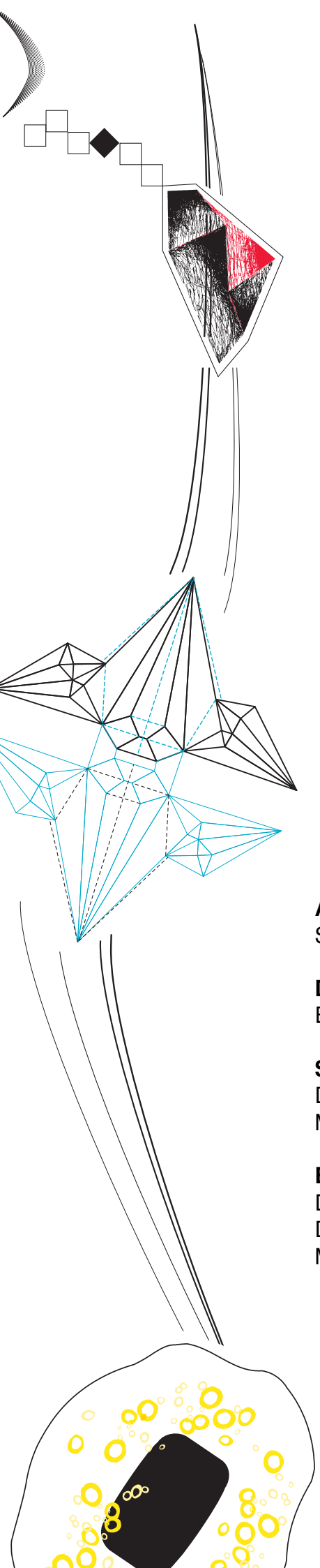


February 14, 2020 BACHELOR ASSIGNMENT

MULTICHANNEL ELECTROCUTANEOUS STIMULUS APPLICATION FOR NOCICEPTIVE SYSTEM CHARACTERIZATION

Faculty of Electrical Engineering, Mathematics and Computer Science
Biomedical Signals and Systems

Author
Silvi Bundo

Department
Electrical Engineering

Supervisors
Dr. ir. Jan Buitenweg
MSc. Niels Jansen

Exam committee
Dr. ir. Jan Buitenweg
Dr. ir. Cora Salm
MSc. Niels Jansen

UNIVERSITY OF TWENTE.

Acknowledgements

I would like express my very profound gratitude to my supervisors, namely, Dr. ir. Jan Buitenweg, Dr. ir. Cora Salm and Niels Jansen Msc., for all the support offered during my journey of writing my bachelor thesis and for allowing me to undertake this research and discover the fascinating field of neuroscience. I am thankful for this great opportunity that made me push my boundaries and face my fears, as well as made me discover abilities of myself, whose potential I did not know. I am also very grateful to Boudewijn van den Berg, who provided technical expertise and advice that greatly assisted the study, your help and guidance were priceless.

Finally, I want to thank all my family and my friends for providing me with unfailing support and continuous encouragement throughout my study and through the process of writing this thesis.

Abstract

Early detection of chronic pain and therapeutic action would mean better treatment outcome and less clinical efforts per patient, but unfortunately appropriate diagnostic methods are lacking. A major limitation of the present nociceptive detection threshold (NDT) and evoked potentials (EP) method is the application of stimulus patterns through a single electrode device using a single channel constant current (the AmbuStim), which can only activate one target population of fibers at a single fixed spatial location.

Yet, since pathological nociceptive processing can involve altered spatial integration or altered multimodal integration, applying a series of electrocutaneous stimuli to more than one location on the skin, or using more than one type of electrode for targeting different fiber populations (e.g. tactile and nociceptive) on the same location, may provide better insight for nociceptive system characterization and may allow early interventions.

In order to permit such experiments, this study presents the full approach of extending the current experiment setup, using a single-channel stimulator (the AmbuStim), with the possibility of multichannel stimulation by implementing the OctoStim, an 8-channel constant current stimulator, wirelessly operated from LabVIEW, which previously used an outdated version of the StimCom command language.

The OctoStim stimulator's embedded software was modified to make it compatible with the present version of the StimCom command language, and likewise the present LabVIEW toolbox was adopted to enable the OctoStim stimulator to be used in the present experiment setup for calibration and human subject experiments. In order to make the necessary modifications, first the electronics of the AmbuStim were examined, and then carefully compared to the electronics of the OctoStim stimulator. After matching the two and detecting the differences at the end-stages, then the new version of the StimCom command language used in the AmbuStim was compared to the old version of StimCom used in the OctoStim's embedded software in order to make the necessary modifications. Finally, the generation of calibrated stimuli is demonstrated for the extended setup via a standard calibration program written in MATLAB.

As a result, the OctoStim stimulator's embedded software was successfully updated, since it shows to properly communicate with LabVIEW via the new adjusted StimCom command language, it generates calibrated stimuli in all 8 channels and proves to be feasible for performing multichannel NDT tracking experiments in human subjects.

List of Acronyms

BSS	Biomedical Signals and Systems
BT	Bluetooth
CP	Chronic Pain
EP	Evoked Potentials
IPI	Inter Pulse Interval
IES	Intra-epidermal Electrocutaneous Stimulation
NDT	Nociceptive Detection Threshold
NOCICEPT	Neurophysiological Observation of Chronification: System Identification Concepts for Effective Pain Treatment
NoP	Number of Pulses
PAINSIGHT	PArameter Identification of the Nociceptive System for Improved monitorinG of cHronic pain developmenT
PW	Pulse Width
QST	Quantitative Sensory Testing
RS	Recommend Standard

Table of Contents

1 Introduction

When pain outlasts the healing process and no longer serves as a protecting mechanism, is then considered to be persistent pain, which can have a negative impact on the quality of life [1]. If this persistent pain is present for more than 3 to 6 months it is then labeled as chronic [2]. In other instances, pain can be considered chronic, when it occurs for no apparent reason and seems to serve no useful purpose [3]. Thus, chronic pain is often thought to be the result of disturbed nociceptive processing in the central nervous system.

According to the studies done by Christa Harstall and Maria Ospina (2003) [4], the prevalence rate of chronic pain (CP) worldwide is estimated to range between 10% and 55% with a higher prevalence rate among females, and according to Bekkering at al. (2011) [5] in the Netherlands alone, it is approximated that 1 out of 5 (one out of five) adults suffer from chronic pain. Also, once chronic pain is established, treatment is relatively ineffective, with – at best – one patient in three or four achieving 50% pain intensity reduction [6]. Besides consequences for individual patients, chronic pain poses a significant burden on health care systems and society in general by taking away from the labour force, for instance, and/or due to increased consumption of medical resources [7]. Even though clinical observation of nociceptive pathways is limited at this point, it could certainly give a better insight into the development of chronic pain, provide better understanding and permit early interventions of this condition.

1.1 Current Issues of Chronic Pain Diagnostic Methods

Appropriate diagnostic methods for detection and better treatment of chronic pain are currently lacking. Several diagnostic methods exist, but they all present issues regarding the optimal approach. Commonly used methods for measuring pain sensitivity include psychophysical Quantitative Sensory Testing (QST) methods, used for observation of stimulus processing and sensory function [8]. Hereby, pain stimuli, either thermal, mechanical or electrical, is exposed to the body to inspect the severity of nerve damage. According to Wilder-Smith et al. (2010) [9] and Backonja et al. (2013) [10] QST can serve as a valuable clinical tool for assessing risk for the development of chronic pain. It remains, however, difficult to distinguish the individual contributions of nociceptive mechanisms, obstructing the diagnosis of chronic pain disorders [8]. Furthermore, QST is not applied in daily clinical practise, because the equipment and the usability of the method do not suit the point-of-care and currently it has to be conducted in a lab [9].

In 2007, alongside other partners, the research department of Biomedical Signals and Systems (BSS), at the University of Twente, as part of the PAINSIGHT project (PArameter Iden-

tification of the Nociceptive System for Improved monitorinG of cHronic pain developmenT) developed the AmbuStim (single channel constant current stimulator). While in the NOCICEPT project, they utilized phasic electrocutaneous stimulation of nociceptive afferents in the skin for estimation of the psychophysical detection threshold using button-press/release responses (yes/no) of the human subjects.

AmbuStim is a portable device, which makes the process of measuring the pain threshold through electrocutaneous stimulus application on the skin much easier and faster, as well as increases the availability of the procedure compared to previously mentioned methods. However, even though it permits well timed activation of nociceptive fibers, using for instance an intra-epidermal needle electrode for nociceptive stimulation (IES-5 electrode) with the ability to specifically stimulate nociceptive $A\delta$ afferents as long as the stimulation current is below twice the detection threshold, in order to minimize the co-activation of tactile $A\beta$ fibers [8] , a major limitation of the AmbuStim is that it only activates one target population of fibers at a single fixed spatial location.

Since we know from literature that pathological nociceptive processing can involve altered spatial integration [11], [12] (e.g. enlarged receptive fields of spinal neurons) or altered multimodal integration (c.f. gate-control theory, allodynia) [6], [13], it would be of interest to be able to apply series of electrocutaneous stimuli to more than one location on the skin, or using more than one type of electrode for targeting different fiber populations (e.g. tactile and nociceptive) on the same location. Though, to permit such experiments, the present experiment setup should be extended with the possibility of multichannel stimulation, i.e. by implementing the OctoStim, an 8-channel constant current stimulator wirelessly operated from LabVIEW.

Figure 1.1 on the right shows the AmbuStim device (single channel stimulator), whereas on the left the OctoStim device is shown, which is the extended setup consisting of 8 channels.

1.2 Research Goal and Research Questions

The goal of this bachelor thesis is to modify the OctoStim stimulator's embedded software to make it compatible with the present version of the StimCom command language and modify the present LabVIEW toolbox to enable the OctoStim stimulator to be used in the present experiment setup for calibration and human subject experiments. The study will focus on comparative analysis between the fully-working embedded code of the AmbuStim and the outdated embedded code of the OctoStim. To this extent, more specific questions must be answered thoroughly:

- How does the embedded code of the AmbuStim work and how is it translated into the functions of its electronics?
- How does the electronics schematics of the OctoStim differ from that of the AmbuStim, especially at the output stage?
- What kind of translations are necessary to perform in the same way in the OctoStim's embedded code when compared to that of the AmbuStim?
- What kind of technical tests are needed to prove the functionality of the OctoStim stimulator after the modifications made in its embedded code?

(a) (b)

Figure 1.1: (a) Multichannel stimulator - the OctoStim (b) Single channel constant current stimulator - AmbuStim

1.3 Thesis Outline

First chapter provides an introduction to the topic, gives the current issues of chronic pain diagnostic methods, presents the research goals and questions and concludes with an outline of the thesis. Second chapter gives an overview of the theoretical background needed to understand the working principle of the electronics behind the AmbuStim and the OctoStim, as well as the functionality of the StimCom command language. Third chapter is a roadmap leading to the core of the research. It describes the procedure that is followed to perform the necessary changes in the OctoStim's embedded code, describes the equipment and software used for doing so and also contains the design validation. The fourth chapter presents the results from the technical tests performed for validating the OctoStim, the generation of stimuli, together with all the discussion necessary to draw a final conclusion. Lastly, the thesis is concluded by presenting answers to the research questions, an overall conclusion of the results and recommendations for future work.

2 Background

2.1 StimCom language

2.1.1 System Overview

StimCom is a command language used in the NociTRACK systems, which plays a crucial role in the communication and controlling of the NociTRACK stimulator device via the Bluetooth controller. This command language is a clear-cut way to implement a set of low-level instructions, that in this case, are used for direct access to stimulus parameters and stimulus generation. StimCom is a connection oriented protocol, meaning each command sent is verified by an acknowledgement which is usually an echo of the command. For communicating, it uses a Bluetooth unit with a serial protocol communication (RS-232), which means the data is transmitted bit by bit. The RS-232 protocol communication is widely used for connections between data acquisition devices and computer systems [14]. The latest available version of the StimCom command language is the 0.3 version. According to this version, the necessary changes in the OctoStim's embedded code were made.

First, a simple StimCom setup is shown in Figure 2.1 consisting of five main blocks. Each of those is briefly explained below.

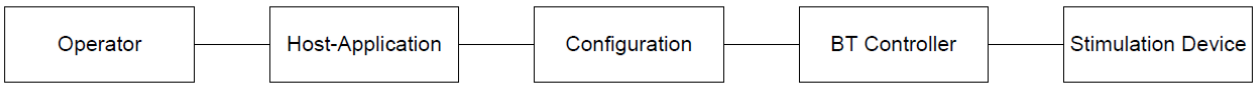


Figure 2.1: System overview of the StimCom setup [15]

- The Stimulation Device is a battery operated stimulator which connects over Bluetooth to the LabVIEW interface (AmbuStim, OctoStim, etc.).
- The Bluetooth Controller (BT Controller) handles all communication aspects of the StimCom. It controls the communication between Host and client and handles the synchronization. It's major tasks can be summarized as: 1) handling the communication between stimulation device and host 2) having access to the StimCom Configuration field 3) listening to the connection between host and stimulation device. The Bluetooth Controller continually checks if there are any un-synced items in the StimCom configuration field and listens to the connection if the link between host and stimulation device is still established. It also searches for stimulation devices and adds their unique Bluetooth addresses to the Bluetooth device list.

- The Configuration building block communicates with the BT Controller. This building block consist of all parameters needed to parse to the controller.
- The Host Application is controlled by an operator/is seen as a representative of the operator. Therefore, it also implements the interface to the operator. It consist of a restricted area which includes the StimCom Configuration field and the Bluetooth Controller. This area is only accessible through 3 functions namely : StimCom Release, StimCom Connect and StimCom Synchronization.
- The Operator is the user which operates the system.

2.1.2 Pulse creating

A pulse is characterized by 3 main parameters: Amplitude (A), Pulse Width (PW) and Inter-Pulse interval (IPI). For the sake of differentiating positive and negative pulses, it is denoted that upper-case “A” is the amplitude of the positive pulse and the lower-case “a” corresponds to the negative amplitude of the pulse. The same goes for the parameter PW where “W” (upper-case) is the pulse width of the positive phase and “w” (lower- case) is the pulse width of the negative phase. While in parameter IPI the "I" variable at the end, stands for interval and shows the elapsed time between successive pulses.

An example of a stimulation pattern is given in Figure 2.2

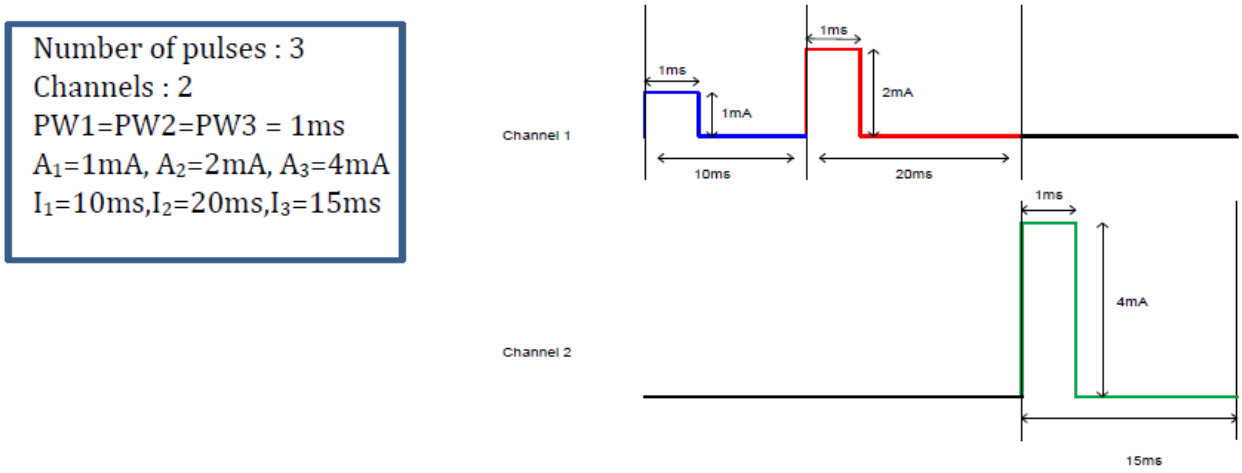


Figure 2.2: Stimulation pattern example [15]

2.1.3 Software Control

Both AmbuStim and OctoStim are microprocessor driven and the embedded software includes a number of simple commands, consisting of strings using a general format:

<ID_char>, <par_1>, <par_2>, . . . , <par_N> $\backslash 0$

With ID_char indicating a case-sensitive command identifier character and <par_int> indicating a scalar integer parameter. The scalar integer parameter after the command type,

can stand for different parameters depending on the command, this will be elaborated a bit further when the command types are explained. The commands are separated from each other using an delimiter "," and the string is terminated by a NULL-character "$\backslash 0$".

Command List and Communication Examples

Below, all the command types are introduced and then a communication example is given to demonstrate the communication between the Bluetooth Controller and the Stimulator Device when a stimulation command "S" is sent.

Command	Description
I,x_1,x_2,x_3,...,x_N	Set interpulse intervals (IPI) for stimulus pattern with x_1 to x_N the number of timer units (35 us). **Note:** (1) a stimulus can be built up from multiple pulses (See S,....).(2) the interpulse interval should always exceed the total pulse duration.
P, x_1,x_2,x_3,...,x_N	Set channel order for stimulus pattern (first set interpulse intervals using I command). With x the channel number.
A,x_1,x_2,x_3,...,x_N	Amplitude of positive phase of the pulse, with x the number of AD units (max 4096 AD-units, 8 uA/AD-unit). **Note:** actual current amplitude depends also on output load.
a,x_1,x_2,x_3,...,x_N	Amplitude of the negative phase of the pulse. See A,x_1,x_2,x_3,...,x_N for details. Note: the command identifier is case-sensitive!
W,x_1,x_2,x_3,...,x_N	Pulse width of positive phase of a pulse, with x the number of timer units (max 4000 units of 35 us each).Note: (1) between positive and negative phase a dead-time of 70 us is maintained for switching and settling of the output stage.(2) the total duration of the pulse (PW_{pos}+dead-time+PW_{neg}) should never exceed the interpulse interval.
w,x_1,x_2,x_3,...,x_N	Pulse width of negative phase of the pulse. See W,x_1,x_2,x_3,...,x_N for more details. **Note:** the command identifier is case-sensitive.
C,x_1,x_2,x_3,...,x_N	Enable or disable channels. Positive and negative pulses can be enabled independently.
M,x,y	Set power On(*x=1*) or Off (*x=0*) (Hold only no control over PD), y = power level.
S,*x*,*y*,*z*	Stimulation command, with x the number of external triggers that is allowed to generate a stimulus (with *x = 0)* an immediate single stimulus is generated), *y* the number of pulses within each stimulus and *z* a delay in time-units. With *x>0*, an echo 'S,x,*y*,*z*' is sent to the client after each trigger, with x the new number of triggers that is allowed.
Q,*start*,*step*,*stop*,*crit*,*index*	Perform Stimulus response series. With Qstart the start value of the pulse amplitude, Qstep the stepsize, Qcrit ramp generation when Qcrit = 1 (button is pressed). Qindex, selection of which pulse had to be ramped according to the settings.
R,x,y,z	Check response button. Send x=y=z=0 ,the stimulator will return its status in parameters: x = Resp signal, y = Trig signal, z = IOK signal.
V,x,y,z	Version Query where x = Version1, y = version2 and z = Serial Number
F,*nCh*,*mPattern*,*nCal*,*nTime*	Features query where *nCh* = The number of channels, *mPattern* = the max. pattern length, *nCal* the calibration factor AD-units to mA and *nTime* the calibration factor timer-units to ms.

Figure 2.3: Command list and their description [15]

As can be seen from Figure 2.3 the S-command, which stands for Stimulation command consists of three parameters:

- nT = number of Trigger inputs
- nP = number of Patterns (number of pulses within each stimulus)
- D = delay (in time units)

With $x = nT$, $y = nP$, $z = D$, it takes the form: S, nT, nP, D. A schematics of the actions happening after a stimulation command is sent to the stimulator by the Bluetooth controller is shown in Figure 2.4.

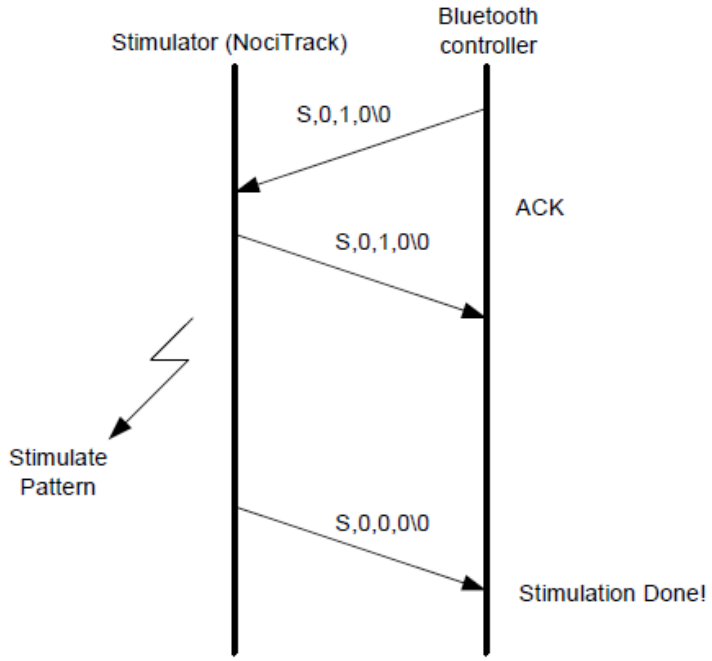


Figure 2.4: Communication between Bluetooth Controller and Stimulator Device when "S" command is sent, without external trigger [15]

In Figure 2.4 the string $"S, 0, 1, 0\backslash 0"$ is sent to the stimulator from the Bluetooth Controller, with no external trigger. Decoding this command would yield: Command ID: Stimulation, Trigger Input: 0, Number of Patterns: 1, Delay: 0. If this command is correctly received at the stimulator, the stimulator answers with an ACK, which is an echo of the command sent, hence in the figure above the exact same command $"S, 0, 1, 0\backslash 0"$ is seen to be sent back to the Bluetooth Controller. Since the delay is zero, the stimulator then immediately starts applying the stimulation pattern and when the stimulation is done it returns back the string $"S, 0, 0, 0\backslash 0"$, meaning "Stimulation Done", or all patterns stimulated. In case of $nP > 1$, it will repeat the "Stimulate Pattern" phase till all patterns are stimulated, then again return back the string $"S, 0, 0, 0\backslash 0"$ when it is done doing so.

In case of a stimulation command using an external trigger, the communication schematics would look as shown in Figure 2.5.

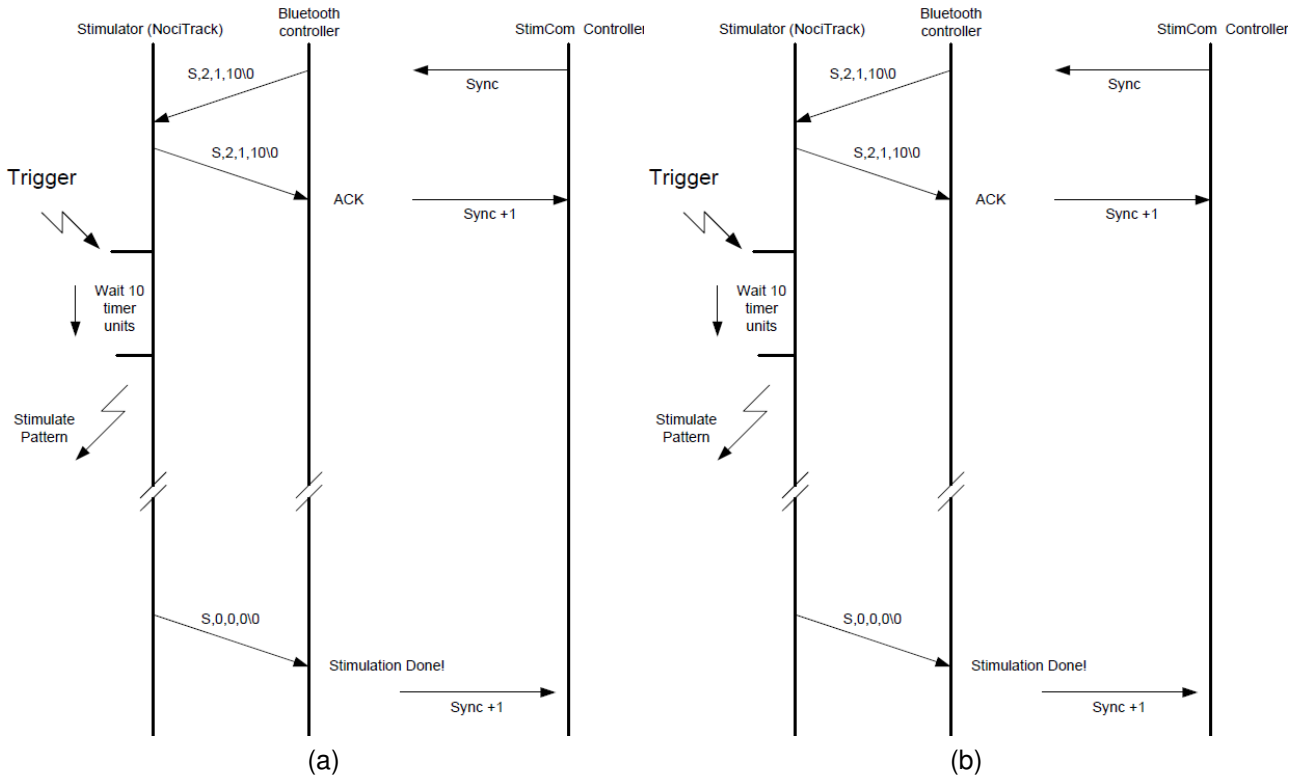


Figure 2.5: Communication between Bluetooth Controller and Stimulator Device when "S" command is sent, with external trigger [15]

In this case, the command sent from the Bluetooth Controller is $"S, 2, 1, 10\backslash 0"$, thus Trigger Input: 2, Number of Patterns: 1, Delay: 10 (timer units). After the stimulator answers with an ACK, it then waits 10 timer units before applying the stimulation pattern. Finally, it returns back the string $"S, 0, 0, 0\backslash 0"$ for implying that the stimulation is done. On the right hand side, "Sync" handles the synchronization state, where the system synchronizes with the stimulation device and waits for its response or timeout. It first checks if there are any other left un-synced flags and then parses commands according to the highest priority list or if all is synced then it goes to the next state "Update Config."

Similar to the case of the S-command given above, the function of the Q-command has also been further explained (graphically) and can be found in Appendix A.1. This command has 5 parameters, namely:

- $Qstart$ = the start value of the pulse amplitude in mA
- $Qstep$ = the stepsize in mA
- $Qstop$ = the end value of the pulse amplitude in mA
- $Qcrit$ = ramp generation when Qcrit = 1 (button is pressed)
- $Qindex$ = selection of which pulse had to be ramped according to the settings

However, there are a few other commands that when sent to the stimulator, instead of an echo they return back some other parameters. Those commands are: the F-command, the V-command and the M-command. For instance, when the string $"F, 0, 0, 0, 0"$ is sent to the

stimulator, it replies back with an F-command in the form "$F, nCh, mPattern, nCal, nTime$", where:

- nCh = number of Channels
- $mPattern$ = the maximum Pattern length
- $nCal$ = the Calibration factor AD-units to mA
- $nTime$ = the calibration factor timer-units to ms

For the OctoStim stimulator the correct F-command sent back from the stimulator would be: $F, 8, 20, 80, 35$.

Similarly, when "$V, 0, 0, 0$" is sent to the stimulator the V-command that is sent back should be of the form "$V, version1, version2, SerialNumber$". The embedded software version and serial number of stimulator device is defined on the source code of the stimulator.

The same applies for the M-command. When "$M, 0, 0$" is sent to the stimulator, it replies with a "M, x, y" form, for:

- x = set power (Set Power On (x=1), Set Power Off (x=0). HOLD only, no control over PD)
- y= power level (DAC has always power On)

Command Generation According to Priority List

Each time a value is un-synced the program looks which command it has to create according to the priority list as shown below. For instance, if the amplitude has changed and also the channel selection, it first looks at the list and sees that the variable channel has a higher priority than the amplitude. Priority list of commands (the lowest number indicates the highest priority) is shown in Table 2.1:

Table 2.1: Priority list of commands [15]

1	F	8	A
2	V	9	a
3	R	10	W
4	M	11	w
5	C	12	S
6	I	13	Q
7	P		

2.2 AmbuStim - Single Channel Stimulator

2.2.1 Description and Specifications of the AmbuStim

AmbuStim is a single-channel constant-current stimulator that delivers short duration current pulses of up to 20 mA with a maximum frequency of 100 Hz [16]. It is a handheld instrument for non-invasive transcutaneous electrical stimulation through skin electrodes. It is developed to be used in an outpatient environment for medical, scientific research and the potential use of future clinical practices in the area of pain diagnostics.

The AmbuStim can be relevant for patients who are at risk of developing or being treated for chronic pain, but not recommended for those with a pacemaker or an implanted electrical defibrillator. Clinical scientific studies have shown that generalized hyperalgesia, which causes an electrical stimulus to be experienced as annoying at lower currents due to pain hypersensitivity, plays a crucial role in the development and maintenance of chronic pain complaints. The AmbuStim makes possible to measure pain thresholds in competent patients who are able to operate the response button on the device according to the instruction given.

AmbuStim contains a current generator which generates current pulses with a pulse width (PW) of 210 microseconds and with a pulse frequency of 100 Hz during a threshold measurement [16]. The procedure of using the AmbuStim is as follows. During the measurement process, the patient is exposed to electrical pain stimuli through some electrode/s on the skin. As the patient is holding the button, a current will start to flow. The operator can ask the patient to release the button at three different sensations. The first is as soon as the patient starts to feel the incoming stimuli (detection threshold, 3 mA), to measure the sensation threshold. Second, to release once this feeling starts to become unpleasant (pain threshold, 8 mA), to measure the pain threshold. Third, to release when the pain becomes unbearable, to measure the pain tolerance. The amplitude of the pulses increase from zero by 0.3 mA/s. When the button is released or when the maximum set amplitude of 20 mA is reached, the power generator stops and the last given amplitude (the threshold value) is reported back via the wireless connection to the PC, tablet or smartphone of the user. The second value, the pain threshold, is relevant for diagnosing an increased sensitivity for pain.

Since the stimulator is battery powered and controlled via a wireless connection, there is no leakage to the environment (as in the case of a mains supply) and the stimulator is inherently safe in terms of electrical skin stimulation.

2.2.2 The Function of the Electronics

AmbuStim is a microprocessor-driven device and the control of functions and reading of settings is done wirelessly (via Bluetooth) from LabVIEW. It consists of a controllable current source, which is powered from an internal high-voltage source (max. 92 volts [16]). This current source is also controlled from the internal microprocessor. The microcontroller used in this NociTRACK system is a high-performance, low-power AVR® 8-bit Microcontroller, the ATmega162 [17]. By putting the ports of the ATmega162 in high or low signals, the control of the electronics is possible. A schematics of the internal circuitry of the AmbuStim is used for examining the electronics, reading the port numbers of the microcontroller and

their functions. The full schematics can be found in Appendix A.2.

Figure 2.7 shows a block diagram of the most important functional elements of the AmbuStim.

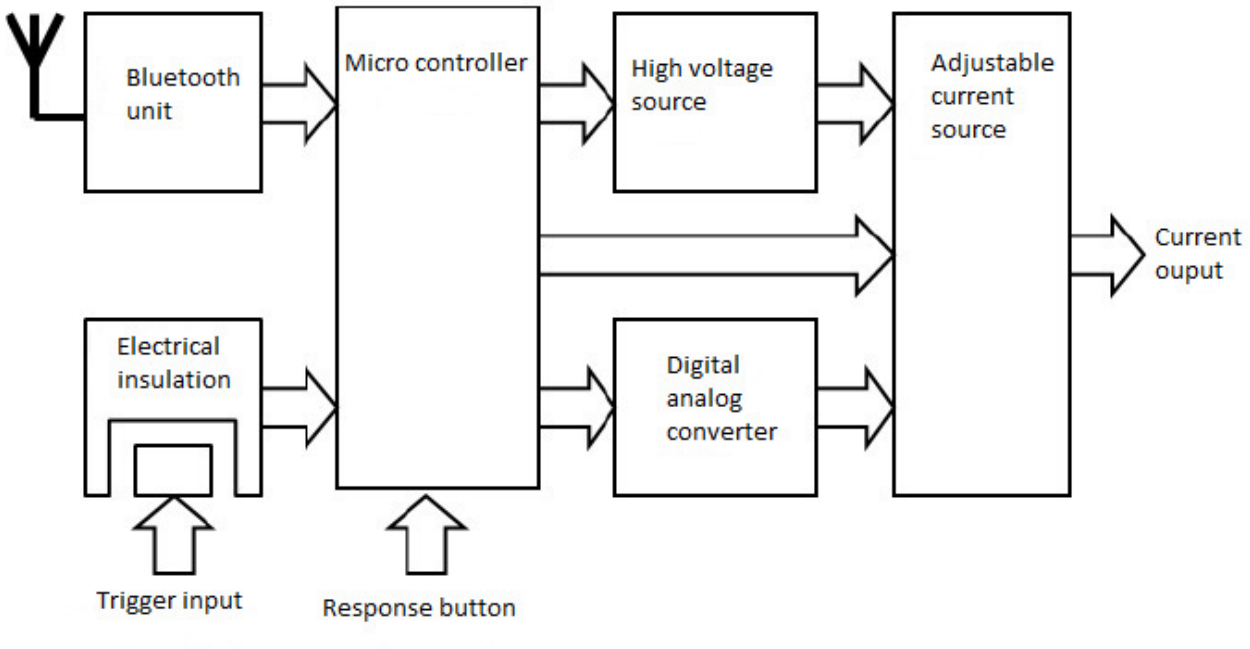


Figure 2.6: Block diagram of the main components of the AmbuStim circuitry[16]

One important electronics shown in the AmbuStim schematics is the Bluetooth unit. A consumer Bluetooth unit is used for the communication of the AmbuStim device with the Bluetooth Controller. The Bluetooth unit used in the AmbuStim device is a Parani-ESD 200, consisting of 8 pins. The Parani-ESD Class 2 Series enable RS-232-based serial devices to communicate wirelessly throughout the range of 30m - 300m [18]. The RS-232 itself is a serial protocol, which is widely used for connections between data acquisition devices and computer systems. It is an old communication protocol, that transmits the data one bit at a time over a single communication line/ channel. Thus the Parani-ESD Series (bluetooth units) are commonly used for RS-232 cable replacement. It is apparent that there is a data link between the microcontroller and the Bluetooth unit, since the microcontroller can send Bytes to the Bluetooth unit and the Bluetooth unit transmits those back.

As a matter of fact, AmbuStim uses a rechargeable lithium polymer (Li-Poly) battery pack, the Varta EasyPack, with a battery supply voltage of $3.7V$. However, for stepping up this voltage, a high-voltage DC-DC converter is used, the 5SMV100D Series Dual-Output [19]. This Dual-Output DC-DC converter steps up the battery voltage to $\pm$ 100 V. Therefore, it is apparent that it becomes a high voltage source. This DC-DC converter is widely used in devices which are primarily supplied with battery power and it is a good way to save space instead of using multiple batteries to accomplish the wanted output voltage, by storing the input energy temporarily using capacitors and then releasing that energy to the output at a different voltage. The AmbuStim is also equipped with an electrically isolated trigger input (however, not used in this study). It is electrically isolated since the diode-transistor optocouplers, use an insulating layer between a LED and an integrated photodetector to provide electrical insulation between input and output.

Another important electronics seen in the AmbuStim schematics is the Digital to Analog Converter. The DAC used in the AmbuStim is the AD5341, a single 12-bit DAC [20]. The GAIN pin of the DAC controls the output range from the DAC to be from 0 to 2*VREF, and since VREF=1/2VCC, it equates to Gain=Vcc, thus Vcc is connected to the GAIN pin of the DAC. An asynchronous CLR input is also provided, which resets the contents of the Input Register and the DAC Register to all zeros. The DAC has also 8 parallel data inputs D0-D7, with D7 being the MSB of those 8 bits and they can be controlled from Port C of the ATmega162 microcontroller. The DAC then outputs an analog pulse with a certain amplitude measured in volts. This Vout from the Digital Analog Converter has a positive value. However, since AmbuStim uses biphasic pulses, both positive and negative pulses are of needed, therefore another important component that comes to use is the logic controlled analogue switch, the CD4066BC. The output coming from the DAC (Vout) severs as input for this switch. In itself this component contains two switches, +S1 and -S1, which are also controlled (opened or closed) from the microcontroller Atmega162 by setting high or low the pins A0 or A1. By opening the switches S1 or -S1, the desired pulse width (positive or negative) is let in the logic controlled analogue switch. Then the output of the analogue switch gives two pulse amplitudes measured in volts, +A1, -A1 (positive or negative). The analogue switch used here, does the so-called "orchestration" of a pulse.

Furthermore, a voltage to current converter is needed, so that it can convert this voltage-amplitude pulse into a current-amplitude pulse. The voltage to current converter uses +A1 and -A1 as its input and by means of the Op-amps and two BAV103 diodes it outputs a negative current amplitude pulse.

Another switch at the output stage of the AmbuStim is used for controlling the +/-100V flowing in. By controlling again the switches +S1 and -S1 from the microcontroller either $\pm$ Out1 is let through the patient. This pulse has finally an amplitude measured in current.

Another components of the AmbuStim is an Inverting Oscillator amplifier with XTAL1 (X1) as input, which is also input to the internal clock operating circuit and XTAL2 (X2) as its output. This is used to provide a stable clock signal and is else known as the Crystal Oscillator. In Standby mode, the crystal Oscillator is running while the rest of the device is sleeping. In the schematics this oscillator is noted as X1X2. The load capacitors C1 and C2 should always be equal for a crystal oscillator to properly operate and their optimal value depends on the crystal in use, the amount of stray capacitance, and the electromagnetic noise of the environment [17].

Other components of the AmbuStim include: 1) the CMOS charge-pump voltage converter operating from a wide 1.5-V to 5.5-V supply voltage. It uses two low-cost capacitors to provide 100 mA of output current. 2) The fixed-voltage monolithic micropower voltage regulators, which are designed for a wide range of applications and are an excellent choice of use in battery-power application. The current increases on slightly at dropout, which prolongs battery life. They are also fully protected from damage due to fault conditions.

2.3 OctoStim - 8-channel Stimulator

2.3.1 The End-Stage

The majority of the OctoStim electronics have the same or similar functions to that of the AmbuStim. However, the end stage of the OctoStim differs a bit from that of the AmbuStim, since there are now 8 current outputs. A schematics of this is shown in the Figure below.

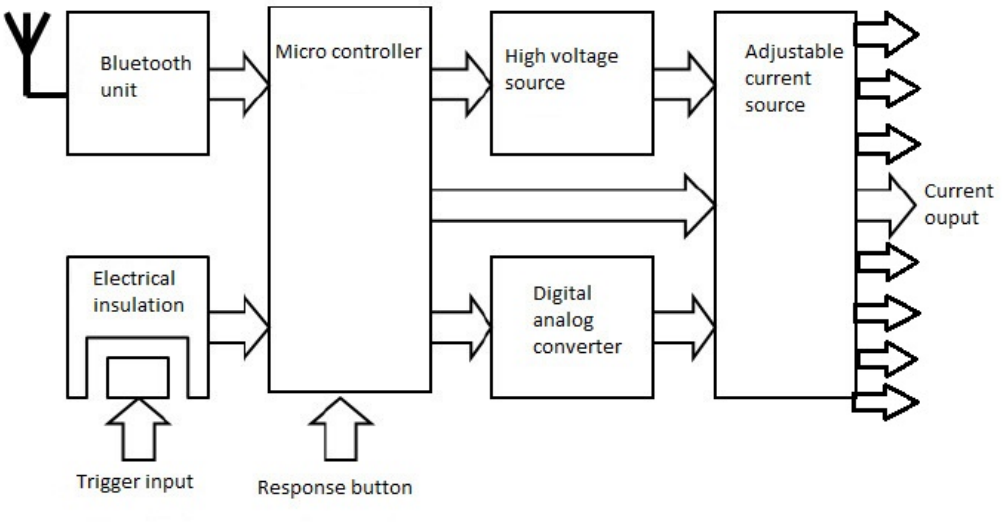


Figure 2.7: Block diagram of the main components of the OctoStim circuitry.

At the output stage there are 8 parallel DC-DC converters for stepping up the voltage. There are 8 switches S01-S08 for enabling the channels where current can go through. Those switches are controlled by means of a 8-Bit Addressable Latch. This device is capable of storing single line data in eight addressable latches, and also a 1-of-8 decoder and demultiplexer with active HIGH outputs. It also has an active LOW common Clear for resetting all latches, as well as, an active LOW Enable. The address inputs of this latch are connected to D0, D1, D2, which can be controlled from pins A0-A2 of Port A of the microcontroller. CS-SN is connected to ground of the latch and is controlled by Port C1 of the microcontroller. D3 is connected to the data input. By setting the D3 high or low (Port A3), the data coming from the combination of the address inputs is let through the latch. This controls also in which channel the current is let through, by activating the switches. When a switch is activated in this latch, the switch opens at the output stage to let current through. The electrodes are also connected at this stage, by means of which the patient feels the current which was let in. In the OctoStim only -100V is generated by the Single-Output DC-DC converter and the other output of it is not being used. This is because OctoStim is not designed to use biphasic pulses. The channels of the OctoStim are all connected to the same anode, hence the OctoStim has a common anode.

3 | Design

3.1 Equipment and Software Use

3.1.1 LabVIEW

For stimulating and wirelessly communicating with the OctoStim a computer with LabVIEW software 2013 SP or higher is needed. In this study, the NI LabVIEW 2019 SP1 (64 bits) version was used and a folder with all the StimCom interfaces (build in LabVIEW), available at BSS group of University of Twente. Furthermore, the computer that was used had at least the following specifications: Windows 764-bit (or higher), Intel Core i5 1.6 GHz (or higher), 8GB RAM (or more) and Bluetooth adapter. [7]

Bluetooth Interface

For starting the program, some steps were followed. First the "single_stimulus_with_trigger" virtual interface, found in the "LabVIEW" folder (containing all the StimCom interfaces), under "Application", is loaded in LabVIEW. Once this panel is opened the next step is to run it, by clicking on the white arrow at the top left corner. Subsequently, "StimCom_Bt_Control_v2" opens automatically, which starts the Bluetooth interface. At this point, it is important to make sure that the Bluetooth module is properly placed on the stimulator and the stimulator is sufficiently charged and turned on. Afterwards, "Search" should be pressed for the Bluetooth Interface to start looking for available Bluetooth connections in the area. After it is done searching, it is then possible to connect to the wanted stimulator. If the connection is correctly established, at the "Stim log" it is possible to keep track of all the communication that took place between the stimulator and the Bluetooth Controller. For a correct communication, the commands sent from the Bluetooth Controller to the stimulator should receive an echo, as explained in section 2.1.3. If correctly communicating, it is then possible to press "Stimulate" at the "single_stimulus_with_trigger" interface, for stimulating the OctoStim. If the stimulation took place correctly, OctoStim is expected to return back the string "$S, 0, 0, 0 \backslash 0$" for implying that the stimulation is done as shown in Figure 2.4. Figure 3.1 shows this Bluetooth interfaces.

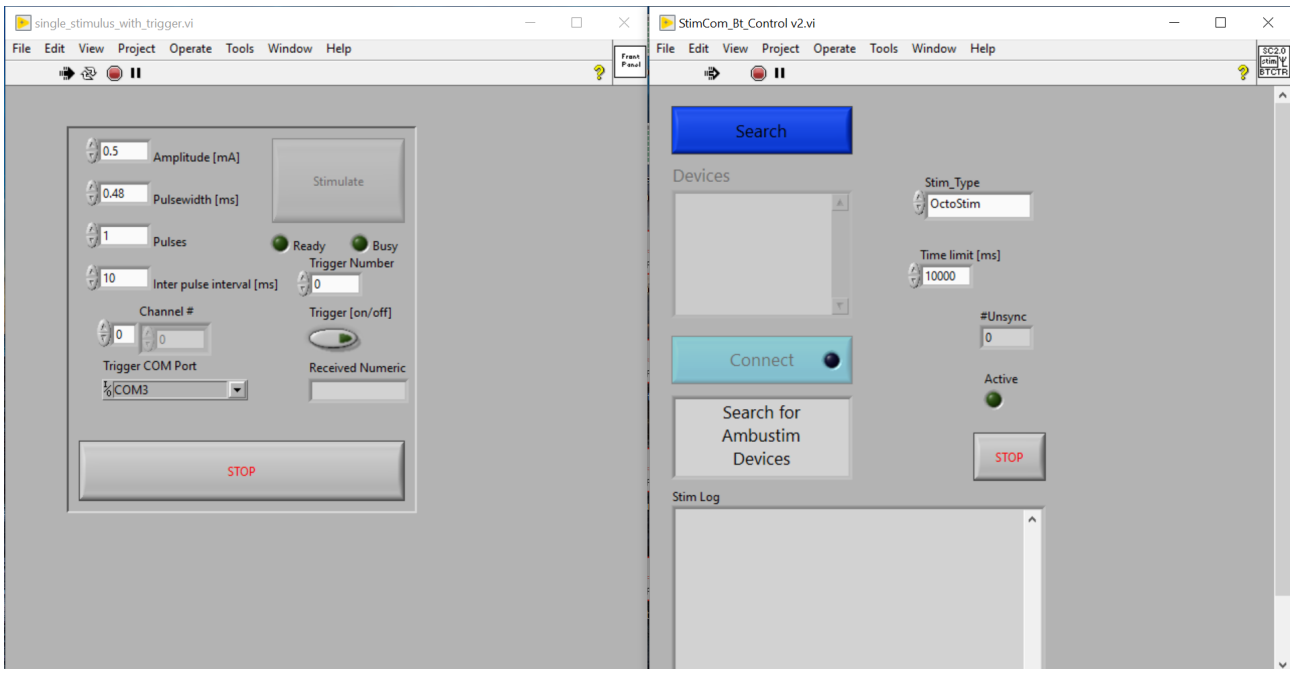


Figure 3.1: Bluetooth interfaces

3.1.2 Embedded Software Compiler

The compiler used in this study for the embedded software is the AtmelStudio 7.0, part of the AVR® Toolchain, which is a a collection of tools/libraries used to create applications for AVR microcontrollers. It provides everything needed to seamlessly edit, build, download and debug applications on the ATmega162 microcontroller. The embedded source code of the NociTRACK systems is written in standard C-language.

When working on a new project it is important to make sure that the configuration of the "make file" is properly done. This can be done/adjusted after opening the compiler and going to "Project" -> "Project Properties" -> "Build" -> "Make File Name". In the "Make File Name" it is important to check that the make file is put in the same directory as the source code files. Afterwards any changes performed at the C-code in the compiler will automatically be saved at the right source code. For programming the OctoStim the AVRISP mkII programmer device is utilized. This tool is used for field upgrades of 8-bit AVR microcontrollers with ISP or PDI interfaces. Furthermore, it supports target voltages from 1.8V to 5.5V and has short-circuit protection. It can program both flash and EEPROM memories, however only the first one is relevant for the OctoStim stimulator. This tool is USB 2.0 compliant and does not require external power supply, since it powers from the USB, which connects to the computer. The other end of this device is a 3x2 Header, which gets connected to the stimulator.

After successfully building the code by going to "Build" -> "Build Solution", the device can then be programmed by going to "Tools"-> "Device Programming". In the panel that opens, several properties needs to be selected and then applied, such as: Tool (AVRISP mkII), Device (ATmega162) and Interface (ISP). For successfully programming the stimulator also the device signature should be read, which refers to the stimulator's "battery level". The stimulator should be sufficiently charged for this to work. After applying all the changes, the device can finally be programmed by going to "Memories" -> "Flash" and then "Program",

which will erase the previous embedded code and reprogram the device with the new one. Below "Flash" the right .elf file with the right source code path should be inserted. It is also important to remove the Bluetooth module out of the stimulator when programming, in order to prevent overwriting it.

3.1.3 Design Validation

For validating the design, the generation of calibration curves for each of the channels, via a standard calibration program written in MATLAB (The Math-Works Inc., version 2017b) was used, developed by B. van der Berg, part of the BSS group, University of Twente. For doing so, an arduino-based trigger generation system was used, which can be connected to the computer (input) via an USB A to B cable, connected to the OctoStim stimulator via a BNC-to-OctoStim (3 pins) connector and connected to the Oscilloscope via another parallel cable. The channels of the OctoStim were connected to the Oscilloscope via some cables connected to a 10K ohm resistor.

For generating the calibration curves the "calibration_full" MATLAB code was used, to measure, analyze and validate the OctoStim waveforms. The Oscilloscope used for the calibration of the OctoStim is a Tektronix MSO 3014 Oscilloscope. Per channel a pulsetrain of 8 pulses was used with a pulsetrain repetition of 10, equating to 80 pulses generated per channel. A pulse width (PW) of 0.52 ms and a set of predefined amplitude values ranging from 0.2 to 0.62 mA was used. All amplitudes in this set are increased by a step-size of 0.06 mA. The inter pulse interval (IPI) used was 1 ms.

After calibrating the stimulator, a report was automatically generated by the MATLAB program, containing all the data and graphs collected during the calibration process.

3.2 Implementation of StimCom 0.3 for OctoStim

3.2.1 Procedure

The following procedure was followed for making the necessary modifications in the OctoStim simulator's embedded software:

- **First**: The electronics of the AmbuStim (the single-channel stimulator) were examined.
- **Second**: The electronics of the OctoStim (8-channel stimulator) were examined.
- **Third**: First step was compared to the second step to point out the differences at the end-stages between the two.
- **Fourth**: The fully-working embedded code of the AmbuStim was studied to figure out how the new version of the StimCom language was implemented there.
- **Fifth**: The old embedded code of the OctoStim was studied to see how this code is translated into the functions of its electronics.

- **Sixth**: By comparing both embedded codes of the AmbuStim and the OctoStim, and by keeping in mind how the translation of the OctoStim's electronics is done, the implementation of the new version of the StimCom language was also performed for the OctoStim stimulator.

The electronics of the AmbuStim and the OctoStim in Step 1 and 2 were examined by means of the schematics given in Appendix A.2 and A.3, respectively. Please refer to Section 2.2.2 and 2.3.1 for more elaborate explanation regarding the functions of the electronics and their differences at the end-stage.

3.2.2 OctoStim's Previous Embedded Code Issues

The previous version of the OctoStim's embedded software presented several issues that caused the stimulator to be dysfunctional. The two main issues are presented below listed by the order of when they emerged, and afterwards several sub-issues are discussed as being the reasons behind those two main issues. Then, the following section 3.2.3 shows the necessary adaptations that were made in the embedded code of the OctoStim stimulator for fixing those issues.

- **Issue 1**: Communication protocol issues. The commands sent from the Bluetooth Controller to the stimulator did not receive an echo command back, as explained in section 2.1.3.
- **Issue 2**: Stimulation issues (the S-Command). When stimulating, the OctoStim did not return back the string "$S, 0, 0, 0\backslash 0$" for implying that "Stimulation is Done".

The reasons behind the aforementioned issues were found to be that:

- **Sub-issue 1**: Old vector names were used.
- **Sub-issue 2**: The StimPhases were not implemented.
- **Sub-issue 3**: The "Parse Command" used an old version of StimCom language for most of the commands.
- **Sub-issue 4**: The max. response time from the "S-command" had changed.

3.2.3 The Performed Changes in the OctoStim's Embedded Code

1. Implementing the new vector names

The ATmega162 datasheet was used for updating the old Interrupt vector names used in the previous OctoStim code. [17].

Old vector name	Updated vector name
SIG_INTERRUPT0	INT0_vect
SIG_USART0_RECV	USART0_RXC_vect
SIG_OUTPUT_COMPARE3A	TIMER3_COMPA_vect
SIG_OUTPUT_COMPARE3B	TIMER3_COMPB_vect

2. Implementing the StimPhases

The Stimphases were not yet implemented in the old version of the OctoStim's embedded code, therefore the AmbuStim's approach for the Stimphases served as an example. In the AmbuStim's embedded code 6 cases are used for thoroughly describing all the StimPhases, however for the OctoStim only cases 3-6 were relevant for the Stimphases implementation. This was due to the fact that OctoStim does not have biphasic pulses, meaning that there is no interphase deadtime between the positive and negative phase of the pulse and neither is there a negative phase for the implementation of the "Start the negative phase" case. However, the relevant cases for realizing the OctoStim's Stimphases were:

- case 3: "Start InterPulse Deadtime (between pulses in the pattern)"
- case 4: "Start next pulse of the pattern"
- case 5: "Terminate S-command induced stimulation"
- case 6: "Conditionally start next pattern or terminate Q-command induced stimulation"

For the StimPhases implementation code please refer to Appendix A.5, line 135-279.

3. Updating the "Parse Command"

The commands in the "Parse Command" were outdated and used an old version of the StimCom command language. For example in the old StimCom language there used to be several "A" or "W" commands, which first specified the channel in which a pulse with a certain amplitude and pulse width was going to be and then applied the pulse amplitude or pulse width in that channel. In the new StimCom command language, all the amplitudes and widths of the pulses are defined within one "A" or "W" command and then the pattern played checks the priority lists of commands and then applies the amplitude/width of the pulse. So for example it first checks the "I" command, the "P" command and then "A" and "W".

Old version :

```
1    case 'A':// Set Positive Pulse Amplitude

3  if( (NumPar==2) & (Par[0]>0) & (Par[0]<=NUMCHANNELS) )
   {
5    if(Par[1]>PA_MAX) Par[1]=PA_MAX;
     if(Par[1]<PA_MIN) Par[1]=PA_MIN;
7    Par[1]=Par[1]&(0xffff);            //Resolution off AD
     NegAmp[Par[0]-1]=Par[1];
9  }
   else {KeyWord[0]=NOTACK;NumPar=0;}
11 break;
```

New adjusted version:

```
case 'A':// Set Positive Pulse Amplitude
if(NumPar==PatternLength)
{
  for(ParNo=0;ParNo<NumPar;++ParNo)
  {
    Par[ParNo]=Par[ParNo]&(0x0fff);
    PosAmp[ParNo]=Par[ParNo];
  }
}
else {KeyWord[0]=NOTACK;NumPar=0;}
break;
```

Similarly for most of the commands such adjustments were made. The new code of the "Parse Command" can be found in Appendix A.5, line 584-782.

4. Changing the "S-command"

The S-command is composed by three parameters S(x,y,z). Parameter z denotes the delay in time-units. This delay stands for the maximum response time. Since no response button is yet connected to the stimulator, and no response time is being measured, there is no need for a maximum response time, therefore this parameter was adjusted all over the code, as well as in the "S" command itself. The z parameter was set to zero.

```
case 'S':// Set stimulation mode (NumTriggers, NumPatterns, StimDelay)
if(NumPar==3)
{
  NumTriggers=Par[0];
  NumPatterns=Par[1];
//  Par[2]=0;
  StimDelay=Par[2];

  if(NumTriggers==0)
  {
    PatternCount=NumPatterns;
  }

  Par[0]=NumTriggers;
  Par[1]=NumPatterns;
//  Par[2]=StimDelay;
  Par[2]=0;
}
else {KeyWord[0]=NOTACK;NumPar=0;}
break;
```

4 Results and Discussion

4.1 Results of StimCom 0.3 Implementation

4.1.1 Communication Establishment

The first issue encountered with the old version of the embedded code was the communication protocol issue. However, after the changes performed in the code, as explained in Section 3.2.3 it was achieved to establish the communication between the Bluetooth Controller and the OctoStim stimulator. A demonstration of the established communication is shown in Figure 4.1.

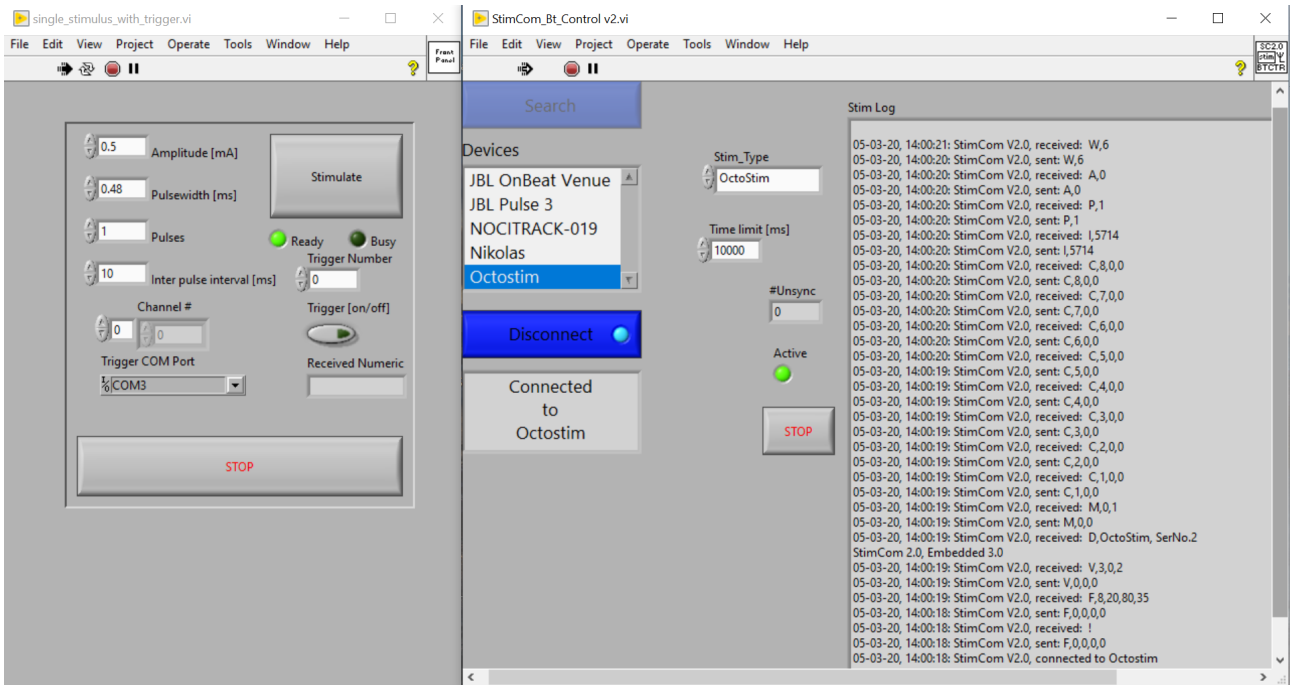


Figure 4.1: Established connection and communication between the LabVIEW interface and the OctoStim stimulator via Bluetooth. The procedure followed to launch this Bluetooth interface is step-by-step explained in section 3.1.1.

As stated in section 3.1.1, when the connection is correctly established, at the "Stim log" it is possible to keep track of all the communication that took place between the stimulator and the Bluetooth Controller. Figure 4.1 clearly shows an established connection and communication, since the commands sent from the Bluetooth Controller to the stimulator indeed receive the right command/echo, as explained in section 2.1.3.

4.1.2 Stimulation Establishment

After the communication was established as seen in the previous section, the next step for verification was trying to stimulate the OctoStim, by sending an "S" command. This was also achieved and it is shown in Figure 4.2. The string "$S, 0, 1, 0$" is send to the stimulator from the Bluetooth Controller and the stimulator correctly answers with an ACK, echo of the command. Since the given delay is zero, the OctoStim then immediately starts applying the stimulation pattern and returns back the string "$S, 0, 0, 0$", for implying that the stimulation is done and all patterns are stimulated. This is also the case shown in the Figure below.

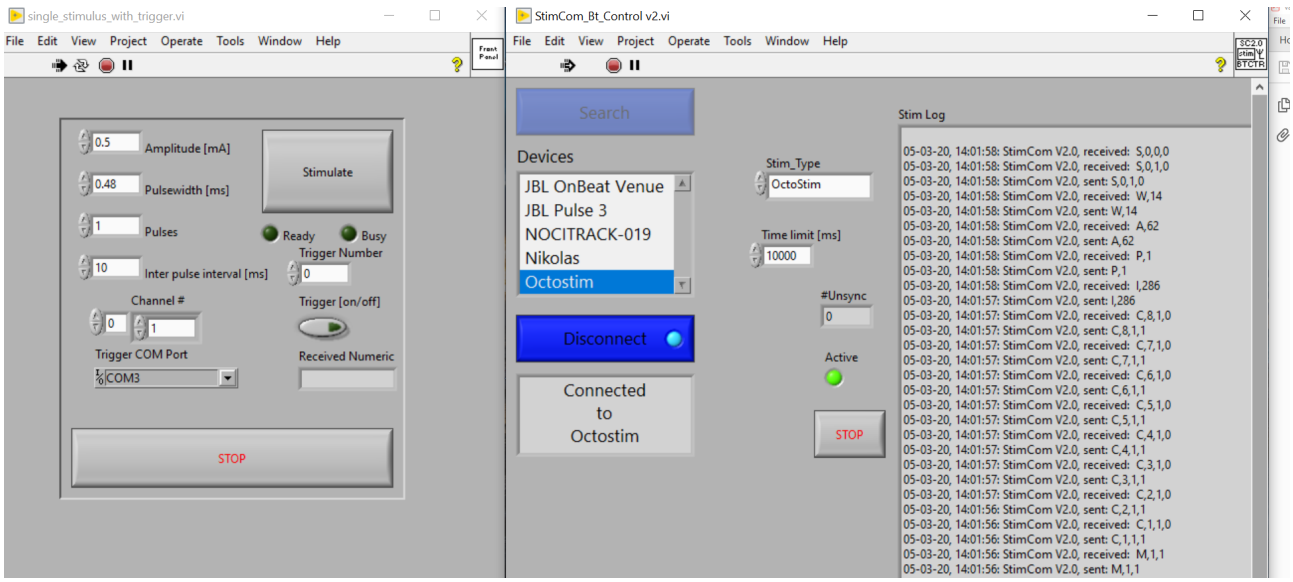


Figure 4.2: Established Stimulation. A Communication demonstration between the Bluetooth Controller and the OctoStim when stimulating (when stimulation command "S" is sent)

4.2 Stimulus generation in multiple channels

Below it is demonstrated that it is possible to make a stimuli sequence, with stimuli in different channels of the OctoStim. Two examples are shown and illustrated in Figure 4.3 and Figure 4.4. So for instance, in Figure 4.3 the first stimulus happens in channel number one followed by a second stimulus after an IPI of 2 ms in channel number two and finally another stimulus after again 2 ms in channel number three. In Figure 4.4 this sequence is done in an decreasing order and then in an increasing order, so the stimuli was sent in the following channel order: channel 1, channel 2, channel 3, channel 3, channel 2 and channel 1 again. Th PW used was 2 ms and the amplitude of the pulses 1 mA. OctoStim seems to generate stimuli with negative phase, despite the expectations of generating positive phase stimuli. This can be due to reverse connection of anode-cathode cables when connecting the channels of the OctoStim to the Oscilloscope.

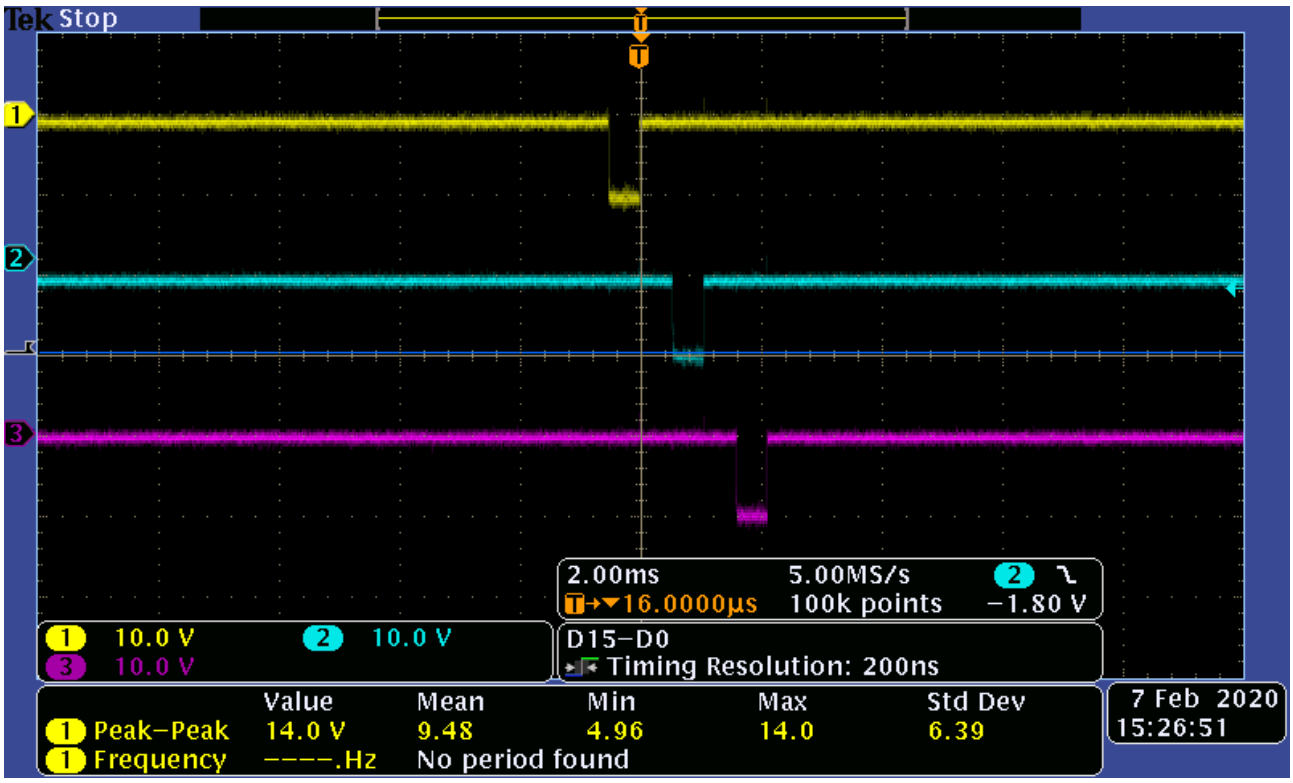


Figure 4.3: Stimuli sequence of 3 stimuli

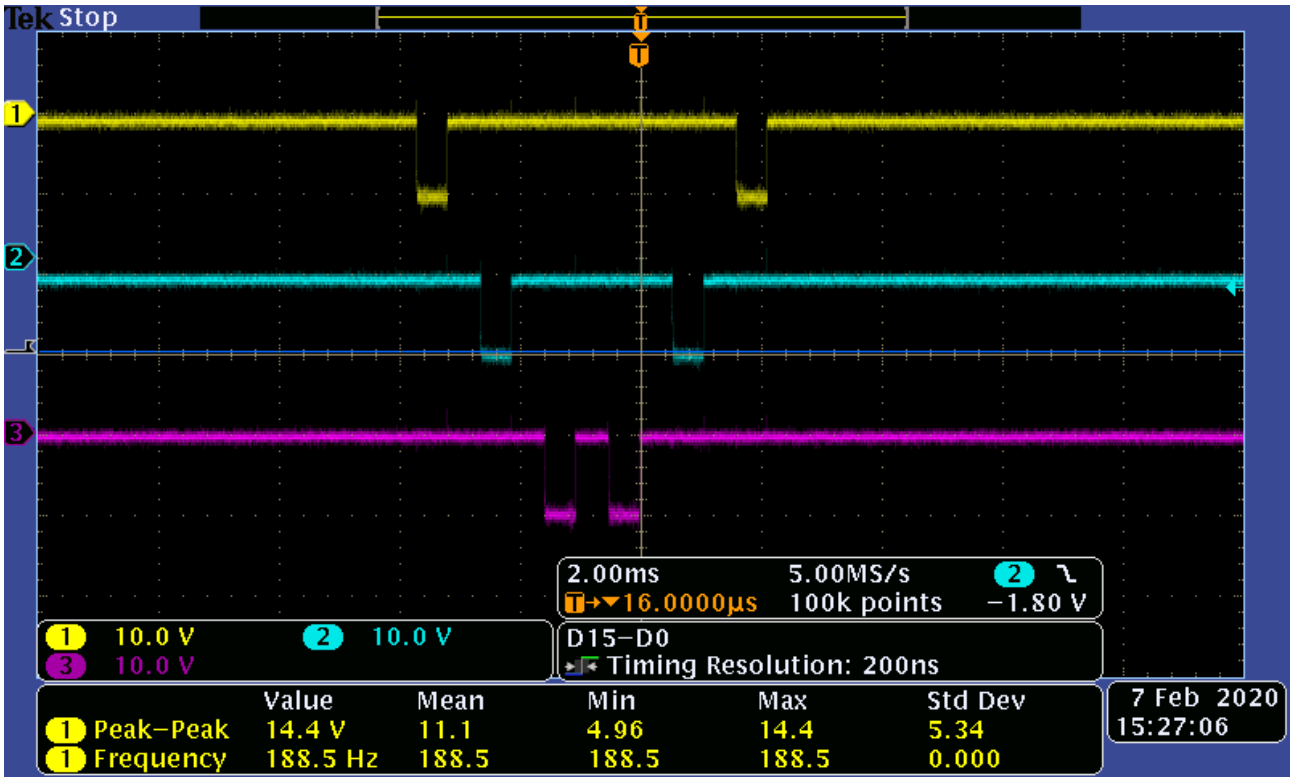


Figure 4.4: Stimuli sequence of 6 stimuli

4.3 Calibration curves

All channels are calibrated using the method described in section 3.1.3. However, only the results of the first calibrated channel of the OctoStim are discussed and summarized below by the graphs. Since all the other channels showed similar behaviour and similar results, it is assumed that this discussion also applies for those channels in the same manner. All the other channel's calibration reports can be found in Appendix A.6.

The plots in Figure 4.5 show the average pulse amplitude waveform and the average glitch amplitude against the number of samples. The OctoStim appears to have no glitches, since there is no overshoot detected in the glitch amplitude plot and the waveform seems to directly converge to the right amplitude value. The glitch amplitude is measured as the maximum absolute value of the blue waveform (measured amplitude) minus the maximum absolute value of the red square-wave form (fitted waveform).

The average amplitude waveform is measured by fitting a square wave form to the measured waveform. After doing so, it optimizes the square waveform to fit as well as possible to the measured waveform, by looking at its amplitude values.

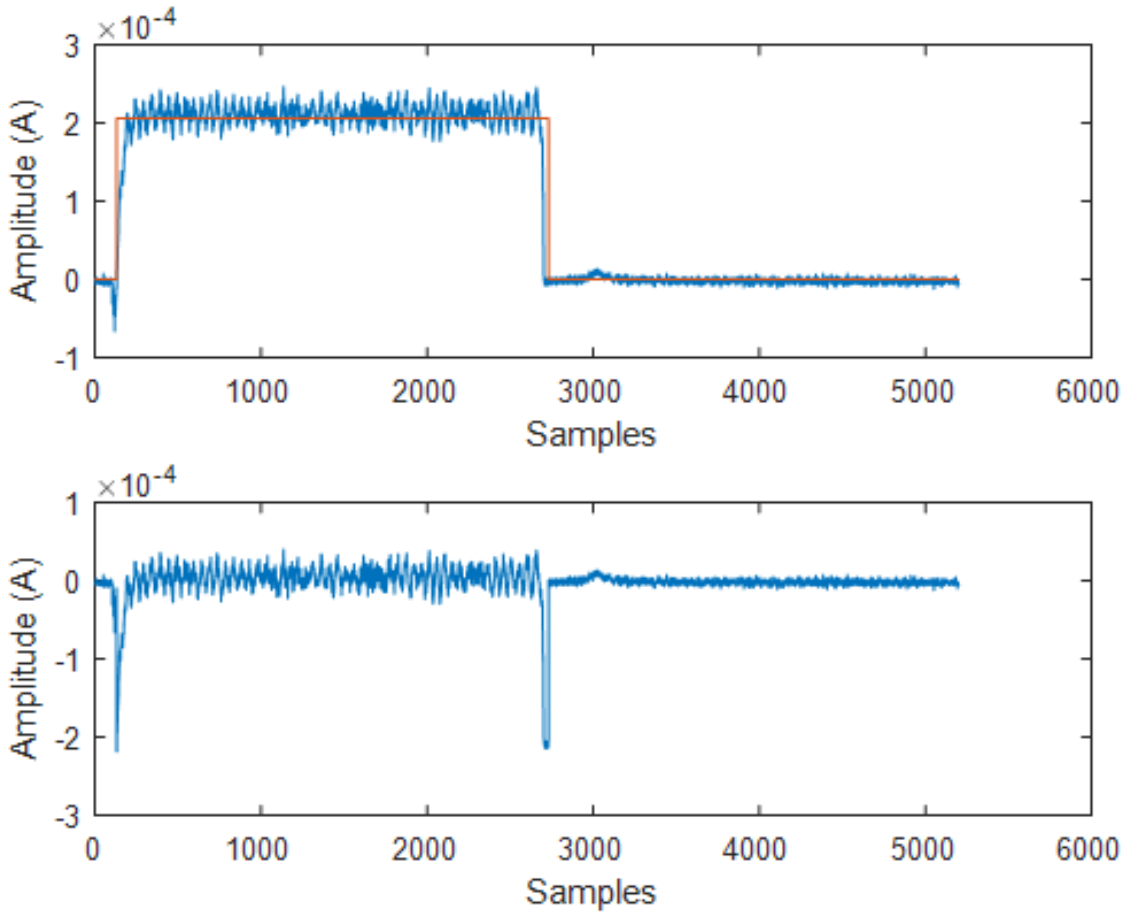


Figure 4.5: Average waveform (top), Average glitches (bottom)

The mean bias and standard deviation of the uncalibrated error as calculated by the MATLAB program are shown below:

mean bias	-0.20378 mA
standard deviation	0.074808 mA

The plot below 4.6 shows the error between the desired amplitude and the measured amplitude, when the first channel of the stimulator is still not calibrated. As mentioned in section 3.1.3, the amplitudes sent to the stimulator are from 0.2 to 0.62 mA with a step size of 0.06 mA. The plot suggest that the error is indeed large when the channel is still not calibrated since, for instance for a desired amplitude of 0.2 mA, the stimulator measures an amplitude with an error bias of approximately -0.09, so the measured amplitude would be approximately 0.11 mA, which makes for an error of approximately 45%, which is almost half the desired value. So, the error starts with a bias of -0.09 and then it goes higher, with the increase of the amplitude. However, this linear relation shows a predictable behaviour and a systematic error, which can easily be fixed when the device is calibrated. If the program can pick this predictable behaviour, it then knows that it can subtract this value for calibrating the device and lessening the error. The large error when the device is still not calibrated may be due to faulty or old components in the circuitry of the OctoStim.

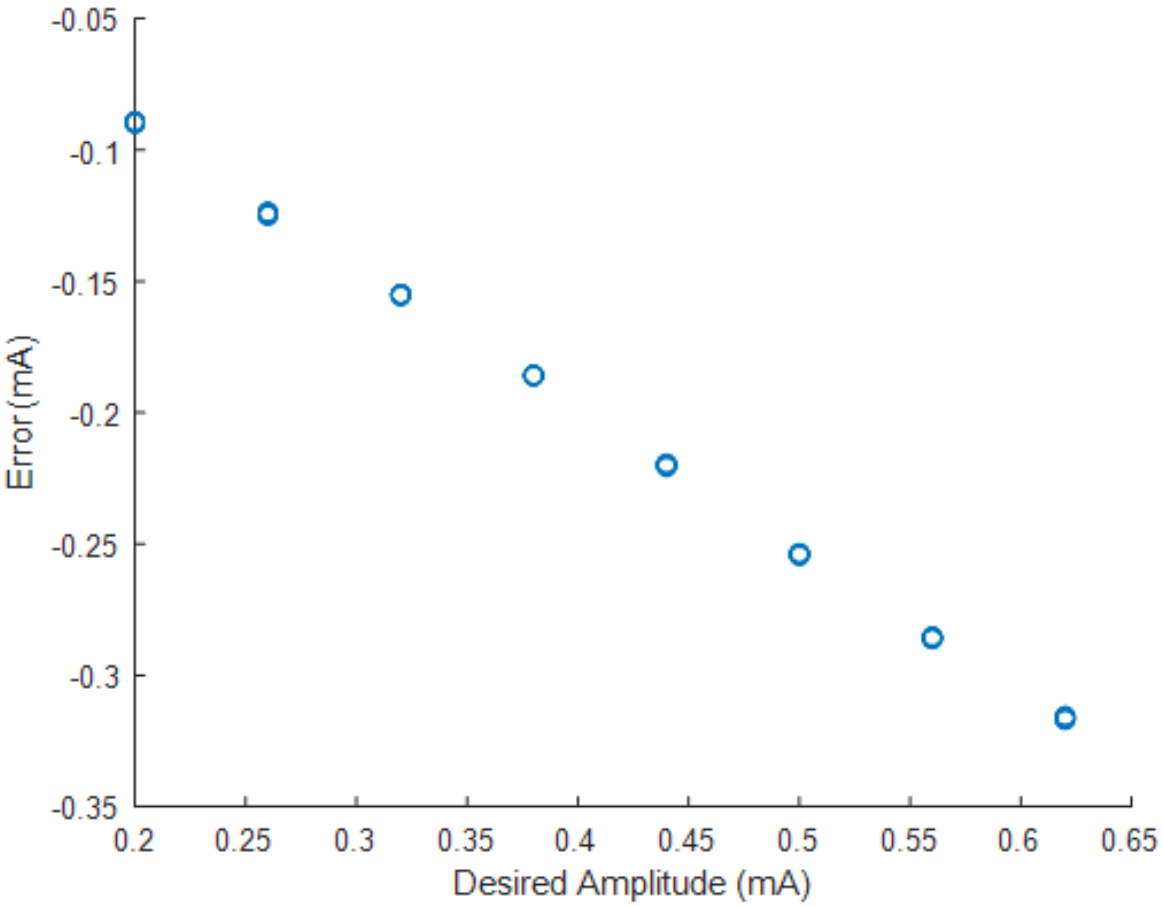


Figure 4.6: Error plot when the stimulator is still uncalibrated

The following table shows the values of the intercept and slope by using the equation of the line (y = intercept + slope * x): With (x,y) being the calibration constants predefined in the program (0.027319, 0.439674).

intercept	0.017888
slope	0.45933

The mean bias and standard deviation of the calibrated error as calculated by the MATLAB program are shown below:

mean bias	-0.0018048 mA
standard deviation	0.0033416 mA

Figure 4.7 below shows the calibrated error plot between the desired amplitude and the measured amplitude when the stimulator is calibrated. After calibration the error appears to be quite small, around 1%. For a desired amplitude of 0.2 mA, the calibrated device gives a measured amplitude value of around 0.198 mA, which is very close to the desired value.

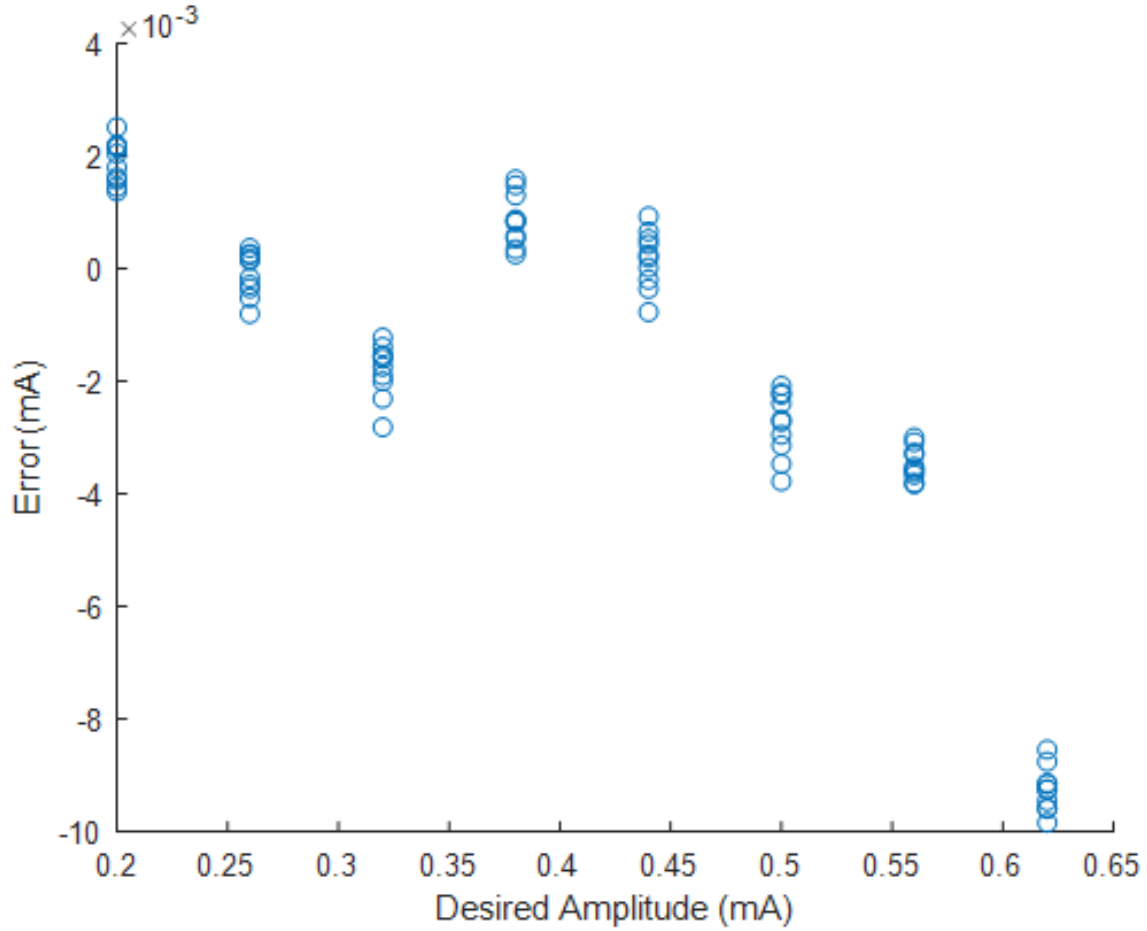


Figure 4.7: Error plot when the stimulator is calibrated

After calibration the program also analyses the delay with respect to the trigger, so how much time after triggering does the device require to stimulate. The mean delay, standard deviation and the largest delay that it measures are given in the table below. The delay time shows to be quite small, and within the wanted value, which is below 1 ms.

Trigger delay:

mean	4.1945e-05 s
standard deviation	1.4898e-05 s
latest	5.28e-05 s

The results show that the device and software function correctly and that the device has been successfully calibrated for channel one.

5 | Conclusion and Further Recommendations

This study focused on the comparative analysis between the fully-working embedded code of the AmbuStim and the outdated embedded code of the OctoStim. Several changes were made in the OctoStim's embedded software to make it compatible with the new version of the StimCom language. It was achieved to modify the previous code and implement the new version of the StimCom command language, to establish the communication between the OctoStim and the Bluetooth Controller and to modify the S-command for a proper stimulation. Also the generation of stimuli and calibrated stimuli in all 8 channels was demonstrated for this extended setup. Sending a sequence of stimuli in different channels of the OctoStim was also proved to be possible. Hence, the objective of this research was achieved. The calibrated stimuli and the generation of multiple stimuli on multiple channels hints for feasible multichannel NDT tracking experiments in human subjects.

It was found that before calibration the error between the desired amplitude of the pulse and the measured amplitude of the pulse was indeed considerably big, around 45%. The bigger the amplitude values, the bigger the error found, thus it showed a linear behaviour. However, after calibration it was seen that the error was significantly reduced and became quite small, around 1%. It was also found that OctoStim seems to not have glitches, since there is no overshoot detected in the glitch amplitude plot. Also, when generating pulses via connecting the OctoStim to the Oscilloscope, it is noticed that OctoStim generates negative-phase pulses.

For further research, it may be useful to create an Investigational Medical Device Dossier (IMDD) for the OctoStim, using the results provided by this thesis, in order to permit human subject experiments. Moreover as a next step, it would be convenient to conduct multi-channel NDT tracking experiments in human subjects to confirm the feasibility of performing such experiments, based on the assumptions made in this thesis. It would also be of interest to investigate further into the generation of negative pulses on the OctoStim instead of positive ones, even though this is believed to happen because of reverse connection of anode-cathode cables when connecting the channels of the OctoStim to the Oscilloscope.

6 | Bibliography

[1] J. A. Hunfeld, C. W. Perquin, H. J. Duivenvoorden, A. A. Hazebroek-Kampschreur, J. Passchier, L. W. van Suijlekom-Smit, and J. C. van der Wouden, "Chronic pain and its impact on quality of life in adolescents and their families," *Journal of Pedriatic Psychology, vol. 26, Issue 3, pp. 145-153*, 2001.

[2] R. D. Treede, W. Rief, A. Barke, Q. Aziz, M. I. Bennett, R. Benoliel, ..., and S. J. Wang, "A classification of chronic pain for ICD-11," *Pain, vol. 156, Issue 6, pp. 1003-1007*, 2005.

[3] "Chronic pain: Backgrounder," *Boston Scientific: Advanced science for life*, 2012. [Online]. Available: https://pae-eu.eu/wp-content/uploads/2015/03/NM-86505-AA_INTL-Chronic-Pain-Backgrounder_Final-2.pdf

[4] C. Harstall and M. Ospina, "How prevalent is chronic pain?" *Pain: Clinical Updates, vol. 11, Issue 2*, 2003.

[5] G. E. Bekkering, M. M. Bala, K. Reid, E. Kellen, J. Harker, R. Riemsma, . . ., and J. Kleijnen, "Epidemiology of chronic pain and its treatment in the netherlands," *Netherlands Journal of Medicine, vol. 63, Issue 3, pp. 141-153*, 2011.

[6] R. Doll, "Psychophysical methods for improved observation of nociceptive processing," *University of Twente*, 2016. [Online]. Available: https://doi.org/10.3990/1.9789036540377

[7] B. van den Berg, "Stimulus related evoked potentials around the nociceptive detection threshold," *University of Twente*, 2018.

[8] R. Doll, P. Veltink, and J. Buitenweg, "Effect of temporal stimulus properties on the nociceptive detection probability using intra-epidermal electrical stimulation," *Experimental Brain Research 234(1) pp. 219-227*, 2016.

[9] O. H. Wilder-Smith and et al, "Patients with chronic pain after abdominal surgery show less preoperative endogenous pain inhibition and more postoperative hyperalgesia: A pilot study," *Pain Palliative Care Pharmacotherapy 24(2), pp. 119-128*, 2010.

[10] M. M. Backonja and et al, "Value of quantitative sensory testing in neurological and pain disorders: NeuPSIG consensus," *Pain 154(9), pp. 1807-1819*, 2013.

[11] A. S.Quevedo and R. C. Coghill, "Attentional modulation of spatial integration of pain: Evidence for dynamic spatial tuning," *JNeurosci. 27(43), pp. 11635–11640*, 2007. [Online]. Available: https://doi.org/10.1523/JNEUROSCI.3356-07.2007

[12] P. Steenbergen, “Somatosensory perceptual maps of electrocutaneous stimulation,” *University of Twente*, 2013. [Online]. Available: https://doi.org/10.3990/1.9789036516662

[13] J. Sandkühler, “Models and mechanisms of hyperalgesia and allodynia,” *Physiological Reviews 89(2), pp. 707-758*, 2009. [Online]. Available: https://doi.org/10.1152/physrev.00025.2008

[14] “The RS-232 protocol,” *Omega*, 2019. [Online]. Available: https://www.omega.com/en-us/resources/rs232-serial-communication

[15] W. Hofman, “Stimcom protocol specification used in nocitrack systems, stimcom version 0.3,” n.d., unpublished.

[16] J. R. Buitenweg, “Investigational Medical Device Dossier Nocitrack Ambustim PT,” University of Twente, Tech. Rep., 19 February 2020, unpublished.

[17] “8-bit AVR® Microcontroller with 16K Bytes In-System Programmable Flash,” *Atmel*, 2013, ATmega162, ATmega162V datasheets. [Online]. Available: https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-2513-8-bit-AVR-Microntroller-ATmega162_Datasheet.pdf

[18] “Parani-ESD100/110/200/210,” *Sena*, n.d. [Online]. Available: http://www.er-soft.com/files/media/files/Sena--Bluetooth--Serial--RS232--RS422--RS485--OEM--Module--Parani--ESD--100-110-200-2pdf

[19] “SERIES AV/SMV SURFACE MOUNT and PLUG-IN,” *Pico Electronics*, 2007. [Online]. Available: https://www.digchip.com/datasheets/parts/datasheet/1603/5SMV100-pdf.php

[20] “2.5 v to 5.5 v, 115 a, parallel interface single voltage-output 8-/10-/12-bit dacs,” *Analog Devices*, n.d. [Online]. Available: https://pdf1.alldatasheet.com/datasheet-pdf/view/48031/AD/AD5341.html

A | Appendices

A.1 Q-Command

Q Command:

The Q command performs a stimulus response serie according to the 5 parameters which are:

Qstart,Qstep,Qstop,Qcrit,Qindex.

Qstart is the start value of the pulse amplitude in m A,

Qstep the stepsize in mA,

Qstop the end value of the amplitude,

Qcrit When this value is 1 (external button is pressed) a ramp is generated ranging from Qstart to the moment the button is released or Qstop is reached.

Qindex, number of the pulse which has to be ramped.

Example Q Command:

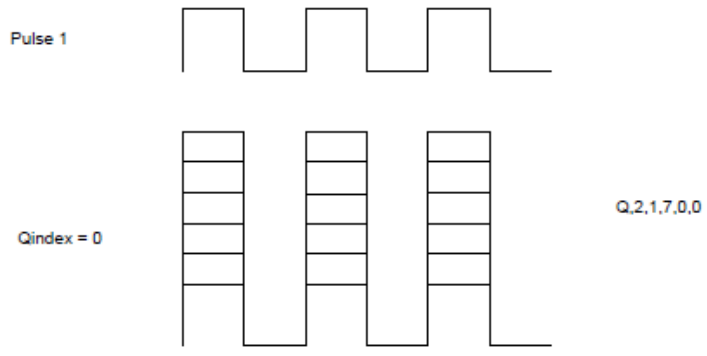
Qstart = 2mA

Qstep = 1mA

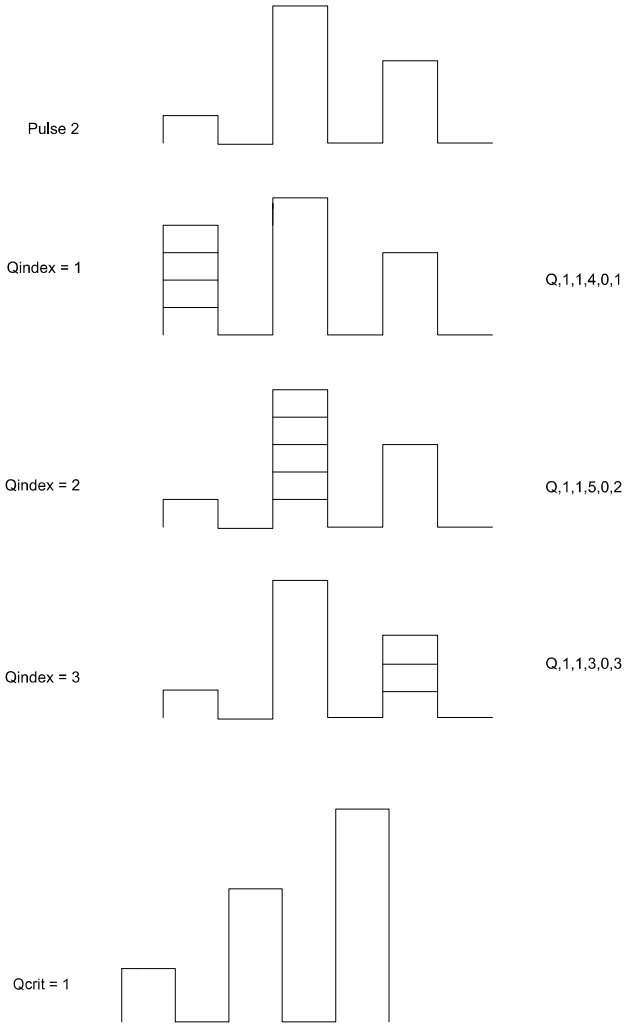
Qstop = 7mA

Qcrit = 0

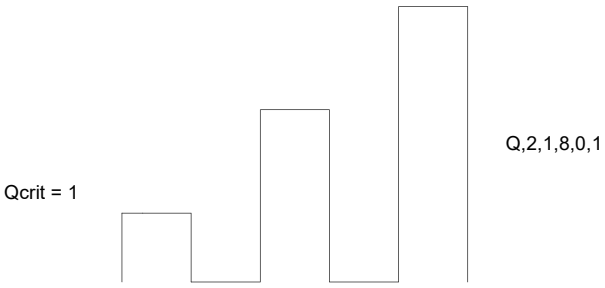
Qindex = 0



When Qindex = 0 all pulses are ramped according to the parameters. In this case starting from a current from 2mA to 7mA in steps of 1mA.



FIGUUR 2



FIGUUR 3

A.2 AmbuStim Schematics

CONFIDENTIAL

This Appendix has been marked confidential. Further information can be requested at the Biomedical Signals and Systems (BSS) department, University of Twente, NETHERLANDS.

A.3 OctoStim Schematics

CONFIDENTIAL

This Appendix has been marked confidential. Further information can be requested at the Biomedical Signals and Systems (BSS) department, University of Twente, NETHERLANDS.

A.4 Investigational Medical Device Dossier (IMDD) Noci-TRACK AmbuStim

CONFIDENTIAL

This Appendix has been marked confidential. Further information can be requested at the Biomedical Signals and Systems (BSS) department, University of Twente, NETHERLANDS.

A.5 OctoStim Source Code

CONFIDENTIAL

This Appendix has been marked confidential. Further information can be requested at the Biomedical Signals and Systems (BSS) department, University of Twente, NETHERLANDS.

A.6 Calibration curves

Report: NociTRACK Calibration

Device: OctoStim_02_chan01

Time: 12-Feb-2020 15:51:25

Method

The device was calibrated using a Tektronix MSO 3014 Oscilloscope. The properties of the calibration stimuli are listed below.

Pulse-width: 0.52 ms

Range: 0.2 to 0.62 mA

Pulse increment: 0.06 mA

Pulsetrain repetitions: 10

Pulses generated: 80

Results

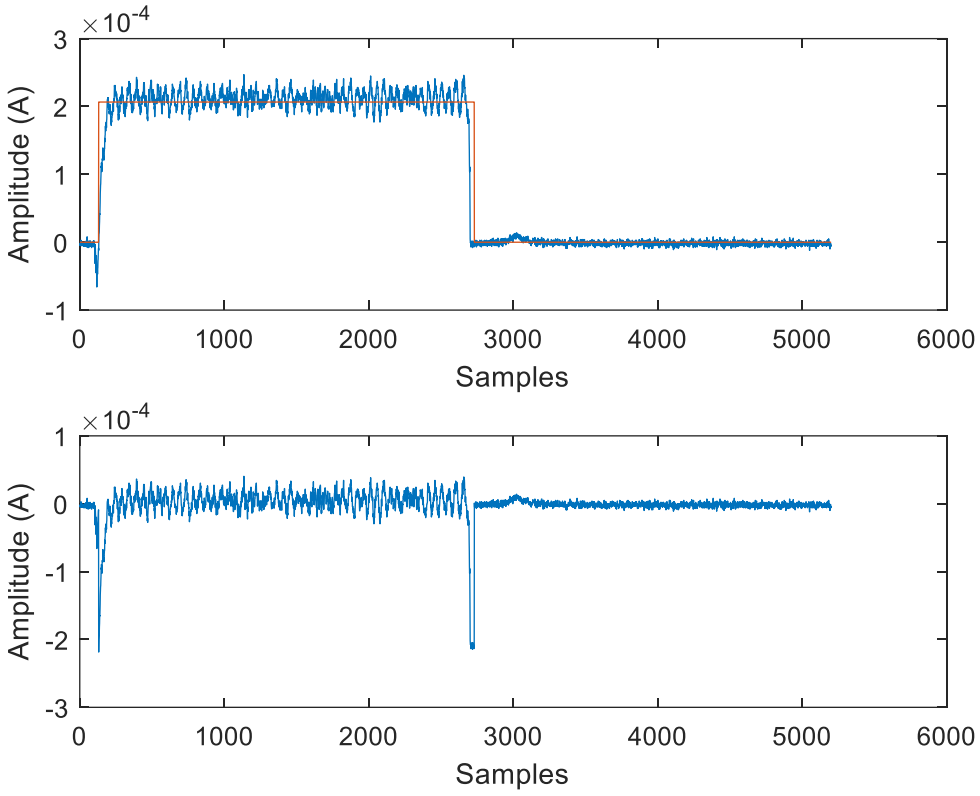
The device has been calibrated succesfully and calibration constants are saved to "calibration.txt". The results are summarized by the graphs below.

The results are summarized by the graphs below.

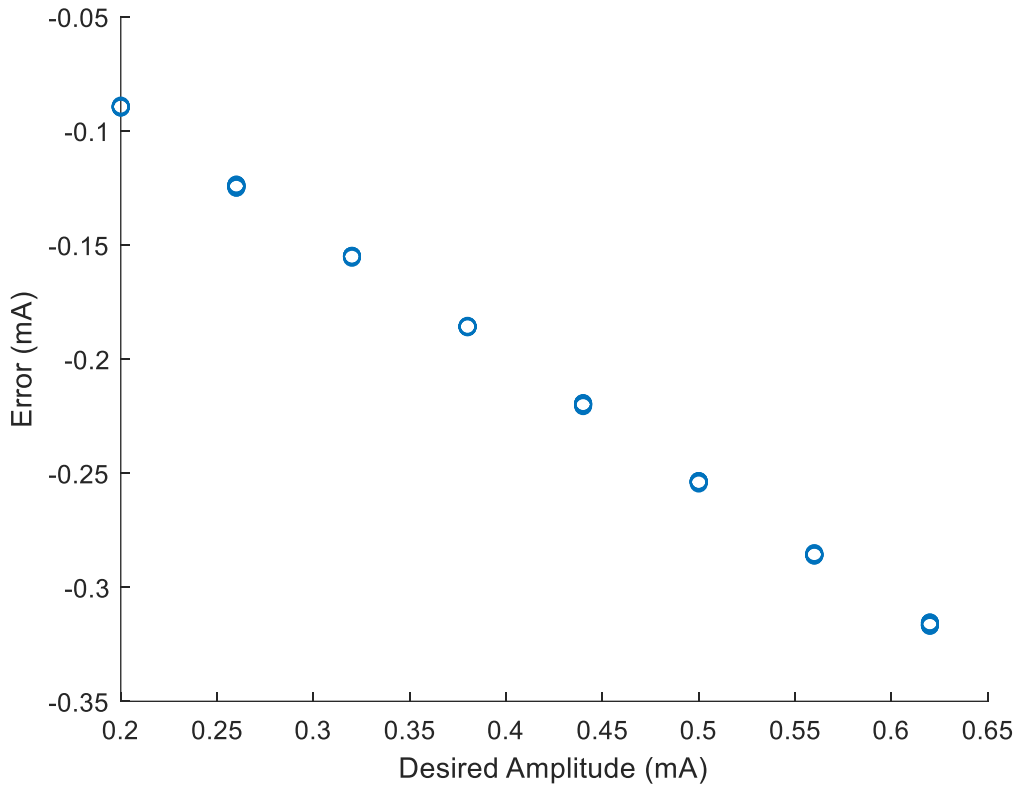
Error (uncalibrated):

mean bias	-0.20378 mA
standard deviation	0.074808 mA

Average waveform (top) and average glitches (bottom):



Error plot (uncalibrated):



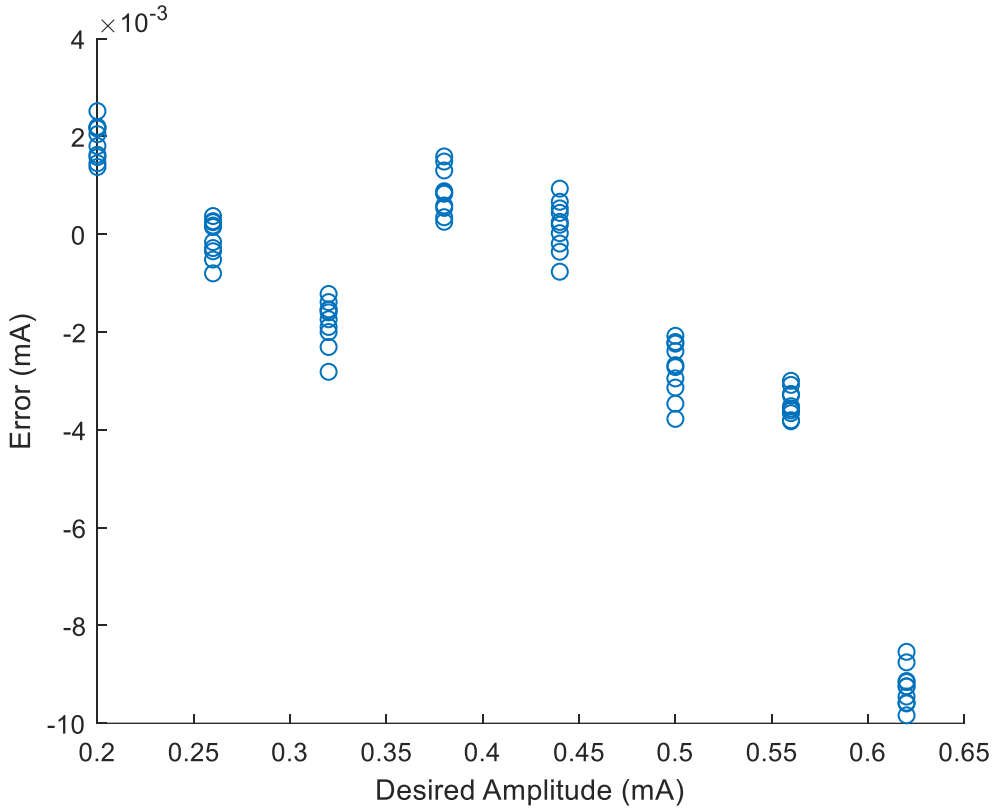
Calibration constants (y = intercept + slope * x):

intercept	0.017888
slope	0.45933

Error (calibrated):

mean bias	-0.0018048 mA
standard deviation	0.0033416 mA

Error plot (calibrated):



Trigger delay:

mean	4.1945e-05 s
standard deviation	1.4898e-05 s
latest	5.28e-05 s

Conclusion

The device and software function correctly. The device has been calibrated succesfully.

Calibration performed by: .....

Signature: .....

Report: NociTRACK Calibration

Device: OctoStim_02_chan02

Time: 12-Feb-2020 16:19:05

Method

The device was calibrated using a Tektronix MSO 3014 Oscilloscope. The properties of the calibration stimuli are listed below.

Pulse-width: 0.52 ms

Range: 0.2 to 0.62 mA

Pulse increment: 0.06 mA

Pulsetrain repetitions: 10

Pulses generated: 80

Results

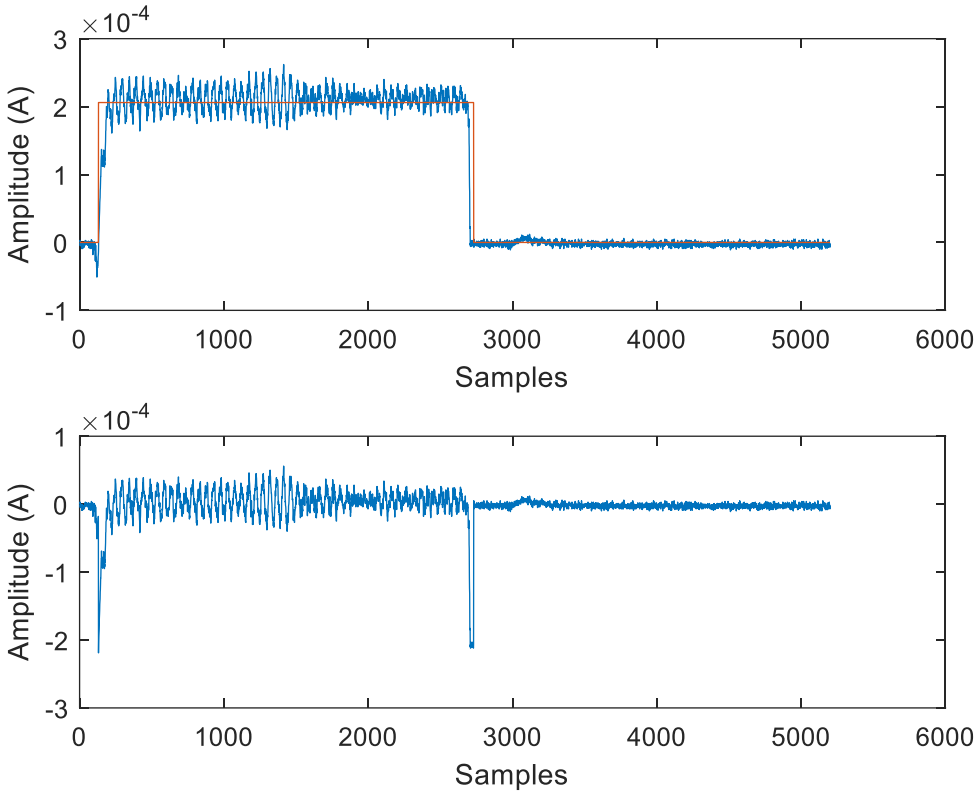
The device has been calibrated succesfully and calibration constants are saved to "calibration.txt". The results are summarized by the graphs below.

The results are summarized by the graphs below.

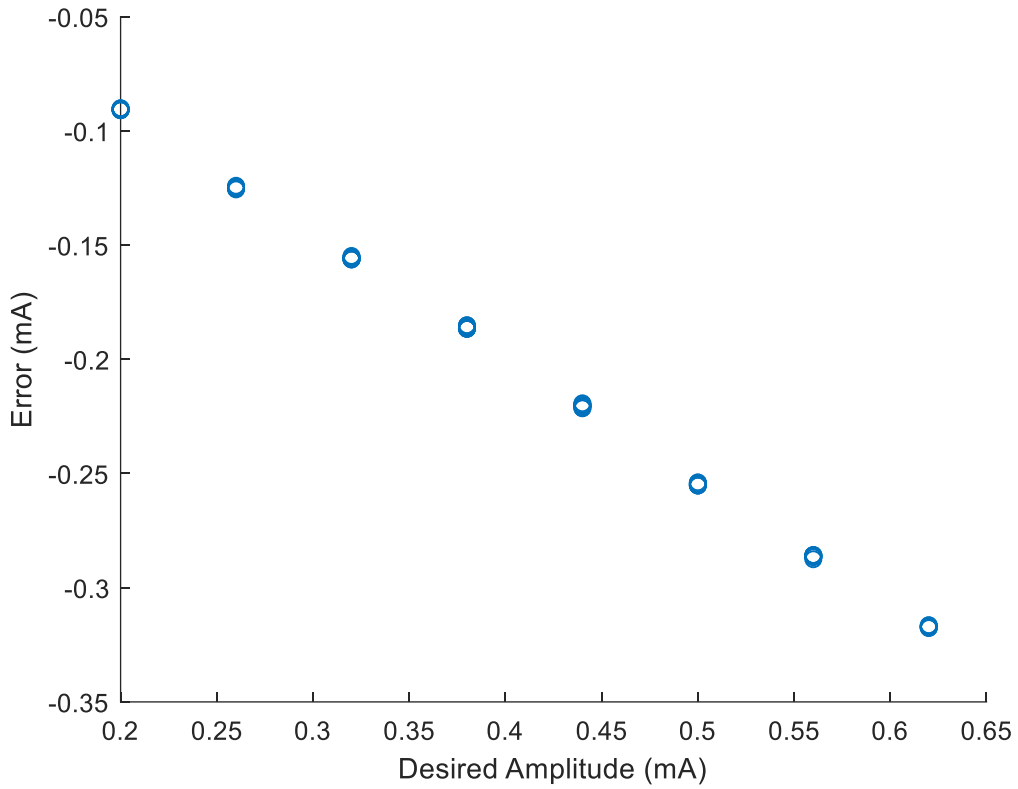
Error (uncalibrated):

mean bias	-0.20447 mA
standard deviation	0.074777 mA

Average waveform (top) and average glitches (bottom):



Error plot (uncalibrated):



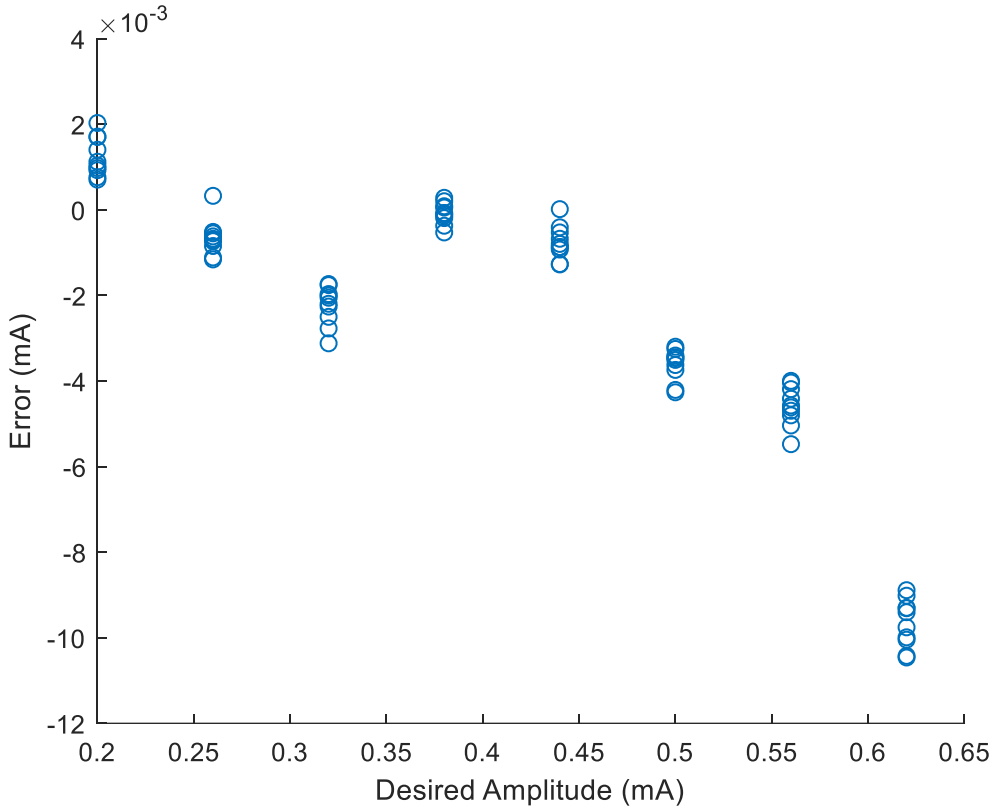
Calibration constants (y = intercept + slope * x):

intercept	0.017114
slope	0.45956

Error (calibrated):

mean bias	-0.0025449 mA
standard deviation	0.0032691 mA

Error plot (calibrated):



Trigger delay:

mean	4.6727e-05 s
standard deviation	9.0956e-06 s
latest	5.56e-05 s

Conclusion

The device and software function correctly. The device has been calibrated succesfully.

Calibration performed by: .....

Signature: .....

Report: NociTRACK Calibration

Device: OctoStim_02_chan03

Time: 12-Feb-2020 16:25:15

Method

The device was calibrated using a Tektronix MSO 3014 Oscilloscope. The properties of the calibration stimuli are listed below.

Pulse-width: 0.52 ms

Range: 0.2 to 0.62 mA

Pulse increment: 0.06 mA

Pulsetrain repetitions: 10

Pulses generated: 80

Results

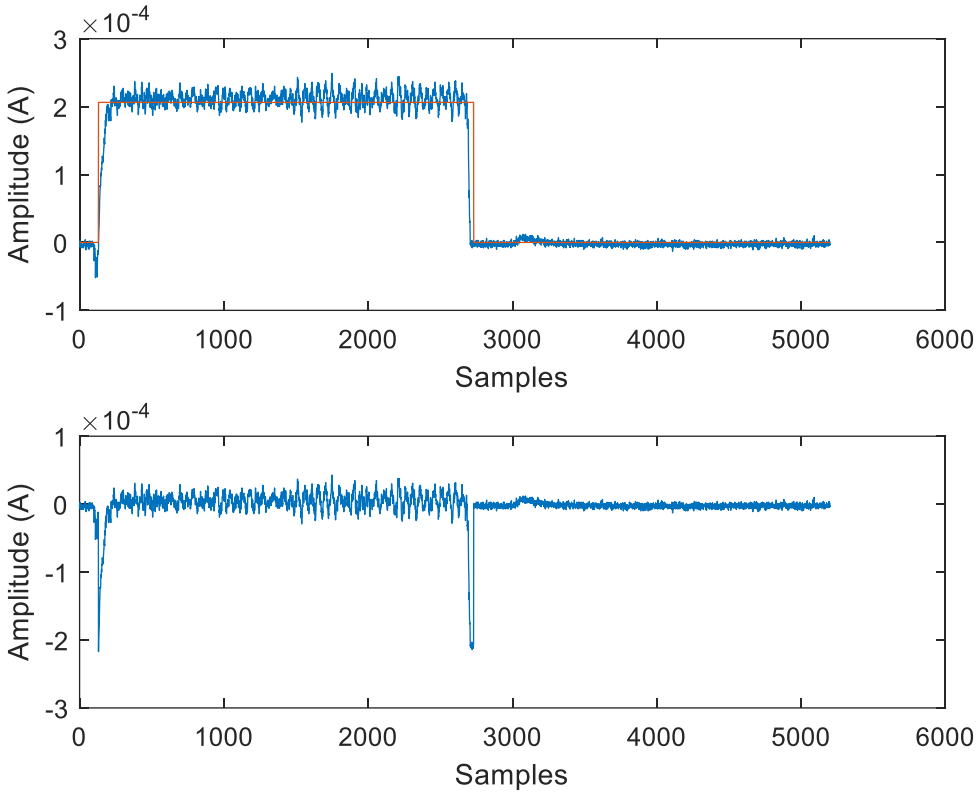
The device has been calibrated succesfully and calibration constants are saved to "calibration.txt". The results are summarized by the graphs below.

The results are summarized by the graphs below.

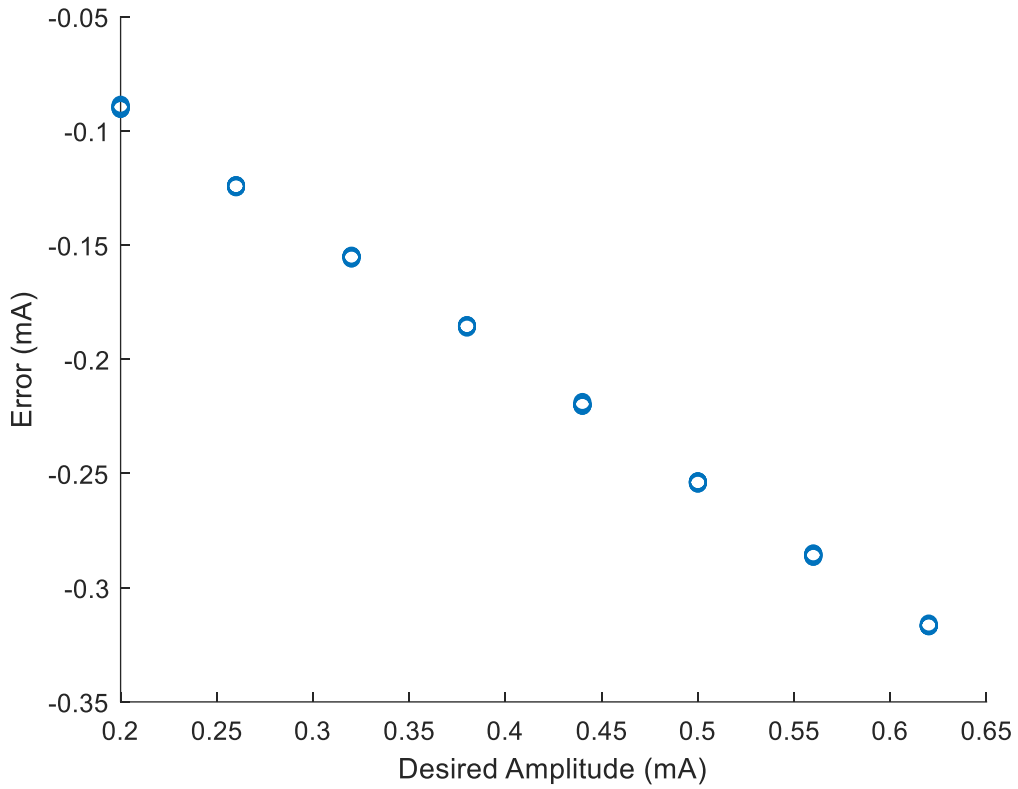
Error (uncalibrated):

mean bias	-0.2038 mA
standard deviation	0.074852 mA

Average waveform (top) and average glitches (bottom):



Error plot (uncalibrated):



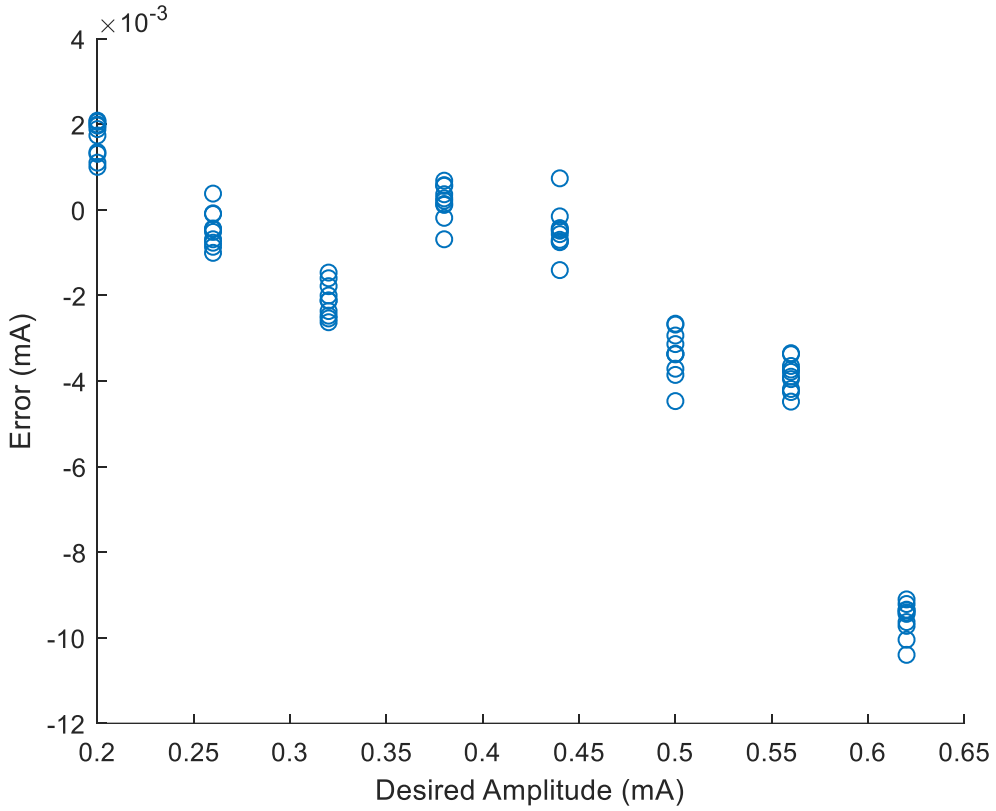
Calibration constants (y = intercept + slope * x):

intercept	0.018003
slope	0.45902

Error (calibrated):

mean bias	-0.0022506 mA
standard deviation	0.0033057 mA

Error plot (calibrated):



Trigger delay:

mean	3.7745e-05 s
standard deviation	1.3677e-05 s
latest	5.12e-05 s

Conclusion

The device and software function correctly. The device has been calibrated succesfully.

Calibration performed by: .....

Signature: .....

Report: NociTRACK Calibration

Device: OctoStim_02_chan004

Time: 12-Feb-2020 16:40:00

Method

The device was calibrated using a Tektronix MSO 3014 Oscilloscope. The properties of the calibration stimuli are listed below.

Pulse-width: 0.52 ms

Range: 0.2 to 0.62 mA

Pulse increment: 0.06 mA

Pulsetrain repetitions: 10

Pulses generated: 80

Results

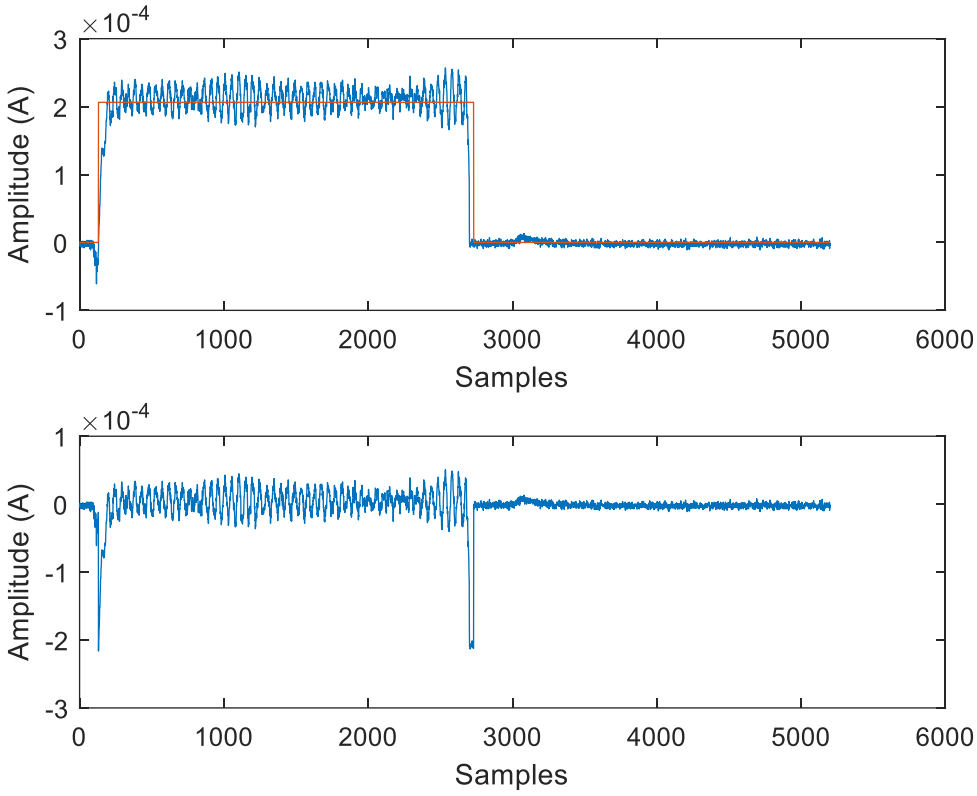
The device has been calibrated succesfully and calibration constants are saved to "calibration.txt". The results are summarized by the graphs below.

The results are summarized by the graphs below.

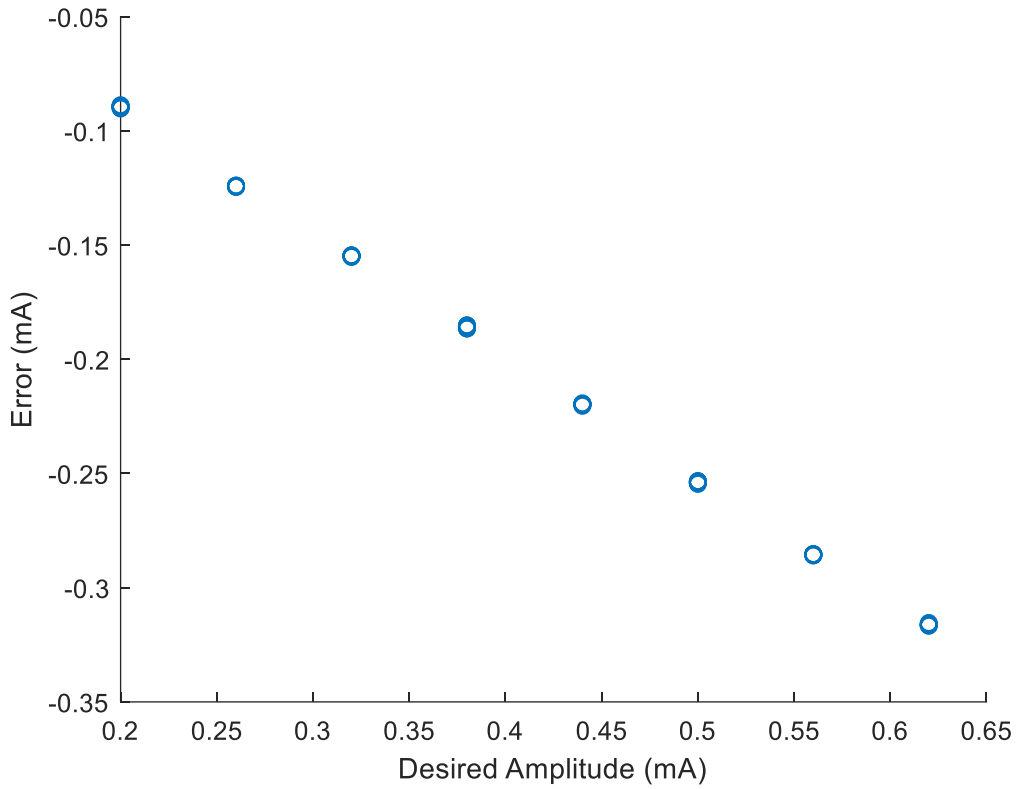
Error (uncalibrated):

mean bias	-0.20368 mA
standard deviation	0.074813 mA

Average waveform (top) and average glitches (bottom):



Error plot (uncalibrated):



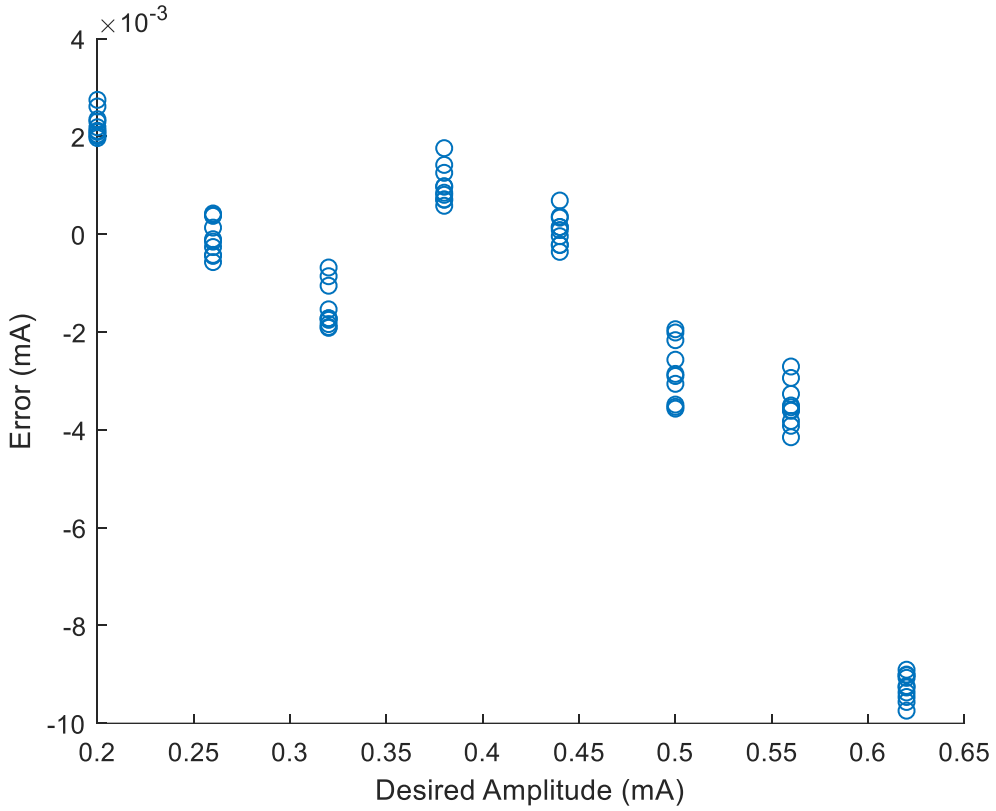
Calibration constants (y = intercept + slope * x):

intercept	0.018006
slope	0.45929

Error (calibrated):

mean bias	-0.0017241 mA
standard deviation	0.0034062 mA

Error plot (calibrated):



Trigger delay:

mean	3.48e-05 s
standard deviation	1.8886e-05 s
latest	5.22e-05 s

Conclusion

The device and software function correctly. The device has been calibrated succesfully.

Calibration performed by: .....

Signature: .....

Report: NociTRACK Calibration

Device: OctoStim_02_chan005

Time: 12-Feb-2020 16:49:26

Method

The device was calibrated using a Tektronix MSO 3014 Oscilloscope. The properties of the calibration stimuli are listed below.

Pulse-width: 0.52 ms

Range: 0.2 to 0.62 mA

Pulse increment: 0.06 mA

Pulsetrain repetitions: 10

Pulses generated: 80

Results

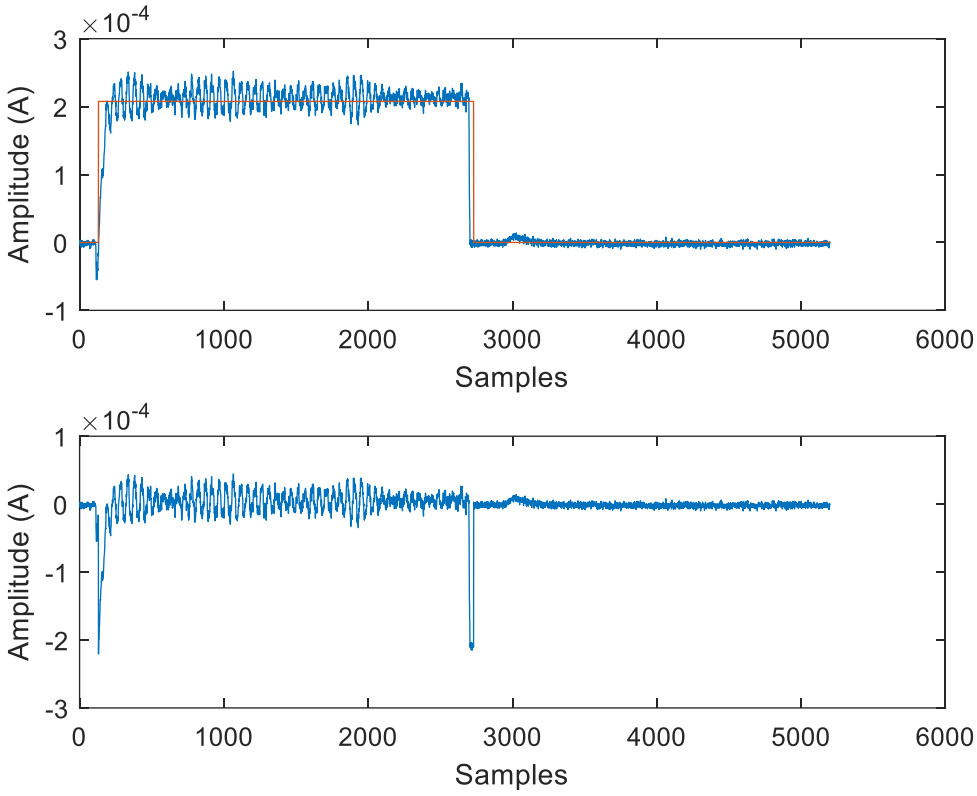
The device has been calibrated succesfully and calibration constants are saved to "calibration.txt". The results are summarized by the graphs below.

The results are summarized by the graphs below.

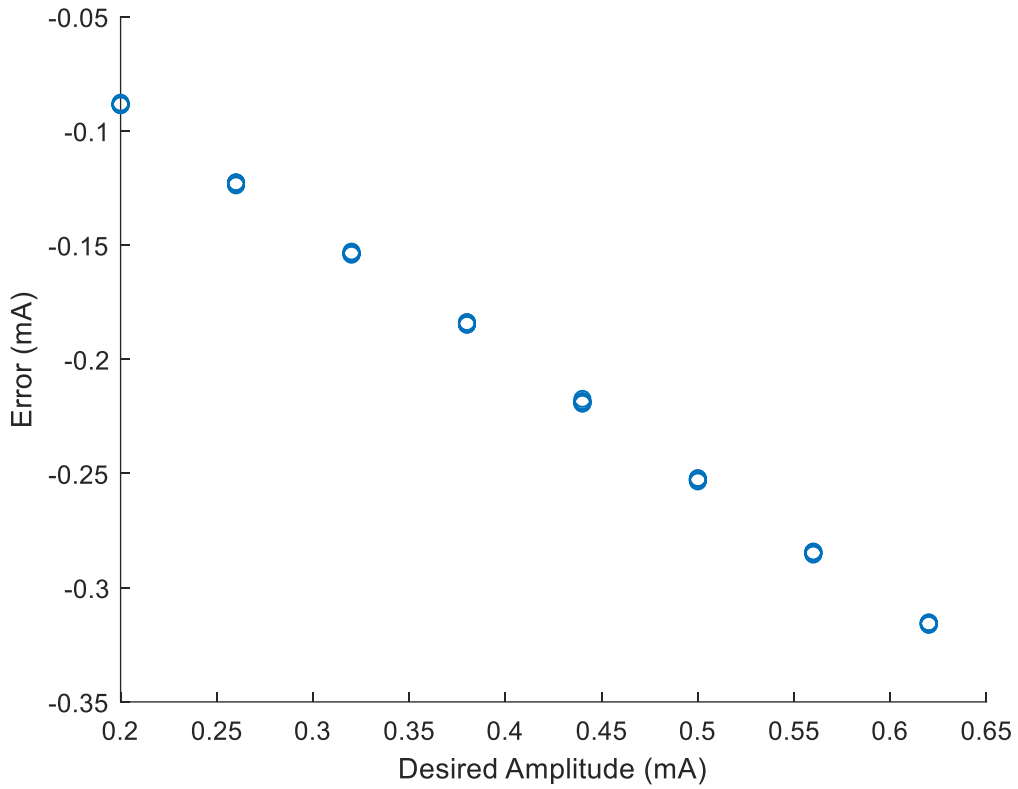
Error (uncalibrated):

mean bias	-0.20266 mA
standard deviation	0.074972 mA

Average waveform (top) and average glitches (bottom):



Error plot (uncalibrated):



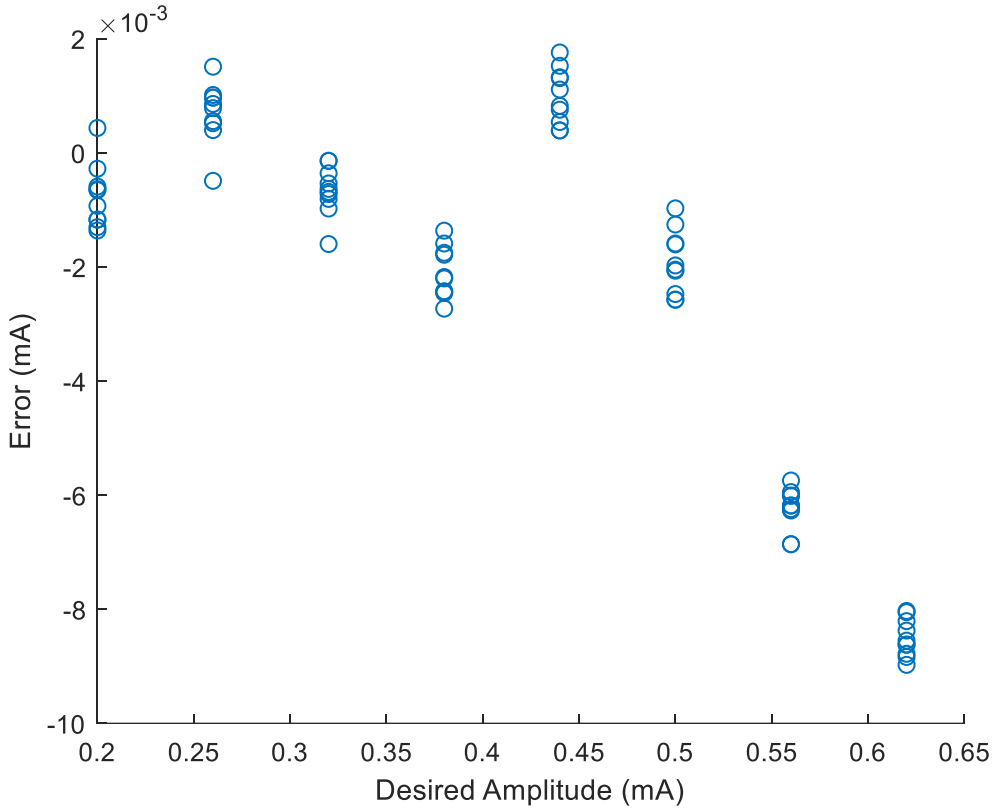
Calibration constants (y = intercept + slope * x):

intercept	0.019499
slope	0.45815

Error (calibrated):

mean bias	-0.0023097 mA
standard deviation	0.0031948 mA

Error plot (calibrated):



Trigger delay:

mean	3.6236e-05 s
standard deviation	1.9387e-05 s
latest	5.34e-05 s

Conclusion

The device and software function correctly. The device has been calibrated succesfully.

Calibration performed by: .....

Signature: .....

Report: NociTRACK Calibration

Device: OctoStim_02_chan006

Time: 12-Feb-2020 16:11:50

Method

The device was calibrated using a Tektronix MSO 3014 Oscilloscope. The properties of the calibration stimuli are listed below.

Pulse-width: 0.52 ms

Range: 0.2 to 0.62 mA

Pulse increment: 0.06 mA

Pulsetrain repetitions: 10

Pulses generated: 80

Results

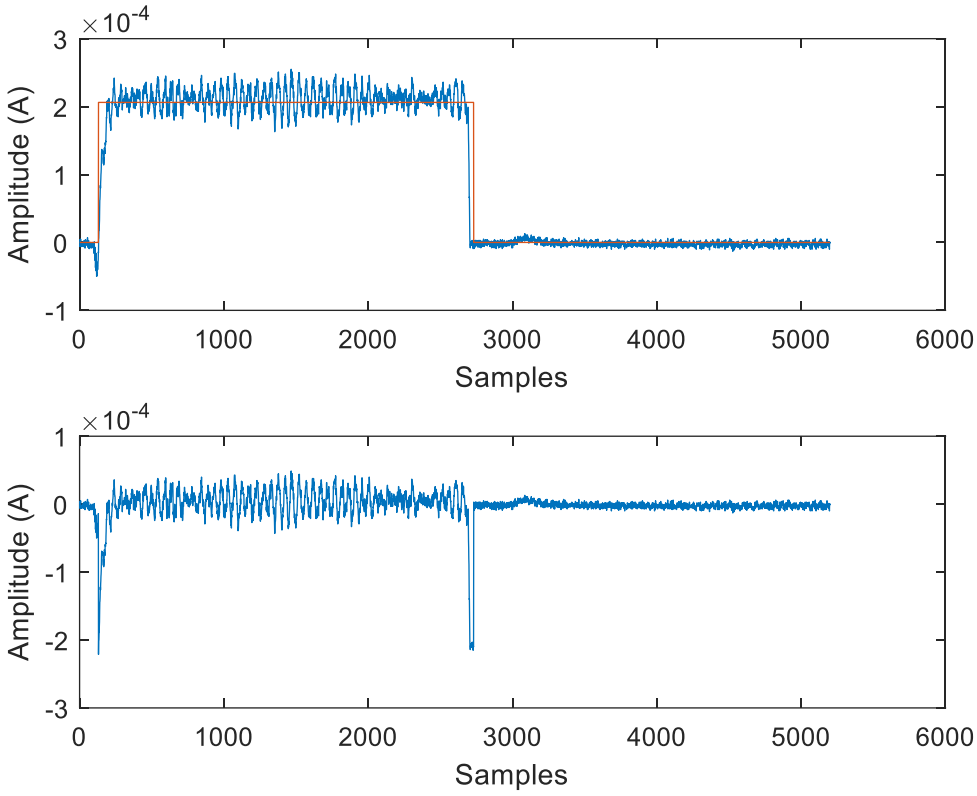
The device has been calibrated succesfully and calibration constants are saved to "calibration.txt". The results are summarized by the graphs below.

The results are summarized by the graphs below.

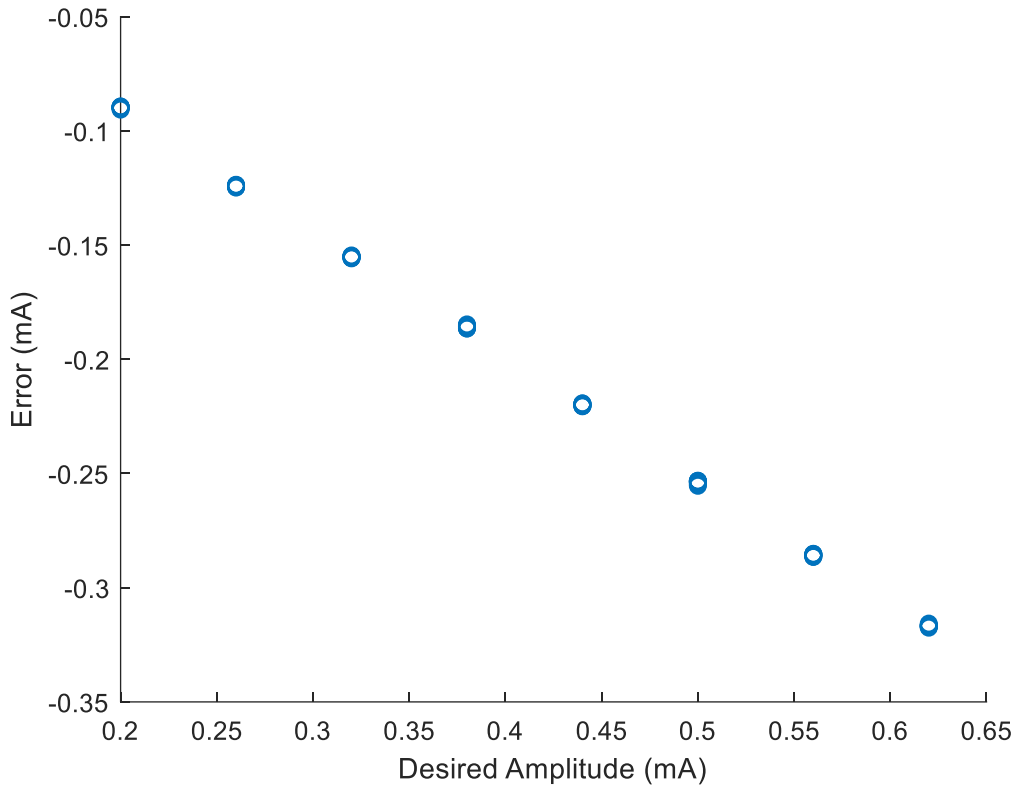
Error (uncalibrated):

mean bias	-0.2039 mA
standard deviation	0.074848 mA

Average waveform (top) and average glitches (bottom):



Error plot (uncalibrated):



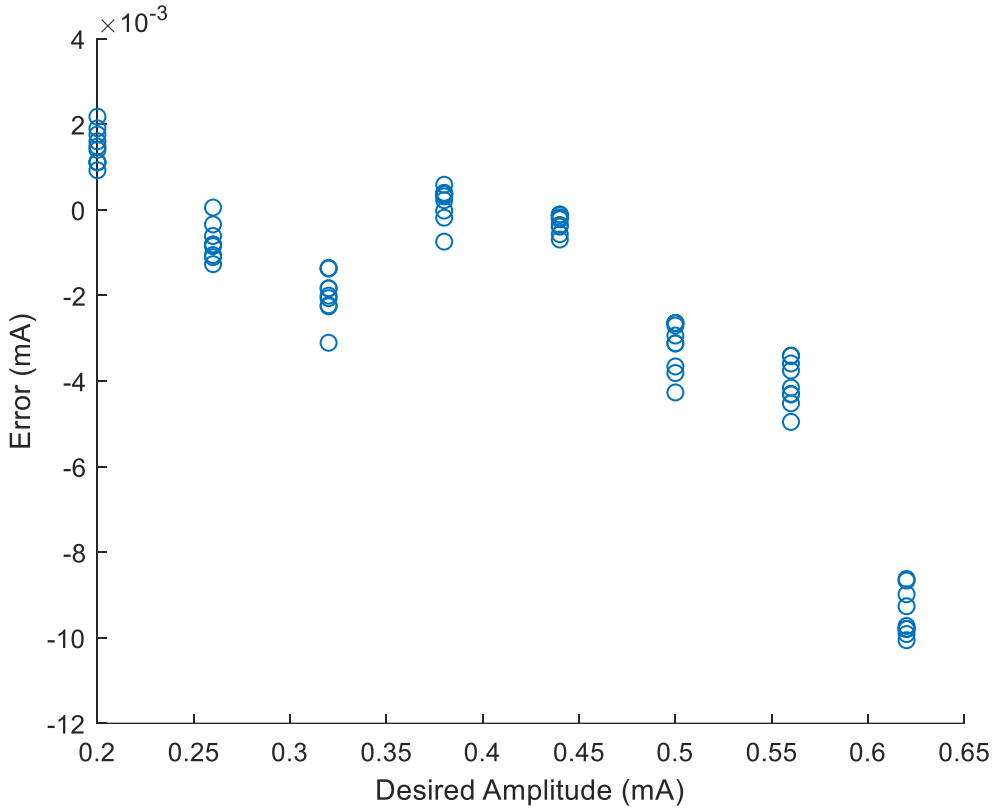
Calibration constants (y = intercept + slope * x):

intercept	0.017892
slope	0.45904

Error (calibrated):

mean bias	-0.0022474 mA
standard deviation	0.0032432 mA

Error plot (calibrated):



Trigger delay:

mean	3.7455e-05 s
standard deviation	1.9585e-05 s
latest	5.48e-05 s

Conclusion

The device and software function correctly. The device has been calibrated succesfully.

Calibration performed by: .....

Signature: .....

Report: NociTRACK Calibration

Device: OctoStim_02_chan07

Time: 12-Feb-2020 16:07:34

Method

The device was calibrated using a Tektronix MSO 3014 Oscilloscope. The properties of the calibration stimuli are listed below.

Pulse-width: 0.52 ms

Range: 0.2 to 0.62 mA

Pulse increment: 0.06 mA

Pulsetrain repetitions: 10

Pulses generated: 80

Results

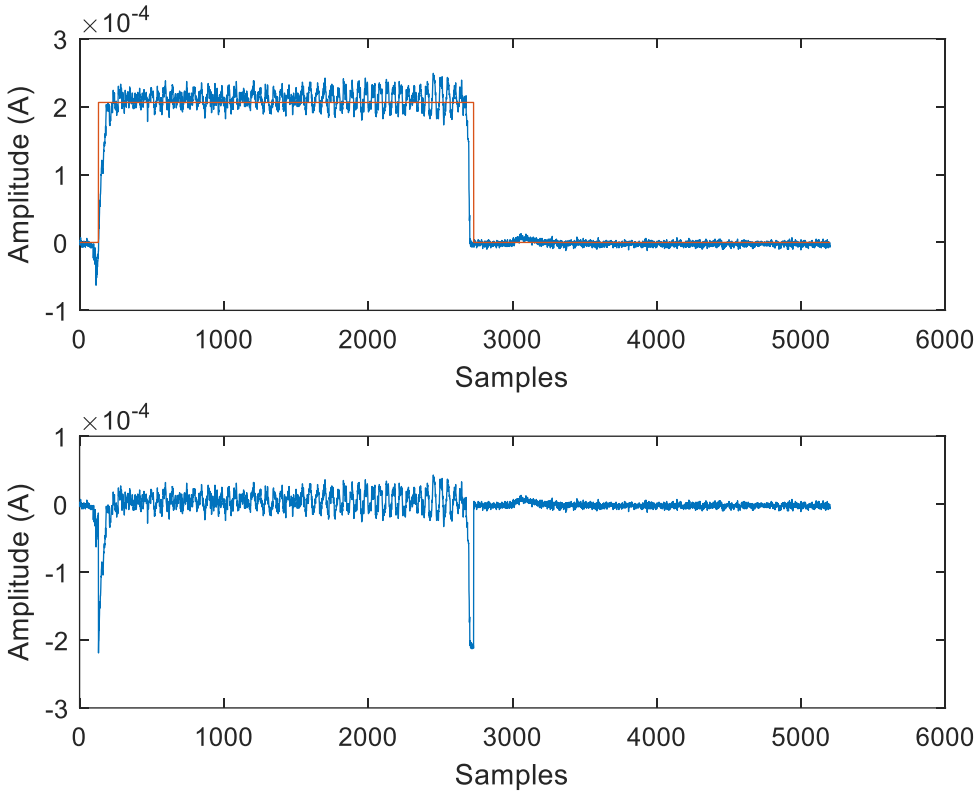
The device has been calibrated succesfully and calibration constants are saved to "calibration.txt". The results are summarized by the graphs below.

The results are summarized by the graphs below.

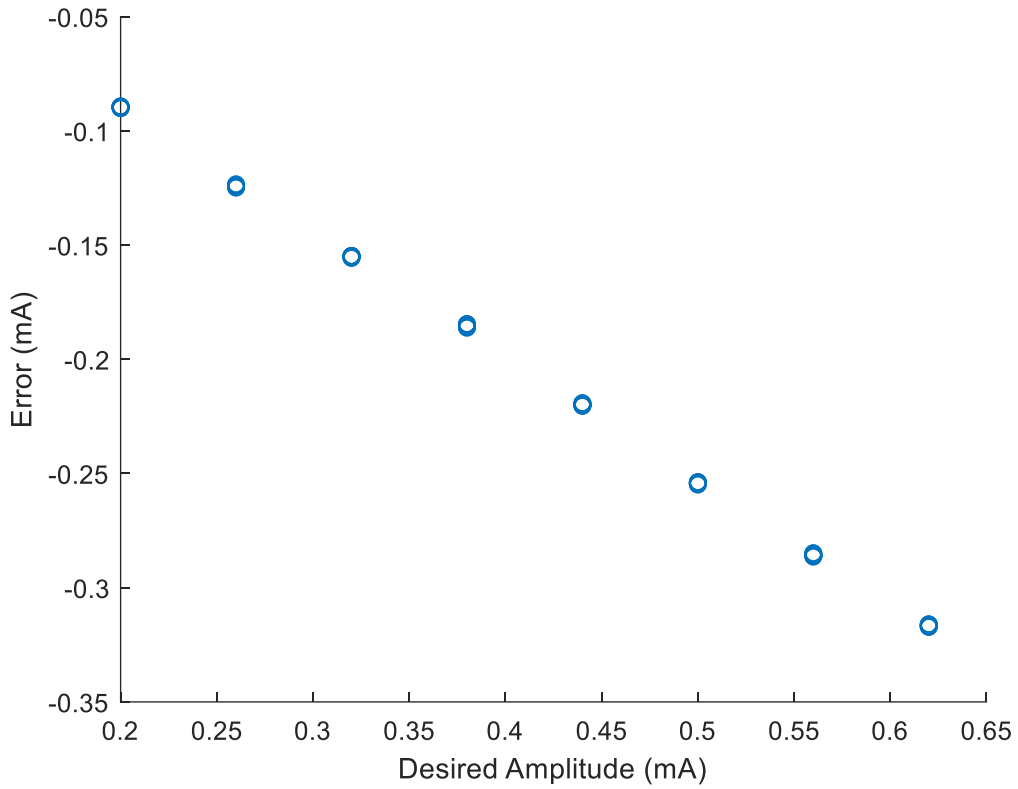
Error (uncalibrated):

mean bias	-0.2038 mA
standard deviation	0.074882 mA

Average waveform (top) and average glitches (bottom):



Error plot (uncalibrated):



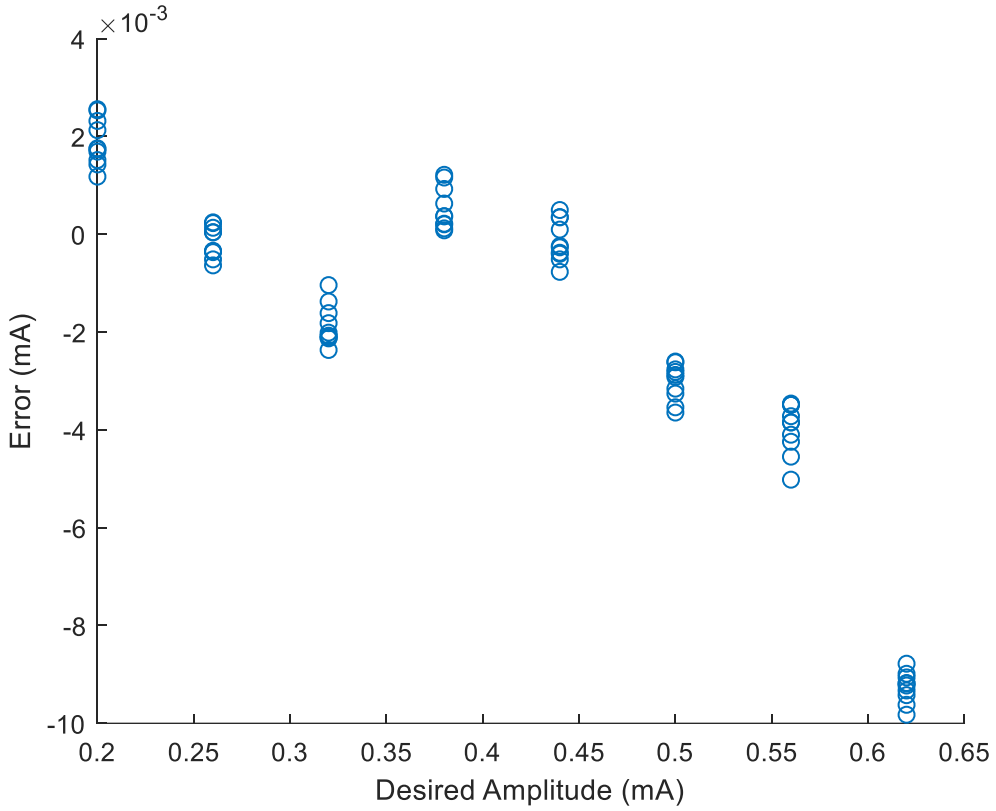
Calibration constants (y = intercept + slope * x):

intercept	0.018088
slope	0.45881

Error (calibrated):

mean bias	-0.0019986 mA
standard deviation	0.0033245 mA

Error plot (calibrated):



Trigger delay:

mean	3.54e-05 s
standard deviation	1.4818e-05 s
latest	5.36e-05 s

Conclusion

The device and software function correctly. The device has been calibrated succesfully.

Calibration performed by: .....

Signature: .....

Report: NociTRACK Calibration

Device: OctoStim_02_chan8

Time: 12-Feb-2020 15:21:02

Method

The device was calibrated using a Tektronix MSO 3014 Oscilloscope. The properties of the calibration stimuli are listed below.

Pulse-width: 0.52 ms

Range: 0.2 to 0.62 mA

Pulse increment: 0.06 mA

Pulsetrain repetitions: 10

Pulses generated: 80

Results

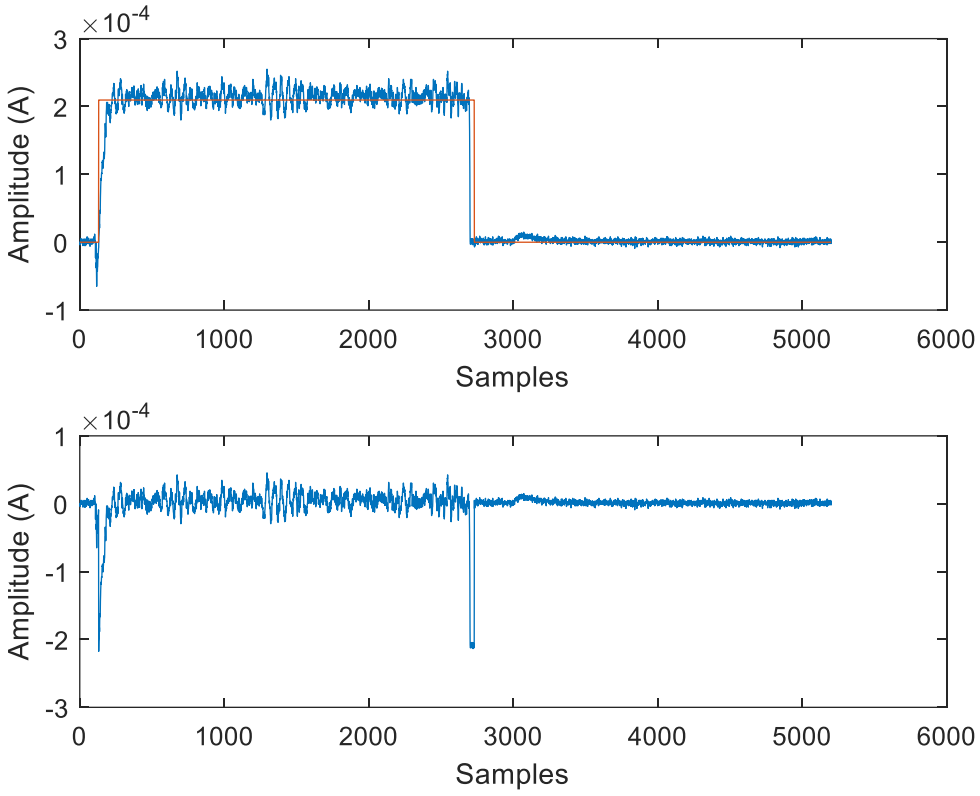
The device has been calibrated succesfully and calibration constants are saved to "calibration.txt". The results are summarized by the graphs below.

The results are summarized by the graphs below.

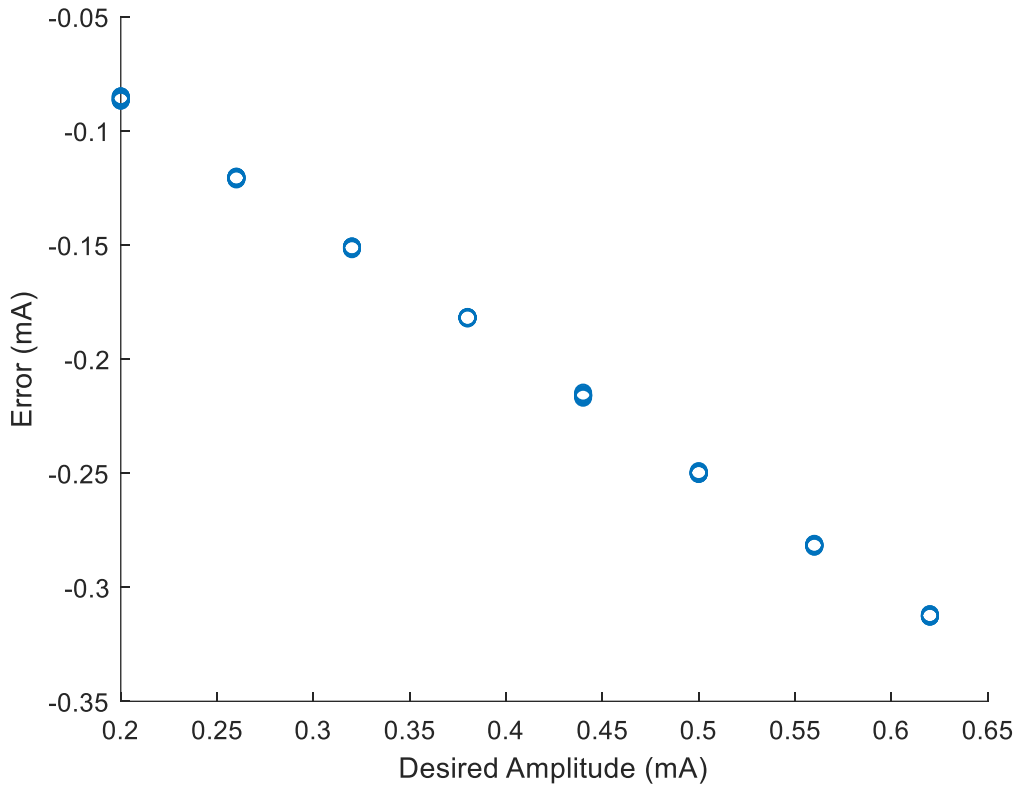
Error (uncalibrated):

mean bias	-0.19995 mA
standard deviation	0.074691 mA

Average waveform (top) and average glitches (bottom):



Error plot (uncalibrated):



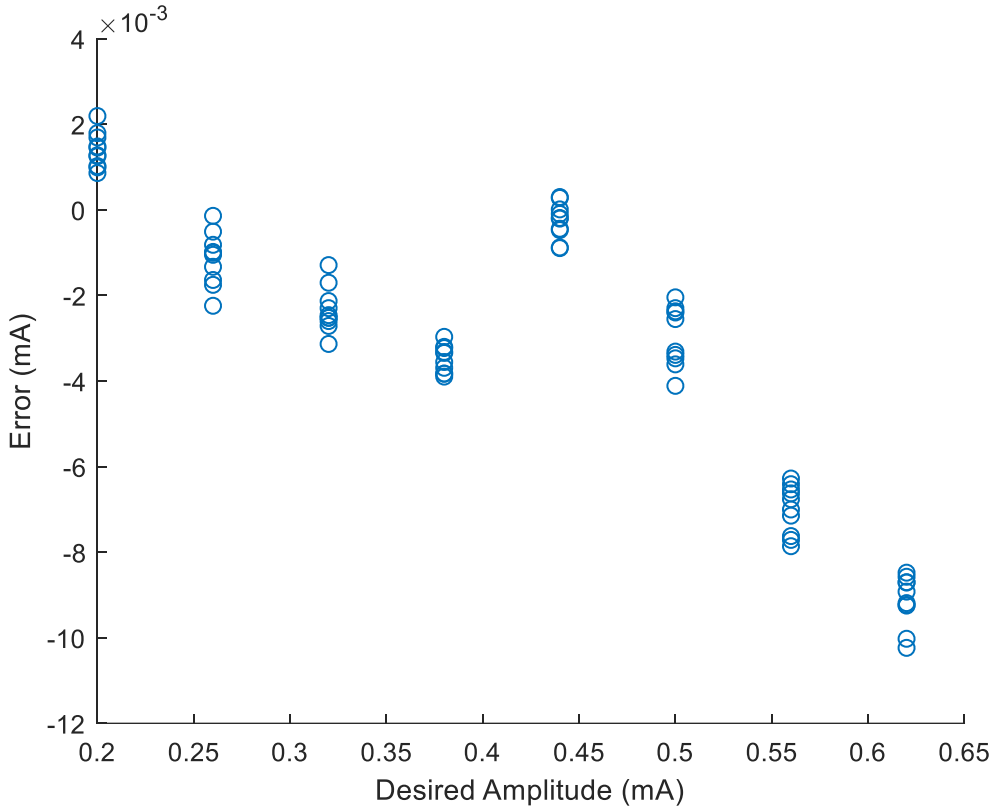
Calibration constants (y = intercept + slope * x):

intercept	0.021377
slope	0.46018

Error (calibrated):

mean bias	-0.0031151 mA
standard deviation	0.0033089 mA

Error plot (calibrated):



Trigger delay:

mean	3.7564e-05 s
standard deviation	1.7214e-05 s
latest	5.22e-05 s

Conclusion

The device and software function correctly. The device has been calibrated succesfully.

Calibration performed by: .....

Signature: .....