dipl.-Ing. Kampusch Stefan BSc.**, who gave me more time for this work than I could have bargained for, kept up with my tight time schedule and always had an open door and an advice for me, when I needed one.

- **My fiancée Hannah**, for all the love a man could wish for and giving me the strength to pull through, even in the hardest of times

- **My mother Margit Knor**, for all the positive energy and happy thoughts.

- **My father Emil Knor**, for the support through all this years.

- **In remembrance of my grandmother Hildegard Pelczmann**, for even though you are no longer with us, I will keep you in my heart.

Table of Contents

1 Introduction

With the ongoing societal evolution, not only the average lifespan of humans increases but also the causes of death. Further, most common diseases shift towards new ones. In a developed country the main causes of death are linked to an unhealthy every day routine [1]. Due to the advancements in medicine many diseases don't have to be fatal anymore. However there are many new illnesses on the rise which have been identified and quantified in many studies e.g. [2]. The main risk factors for being affected by one of the modern diseases are not only limited to the lifestyle but also to age and body mass index [2]. Figure 1 and Figure 2 show the causes of death in percentage of the world's population split into male and female in relation to age.

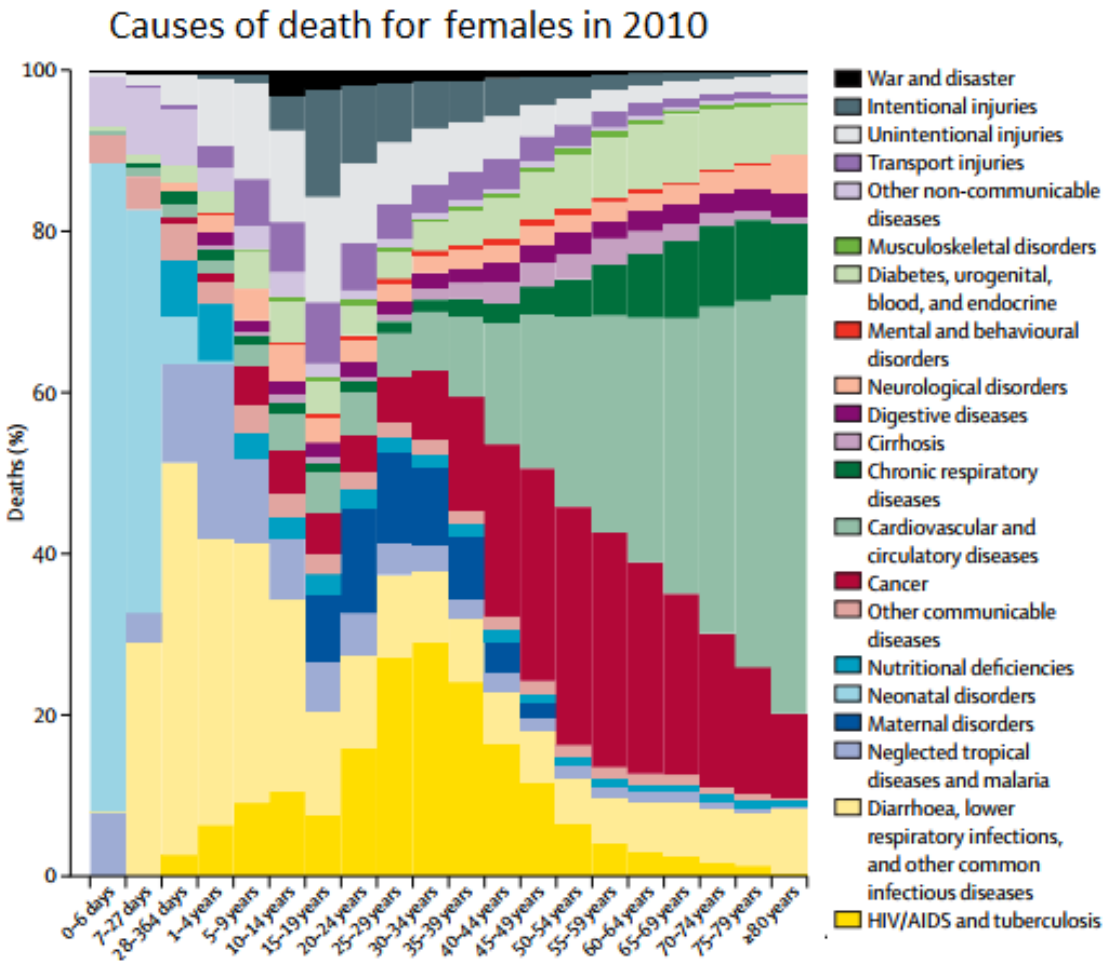

Figure 1: Causes of death on a global scale for female subjects in 2010. This figure can be created online at [3].

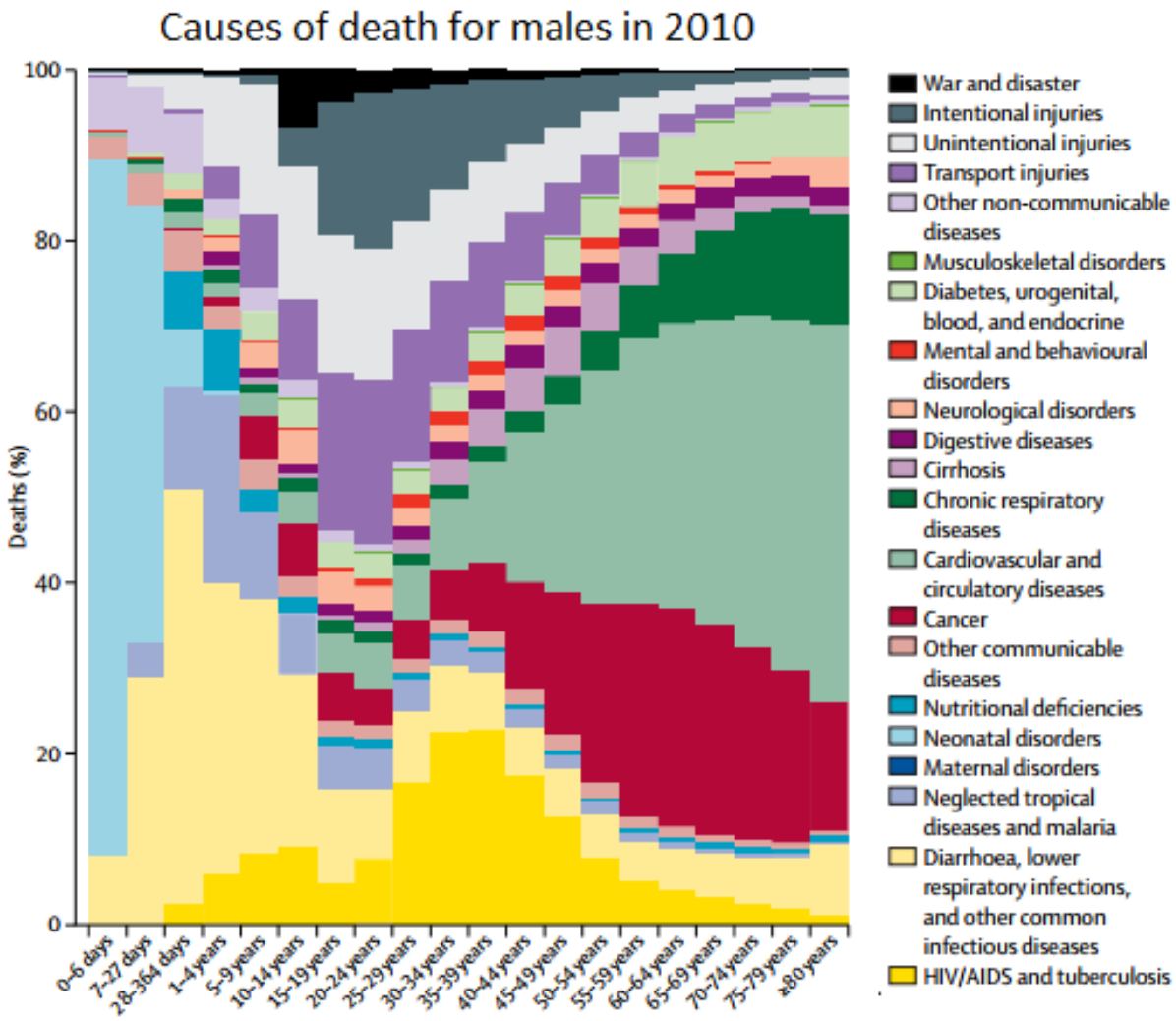

Figure 2: Causes of death on a global scale for male subjects in 2010. This figure can be created online at [3].

In developed countries with high income, the leading causes of death are: stroke, coronary artery disease and peripheral artery disease [4]. The recent years show that there has also been a noticeable rise of illnesses that, while no longer leading to certain death, due to their treatability, still leaves the affected patients with need of constant therapy and a change in their daily life routine. Diabetes mellitus is one of the leading examples of those illnesses, and how it changes the life of affected patients and can even lead to death if not treated right [5].

There are many treatments for those “civilization diseases”. One therapeutic option is gaining attention in that field, the vagus nerve stimulation (VNS). This therapy shows promising results in dealing with a range of chronic neurological, mental or

inflammatory disorders[6]. Also this treatment can be adapted to the severity of the illness. There are currently many different methods for the stimulation of the vagus nerve ranging from minimal invasive stimulation via needle electrodes, to an implanted stimulator with a cuff electrode around the nerve [7].

To ease the treatment process for the patient and the attending physician the treatment can be facilitated by using wireless communication protocols with the stimulators and a smartphone. Since nearly everyone in a developed country owns a smartphone, it can be used for breaking down complex tasks into easy to understand figures. This in return will help to raise the awareness of the patients for their current health status and the physician will be assisted in therapy guidance, via an easy application running on his smartphone, e.g. to access collected patients data and to personalize the treatment [8]. In terms of security, near field communication (NFC) possesses the highest reliability, because both the reader and tag have to be in direct proximity to each other. Therefore, the threat of theft, cloning, man in the middle, relay attacks and loss of a mobile device poses only a minimal risk. Security is a big issue when it comes to personal data, especially when considering medical records [9].

This thesis presents an implementation of such system to personalize the treatment with VNS. A NFC enabled microcontroller, in charge of the stimulation, and an Android based smartphone app, which transforms the smartphone of the medical professional into a personal assistant, build up the whole system. The application will help the doctor with feedback regarding the electrode status, history of the therapy with regards to on/off time and stimulation amplitudes, or the remaining battery lifetime.

2 Physiological and Technical Background

This section will provide knowledge of all the basic principles needed to understand a microcontroller supported stimulation of the Vagus nerve, and the state of the art of telemedicine and wireless communication in the medical field.

2.1 Percutaneous Auricular VNS

The cranial nerve X, previously called the pneumogastric nerve, due to its connection to the digestive tract and the lungs, is now better known as the vagus nerve. It is part of the autonomic nervous system and the parasympathetic control. Furthermore the nerve is highly entwined with the cranial nerve IX [10].

2.1.1 Anatomical Background

Through the nervous system the body is able to receive input from its environment and interact with it through the somatic nervous system. Input from the inside of the body, from its intestines, is processed and influenced through the autonomic nervous system. The somatic and the autonomic nerves together build up our whole nervous system. For conscious and voluntary perception and interaction the network of the somatic system is responsible, while for tasks that are conducted subconsciously the autonomic nervous system processes the input and regulates output to the organs [11] p. 608.

The autonomic nerves can further be divided into the sympathetic nervous system (SNS), parasympathetic nervous system (PNS) and the enteric nervous system (ENS). While the independent definition of the enteric nervous system as distinct system is relatively new, it is justified through various facts like the independent function of the gastro-intestinal system and the high number of nerve cells. [11] p.690.

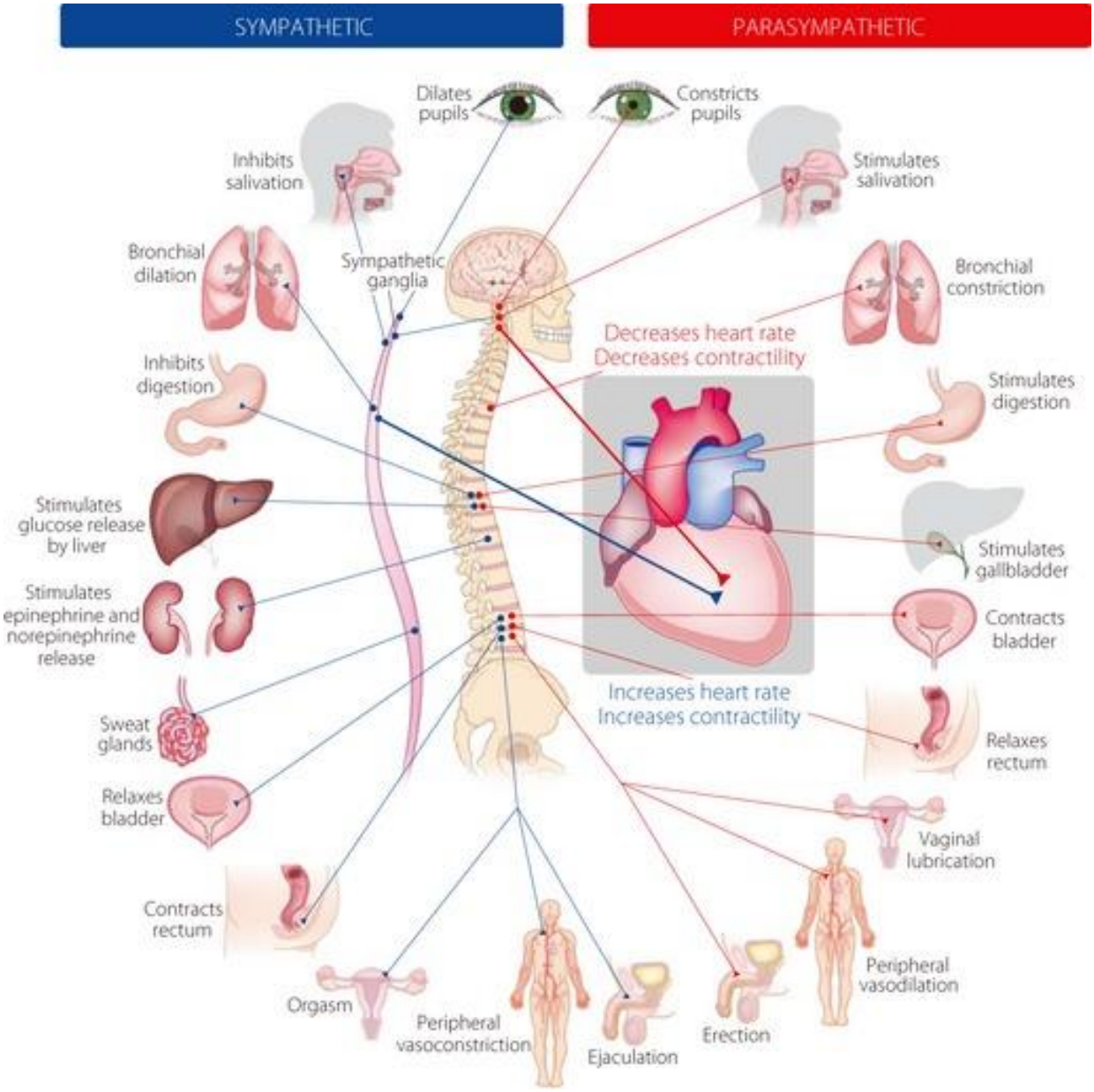

Figure 3: Overview of the sympathetic and parasympathetic innervation route. [12]

All nerve cells can furthermore be differentiated into afferent and efferent nerves. While afferent nerves carry information from the periphery to the spinal cord and brain (i.e. the central nervous system), efferent nerves carry impulses from the brain and spinal cord to the periphery. In the special case of a reflex arc the impulses come from the afferent fibers to the spinal cord where they immediately trigger the efferent fibers [13]p.78 [14]p. IX.

The sympathetic and parasympathetic systems are antagonistic systems. The sympathetic nerve activity dominates the body during time of physical or psychological stress and prepares the body for physical exertion. This happens through many processes, which are shown in Figure 3 through the blue lines. All those actions are set to provide energy and oxygen to the skeletal muscles and prepare for a fight or flight situation. On the contrary, if the body wants to regenerate, the parasympathetic system takes over. The controlled processes can be seen in Figure 3 on the red lines. Most of those actions are in the opposite of the sympathetic stimulation [15].

As the most important brain nerve, most of the fibers of the Vagus nerve are parasympathetic neurons, which reach out to the most important vital organs [11] p.699-700. The earliest studies tracing the vagus nerve were conducted 1987 [16]. Since this nerve runs form the brainstem to the abdomen, the vagus nerve also is known as the longest nerve in the human body [7]. The vagus nerve consists of a left and a right branch, which are formed by merging afferent and efferent rootlets. This happens as the nerves leave the cranium through the jugular foramen. It then continues running downwards connecting to several important vital organs like the skin, heart, lung, spleen, diaphragm, liver and the intestines which can be seen in Figure 4. Most of this knowledge is derived from studies in rats, which is very similar to other species including the human [17]. However meanwhile the structure of the vagus nerve has already been studied in the human body [18].

The vagus nerve branches out to nearly all organs from the transverse colon up to the neck and also deploy parasympathetic nerve fibers to a few skeletal muscles. This indicates that the vagus nerve is in charge of a series of highly vital functions, such as influencing the heart rate. The Vagus nerve even branches out to the outer ear with some afferent fibers also known as Alderman's nerve [19] p. 66 .

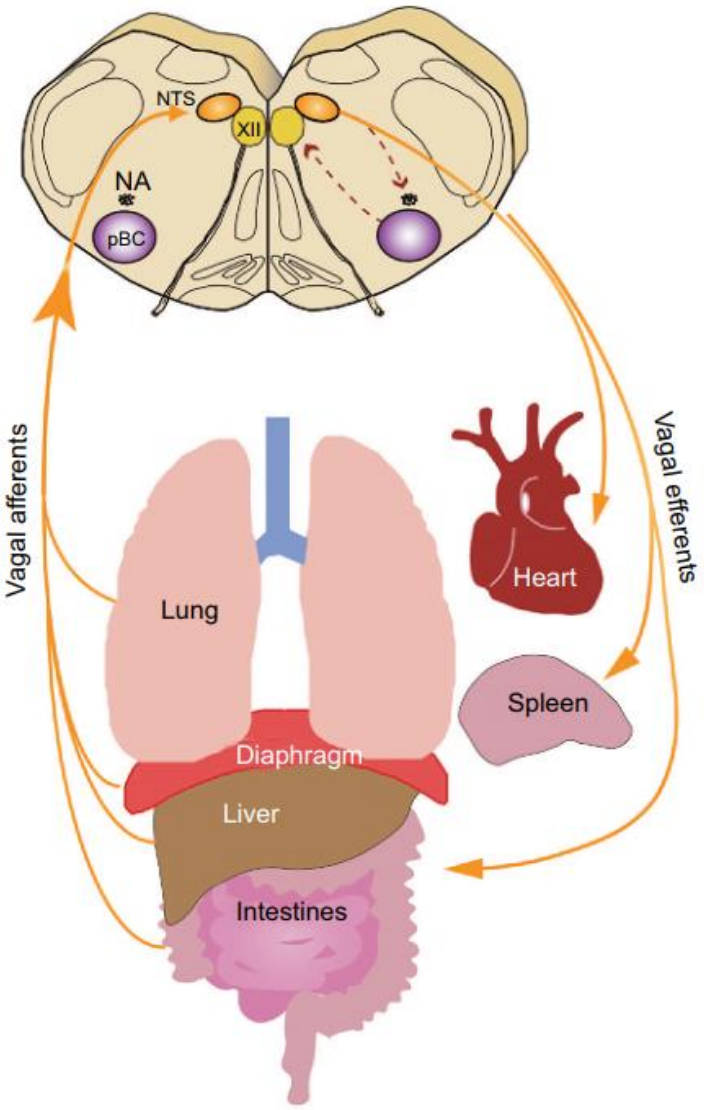

Figure 4: Overview of the Vagus nerve circuit. [20]

2.1.2 Effects of VNS

After finding the paths of this nerve, many studies have been conducted to find out what function could be triggered by electrical stimulation of this nerve. Since the activity of the vagus nerve is coupled to lowering morbidity and mortality, it may be an excellent target for various therapies, e.g., via electrical stimulation [21]. For stimulation, in the area of the throat and head, the left Vagus nerve is always

preferred (because of less cardiac side effects) and the right side nerve is only used when the left one is not accessible.

Currently there is a variety of vagus nerve stimulators, with different approaches, on the market. Some try to stay minimal to non-invasive, while others are implanting stimulators to raise the effectiveness while increasing the risk [22], [23], [24]. Many studies have shown the efficacy of stimulating the extensive afferent and efferent network provided by the Vagus nerve. However, the complete innervation extent remains unknown. One of the useful known applications is, that by stimulating the Vagus nerve, a cholinergic anti-inflammatory response can be triggered through a loop of the Vagus afferents, which can be used in treatment for diabetes [20]. Furthermore this treatment is used as a clinical method in treating epilepsy, depression, pain and for cardiovascular diseases. Research is ongoing to use the VNS as treatment for sepsis, obesity, lung injury, stroke and many other chronic diseases with inflammatory response [6], [25], [20].

2.2 Microcontroller Guided Stimulation

With a history of more than 50 years, electrical stimulation is a well-known and established approach to stimulate nerves. On the 8th of October 1958 the first electronic pacemaker was implanted into a patient. Since then electronics made huge steps and become more advanced, than people back than could have dared to dream about [26].

Nowadays, implanted microcontrollers are state of the art and no novelty. A microcontroller is a low cost single chip computer. The main purpose of a microcontroller is to hold a program and execute it. Like most computers a microcontroller is built from a central processing unit (CPU), some memory in form of random access memory (RAM) and read only memory (ROM), input/output (I/O) lines, serial and parallel ports for communication with its environment and some timers. Optionally some are equipped with analog digital converters (ADC) and digital analog converters (DAC). Microcontrollers put the smart in most devices on the market [27] p.3-4.

However progress for better, smaller, more efficient and safer systems in medicine has not come to a halt. Chih-Kuo et al. have proposed in 2003 a new implant technology, which is equipped with a coil to receive data and get charged through inductive coupling and a cuff electrode around a nerve for innervation. The system block diagram of this system can be seen in Figure 5 [28].

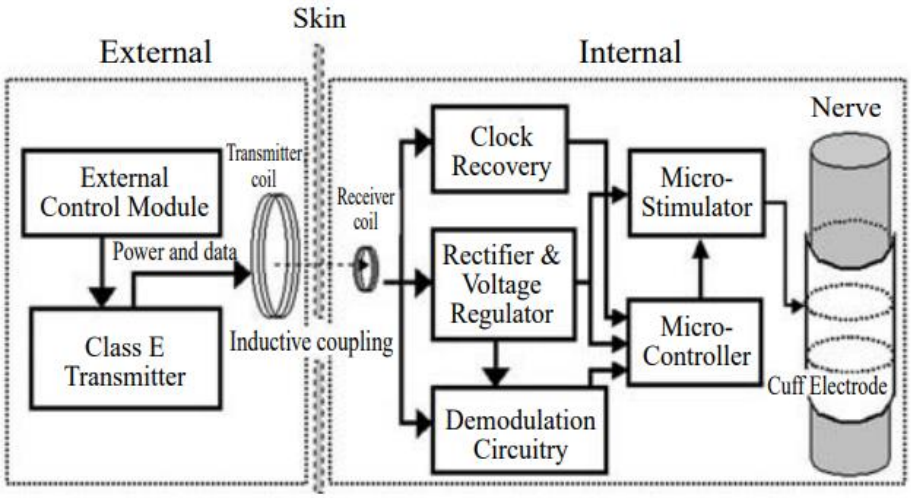

Figure 5: Overall system block diagram of an implantable micro-stimulation system. [28]

Microcontrollers are versatile, have a lot of power and their field of application is only bound by the limits of imagination and program space on the chip. The programming language depends on the underlying architecture but the most common are either assembler or C. The microcontroller projects that have boosted the image of microcontrollers are without a doubt Arduino and RaspberryPI. Those are special microcontrollers developed for a broad audience, which allows their users easy and guided projects for robotics and automatization, like smart home solutions [29].

Microcontrollers are used for research and development of methods in various different application domains. The controller always has to match and follow the needs of the application, and resources have to be allocated to fit and guarantee the best performance. The main aspects one has to decide upon are program space and speed of the program [30] p. 2-4.

In biomedical systems microcontrollers are used to either record biomedical signals from the body or to stimulate specific body structures. Many signals can be

transmitted and or received through electrodes, ranging from electromyograms to the electroencephalogram. The microcontroller can be equipped with a radiofrequency antenna, some amplifiers and some filter circuits and is ready for use [31].

The vagus nerve offers an effective suppression for a wide range of various diseases and therefore presents itself a perfect target for microcontroller aided stimulation. Normally the nerve would react automatically to internal or external stimuli, but a microcontroller can be used to initiate impulses on the nerve to stimulate the natural responses of the underlying system of the Vagus nerve [32].

Since microcontrollers are not only bound to static routines the stimulation can be modulated in real time. Input for the modulation of the stimulation signal can be received through all channels of the microcontroller. This could be achieved by continuously transforming vital signs into electrical signals, by adding a NFC chip, or by taking magnetic fields as input signals. A schematic stimulation system with coupling to vital signs and stimulation electrodes is shown in Figure 6.

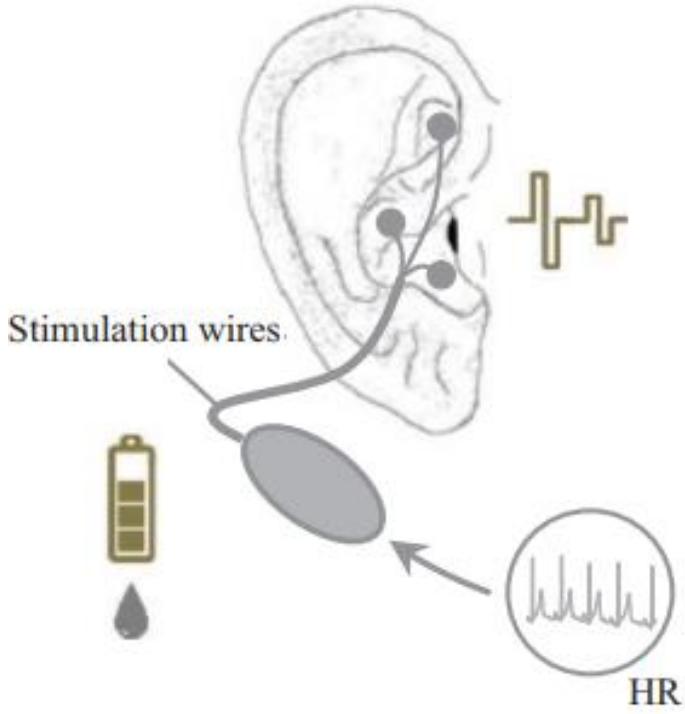

Figure 6: Connection of a percutaneous VNS device with three pin electrodes and feedback via biosignals. [33]

2.3 Mobile Health and Wireless Connection

In progressive times like this, the need for faster, more affordable and more flexible ways to deliver treatment to the patient, is on the rise. Therefore the whole healthcare sector is currently shifting into a new direction, which is closer intertwined with modern technology. The medical treatments get more support from different electronic devices to achieve this goal [34].

Telemedicine is the linkage, advances in technological research enabled, between the medical staff and the patients, where the doctor is no longer bound by distance to treat the patient. Telehealth is the next step and is not only limited to deliver treatment by physicians but also other services like telepharmacy and telenursing. For these systems mobile health is a huge aid [34].

Mobile health is the use of wearable devices to assist patients with their treatment and or administer the treatment. Wearable devices are also used to record data for later evaluation by the physician and allow optimizing the treatment. Figure 7 shows the necessary systems in form of web services to ensure the full functionality and also the interaction between the patient and the possible devices [8].

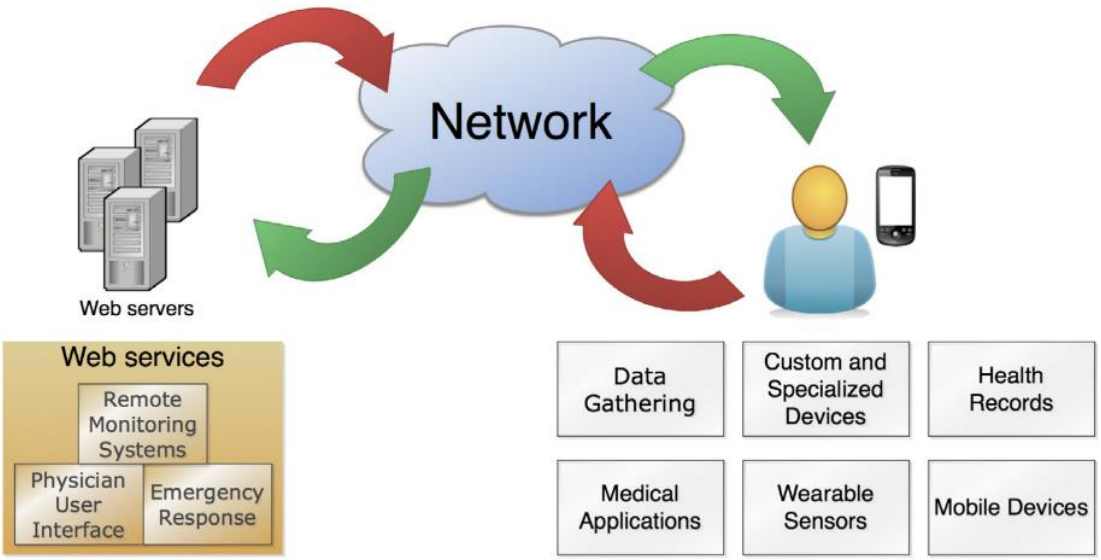

Figure 7: A look behind the curtain of mobile health. [8]

Currently many companies and hospitals are utilizing the mobile health apps to achieve a virtualization of the healthcare industry, as the next step in telemedicine. Often those apps are only the interface to specialized devices like a stimulator, drug administrator or recording device like an ECG [34].

Part of this “health care revolution” is still only predominant in first world countries due to the requirement of access to fast mobile internet, with a decent download volume. However, there also are many offline application for mobile health [8].

One of the biggest advantages of mobile health lies within the possibility to treat chronic conditions not only more constant but also to simplify the logging of medical data. In 2013 one out of 3 adults in the United States suffered from hypertension, which resulted in a cost of 93 billion dollar to the health care system. Those costs could be cut drastically by mobile monitoring using the connection of a smartphone and blood measuring device and only check in at the doctors if it is necessary [35].

Medical effectiveness rises with the amount of information a physician has at his/her disposal. This amount can be drastically increased by the use of mobile health devices that store or extract the patient data. With the increase of data, the importance of the linkage rises as well. Therefore data without reference is at best waste, but in worst case, like if responsible for drug delivery, can lead to suffering and should be prevented at all cost. Radio frequency identification (RFID), which is used in NFC tags, is already used to create such a linkage [36]. A possible set up to help nurses connect a patient to the right medication is shown in Figure 8.

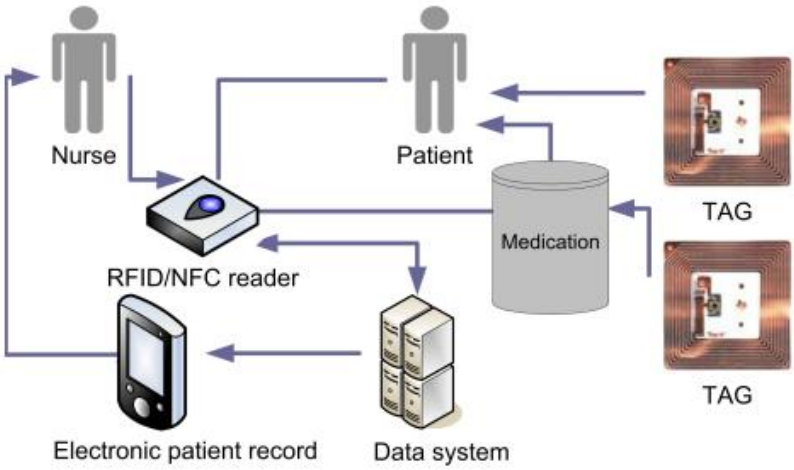

Figure 8: A possible RFID/NFC system set up. [36]

NFC has a wide range of applications, from paying wireless to exchanging contact information by holding two smartphones close to each other. The connection involves an active and a responsive tag/device. NFC has 3 modes of operation: Reader/Writer mode, Card Emulation mode and Peer-to-Peer mode. In Reader/Writer mode data is read from or written to the passive tag, while in Card Emulation mode the NFC device acts as an NFC tag so that other devices can send and receive data. In Peer-to-Peer mode a link between two devices is established for active data transfer in both ways. Around 80% of NFC apps utilize the Reader / Writer mode [37].

Most NFC applications in medicine are about identification and linking information together. For these tasks, dependability and security are the top priorities. Those two requirements are easily fulfilled by NFC applications. Since every NFC tag receives a permanent unique identifier during manufacturing, which can't be changed afterwards, this prevents misidentification. The tag itself doesn't consist of many components and is always operative, since it is powered by the active device. Security of the connection is physically given, due to the short range of the connection, which makes attacks very difficult [36]. Furthermore, a strong cryptographic framework can be used to tighten the security of the connection even further. In Figure 9 a security model of a NFC enabled "Healthcard" is shown, which can hold confidential medical records and identifies the patient [9].

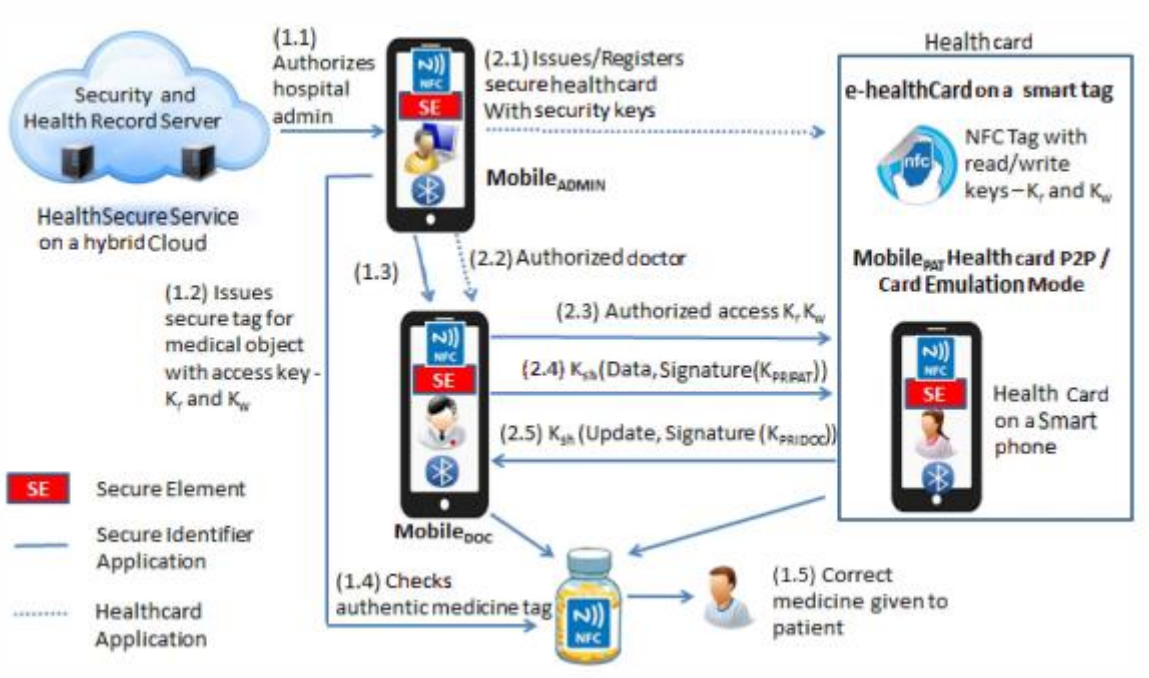

Figure 9: Security flowchart of a NFC Healthcard mock up. [9]

3 Methods

In the following section the main tools necessary for this thesis are explained very shortly.

3.1 Materials

For all the experiments, tests and development the following instruments and software have been used. Most of these devices were provided by TU Wien. The stimulation device was provided by SzeleSTIM GmbH.

Table 1: Physical devices used.

Table of physical devices	
Battery	CR2032MFR (Renata,Swiss)
Power Supply	Konstanter t2 k18 ru2 (Gossen Metrawatt, Germany)
Digital Multimeter	EM393B (Hama, Germany) & Fluke 85 III True RMS Multimeter (Fluke, USA)
Oscilloscope	tds2024B (Tektronix, USA)
Programmer	PIKIT 3 (Microchip Technology, USA)
Smartphone	One (OnePlus, China) with Android 9:Pie
Stimulator	Prototype (SzeleSTIM GmbH, Austria)

Furthermore a PC with windows 10 installed was the base for following programs.

Table 2: Software used.

Table of software	
Android Studio	Version: 3.4.1 Build #AI-183.6156.11.34.5522156, built on May 1, 2019 JRE: 1.8.0_152-release-1343-b01 amd64 JVM: OpenJDK 64-Bit Server VM by JetBrains s.r.o
MPLAB X IDE	Version: 5.20 Java: 1.8.0_181; Java HotSpot(TM) 64-Bit Server VM 25.181-b13 Runtime: Java(TM) SE Runtime Environment 1.8.0_181-b13 Compiler: XC8(v2.00) Pro Version (trial)
Source Tree	Version: 2.6.10.0

3.2 Android Studio

3.2.1 Android

Android is an open source operating system (OS), established as Android Open Source Project [38]. However, Android is also a software platform for mobile devices, such as smartphones or tablets and therefore is only applicable for certain devices which were built for Android. Although Androids seems like its own standalone OS, in reality it only is another distribution of Linux. This secures its position as open source software and adds up to its popularity through low cost, high transparency and flexibility to be further developed for single uses (e.g. home entertainment vs smartphone) [39]. The popularity of Android can be seen in Figure 10, as its market share in 2018 rises to 88%.

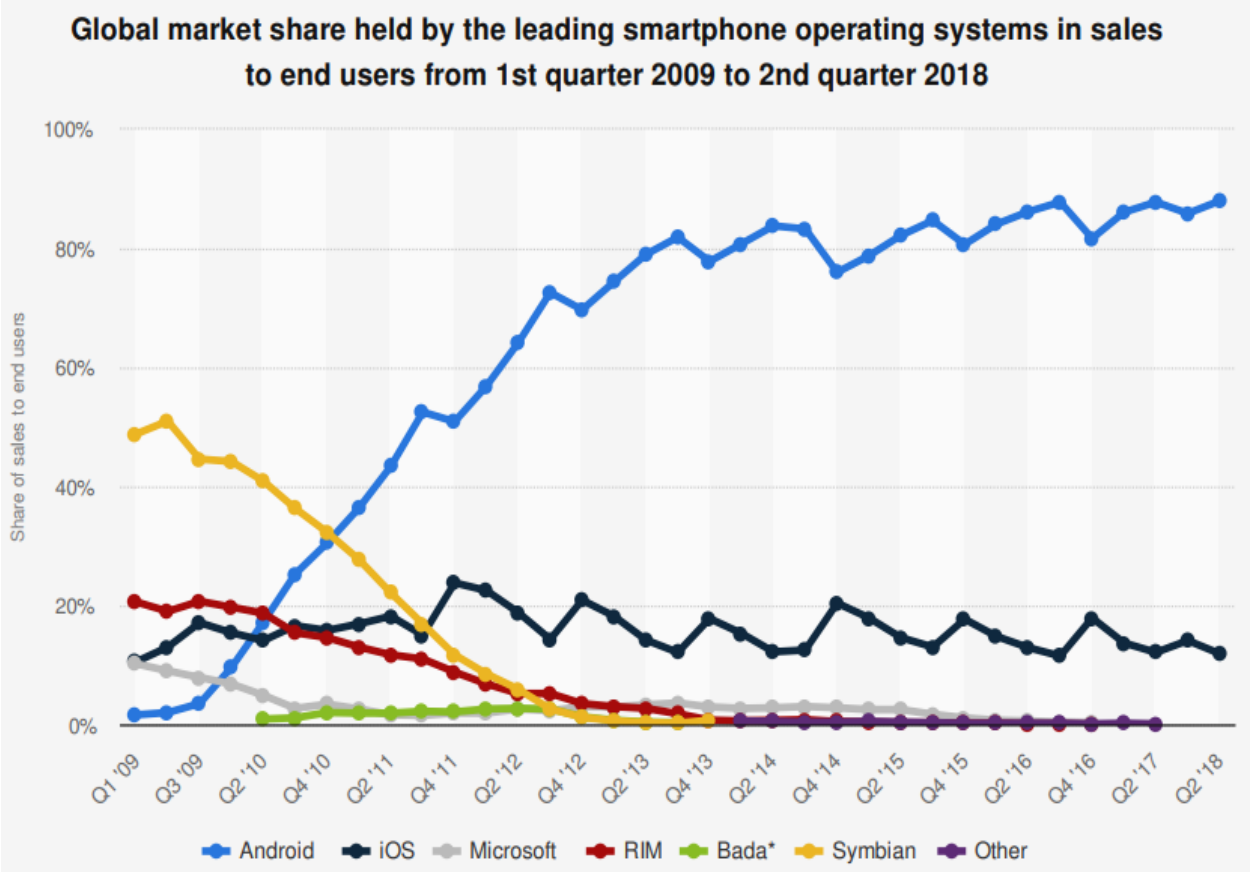

Figure 10: Percentage of mobile phone OS market share [40].

3.2.2 Set Up

Android Studio is a free integrated development environment for Android applications, which is based on IntelliJ IDEA from Jet Brains. In 2015 Android Studio replaced the android developer tool plugin for Eclipse as standard development environment [41]. Figure 11 shows the user interface of Android Studio which reassembles the UI of the IntelliJ IDEA.

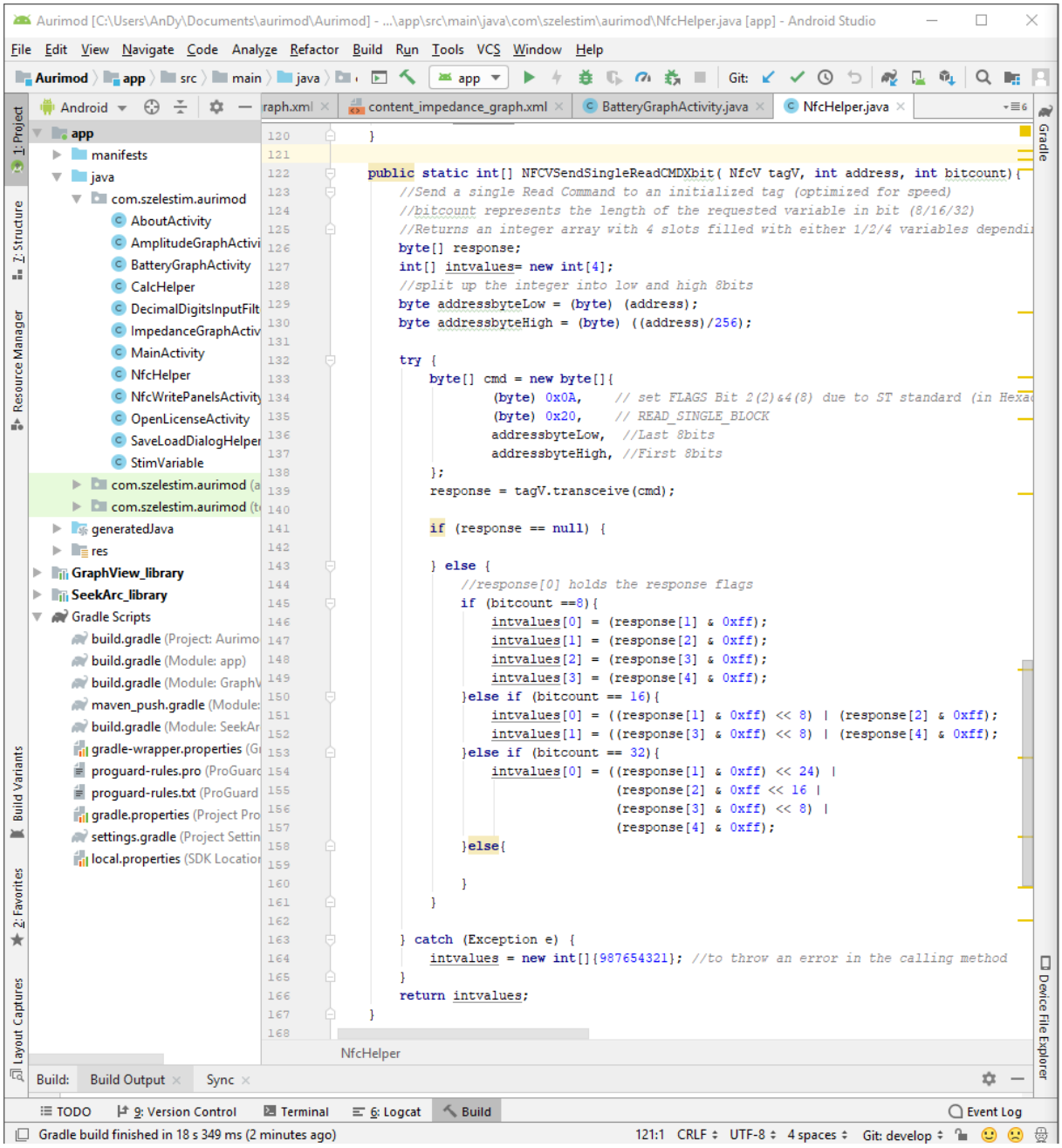

Figure 11: Android Studio User Interface.

The software is equipped with many tools such as a debugger, or a compiler and an installer via the android debug bridge to a real device which is connected through USB or a virtual emulator. Furthermore, Android Studio has a fully built in version control. Android projects use Gradle as build management tool. The Gradle build files can also be accessed directly with Android Studio.

3.3 Source tree

Source tree is a free client for visualizing the status of Git and, or Bitbucket repositories for Windows and Mac [42].

3.3.1 Set Up

First you have to create a repository using Github or Bitbucket to create the initial state. This repository can be easily linked to the source tree desktop app to make future changes in the repository [42].

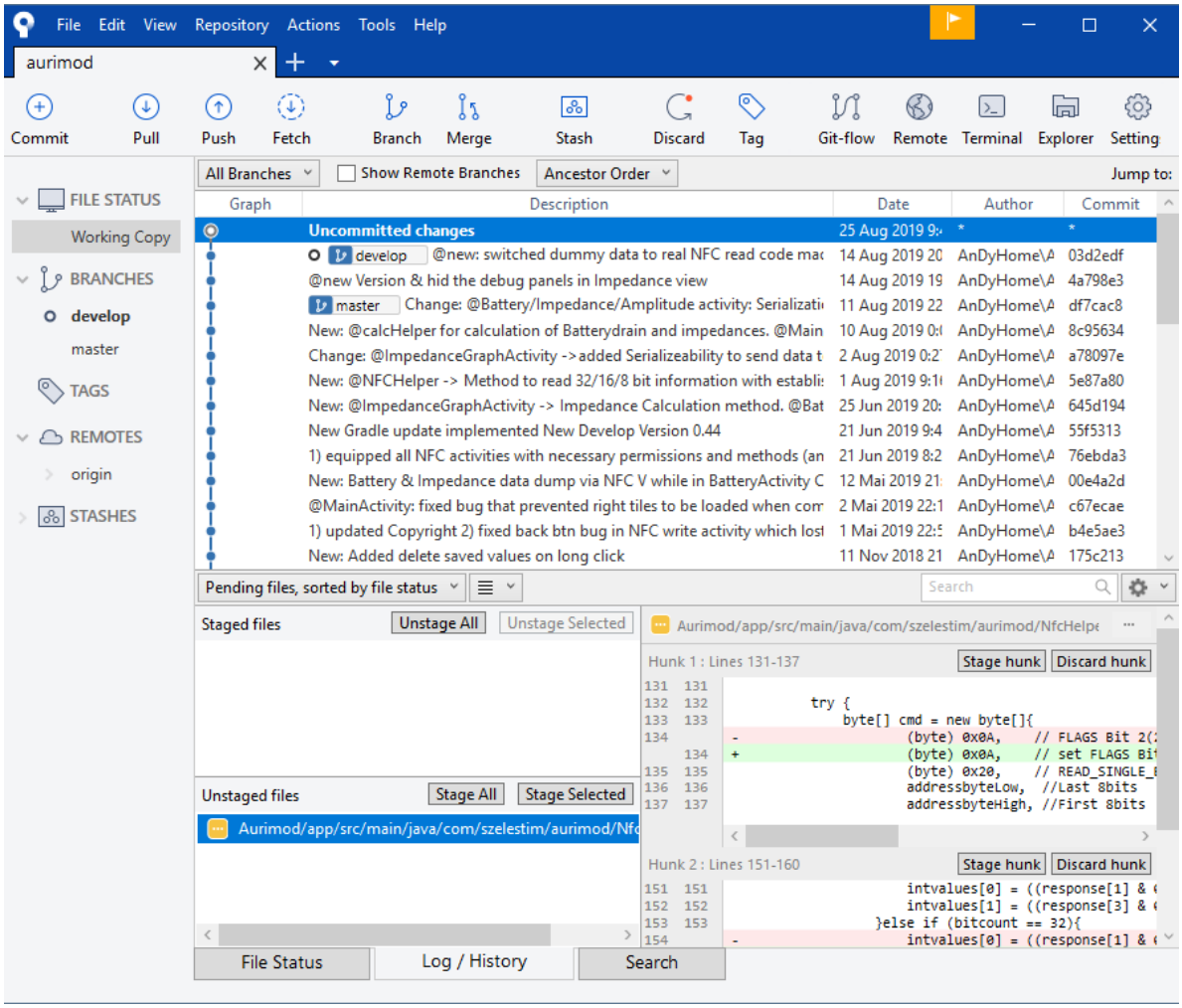

Figure 12: Sourcetree User Interface.

Figure 12 shows the user interface of source tree. The main window shows the versions and the comments of the developer to these changes, as well as the status of the branches (here titled develop and master). The bottom windows show changes, which can be uploaded to the repository.

3.3.2 GIT / Bitbucket

GIT is a distributed version control system used in software development. It is used to keep track of changes to the source code. Even during phases when more than one person is editing code and provides ways to merge changes to create the final code [43] p.351. Bitbucket is similar to GIT, however since it is software coming from Atlassian it also offers some features to interact with other Atlassian software. For teams of no more than 5 people the software is free to use and offer a Jira, Trello and of course Sourcetree Integration.

3.4 MPLAB X

MPLab X is an integrated development environment for C, developed by Microchip Technology and most of its features are free to use. This software shows most affinity to the microcontrollers developed also by Microchip technology. Its field of application lies within editing, debugging and programming of 8-, 16- and 32-bit microcontrollers [44].

3.4.1 Set Up

After installing the software, the XC 8 compiler v2.00 had to be downloaded and installed, then the plugin “Code Configurator” can be added via -> Tools-> Plugins. It is important to run the compilation in Pro-mode, otherwise the timing of code execution will be delayed. The slightest delay in the measuring routines will lead to wrong values in a time variant signal. Also the safety routines will be affected and will cause a shutdown of the device if the measured values are out of bounds. With the

time optimized compilation those errors can be avoided. Once finished with the installation the PICkit 3 was connected and the Project Properties were set up to provide a voltage level of 3V to simulate the battery.

The interface shows the overview of the different projects currently open in the upper left window, the specifications of the target device in the bottom left window, the code in the upper right window and results in the bottom right window.

3.5 Interface NFC-V

3.5.1 Set Up

For communication with the wearable stimulation device, it was equipped with a M24LR64E-R chip produced by STMicroelectronics. This microchip comes with a dynamic NFC/RFID tag and 64 Kbit EEPROM and is able to harvest energy from the Radiofrequencies (RFs), which allows data transfer even without a power supply. The energy harvesting system can also be used for loading a rechargeable battery, if an external coil is connected. For the communication the chip takes the role of the passive element only answering to correct commands, received by the active element. As active element in this setup a mobile phone running an android system of API 16 (“Jelly Bean”) or higher was used. The mobile phone sent the commands through the build in NFC coil to the chip, which then replied with an answer, and optionally changes the values in the EEPROM managed by the chip.

The EEPROM of the M24LR64E-R is divided into 64 sectors of 32 blocks of 32 bits. A read - / and or write protection can be assigned individually to each sector with a password command specified in the ISO 15693. The chip comes with a pre-set area which is read only access and is called the “type plate”. This area, consisting of 4 sectors spanning from the addresses 0 to 127, is filled on the production line during production of the chip and holds information of the chip like the 64-bit unique identifier (UID). Those 4 sectors are modified to read only access also during production.

3.5.2 Unique Identifier

Each chip compliant with the ISO/IEC 15963 standard is identified through a unique identifier. The ISO specification states that each UID has to consist of 64bit constructed in the following way:

1) Eight most significant bits (MSBs) with a value of “0xE0”
2) The code consisting of 8 bits, which identifies the manufacturer. For example STMicroelectronics was given the code “0x02”. This code is imprinted onto every chip from that manufacturer.
3) A unique serial number consisting of 48bits

If read from address 0 (LSB) to 64 (MSB) the UID will be displayed in reverse order and needs to be reversed again. With this UID, traceability of the RF tag, the IC itself and everything that is attached to the tag, for quality control can be ensured. Furthermore the UID can be used in commands to target one specific tag if more than one tag is in the readers RF field [45].

3.5.3 Transmission Protocol

The integrated circuit (IC) used for this set up, is equipped with an 64 Kbit EEPROM, an I²C bus and an NFC/RFID tag, to support the microcontroller with communication. The IC is built to utilize NFC-V connections standardized through the ISO 15693 and ISO 18000-3 mode 1. Also the supplier expanded the command list by adding custom commands to the NFC-V transmission [46]. The added commands where designed to give the user more control and also faster transmission rates. However those commands could not be called correct from the android app, maybe due to formatting error during conversion from signed to unsigned, or an error in the specification itself. The supplier could be contacted to solve this problem. For the android application only the original ISO 15693 commands have been used, since

they worked reliable even with existing background noise produced through the stimulation.

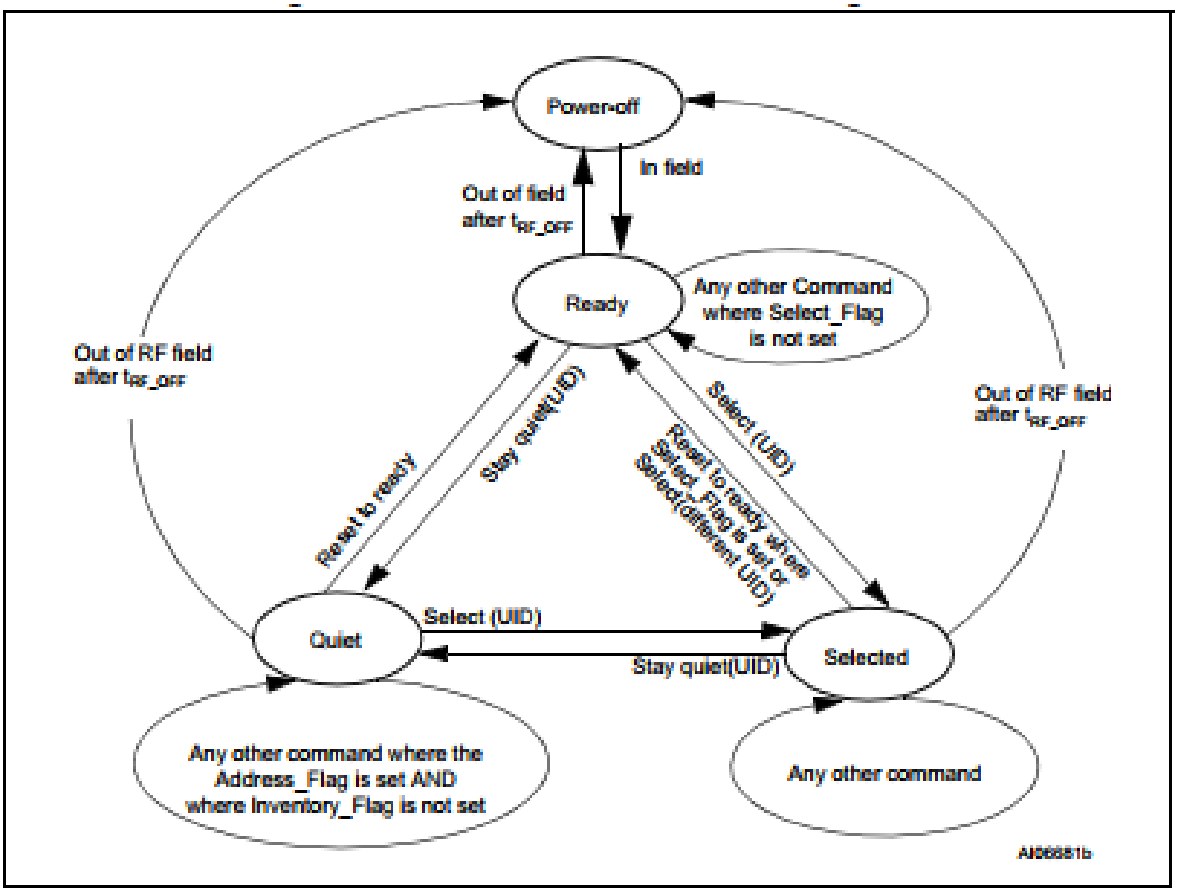

Figure 13: State Transition Diagram shows the three general states of the tag and the commands to switch between them. [46]

First, the smartphone, from now on called the vicinity coupling device (VCD), needs to activate the RF tag with the RF created by the VCD. The second step is to send the command from the VCD to the M24LR64E-R chip. The chip then sends a response, if the command from the VCD was formatted correctly. Those functions work even while no battery is attached or the battery is depleted, by harvesting the energy in the RF field. This process is named “reader talks first”. The carrier frequency f_c is standardized per ISO 15693 to 13.56 MHz ± 7 kHz. This transmission protocol is shown graphically in Figure 13 for normal use switching between “Power-off”, “Ready” and “Selected” state. While a tag in “Ready” state will answer to any command sent without an UID. A tag switches to the “Selected” state when it

receives a command with its UID. If the VCD sends a command with a UID, only the correct tag will answer. [45]

3.5.4 Data Rate and Data Coding

The data is transferred by using the position modulation technique. Each possible value is assigned to a position of a pause of the signal. The duration of a value slot is defined as 18.88 µs, which equals 256 oscillations of the f_c. The pause only occurs during the 2nd half of the slot and therefore lasts 9.44 µs. To get a value with this method the receiver needs to check for the missing, or weak amplitude oscillation out of a pre-set range of oscillations. In the 1 out of 256 mode a byte is transferred and in the 1 out of 4 two bits are transferred. The coding mode is signaled by the “Start of frame” (SOF) which leads the command. In Figure 14 the coding for the 2 modes is displayed with the timing of the pauses. The SOF is chosen by the VCD by setting the “Data_rate_flag” or not setting the flag in the command.

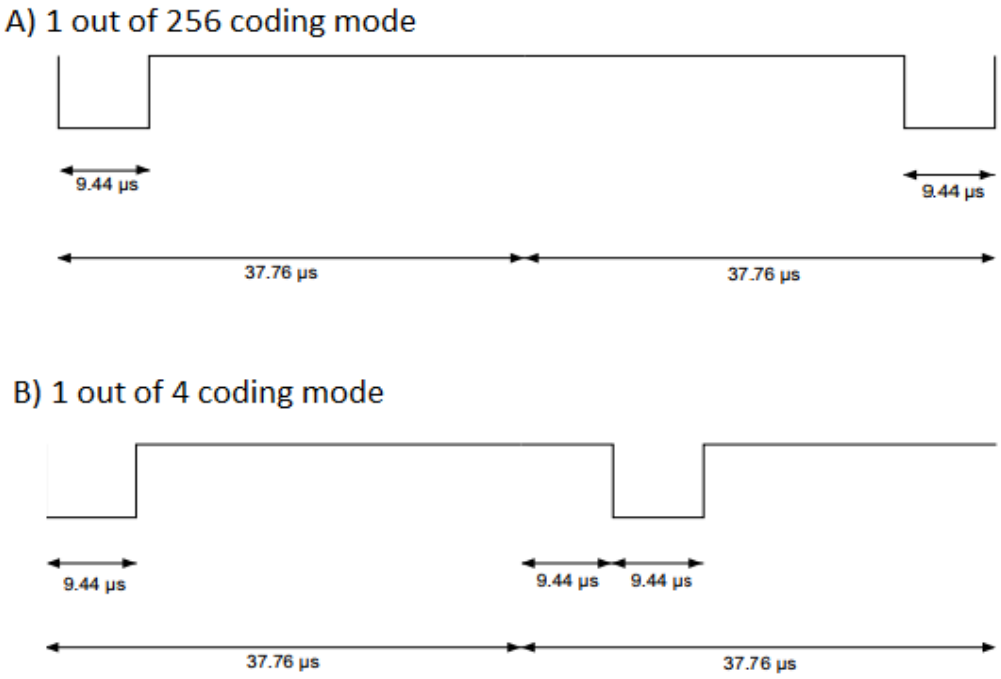

Figure 14: SOF coding of A) 1out of 256 and B) 1 out of 4. [46]

3.5.4.1 “1 out of 256” Mode / Low Datarate

A byte can hold a value from 0 to 255, which results in a transmission time of 4.833 ms per byte and a data rate of 1.65 Kbits/s. The transmission time can be calculated by adding 256 value slots together ($= 256 * 18.88\ \mu s$) and the data rate is calculated by dividing 1 second by 8 times the transmission time ($= \frac{8\ bit}{4.833\ ms}$), since one byte consists of 8 bits. An example of an encoded byte with the value 0xE1 (225 in decimal) is displayed in Figure 15.

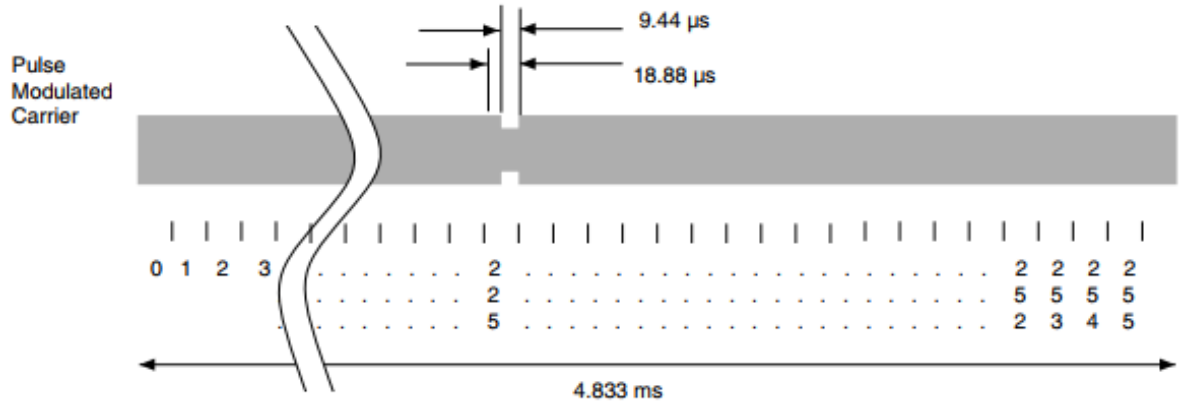

Figure 15: Example of position modulation, with the “1 out of 256” method, of the value 0xE1 (225). [46]

3.5.4.2 “1 out of 4” Mode / High Data rate

With this method 2 bits are transferred with one pause. During one transmission the pause happens during 1 of 4 subsequent time periods and code for 00, 01, 10 or 11. To form a byte, with this method 4 successive pairs are added. The transfer of 2 bits takes 75.52 µs ($= 4 * 18.88\ \mu s$) and therefore the transfer of a byte takes 302.08 µs ($= 4 * 75.52\ \mu s$). This transfer rate can be calculated to be 26.48 Kbits/s ($= \frac{1\ bit}{8*75.52\ \mu s}$). The 4 time slots possible with this method can be seen in Figure 16.

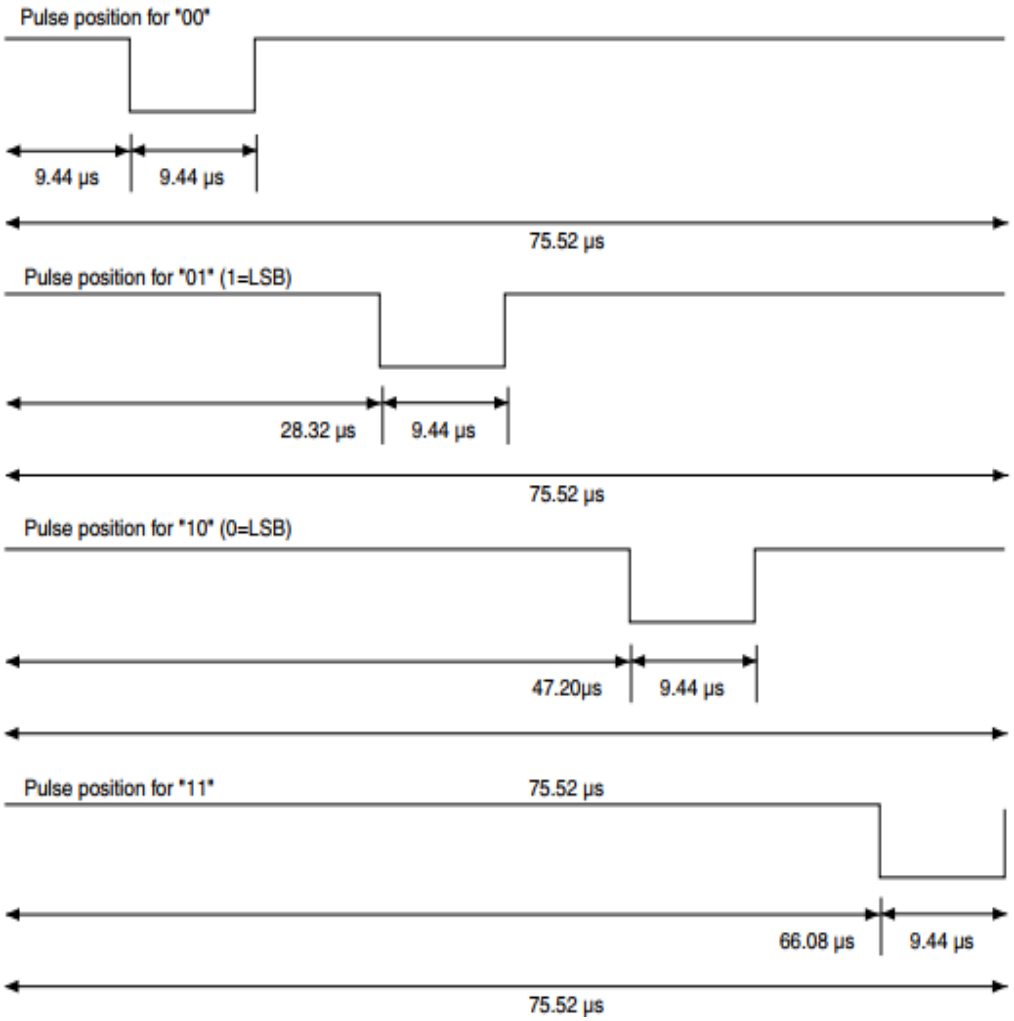

Figure 16: Possible positions of the pauses for the “1 out of 4” method. [46]

An example of an encoded byte with the value 0xE1 (225 in decimal / 11100001 in binary) is displayed in Figure 17. [46]

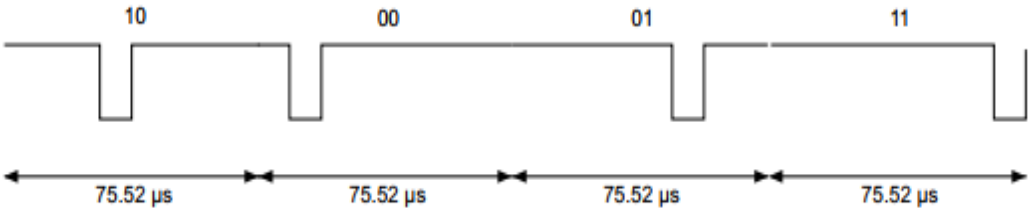

Figure 17: Example of position modulation, with the “1 out of 4” method, of the value 0xE1 (225). [46]

Here again one has to keep in mind that the LSBs are transmitted first followed by the MSBs.

3.5.5 Command Format

Like mentioned in the previous chapters, the VCD sends a command to the tag to get a certain answer. For the given application the Read and Write commands are two essential commands. The command has a structure defined as in the ISO 15693. In Table 3 the required structure of a command is shown, while in Figure 18 an example of a command in the application is shown. The SOF and end of frame (EOF) don't have to be manually added. They will be calculated, taking the flag bits into account by the android framework, and will be automatically added. The cyclic redundancy check (CRC) is a tool to check, if the full command has been transmitted and nothing has been lost along the way. The value of the CRC depends on the rest of the sent command and therefore will also be calculated by the android system and added automatically. The command code can be taken either from the ISO specification or the STMicroelectronics specification. Some more important command functions can be found in Table 4. Both the "Parameters" space and "Data" space in the command are optional and only necessary if the command requires it to function. For example a "Stay Quiet" would not need any further bytes added to the command, while a read or write command always "Data" in form of a target address needs. Special care has to be taken while writing bytes manually, since the simple java byte class handles them as a signed value from -128 to 127 and C handles bytes as unsigned values. Therefore, values of 128, or above need the suffix: "& 0xFF", to shift the value to their right place. Also remarkable is the fact that the read and write commands always address 4 bytes, for example, a read command with the address 128 would result in a response of the values from 128/129/130/131.

Table 3: Request Command Structure

SOF	Request Flags	Command code	Parameters	Data	CRC	EOF

```
29  byte[] cmd = new byte[]{
30          (byte) 0x2A,    // FLAGS Bit 2(2)&4(8)&6(32) has always to be set for this cmd due to ST standard (in Hexadecimal)
31          (byte) 0x20 & 0x0FF,    // READ_SINGLE_BLOCK command
32          (byte) 0x00, (byte) 0x00, (byte) 0x00, (byte) 0x00, (byte) 0x00, (byte) 0x00, (byte) 0x00, (byte) 0x00,
33          address,// block number
34          (byte) (0x00)
35  };
36  System.arraycopy(uid, srcPos: 0, cmd, destPos: 2, uid.length);
```

Figure 18: Code snipped READ SINGLE BLOCK command in 16 bit range.

```
133 byte[] cmd = new byte[]{
134         (byte) 0x0A,    // set FLAGS Bit 2(2)&4(8) due to ST standard (in Hexadecimal)
135         (byte) 0x20,    // READ_SINGLE_BLOCK
136         addressbyteLow,  //Last 8bits
137         addressbyteHigh, //First 8bits
138 };
```

Figure 19: Code snipped READ SINGLE BLOCK command in 32 bit range without ID necessary.

Table 4: Examples of some important NFC-V commands

Command Code	Function
0x01	Inventory
0x02	Stay Quiet
0x20	Read Single Block
0x21	Write Single Block
0x23	Read Multiple Block
0x25	Select
0x26	Reset to Ready
0xB1	Write-sector Password
0xB2	Lock-sector
0xC0	Fast Read Single Block
0xC3	Fast Read Multiple Block

3.5.6 Response Format

The structure of the response is depended on the received command and will always lead with the error flag. If the error flag is set, only the first 8 bits will the conveyed, which hold the value of the error flag. On the other hand if no error occurred the first 8 Bits will contain “0x00” in Hexadecimal representation or “0000 0000” in bit representation. Error flags can result from either a bad command or bad connection with too much background noise. The response from the tag is again transferred from LSB to MSB and needs to be inverted.

3.5.7 Storage Address

Since the first 128 bytes (0-127) are locked, only the bytes with higher address are suitable for use. The stimulator uses the EEPROM storage addresses from 128 to 135 for the stimulation values and the electrical current pointer, which only gives the current address of the latest measurement. In the addresses ranging from 400 to 4.399 electric current measures and the history of the stimulation parameters are stored.

3.5.8 Transmitted Data

As described, the Data transmitted via NFC-V consists of 32 bits per transmission which are further divided into 8 bit segments:

1) Amplitude
2) Frequency (relax time (relax sub slot length EVEN/ODD))
3) Pulse width
4) Symbols per burst
5) History of all the above values
6) Pointer to the address of the latest electrical current values
7) 1,000 pieces of electrical current values (CH1/CH2/CH3)

Those values are implemented into both the java source code as well as the C code to be deconstructed into 8 or 16 bit constructs transmitted and reconstructed by the VCD.

3.6 Test Set Up

To test the function of the stimulator with the updated firmware, a test procedure was implemented. Figure 20 shows the essential ports accessed for testing of the firmware. The channels can be either accessed through the debugger plate on the right side, or through the normal output-pins 1, 3 and 5. The pins 2 & 4 disconnect the battery from the board, only when those two channels are connected the stimulator receives the voltage from the battery.

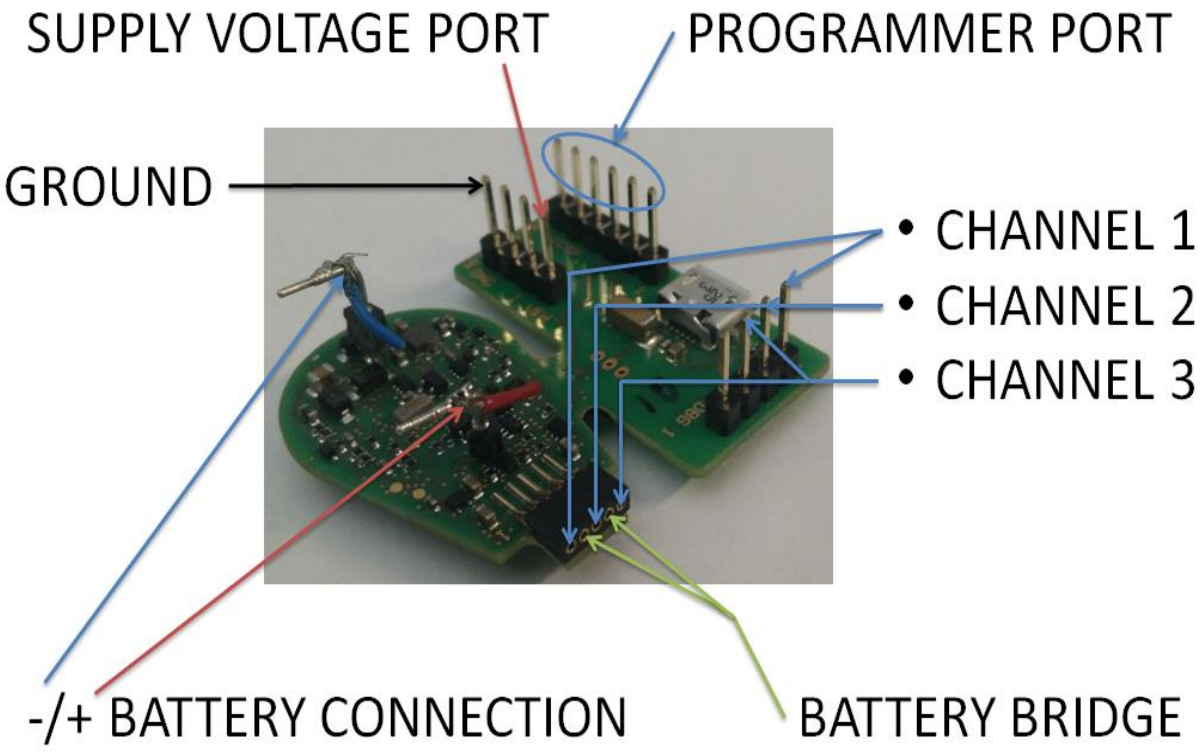

Figure 20: Stimulator Interface Description.

First, the updated C-Code is transferred to the stimulator through the PICkit3, connected to the programmer port. When the internal microcontroller checks are done, the stimulator needs to be disconnected. The end of the internal check is indicated, when the yellow led goes out followed by a flashing of the green led. The

disconnection is important, since the channel of the microcontroller responsible for the output of the stimulation signal channel 1 is also used for programming and debugging of the microchip. As long as the stimulator is powered by the programmer, the channel 1 output will supply a wrong signal.

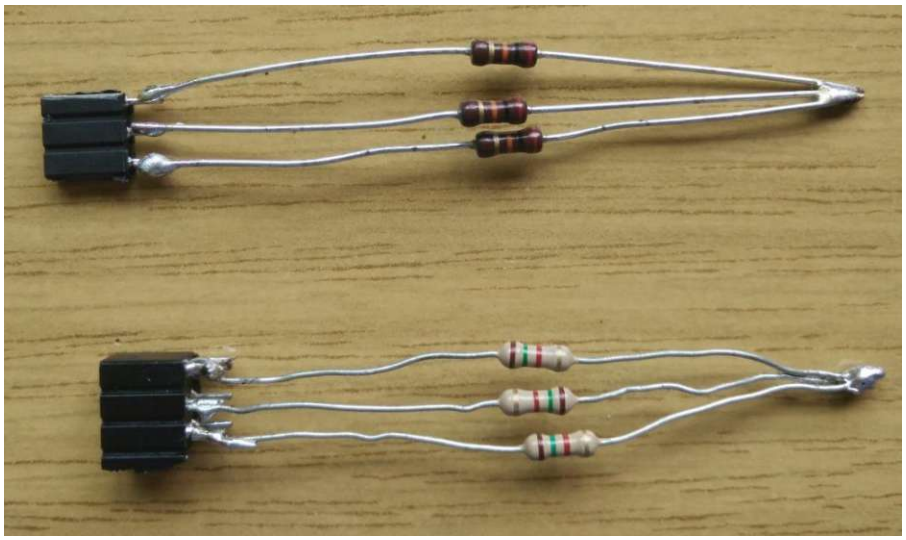

Figure 21: Resistor bridge soldered together with a connection port.

Next, 3 resistors and an oscilloscope are attached to channel 1, 2 & 3. In Figure 21 an easy to apply resistor bridge can be seen, which can be plugged onto the debugger board channels 1-3. The oscilloscope can be attached onto the resistors, on the side of the plug, while the ground can be connected to the ground pin shown in Figure 20.

Then ~3 V should be applied to the battery connection shown in Figure 20, with close attention to the positive and negative connection. Connecting the supply voltage inverted will lead to hardware damage, which needs to be repaired prior to the next usage. The voltage can either be supplied by a battery, or by a DC power supply. A fitting battery and power supply are listed in Table 1. Either before or after the voltage is applied, the battery bridge has to be inserted into output pins 2 & 4, which are shown in Figure 20.

Once the microcontroller receives the necessary voltage the self-test will start, indicated by a yellow led. When the yellow light goes out, the green led will blink every 10 seconds and the signal can be examined on the oscilloscope.

In the end, the stimulator should be connected to the equipment like shown in Figure 22. The oscilloscope should be connected to 2 output channels and calculate the difference to get rid of diverse interferences and spikes in the signal. In Figure 22 the power supply is set to ~3V, and a resistor bridge, like shown in Figure 21, is connected via the debug tool to the simulator.

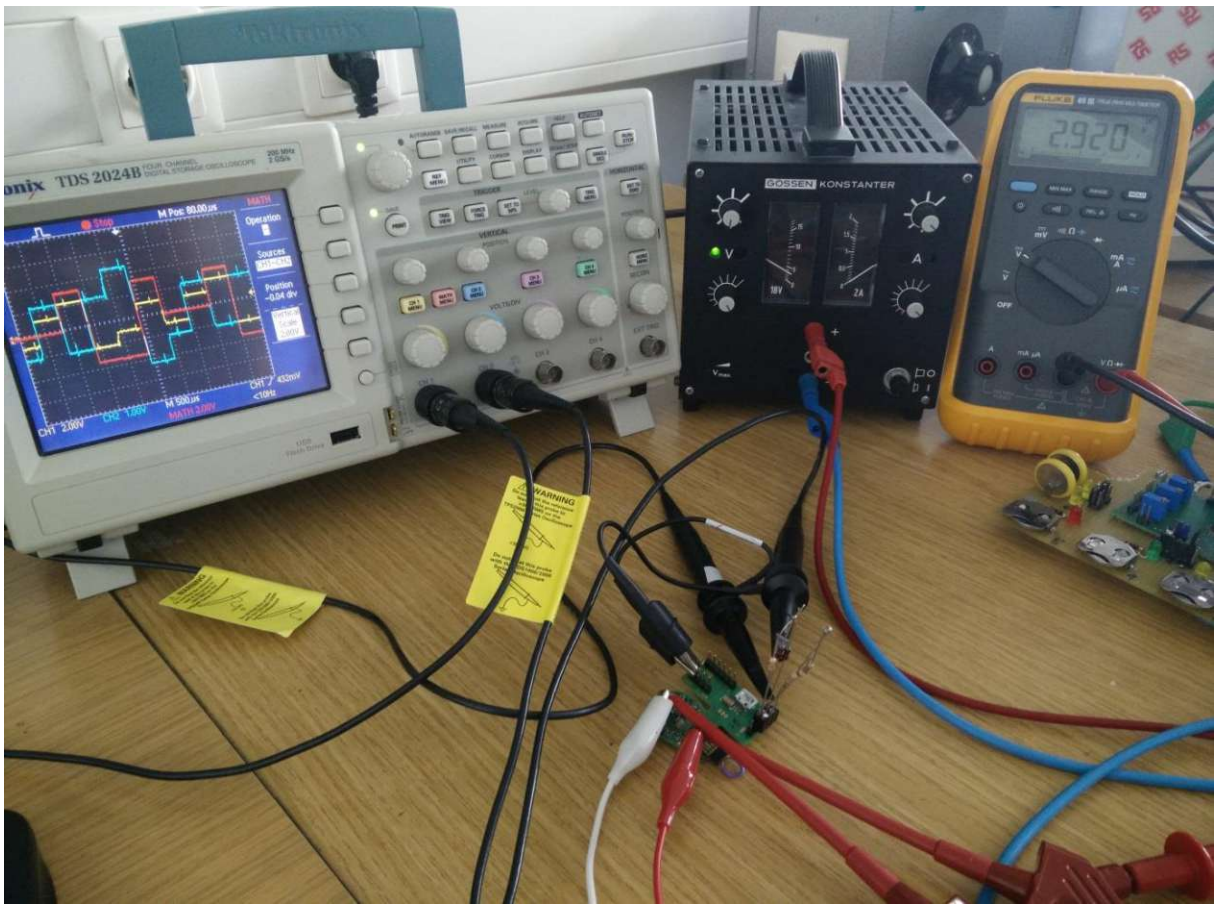

Figure 22: Test Set up. From left to right: Oscilloscope with 2 channels connected to the stimulator output channels, a power supply and a multimeter to monitor the power supply. This equipment was connected to the related ports on the stimulator like shown in Figure 20.

To test the magnetic pen (pen used to set the amplitude of the stimulator via a Hall sensor on the stimulator), the set up shown in Figure 22 will work. However, to test the NFC-V connection the stimulator should be flipped, since the NFC chip is located at the bottom of the board. In addition, it should be kept in mind, that if the stimulator is currently stimulating, the connection can be disconnected due to interference. Therefore, long read or write commands should only be used when the device is not stimulating, or if the stimulator is equipped with an additional NFC antenna.

4 Results

In the following section the development process of the mobile application and its results are shown, including errors during the development and their solutions.

4.1 Front End Interface

The front end interface of this application was programmed for Android, which utilizes the combination of the extensible mark-up language (XML) and the object oriented programming language java. The XML will provide static graphical build up, which will receive its functionality by the underlying java code.

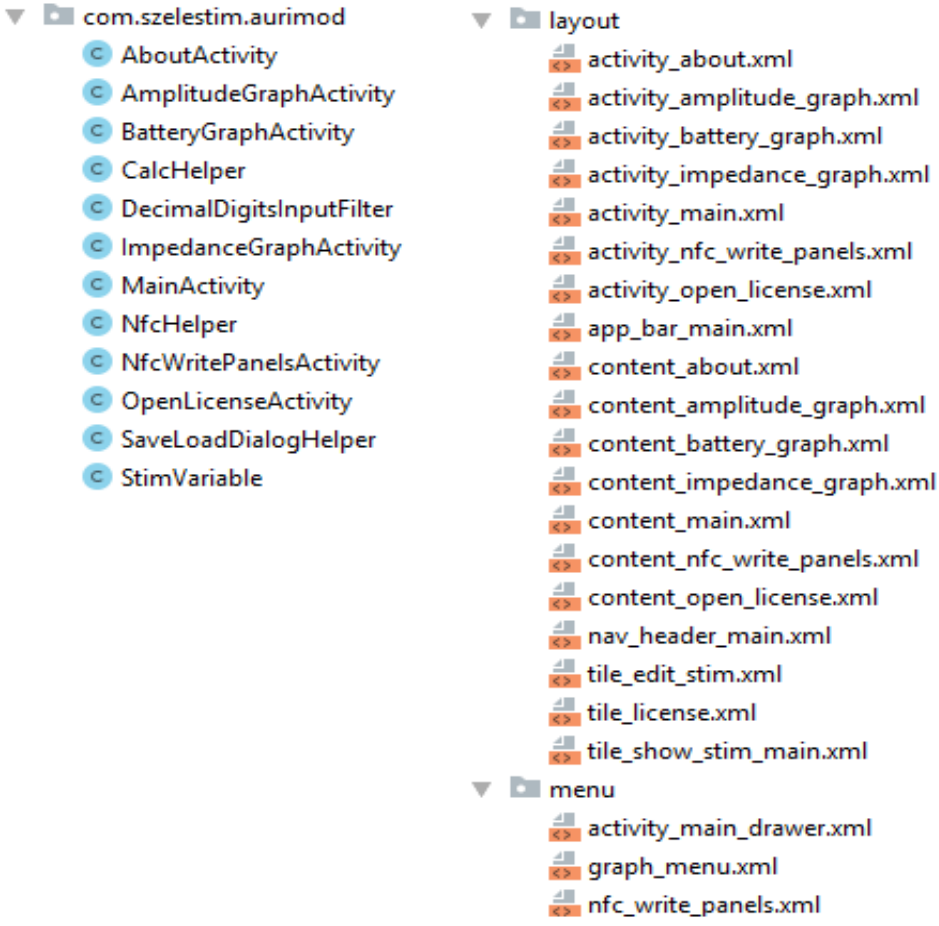

Figure 23: Overview of all the Java classes and XML layouts used for the stimulator - Android application

Android applications use an activity-based system, which links a java class to an xml layout. Each activity consists of a java class and an XML layout, however, standalone java classes are still possible as helper classes and standalone xml layouts are possible as detail views, used by an activity layout. In Figure 23 all necessary Java classes and XML layouts, which were created for this thesis, for the stimulator - Android application version 1.0 are shown.

Two external libraries were used to create this application, the “Seekarc” library and the “Graphview” library. Both are free to use and available at Github (https://github.com/). Those libraries could easily be added with the help of the build.gradle file and a reference to the library. The referencing command used in the code is shown in Figure 24.

```
33      implementation project(':library:SeekArc_library')
34      implementation project(':library:GraphView_library')
```

Figure 24: Reference from the build.gradle file to the external libraries.

The first goal was to achieve a connection with the stimulator via NFC and edit the values inside the EEPROM of the stimulator. Figure 25 depicts the early development of the application.

The version 0.1 was able to read and write the stimulation parameters as well as the ID of the tag. This early version was also able to tell the NFC connection type, which in case of the stimulator, always was NFC-V. In the advancements to Version 0.32, the application dropped the ability to differentiate the NFC types and specialized on NFC-V connections. For feedback to the user about successful or failed connection and data transfer, a text message was implemented, to inform about the success.

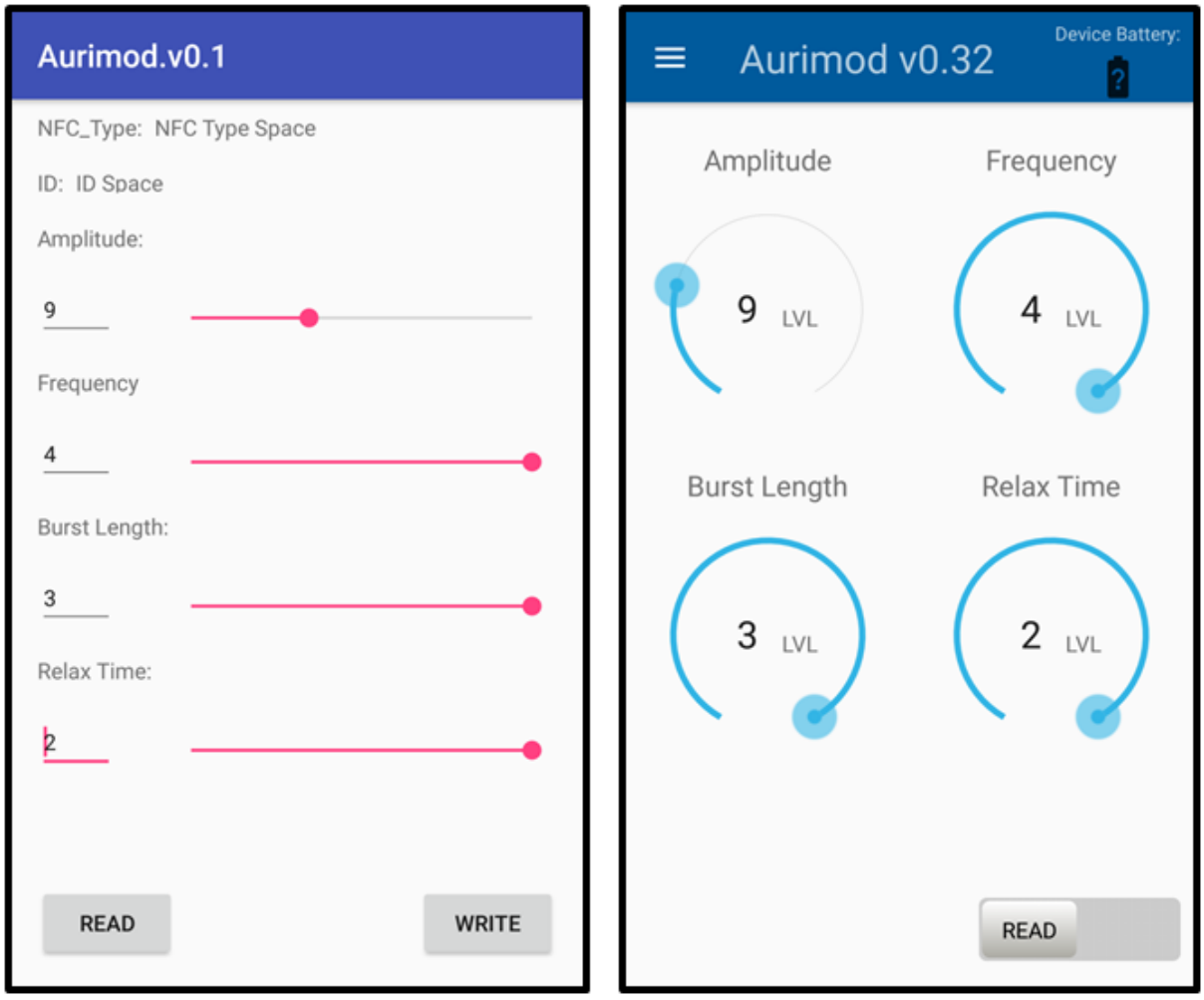

Figure 25: Comparison of early version of the application for Android. Left side shows version 0.1. Right side shows 0.32.

To create a better user experience the font size was increased and the stimulation parameters were divided into panels which can be seen on the right side of Figure 25. Each panel consists of the title, a Seekarc, the value of the parameter and its unit. Those parts are controlled by a “StimVariable.class” object. Also the read and write buttons were merged into a slider.

At the start of the application development, everything was built around the “StimVariable.class”. This java class was designed to generate an object for every stimulation parameter and to create a responsive user interface. Each object was responsible to keep all the views linked together and if the value is changed through

one mean (e.g. read through NFC, changed in the Seekarc, or tipped in through the text field), to keep all the other views up to date. The StimVariable object was furthermore responsible to check the received values for their plausibility, as well as to give feedback if the value was out of bound. All the variables that were necessary to achieve a responsive user interface and written into the StimVariable class can be seen in Figure 26. Also Figure 26 shows that the StimVariable class implements the Serializeable class, which is essential if an object is passed back and forth between activities.

```
27  public class StimVariable implements Serializable{
28      //region Initializing of variables
29      private Integer intVal;
30      private Integer factorVal;
31      private Integer decimaldigits;
32      private Integer maxVal;
33      private Integer minVal;
34      private String hintVal;
35      private String Name;
36      private String Unit;
37      private String IVname;
38      private String EtvTempValue;
39      private boolean etvUserInput = false;
40      private transient View showTileView;
41      private transient View editTileView;
42      private transient ConstraintLayout CLTile;
43      private transient TextView tvLabel;
44      private transient TextView tvUnit;
45      private transient TextView tvValue;
46      private transient TextView tvID;
47      private transient EditText etv;
48      private transient SeekArc ska;
49      private transient ImageView iv;
50      private transient Context CallingContext;
51      private transient Activity CallingActivity;
52      //endregion
```

Figure 26: Code snippet of the all variables a StimVariable object holds.

In version 0.22 the application got equipped with a navigation drawer. The early prototype was configured to present the account currently logged in the app, to

differentiate the users between medical staff and patients, and to manage the permissions. This approach was abandoned for now with the update to version 0.32 and the area was replaced by the logo of SzeleSTIM. The icons for the navigation points were taken from the icons8 website (icons8.com).

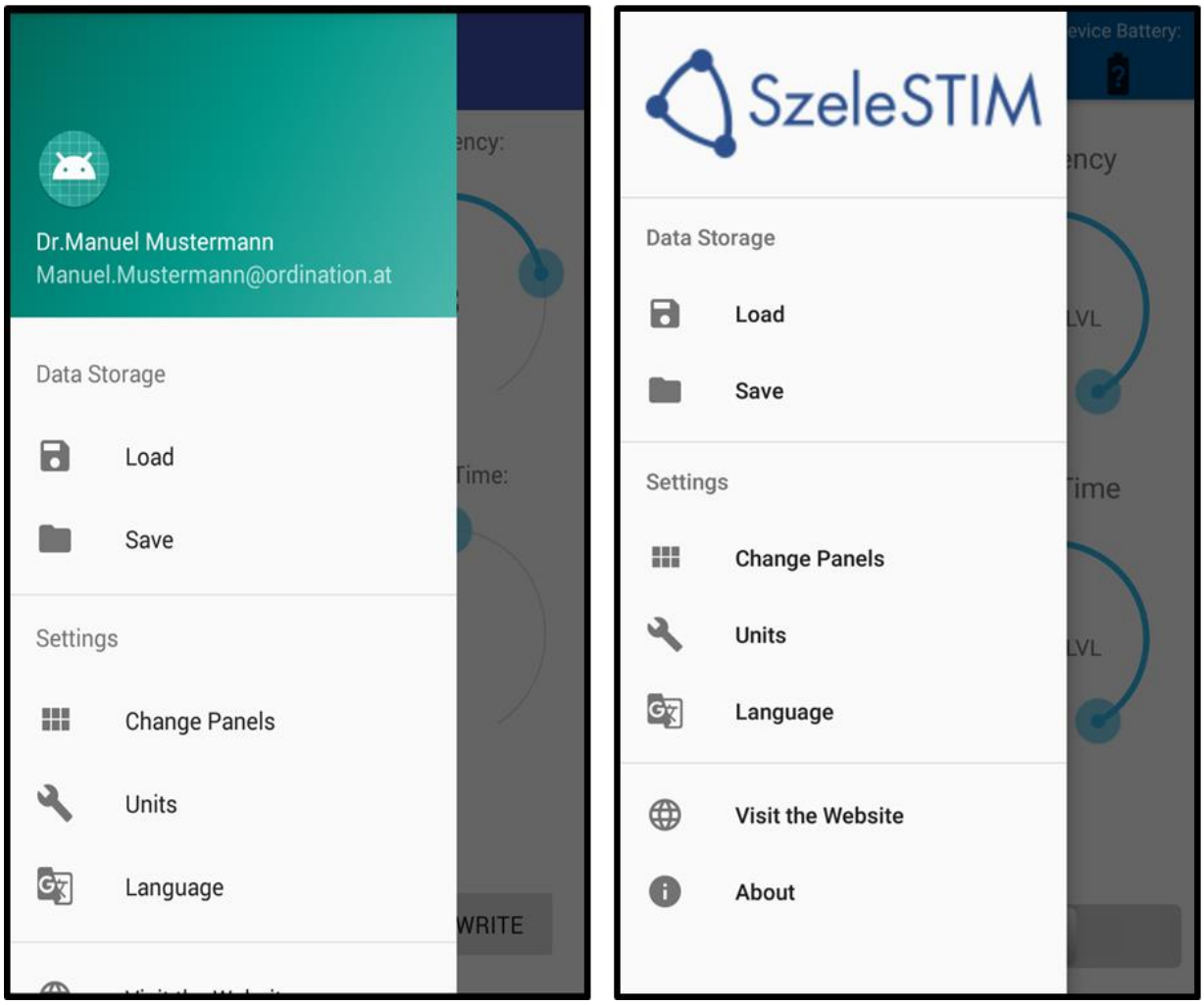

Figure 27: Comparison of early version of the application for Android. Left side shows version 0.22. Right side shows 0.32. Both applications show the implemented drawer.

The application was developed according to the Android design guide. The guide standardizes many elements, to give the user a similar control feeling in all android apps. Therefore the navigation drawer was implemented to be accessible through a swipe from the right edge inwards, as well as a standardized icon on the top left which can be seen in Figure 25 in the top left corner of the version 0.32.

Also an option to save and load stimulation parameters, for quick access to previous used stimulation settings, was implemented. A safety feature only allows to safe values after they have been read from a stimulator, or written to a stimulator without errors. The save dialog shown in Figure 28 is automatically filled in with the stimulator ID and the current date & time but can be changed by the user. Figure 28 moreover shows on the left side the loading screen of previously saved parameters. These data were saved on the on-board memory of the smartphone protected against access from outside the app.

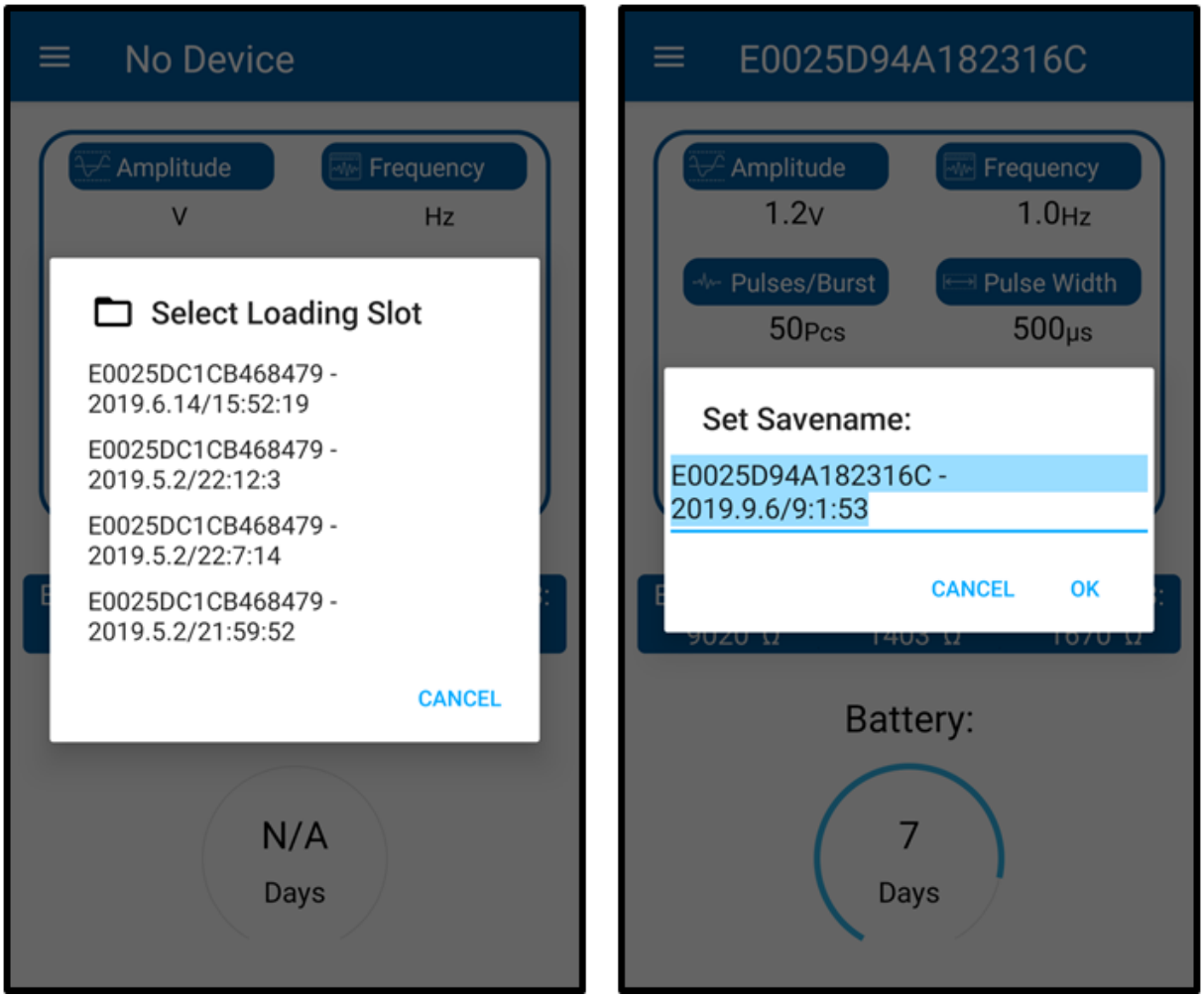

Figure 28: Two screenshots taken from the application version 1.0. The left screen shows the loading menu to restore previously saved vales. The right screen shows the save dialog if a new save slot should be created.

The development was an iterative process, with many little improvements in the user interface. In the end the application was able to read, alter and write the stimulation parameters via an NFC connection. Additionally the feature to roughly calculate the impedance values of the connected stimulation electrodes and to estimate the remaining battery was added. All that information was implemented into the start screen of the application, like shown in Figure 29.

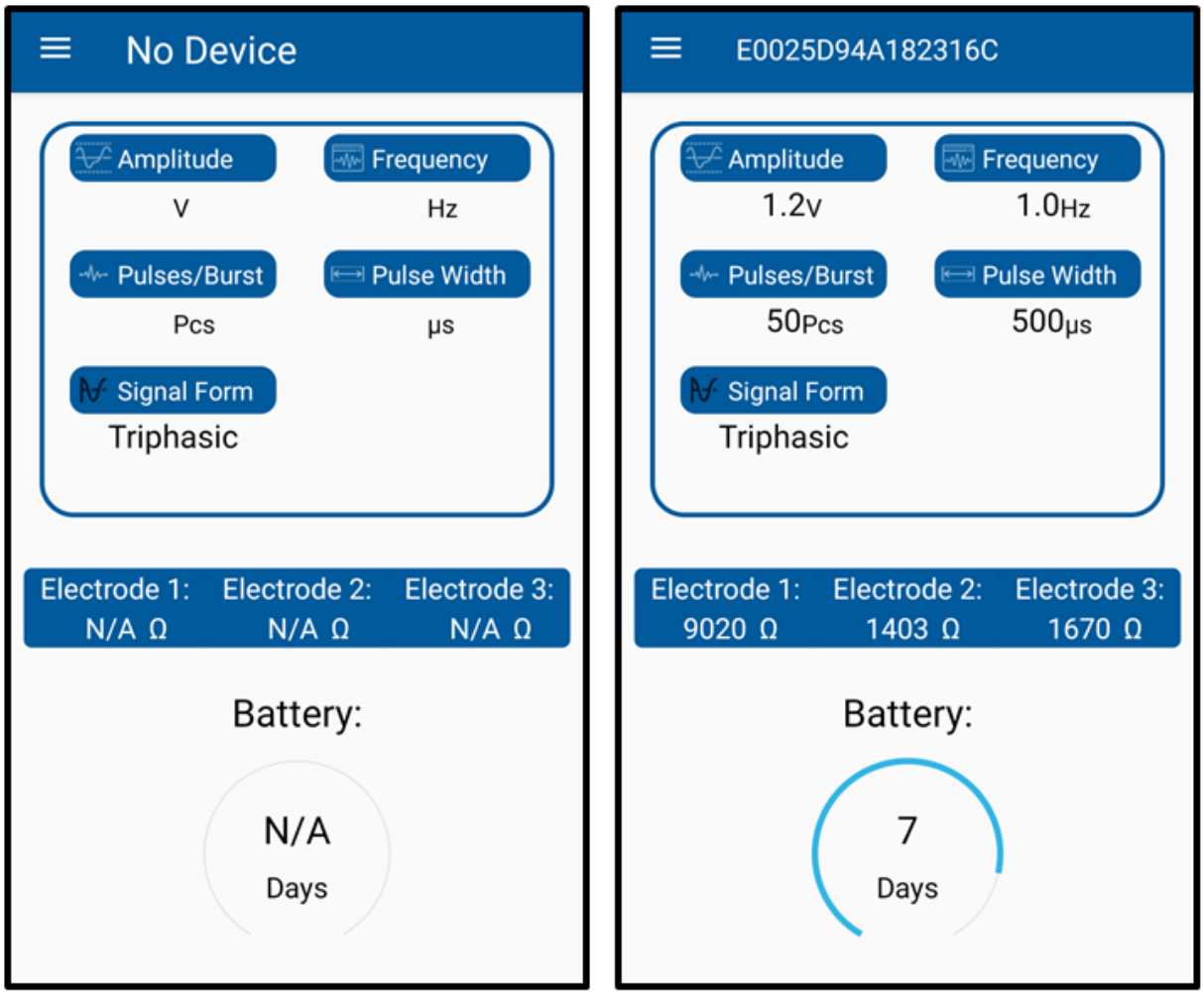

Figure 29: Screenshots of the application version 1.0 start screen. The left screen shows the app before reading a tag and the right one after reading.

On the top bar, the ID of the read or written tag was displayed. In the area below, the panels for the Stimvariable objects are located and they displayed their value after the connection. A bar which shows the impedances was located in the middle of the screen and below the bar the estimated battery life was shown in days and

percentage in form of a Seekarc. The algorithm for the battery and impedance calculation can be found in chapter 4.4 and 4.5. For more details about the shown information new activities were built. Those could be accessed by pressing the area of interest on the main screen or by accessing the navigation drawer, which is shown in Figure 30.

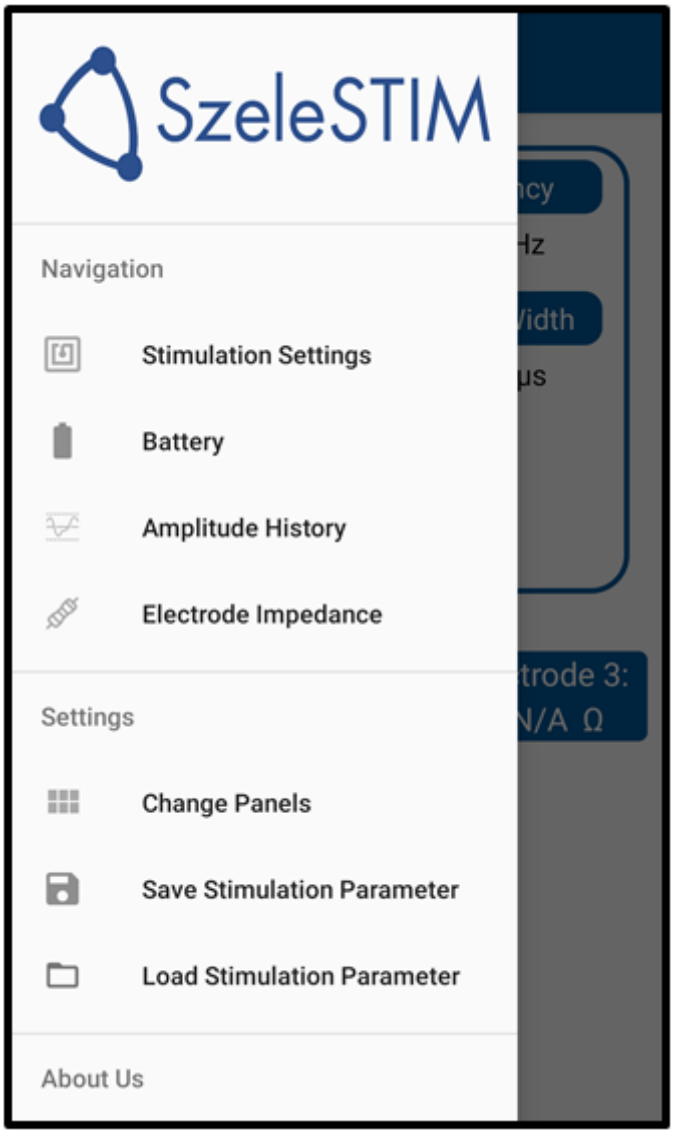

Figure 30: Screenshot of the application version 1.0 opened navigation drawer.

The navigation drawer, like in the previous versions, also allowed access to: the save/load menu, the setting to choose the displayed panels, an about the app section and access to the SzeleSTIM website.

The setting to change the displayed panels was constructed with easy to use checkboxes. The panel setting dialog and its effect on the main activity was depicted in Figure 31. For this example only the Amplitude checkbox was left checked. This change also alters the Panels, shown in the Stimulation Settings screen shown in Figure 32.

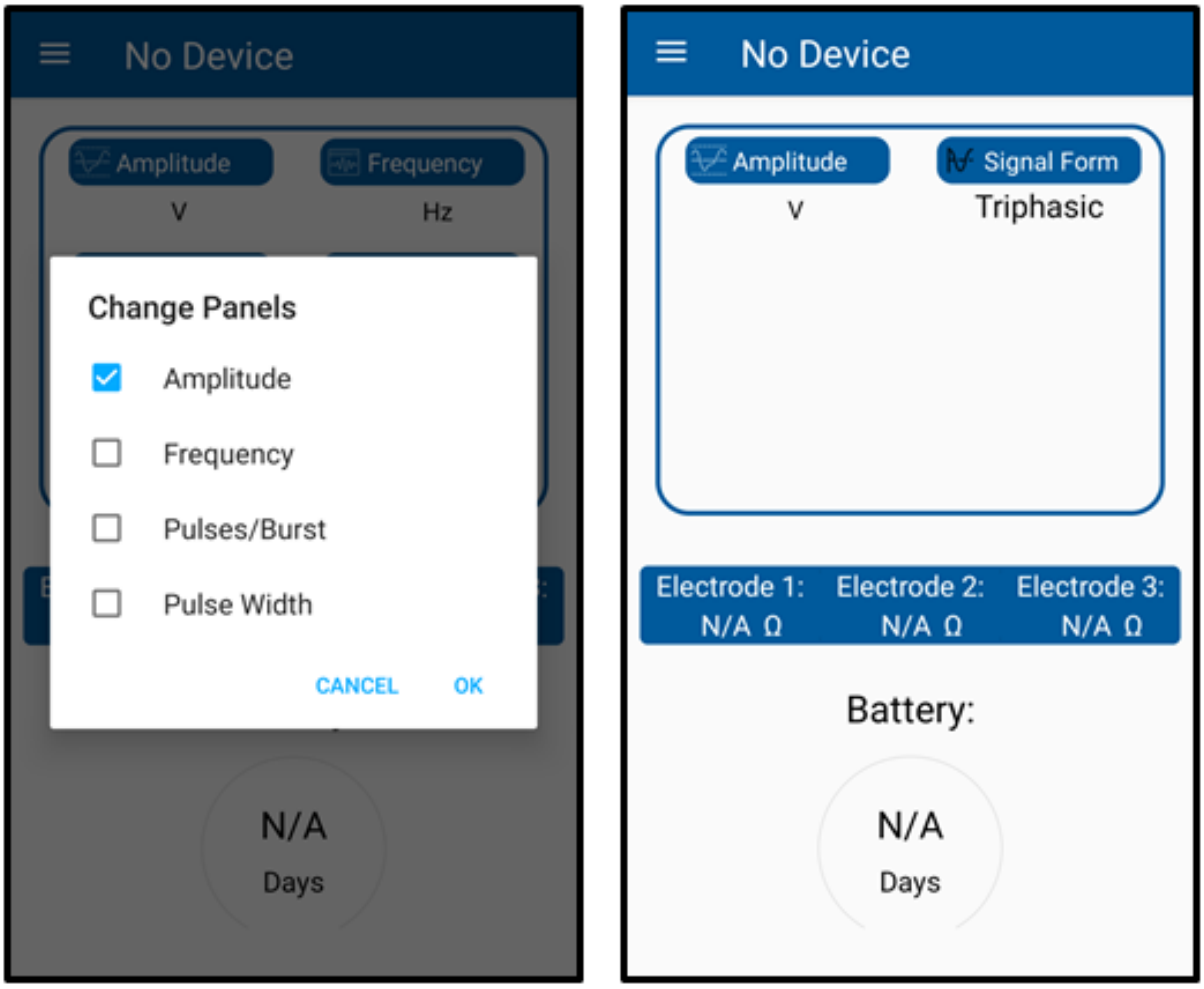

Figure 31: Screenshots of the application version 1.0 change panel dialog. The left screen shows the settings dialog tag and the right one shows the effect of unchecking all boxes except the Amplitude.

Through the navigation drawer five further activities could be called: the NFC write panel activity shown in Figure 32, the battery graph activity shown in Figure 38, the amplitude graph activity shown in Figure 44, the impedance graph activity shown in Figure 42, and the about activity shown in Figure 33.

The NFC write panel activity was devised to decouple the NFC writing process from the main activity and create bigger tiles for easier user interaction. The values can either be adjusted by dragging the button around the circle, or by tapping the value and insert a number with the keyboard. Certain safety measures were implemented to keep the user from inserting invalid input, either by entering invalid numbers or choosing incompatible combinations. Also measures have been taken to prevent writing to a tag, without filling out all values.

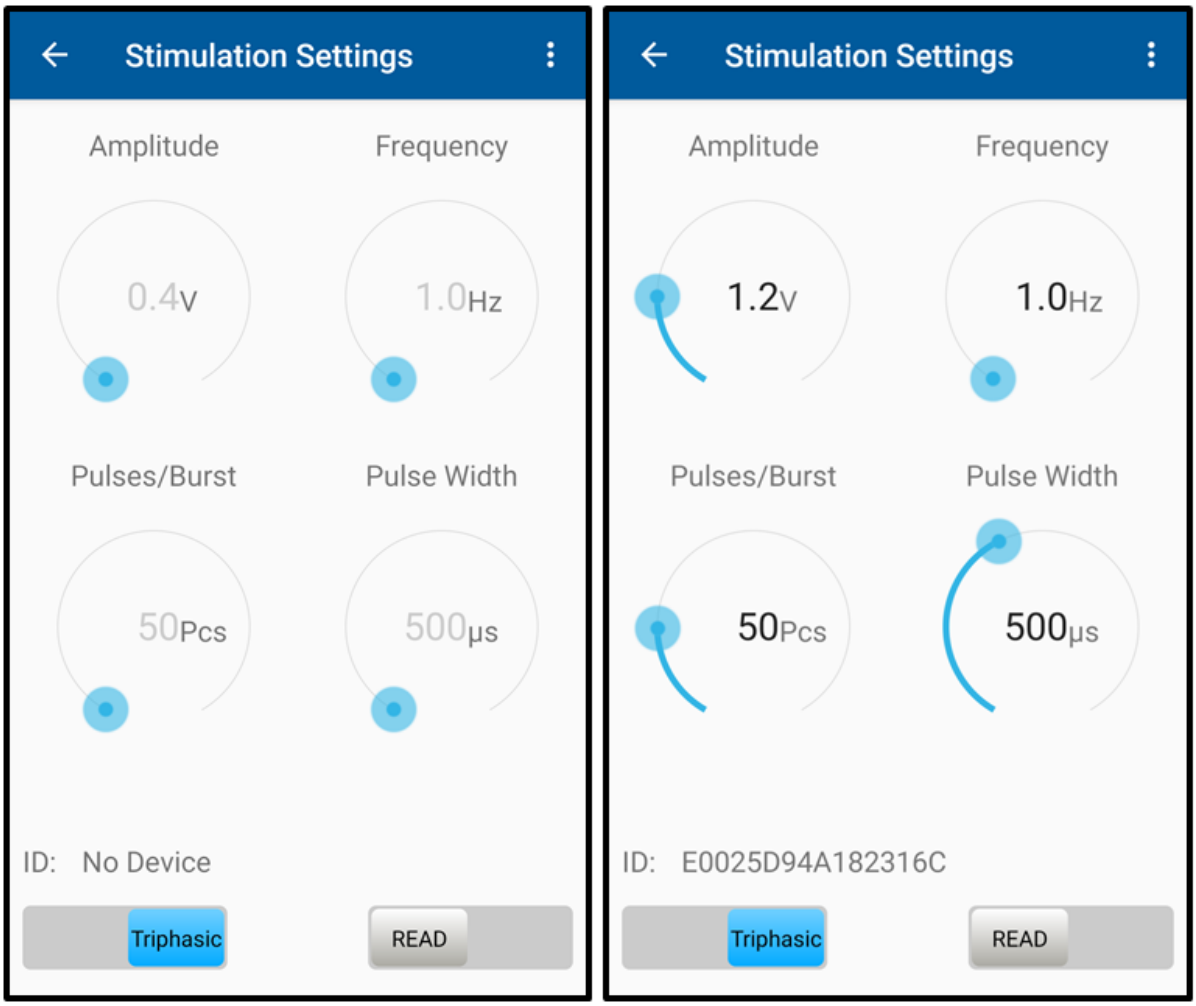

Figure 32: Screenshots of the application version 1.0 Stimulation NFC write panels activity. The left screen shows the app before reading a tag and the right one after reading.

Figure 32 shows on the left screen the initiated NFC write panel activity. When the activity is first opened, default values are shown as hints. The right screen shows the response of the application, after reading a tag with set text views and progress bars.

The about activity was built to be accessible from all activities. It contains the build version, copyright information and the licences that were used for the underlying libraries. The exact details about the licences are shown in the open licence activity which is partly depicted in Figure 33 on the right side.

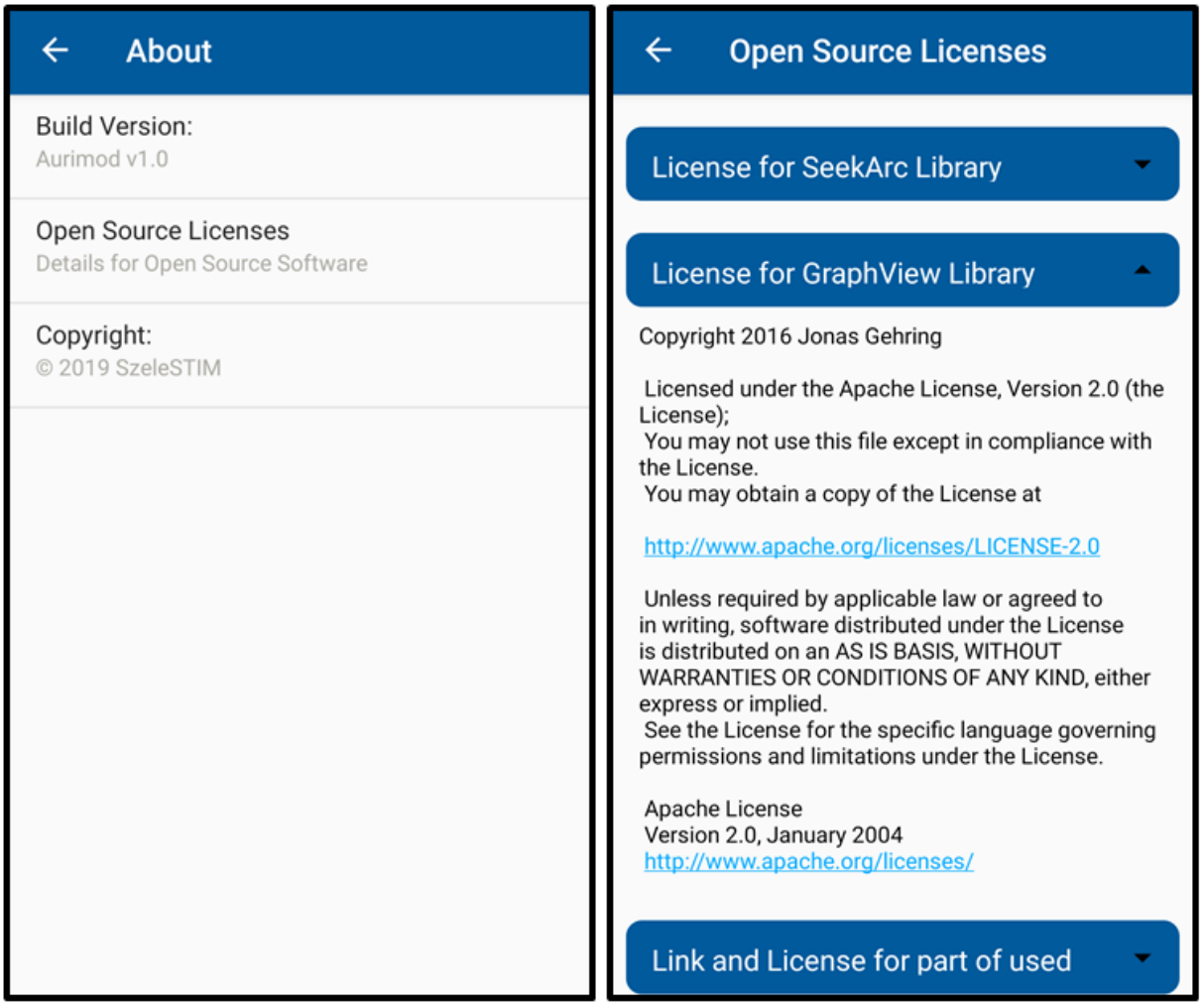

Figure 33: Screenshots of the application version 1.0. The left screen shows the about activity and the right shows the activity containing the references to the open source licences.

4.2 Firmware C

The functioning main C code was supplied by SzeleSTIM GmbH. However, this firmware did not implement any way to communicate via the integrated NFC chip.

The code could only be applied with the pro version of the compiler tool XC8 version 2.00, since otherwise the build would use up more than 100% of the memory of the microcontroller. Furthermore, section 3.4.1 explains why the pro version was used for the firmware.

The following enhancement of the firmware cut down its memory consumption of the original code from 91% of its total memory by around 50%. This refinement needed to be implemented first, since the code, to allow NFC connection and a more precise adjustment of the stimulation parameter, consumed about 25% of the total memory. At the end the memory consumption was down to 66% of the total available.

The cut on the memory consumption was achieved by changing the input variables for the stimulation process, from bulky arrays with hard coded values for every stimulation step, to an agile routine which calculates the input parameters for the ADC. The new array, with the values for the set amplitude, is calculated once the amplitude is changed and overwritten when a new value for the amplitude is adjusted. The calculation for the necessary values was reverse engineered from the existing arrays, which incremented the amplitude in 36 steps by various values to give a specific rise of the amplitude over the steps. The new algorithm always alters the amplitude in steps of 0.1 V from 0 to 6.0 V. This also allows a more precise adjustment of the amplitude through the Android app. The existing solution with a magnetic pen (see above) was reduced to a linear resolution of 0.2 V per step.

```
48  const int16_t gStimPatternTemplate[] = { 2 ,-2 ,-4 ,4 ,-4 ,4 ,2 ,-2 ,-4 ,4 };
49
50  volatile uint16_t gStimPattern[AURI_NUMBER_OF_PULSES_IN_SYMBOL+4] =
51  { 545 ,545 ,545 ,545 ,545 ,545 ,545 ,545 ,545 ,545 };
52  volatile uint16_t gStimPatternSmooth1x[AURI_NUMBER_OF_PULSES_IN_SYMBOL+4] =
53  { 545 ,545 ,545 ,545 ,545 ,545 ,545 ,545 ,545 ,545 };
54  volatile uint16_t gStimPatternSmooth2x[AURI_NUMBER_OF_PULSES_IN_SYMBOL+4] =
55  { 545 ,545 ,545 ,545 ,545 ,545 ,545 ,545 ,545 ,545 };
56  volatile uint16_t gStimPatternSmooth3x[AURI_NUMBER_OF_PULSES_IN_SYMBOL+4] =
57  { 545 ,545 ,545 ,545 ,545 ,545 ,545 ,545 ,545 ,545 };

348 for (uint8_t i=0;i<AURI_NUMBER_OF_PULSES_IN_SYMBOL+4;i++){
349     gStimPattern[i] = (545+((gAmplitudeStep*(4/4)*(90))/(gStimPatternTemplate[i]*10)));
350     gStimPatternSmooth1x[i] = (545+((gAmplitudeStep*(3/4)*(90))/(gStimPatternTemplate[i]*10)));
351     gStimPatternSmooth2x[i] = (545+((gAmplitudeStep*(2/4)*(90))/(gStimPatternTemplate[i]*10)));
352     gStimPatternSmooth3x[i] = (545+((gAmplitudeStep*(1/4)*(90))/(gStimPatternTemplate[i]*10)));
353 }
```

Figure 34: Code snippet of agile amplitude array generator.

In Figure 34 the code, which eliminated huge static arrays with hardcoded values is depicted. The ADC has an offset to create a voltage of 0 V with the magnitude of 545 points. Therefore, the initial arrays were created with the value for 0 V. The demand of the output signal was given by the stimulation method and consisted of a repeating pattern of 6 steps. Each channel had an offset in this pattern of 2 steps to each other. Therefore, a 10 value array was build which repeated itself after the 6^{th} value. This allowed channel 2 to stimulate with the value of i+2 and channel 3 to stimulate with the value of i+4, without getting out of bounds. The Equation in line 349 to 352 in Figure 34 converted the amplitude value to ADC points, while 1 V would count for 50 ADC points. Since the amplitude value was programmed as an integer, it was 10 times the real value to cover a range from 0.1 V to 6.0 V. The factor 0.9 was inserted due to internal conversion errors.

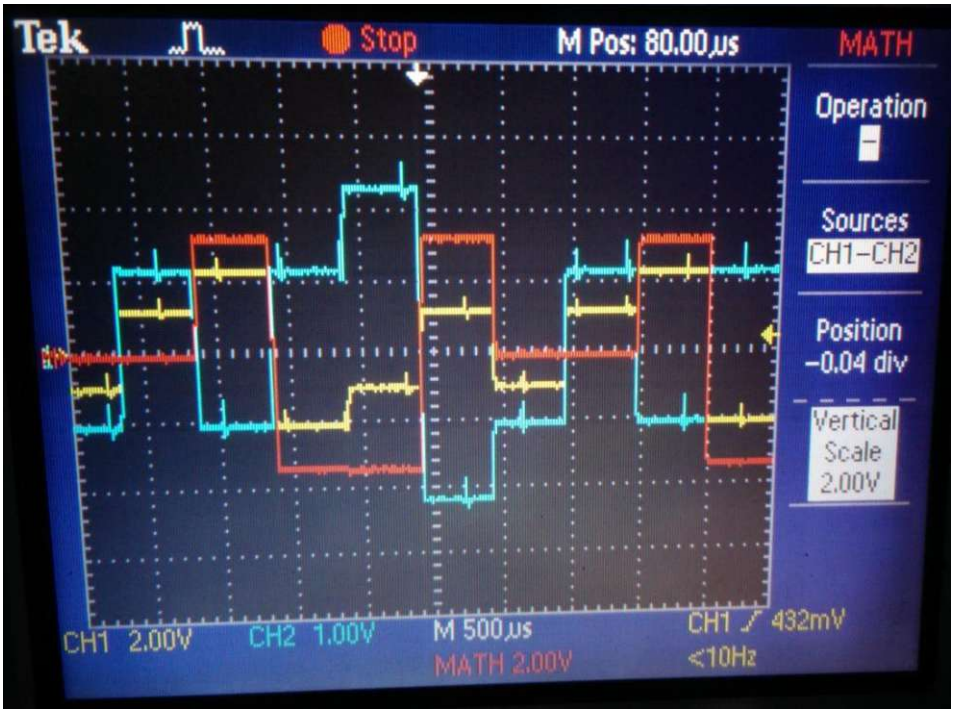

Figure 35: Oscilloscope screen while 2 channels of the stimulator are connected to CH1 (yellow) & CH2 (blue) and the difference between the two inputs is calculated (red).

The output of the stimulator with the implementation of the code in Figure 34 can be seen in Figure 35. In this figure channel 2 only has a resolution half as high as

channel 1, to visualize that the small step is half as high as the bigger one. Although the signal of the stimulator channels are afflicted with artefacts, only the difference between signals matters to the body during the treatment with the stimulator, which is free of interference. For recording the amplitude was set to 2.1 V and the pulse width to 500 µs.

Additionally to the feature to manipulate the amplitude, a function to edit the stimulation parameters like frequency, pulses per burst and pulse width was implemented into the firmware. This was achieved by reading or writing values into the EEPROM of the NFC chip after conversion. It was implemented so that the EEPROM would be checked for changes, after each burst and overtaken if changes would be present.

```
777   if (pageReadings[1]>=1 && pageReadings[1]<=250 &&
778           pageReadings[2]>=1 && pageReadings[2]<=250 &&
779           pageReadings[3]>=15 && pageReadings[3]<=100){
780       if ((2000000UL/pageReadings[1])-(pageReadings[2]*6UL*(pageReadings[3]*10))>AURI_MIN_RELAX_SUBSLOT_DURATION){
781           //symbols
782           gCurrentSymbolsPerOddBurst = pageReadings[2];
783           gCurrentEvenRelaxDuration = (gCurrentEvenRelaxDuration*7)/10;
784           //frequency to relax time
785           gCurrentOddRelaxDuration =
786                   ((2000000UL/pageReadings[1])-(gCurrentSymbolsPerOddBurst*6UL*(pageReadings[3]*10)));
787           gCurrentEvenRelaxDuration =
788                   ((2000000UL/pageReadings[1])-(gCurrentSymbolsPerEvenBurst*6UL*(pageReadings[3]*10)));
789           //symbolwidth
790           TMR5_ChangeReload(0xFB50-((pageReadings[3]-15)*80));//0xFB50 = 150µs
791           gCurrentSymbolsPerOddBurst =(gCurrentSymbolsPerEvenBurst*7)/10;
792       }else{
793           //Do nothing
794       }
```

Figure 36: Code snippet of the conversion algorithm from the written EEPROM values to usable values in the C-Code. The EEPROM readings were stored as 8 bit values into the pageReadings array with 4 slots.

First the read values from the EEPROM were verified. If they were within a plausible range they were converted. The stimulation pulses also switched between an even and odd stimulation pattern, which needed to be considered into the code with a factor of 0.7 for some values (e.g., difference between even and odd burst length, may be modified in future versions). Moreover, the frequency needed to be converted to the time between the stimulation bursts. The timer 5 was responsible for the pulse

width of each signal. The timer input required the value in µs, converted into a point system. This whole process can be seen in Figure 36.

Routines were developed to automatically safe the stimulation parameters and the current of each channel every 30 minutes into the NFC EEPROM, for later evaluation on a smartphone. Due to memory shortage in the EEPROM and the criteria to save data during the whole treatment, redundancies in the saved data were eliminated until only the value of the electrical current and the stimulation parameters remained. The EEPROM storage is divided into 8,192 elements with the size of 8 bit. The requirement for the routine was to safe 12 – 8 bit elements per save point and it should last for around 11 days (264 hours). Since space was limited only 7792 – 8 bit elements were available, which led to a save time interval of 30 minutes. However, the algorithm was designed so that the latest value is overwritten every minute, to ensure that the latest values are always available. The code to write the current values of channel 1 and 2 are shown in Figure 37 as example code. First a 4 times 8 bit array was constructed and then sent to the I2C routine which connected the microcontroller to the NFC EEPROM.

```
848   //save 1)CH1-I 2)CH2-I
849   gSaveBits[0] = (uint8_t) (gMeasurementI1P >> 8);
850   gSaveBits[1] = (uint8_t) gMeasurementI1P & 0xff;
851   gSaveBits[2] = (uint8_t) (gMeasurementI2P >> 8);
852   gSaveBits[3] = (uint8_t) gMeasurementI2P & 0xff;
853   gRetval = DataStorage_EEPROM_WritePage(SaveDataPointer,gSaveBits);
```

Figure 37: Conversion of 16 bit values to 8 bit values to be able to store them via I2C in the EEPROM.

4.3 NFC Data Transfer

The section 4.2 showed how the firmware stored and received the necessary data to and from the EEPROM of the NFC chip via I2C connection. Since the NFC chip takes the role of the passive element no further action of the firmware was required.

The main activity and the stimulation settings activity were conceptualized to perform only a short NFC connection to receive the current, or sent new parameters. The battery, impedance and amplitude activity, on the contrary where designed to read out all of the values written into the EEPROM by the firmware. How the connection was established can be read in section 3.5.

Since for many connection attempts the stimulator will be worn on the neck and therefore out of sight, a tone was implemented, to give feedback about the success or fail of a read or write process, into all activities that are able to perform the short NFC communication. For the long connection two tones were programmed into the application, a short tone that indicates the start of the connection and a long tone signalling the end of the reading process.

The necessary code to establish the connection was stored in the NFC Helper class, to be called from every activity.

4.4 Battery Forecast

The calculation of the remaining battery of a stimulator was first planned to be dependent of the battery voltage. But since the used battery provides a constant voltage until the time of depletion, this method was not applicable and can be seen in Figure 38 on the left screen where the collapsing battery voltage is shown. On the right side of Figure 38 the new calculation is shown. The battery drain per 30 minutes is calculated with Equation 1. In this test the stimulation amplitudes shown on the left side of Figure 44 were used.

$$Bat_{\mathrm{Drain}} = \frac{1}{2}\mathrm{h} \times \left(\left(\frac{\left(\frac{(I_1+I_2+I_3)}{\sqrt{3}} \times 2 \right) + 0.465\ \mathrm{mA}}{1.705\ V} \times V_{\mathrm{supp}} \times T_{\mathrm{Tri}} \times N_{\mathrm{b}} \times f_{\mathrm{rep}} \times 0.75 \right) + 0.641\mathrm{mA} \right) \quad (1)$$

In Equation 1 the calculation of the battery drain per thirty minutes is shown in mAh represented by Bat_{Drain}. The variables I_1, I_2 & I_3 represent the measured currents on the output channels in mA and V_{supp} stands for the supply voltage which has to be

3.38 V for amplitudes below 3 V and 6.47 V for amplitudes above 3 V and below 6 V. Furthermore, in Equation 1 T_{Tri} stands for the length of one symbol which equals to 6 times the pulse width in seconds, N_b represents the pulses per burst times a factor of 0.85 which takes into consideration the ratio of the even to odd pulses and f_{rep} represents the frequency in which the bursts occur in Hz.

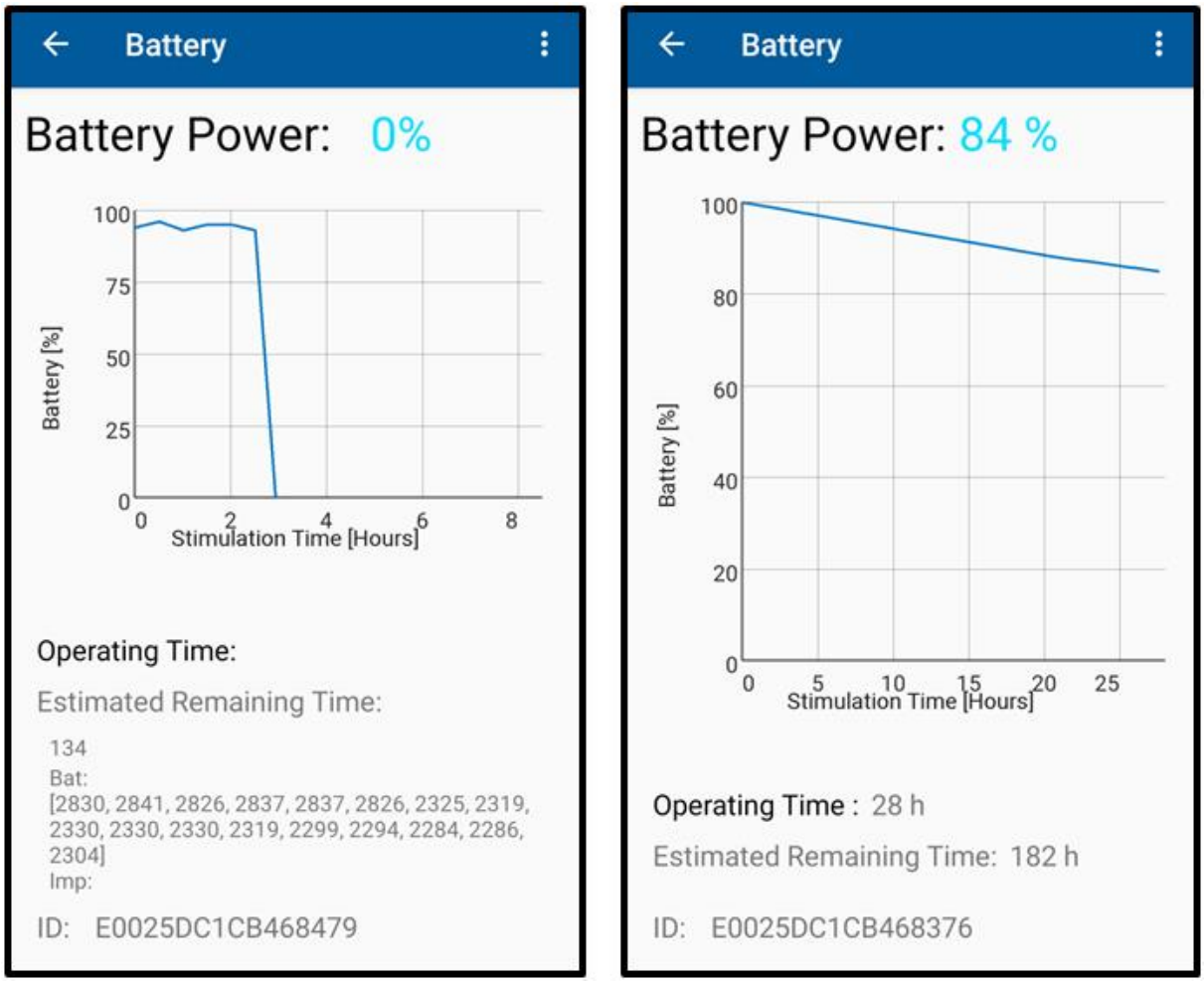

Figure 38: Screenshots of the application version 1.0 battery graph activity. The left screen shows the remaining battery power calculated by the battery voltage and the right screen shows the calculation of the remaining battery power by calculating the used mAh.

An analysis of all the electrical current consumption was used to derive an Equation, which tracked the consumed charge in mAh of the battery. This Equation was only depended on the voltage and electrical current used by the output channels. Once

the value of the mAh of the battery specification was reached, the battery was depleted. The code for the calculation can be seen in Figure 39.

```
21      //Convert I to µA
22      double I1_Val_myA=ConvertI(I1_Val);
23      double I2_Val_myA=ConvertI(I2_Val);
24      double I3_Val_myA=ConvertI(I3_Val);
25
26      //Constants
27      double etha_DCDC = 0.682; //efficiency
28      double I_tot = 0.641; //mA
29      double Iq = 0.155; //mA
30      double V_bat_min = 2.5; //V
31
32      //Semi Constants
33      int T_tri = 6*(PulseWidthpoints*10); //= 6*PulseWidth (µs)
34      double f_rep = (Frequencypoints/2); //Hz
35      double D = 0.75; // (t_on/(t_on+t_off))
36      double N_b = PulsesPerBurst*0.85;//(50+35)/2;(OddPulses + EvenPulses)/2
37
38      //Calc Part
39      double I_pk = (I1_Val_myA+I2_Val_myA+I3_Val_myA)/3; //µA
40      double I_RMS = (I_pk/1000) * 1.1547;//=(I_pk/1000)*root(3)*(2/3)  (1000 to get to mA from µA)
41
42      double V_Supp_min;
43      if(U_Val<=30){
44          V_Supp_min = 3.38;//V
45      }else{
46          V_Supp_min = 6.47;//V
47      }
48      double V_RMS = V_Supp_min * 1.732; //V_Supp_min * root(3)
49      double P_pulse = (V_RMS * I_RMS * 1.732) + (3*Iq*V_Supp_min); //mW
50      //T_tri in µs & frep in 1/seconds therefore "/1000000"
51      double P_total = ((T_tri * N_b * f_rep * D * P_pulse)/1000000);
52      //Q_Battery Drain per half hour in mAh
53      double BatteryDrain = 0.5*(((P_total)/(V_bat_min * etha_DCDC))+I_tot);
```

Figure 39: Code snippet of the battery consumption calculation, located in the Calc Helper class.

At the top of Figure 39 all the constants are gathered and converted to useable values in the lines from 21 to 36. Next the root mean square of the average output current was calculated in the lines 39 to 40. Since the boost converter was only activated for stimulation amplitudes above 3 V the support voltage needed to represent this fact is shown in the lines 42 to 47. Then the computation of the total used power of one pulse and then of the whole burst was calculated with the Equations seen in lines 48 to 51. In the end, the battery drain per half hour in mAh

was calculated. Those values were tracked and summed up and subtracted from the battery mAh, to receive the remaining battery power.

Since the short NFC connections only provides the last stimulation values of the EEPROM only a preview, with limited accuracy, could be given on those screens. For detailed information about the current remaining battery power a long read needed to be performed. After the long read the new value is presented in the main screen and the detailed battery view, like shown in Figure 39 on the right screen. In this figure also the lower energy consumption, by dropping from amplitude of 2 V to 1.2 V, can be seen after 21 hours.

4.5 Impedance Calculation

One of the main topics was to identify electrodes with bad and or lost connection. The impedance of every channel was calculated to rate the quality of the connection, to achieve this goal.

Disconnected electrodes were identified by checking the electrical current on each channel. For disconnected electrodes the current would be 0. The identification of disconnected electrodes was realized by checking if the measured current was beneath 596 ADC points, which equals to 32 µA. To calculate the impedance, the currents from the connected Y circuit were transformed to values of a delta circuit and then the resistors of this circuit were calculated and transformed back into the Y circuit. Those steps are shown mathematically in Equation 2, Equation 3 and Equation 4. The implementation of those Equations into the code can be seen Figure 41.

$$I_\Delta = \begin{bmatrix} I_{1,2} \\ I_{3,2} \\ I_{1,3} \end{bmatrix} = \begin{bmatrix} \frac{I_1 + I_2 - I_3}{2} \\ \frac{-I_1 + I_2 + I_3}{2} \\ \frac{I_1 - I_2 + I_3}{2} \end{bmatrix} \qquad (2)$$

Equation 2 shows the transformation of the current form the Y to triangle circuit, which was derived from an equation system. In Equation 2 the variables I_1, I_2 & I_3 represent the measured currents on the output channels and I_Δ stands for the currents of the triangle system and both of them have the unit A.

$$R_\Delta = \begin{bmatrix} R_{1,2} \\ R_{3,2} \\ R_{1,3} \end{bmatrix} = \begin{bmatrix} \frac{U \times 1.5}{I_{1,2}} \\ \frac{U \times 1.5}{I_{3,2}} \\ \frac{U \times 1.5}{I_{1,3}} \end{bmatrix} \qquad (3)$$

Equation 3 shows the calculation of the R_Δ, which represents the impedance of the triangle circuit in Ω, by applying Ohm's law with the set amplitude represented by U in V and the currents calculated with Equation 2 in A.

$$R_Y = \begin{bmatrix} R_1 \\ R_2 \\ R_3 \end{bmatrix} = \begin{bmatrix} \frac{R_{1,2} \times R_{1,3}}{R_{1,2} + R_{1,3} + R_{3,2}} \\ \frac{R_{1,2} \times R_{3,2}}{R_{1,2} + R_{1,3} + R_{3,2}} \\ \frac{R_{3,2} \times R_{1,3}}{R_{1,2} + R_{1,3} + R_{3,2}} \end{bmatrix} \qquad (4)$$

Equation 4 shows the transformation of the impedances, from the triangle-circuit values calculated with Equation 3, to the Y-circuit values represented by R_Y. Equation 4 has been derived from the wiring diagram, shown in Figure 40.

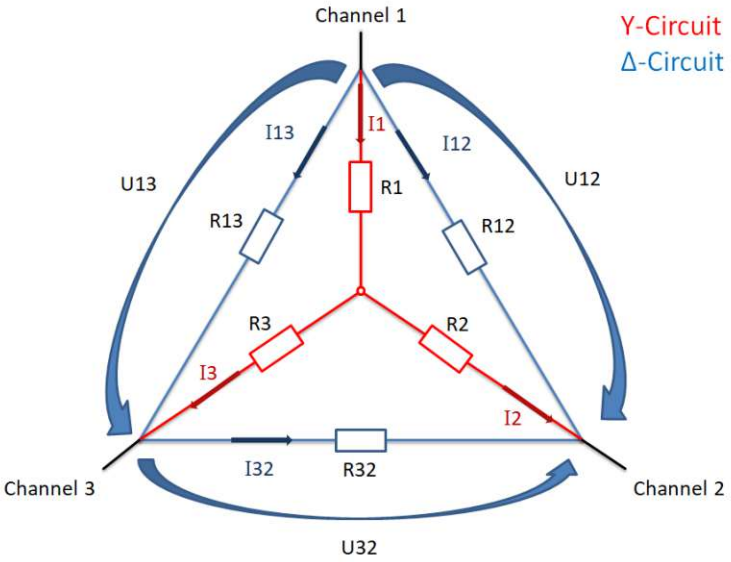

Figure 40: Y-delta circuit for calculations

```
120     //Correlating the current values of the star/triangle to their max values of the clamp
121     //obtained through solving the equation system of the triangle circuit
122     double I_12 = (I1_Val_myA + I2_Val_myA - I3_Val_myA) / 2;
123     if (I_12 < 1) {//negative currents are not allowed
124         I_12 = 1;
125     }
126     double I_23 = (I2_Val_myA + I3_Val_myA - I1_Val_myA) / 2;
127     if (I_23 < 1) {
128         I_23 = 1;
129     }
130     double I_31 = (I3_Val_myA + I1_Val_myA - I2_Val_myA) / 2;
131     if (I_31 < 1) {
132         I_31 = 1;
133     }
134
135     // For CH1 peak -> U2 = U3 -> dU23 =0 &dI23 = 0; U[µV] // I[µA] -> R[Ohm]
136     //U_step 10^U[V] due to int step & positive peak to negative peak = 1.5^U_Val
137     //(Ustep^1.5/10)V^1000m^1000µ -> Ustep^100^1000^1.5
138     double R12 = ((U_step * 1.5 * 100 * 1000) / I_12);
139     double R13 = ((U_step * 1.5 * 100 * 1000) / I_31);
140
141     // For CH2 peak -> U1 = U3 -> dU13 =0 &dI13 = 0; and R23 can be calculated
142     double R23 = ((U_step * 1.5 * 100 * 1000) / I_23);
143
144     // Apply triangle -> Y transformation (taken from Wikipedia )
145     ImpedanceVal[0] = (int) Math.round((R12 * R13) / (R12 + R13 + R23));
146     ImpedanceVal[1] = (int) Math.round((R12 * R23) / (R12 + R13 + R23));
147     ImpedanceVal[2] = (int) Math.round((R13 * R23) / (R12 + R13 + R23));
```

Figure 41: Code snippet of the impedance calculation of the connected electrodes, located in the Calc Helper class.

For testing of these routines, a resistor bridge like shown in Figure 21 was soldered together, consisting of resistors with 6 kΩ, 500 Ω and 2.2 kΩ. The resulting graphs can be seen in Figure 42 and indicate the rough values of the resistors. Since the values of the electrical currents are measured with a tolerance of +/-20%, the values of the impedance are only rough estimates, but are still able to indicate the quality of the signal with a high tolerance.

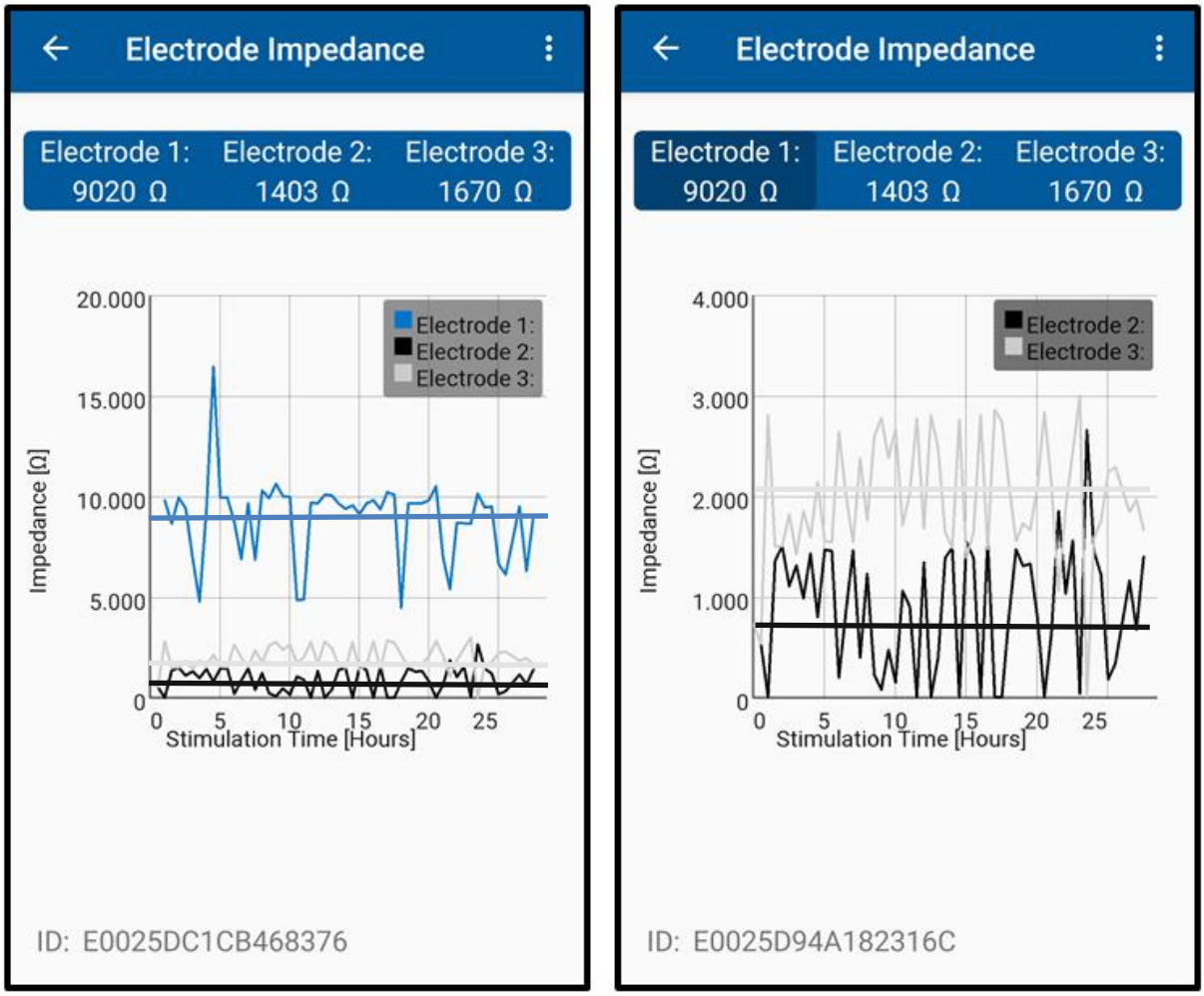

Figure 42: Screenshots of the application version 1.0 impedance graph activity. The left screen shows all the impedances of the connected resistor bridge and the right one showes the details of electrode 2 & 3 by dismissing electrode 1. The connected resistor bridge had values of 6 kΩ, 500 Ω and 2.2 kΩ. Including mean values for each graph which were added for the pictures with approxemately 9212Ω, 836Ω & 2072Ω.

To see the current impedances of the electrodes, only a short connection was needed. To see the full history on the other hand like shown in Figure 42, a long read was necessary to receive all the data needed.

The graph was devised to be able to hide or show single graphs by pressing the electrode button in the bar. This was implemented to show the detailed progress of the curve without interference of other electrodes and is shown in Figure 42, where the curve of electrode 1 was removed from the view.

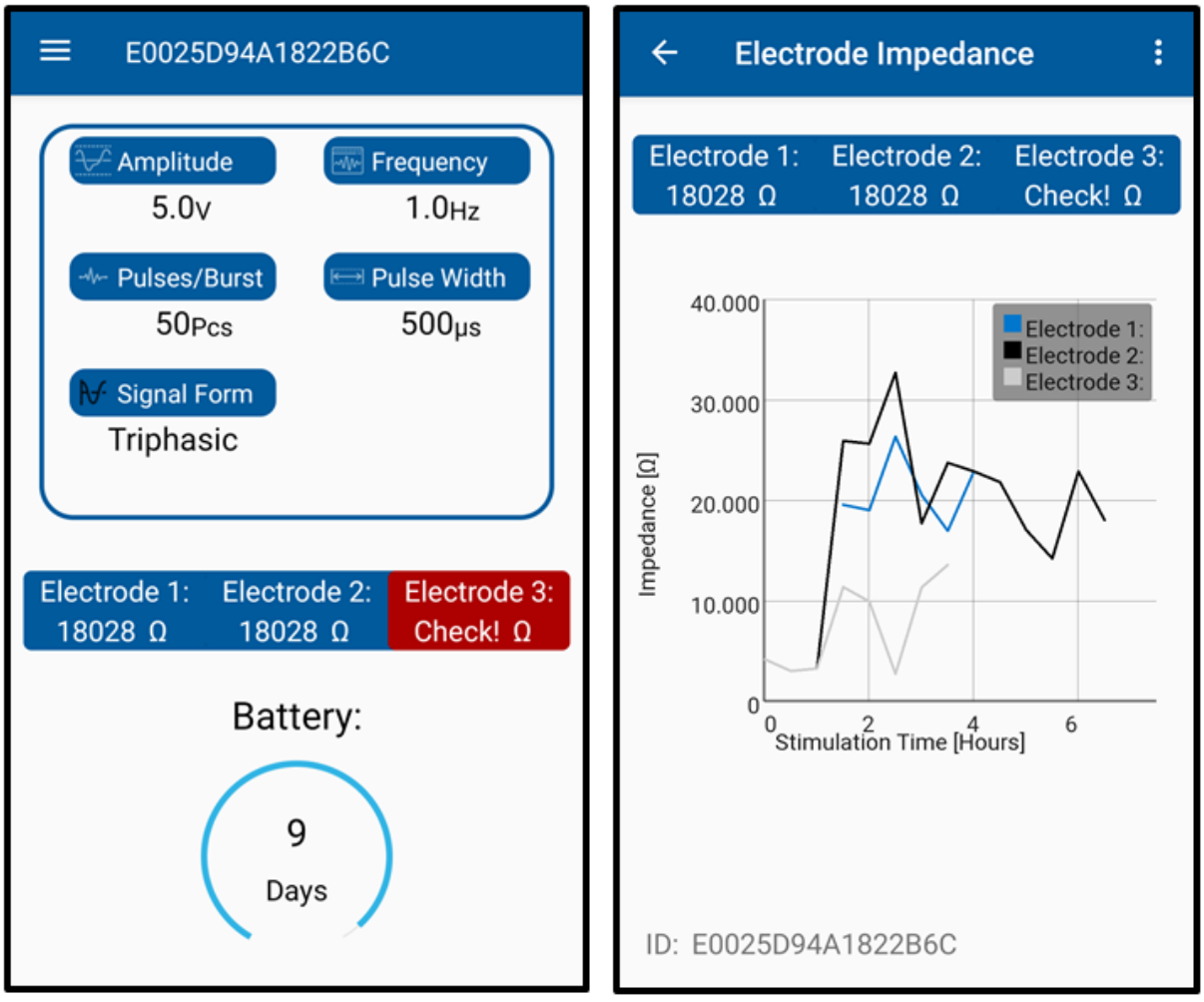

Figure 43: Screenshots of the application version 1.0. The left screen shows the main activity in case of a disconnected electrode resistor bridge and the right one showes the impedance graph activity while at the start electrode 1 was disconnected and in the end electrode 3.

In case of a detected disconnection, the application was programmed to show the disconnected electrode in a red colour to concentre the attention to this issue. In the impedance graph view, the disconnected points were simply removed. This behaviour can be witnessed in Figure 43 where on the left side the application shows an error with electrode 3 and on the right side can be seen that electrode 1 was disconnected for the first 1.5 hours of stimulation and electrode 3 is disconnected for approximate 3 hours.

4.6 Amplitude History

To get an overview of the therapy, an activity was created to see the history of the set amplitude during the treatment.

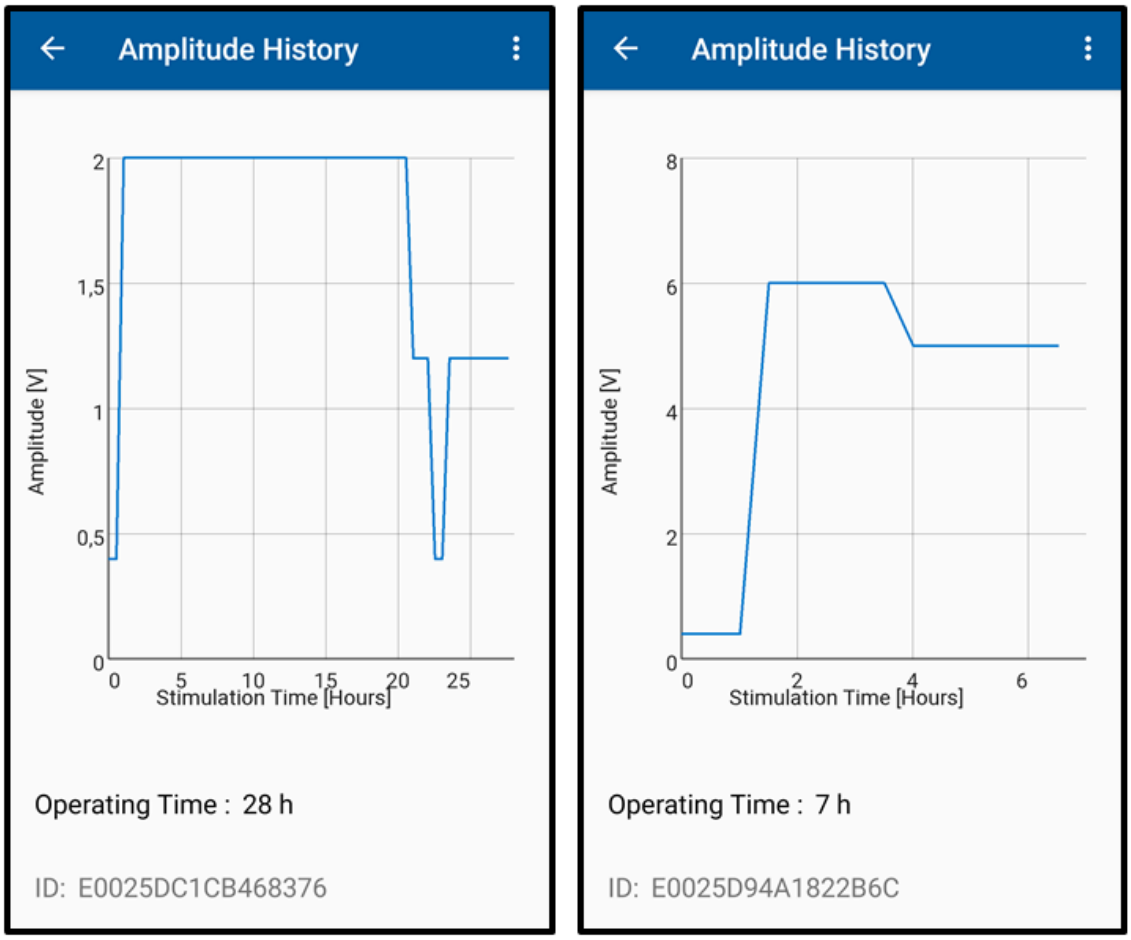

Figure 44: Screenshots of the application version 1.0 amplitude graph activity. The left screen shows all the amplitude progression connected to Figure 42 and the right one the amplitude progression connected to Figure 43.

This activity can be seen in Figure 44, as it shows on the left side the progression of the impedance test shown in Figure 42 and on the right side the amplitude history of the test shown in Figure 43. The drop from 1.2 V to 0.4 V and back was because the power supply of the stimulator was switched off and on, to see if it had any impact on the impedance values. Once the stimulator turned back on, the preset stimulation variables are loaded.

5 Discussion

The aim of this thesis was to develop an application, as a step stone to assist every stakeholder of the VNS treatment with the given stimulator. Until now, the stimulator could only be adjusted through a magnetic pen and only the amplitude could be changed. With a high availability of smartphones, which can provide quite an amount of calculation power, the possibility to develop an application which shows a high variety of functionality was given and performed.

Due to the fact, that a high percentage of all smartphones runs the Android OS and the lower prices of mobile phones with Android OS, this OS has been chosen for the first application development. To reach an even higher percentage of smartphone users, Phonegap, a cross platform framework was considered as the base for the given application. This approach was abandoned, for critical security issues, since this framework is still relatively new and data security is a high priority topic in medicine.

The main screen of the application was designed to give the user a quick overview about the stimulation parameters. For further insights into the details, separate activities were created to give the user more information. This updated the handling of the stimulator from a one way into a two way communication, since the stimulator is now able to give detailed feedback that can be interpreted by the smartphone. For improved usability the application was equipped, not only with visual, but also audial response in form of a tone. This is important for the patients, since for establishing the communication, they are moving the phone out of their line of sight. Many thoughts were put into the easiest handling possible for the application.

There are still some limiting factors that need to be considered. The current NFC connection with the onboard chip, which is located on the bottom of the board, shows a weak signal and tends to lead to communication errors. For this point it is highly suggested to mount a NFC antenna which is attached to the top of the stimulator

cover, to ensure the shortest way between the antenna and the VCD, to eliminate artefacts in the communication.

Also the communication with NFC-V is rather slow. Reading out only the current stimulation parameters is no deal, while reading out the whole tag at the end of a treatment can take up to 30 seconds, which is inconvenient. The supplier of the current chip implemented faster methods, which could cut this time down by half to 15 seconds, however also a change to a different NFC type, like NFC-A, can be considered to increase the transmission speed. Due to the slow communication, simplifications were used inside the app and the activities were divided into long read and short read/write activity based on the information they needed.

Currently the lifecycle ends with the depletion of the battery. If the battery of the device becomes rechargeable, the amount of write cycles will become the new life expectation time. With the current need for always up-to-date data, those write cycles are reached in around 3 recharges. Here maybe a short delay in overwriting will prolong the life cycle of one stimulator.

The current NFC connection and data management system was developed by only one person, but if new applications want to utilize the access, over the same interface, it could come to errors. This could be avoided by issuing a NFC interface description / datasheet. This description should contain the areas and the type of data including the encoding (32/16/8 bit) for the whole NFC EEPROM.

The data shown in the amplitude history does not include the real amplitude values during resting time. This was a solution that was implemented for the moment due to limited data transfer space. This could be resolved in future developments to show the real amplitude value rather than the set value.

The whole application was designed modular, this was done to support future changes in the direction of accessibility linked to a user. For example the NFC write

activity could be locked for all users without the necessary permission. Also some panels could be removed for different users to prevent overstrain of the user.

The battery calculation was designed to give a very rough estimate for the battery time based on the short connection, which only provides the current stimulation parameter and the latest electrical currents. This can lead to wrong results if the stimulation amplitude is changed by a high value during the stimulation. This deviation can be resolved by a long read, which also takes all the previous values into account. If the data rate could be increased for the NFC connection the estimate containing errors could be dismissed by always perform a long read. Also if the battery gets replaced by a rechargeable, the whole calculation could be cut out by just measuring the remaining battery voltage.

For the impedance calculation it had to be considered that during measurement of the positive voltage peak, a negative current would be measured by the stimulator, due to its circuit composition. Therefore particular attention has to be paid at the conversion of the ADC points to real electrical current values. Furthermore the measurement of the electrical current is afflicted by a tolerance of 40% (+/-20%) per channel. This renders the result highly vague, since the 40% tolerance gets added up in the calculation for each output channel, leaving the impedance values with a tolerance of +/- 60%. This also affects the battery calculation since it is also based on the electrical currents. Even so, if the measured values show a uniform distribution and enough values are recorded the mean of all the values should be equal to the applied impedance. Never the less, the measurement routine needs to be improved to reach better accuracy.

This Android application shows potential, to support the treatment with the given stimulator and gives physicians more possibility to adjust the therapy, to personalize it to the needs of each patient. However, even if apple only holds an estimate market share of 18% of the smartphone market, an iOS application should be considered to be modeled after the Android app.

6 References

[1] S. Betlejewski, “Social diseases, civilization diseases or lifestyle diseases,” *Wiad. Lek.*, vol. 60, no. 9–10, pp. 489–92, 2007.

[2] R. Lozano *et al.*, “Global and regional mortality from 235 causes of death for 20 age groups in 1990 and 2010: a systematic analysis for the Global Burden of Disease Study 2010,” *Lancet*, vol. 380, no. 9859, pp. 2095–2128, Dec. 2012.

[3] Institute for Health Metrics and Evaluation, “Life Expectancy & Probability of Death | Institute for Health Metrics and Evaluation,” 2013. [Online]. Available: http://vizhub.healthdata.org/le/. [Accessed: 16-Aug-2019].

[4] F. G. R. Fowkes *et al.*, “Comparison of global estimates of prevalence and risk factors for peripheral artery disease in 2000 and 2010: a systematic review and analysis,” *Lancet*, vol. 382, no. 9901, pp. 1329–1340, Oct. 2013.

[5] K. K. Nielsen, M. de Courten, and A. Kapur, “The urgent need for universally applicable simple screening procedures and diagnostic criteria for gestational diabetes mellitus lessons from projects funded by the World Diabetes Foundation.,” *Glob. Health Action*, vol. 5, 2012.

[6] E. Kaniusas *et al.*, “Current Directions in the Auricular Vagus Nerve Stimulation I - a physiological perspective,” *Front. Neurosci.*, vol. 13, no. August, p. 854, 2019.

[7] B. Bonaz, C. Picq, V. Sinniger, J. F. Mayol, and D. Clarençon, “Vagus nerve stimulation: From epilepsy to the cholinergic anti-inflammatory pathway,” *Neurogastroenterol. Motil.*, vol. 25, no. 3, pp. 208–221, 2013.

[8] B. M. C. Silva, J. J. P. C. Rodrigues, I. de la Torre Díez, M. López-Coronado, and K. Saleem, “Mobile-health: A review of current state in 2015,” *J. Biomed. Inform.*, vol. 56, pp. 265–272, 2015.

[9] D. Sethia, D. Gupta, T. Mittal, U. Arora, and H. Saran, “NFC based secure

mobile healthcare system," IEEE, Jan. 2014.

[10] H. K. Walker, *Cranial Nerves IX and X: The Glossopharyngeal and Vagus Nerves*. Butterworths, 1990.

[11] A. Faller, G. Schünke, and M. Schünke, *Der Körper des Menschen : Einführung in Bau und Funktion*. Thieme, 2012.

[12] A. I. Vinik, T. Erbas, and C. M. Casellini, "Diabetic cardiac autonomic neuropathy, inflammation and cardiovascular disease.," *J. Diabetes Investig.*, vol. 4, no. 1, pp. 4–18, Jan. 2013.

[13] S. Silbernagl and A. Despopoulos, *Color Atlas of Physiology*, 6th editio. Stuttgart-New York: Thieme, 2009.

[14] C. R. Noback, N. L. Strominger, R. J. Demarest, and D. A. Ruggiero, *The Human Nervous System*, 6th editio. Humana Press Inc., 205AD.

[15] L. K. McCorry, "Physiology of the autonomic nervous system.," *Am. J. Pharm. Educ.*, vol. 71, no. 4, p. 78, Aug. 2007.

[16] M. Sato and H. Koyano, "Autoradiographic study on the distribution of vagal afferent nerve fibers in the gastroduodenal wall of the rabbit," *Brain Res.*, vol. 400, no. 1, pp. 101–109, 1987.

[17] H. R. Berthoud and W. L. Neuhuber, "Functional and chemical anatomy of the afferent vagal system," *Auton. Neurosci. Basic Clin.*, vol. 85, no. 1–3, pp. 1–17, 2000.

[18] R. Câmara and C. J. Griessenauer, "Anatomy of the Vagus Nerve," *Nerves Nerve Inj.*, pp. 385–397, Jan. 2015.

[19] S. Eljamel, *Problem based neurosurgery*. World Scientific, 2011.

[20] R. L. Johnson and C. G. Wilson, "A review of vagus nerve stimulation as a therapeutic intervention," *J. Inflamm. Res.*, vol. 11, pp. 203–213, 2018.

[21] R. La Marca, M. Nedeljkovic, L. Yuan, A. Maercker, and U. Ehlert, "Effects of

auricular electrical stimulation on vagal activity in healthy men: evidence from a three-armed randomized trial," *Clin. Sci.*, vol. 118, no. 8, pp. 537–546, Apr. 2010.

[22] S. Kampusch, E. Kaniusas, and J. C. Szeles, "New approaches in multi-punctual percutaneous stimulation of the auricular vagus nerve," in *2013 6th International IEEE/EMBS Conference on Neural Engineering (NER)*, 2013, pp. 263–266.

[23] C. B. Nemeroff *et al.*, "VNS Therapy in Treatment-Resistant Depression: Clinical Evidence and Putative Neurobiological Mechanisms," *Neuropsychopharmacology*, vol. 31, no. 7, pp. 1345–1355, Jul. 2006.

[24] M. Eisenstein, "Neurodevice startups target peripheral nervous system," *Nat. Biotechnol.*, vol. 31, no. 10, pp. 865–866, Oct. 2013.

[25] E. Kaniusas *et al.*, "Current Directions in the Auricular Vagus Nerve Stimulation II – An Engineering Perspective," *Front. Neurosci.*, vol. 13, no. July, pp. 1–16, 2019.

[26] N. M. van Hemel and E. E. van der Wall, "8 October 1958, D Day for the implantable pacemaker.," *Neth. Heart J.*, vol. 16, no. Suppl 1, pp. S3-4, Oct. 2008.

[27] J. Iovine, *PIC Microcontroller projekt book*. McGraw-Hill, 2000.

[28] C.-K. Liang *et al.*, "A microcontroller-based implantable neuromuscular stimulation system with wireless power and data transmission for animal experiments," *J. Chinese Inst. Eng.*, vol. 26, no. 4, pp. 493–501, 2003.

[29] D. Mazzei, G. Montelisciani, G. Baldi, and G. Fantoni, "Changing the programming paradigm for the embedded in the IoT domain," in *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)*, 2015, pp. 239–244.

[30] S. Heath, *Embedded Systems Design*, 2nd editio. Oxford: Newnes, 2003.

[31] C.-N. Chien, H.-W. Hsu, J.-K. Jang, C.-L. Rau, and F.-S. Jaw, "Microcontroller-

based wireless recorder for biomedical signals," in *2005 IEEE Engineering in Medicine and Biology 27th Annual Conference*, 2005, pp. 5179–5181.

[32] S. Kampusch, E. Kaniusas, and J. C. Széles, "Modulation of Muscle Tone and Sympathovagal Balance in Cervical Dystonia Using Percutaneous Stimulation of the Auricular Vagus Nerve," *Artif. Organs*, vol. 39, no. 10, pp. E202–E212, Oct. 2015.

[33] S. Kampusch, E. Kaniusas, F. Thürk, D. Felten, I. Hofmann, and J. C. Széles, "Device development guided by user satisfaction survey on auricular vagus nerve stimulation," *Curr. Dir. Biomed. Eng.*, vol. 2, no. 1, pp. 593–597, 2016.

[34] R. S. Weinstein *et al.*, "Telemedicine, Telehealth, and Mobile Health Applications That Work: Opportunities and Barriers," *Am. J. Med.*, vol. 127, no. 3, pp. 183–187, Mar. 2014.

[35] S. R. Steinhubl, E. D. Muse, and E. J. Topol, "Can Mobile Health Technologies Transform Health Care?," *JAMA*, vol. 310, no. 22, p. 2395, Dec. 2013.

[36] A. Lahtela, M. Hassinen, and V. Jylha, "RFID and NFC in healthcare: Safety of hospitals medication care," in *2008 Second International Conference on Pervasive Computing Technologies for Healthcare*, 2008, pp. 241–244.

[37] O. + Kerem, M. N. Aydin, V. Coskun, and B. Ozdenizci, "Exploring Underlying Values of NFC Applications."

[38] "Licenses | Android Open Source Project." [Online]. Available: https://source.android.com/setup/start/licenses. [Accessed: 17-Aug-2019].

[39] V. Coskun, O. Kerem, and B. Ozdenizci, "NFC Application Development for Android," p. 316, 2013.

[40] A. Holst, "Mobile OS market share 2018," 2019. [Online]. Available: https://www.statista.com/statistics/266136/global-market-share-held-by-smartphone-operating-systems/. [Accessed: 16-Aug-2019].

[41] J. Eason, "Android Developers Blog: An update on Eclipse Android Developer

Tools.” [Online]. Available: https://android-developers.googleblog.com/2015/06/an-update-on-eclipse-android-developer.html. [Accessed: 17-Aug-2019].

[42] “Sourcetree | Free Git GUI for Mac and Windows.” [Online]. Available: https://www.sourcetreeapp.com/. [Accessed: 25-Aug-2019].

[43] A. Scopatz and K. D. Huff, *Effective Computation in Physics: Field Guide to Research with Python - Anthony Scopatz, Kathryn D. Huff - Google Books*, First. O’Reilly Media, Inc., 2015.

[44] R. Barnett, L. O’Cull, and S. Cox, *Embedded C Programming and the Microchip PIC - Richard H. Barnett, Larry O’Cull, Sarah Alison Cox - Google Books*. Delmar Learning Inc., 2004.

[45] I. O. for Standardization, “ISO 15963,” 2009.

[46] STMicroelectronics, “M24LR16E-R Dynamic NFC / RFID tag IC with 16-Kbit EEPROM ,” no. July, pp. 1–144, 2017.

7 List of Figures

8 List of Tables

9 List of Abbreviations

ADC/DAC	analog digital converter / digital analog converter
CPU	central processing unit
CRC	cyclic redundancy check
ENS	enteric nervous system
EOF	end of frame
I/O	input/output
IC	integrated circuit
IDE	integrated development environment
NFC	near field communication
PNS	parasympathetic nervous system
RAM	random access memory
RF	radio frequency
RFID	radio frequency identification
ROM	read only memory
SNS	sympathetic nervous system
SOF	start of frame
UI	user interface
UID	unique identifier
VCD	vicinity coupling device
VNS	vagus nerve stimulation
XML	extensible mark-up language

Eidesstattliche Erklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit gemäß dem Code of Conduct, insbesondere ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegeben Hilfsmittel, angefertigt wurde. Die aus anderen Quellen direkt oder indirekt übernommenen Daten und Konzepte sind unter Angabe der Quellen gekennzeichnet. Die Arbeit wurde bisher weder im In- noch im Ausland in gleicher oder in ähnlicher Form in anderen Prüfungsverfahren vorgelegt.

Wien am 16.09.2019

……..

Andreas Knor