UNIVERSITY OF CALIFORNIA

Los Angeles

A Programmable Wireless Single Channel Neural Interface with Artifact Cancellation Capability

A dissertation submitted in partial satisfaction of the

requirements for the degree Master of Science

in Electrical and Computer Engineering

by

Yu Nong

2022

ABSTRACT OF THE THESIS

A Programmable Wireless Single Channel Neural Interface

with Artifact Cancellation Capability

by

Yu Nong

Master of Science in Electrical and Computer Engineering

University of California, Los Angeles, 2022

Professor Wentai Liu, Chair

In the past decades, the field of neural interface has gained significant amount of attention and advancement. However, some desirable powerful features have not been designed in the current neural interface devices because of technical challenges. First, most current neural interface devices use wire communication to transfer collected data and receive external commands. While wire communication provides better signal quality and data rate, wireless communication enables the free movement of experiment target in chronic studies. Secondly, stimulation artifact becomes a major roadblock for the development of bidirectional neural interfaces. This problem can be further divided into two parts, the saturation of amplifier caused by large-amplitude artifact and the distortion of neural signals. Thirdly, for rodent-based chronic studies, the size and weight become another layer of constraints given the size of experiment animal. Fourthly, various neuromodulation applications usually require different stimulation protocols, thus the stimulation waveforms generated by the interface must be versatile and programmable to support a broad set of applications – a one size fits all concept.

To address the aforementioned features, this thesis introduces a low-power mobile, lightweight, and wireless single channel stimulation and recording system with real-time artifact cancellation capability. The proposed system consists of a remote/implanted device, a wireless communication adapter, and a graphical user interface (GUI) on a laptop/desktop. User can execute commands (e.g., toggle stimulation/recording, adjust stimulation parameters, enable artifact cancellation) to the implanted/remote device through a GUI in real-time. Bluetooth Low Energy (BLE) is used as the wireless communication protocol between the implanted/remote device and the adapter. An adapter is used to accommodate the speed limitation of BLE on laptop/desktop. The size of the current prototype is approximately 8cm×3cm and has the potential to shrink down to 2cm×1.75cm. Real-time artifact cancellation capability is realized through the combination template subtraction followed by Hampel filter. The functionality and robustness of the system has been validated in both in-vitro and in-vivo experiments.

The dissertation of Yu Nong is approved.

Jonathan Chau-Yan Kao

Mani B. Srivastava

Wentai Liu, Committee Chair

University of California, Los Angeles

2022

Table of Contents

List of Tables

Acknowledgement

First, I would like to express my sincere thanks to my advisor Dr. Wentai Liu, for this precious opportunity working in Biomimetic Research Lab to prototype this single-channel stimulation and recording device. I also want to thank you for his guidance, support, affirmation, and patience for the past 1 year while working with you. More importantly, I want to thank you for introducing me into the field of neural engineering in which I found passion and providing the opportunity for me to continue learning after my graduation.

To Dr. Mani Srivastava, many thanks for your instructions and guidance that really open my limited horizon. Things I learned from you are for sure a valuable asset for your future work. Also, I want to thank you for your time, patience, help, and encouragement along the way.

To Dr. Jonathan Kao, I would like to thank he for the excellent courses where I picked up incredibly useful tools and knowledge, and for the patient and genuine guidance which helps me identify my career path. It is truly an honor to have you to be a part of my thesis committee.

To my fellow student Yanpeng Chen, it is a pleasure to collaborate with you on this project. Your guidance at the beginning of the project and your suggestion along way are crucial for this project. To all other fellow students in Biomimetic Research Lab, many thanks for welcoming me into this group, and for helping me in study and life.

Finally, to my parents and my wife, I couldn’t come this far without you all. Many thanks to your unreserved support and love.

My sincere appreciation for anyone who helped along the way.

This work is partially supported by NIH 1RF1NS118301-01 and NIH OT2 OD024899 (Subawards: WL, MM)

Chapter 1 Introduction

1.1. Introduction to Neural Interface

Neural interfaces are devices that interact with the nervous system in two ways [1]. In one way, information in the nervous system is encoded in the form of electrical signal, which is recorded by neural interface to recover useful information [1]. In the other way, information can be injected into the nervous system by eliciting electrical stimulation with special patterns from outside of neurons, which generates voltage gradients across membrane and subsequently triggers the response of the stimulated neurons [1]. While many neural interface devices are only capable of establishing one way communication reliably, a bidirectional neural interface can record neural signals and execute electrical stimulation to nervous system simultaneously. Such feature is ideal for medical applications to restore patient's functionality which is disabled by disease or injury. To restore the functionality through electrical stimulation which mimics the natural neuronal behaviors, precise modulation of stimulation pulses needs to be determined, ideally in an adaptive way in real-time based on the feedback of the recorded neural response. Such a stable close-loop mechanism in a dedicated integrated system is a basis for successful prosthetic devices [1].

1.2. Challenges for Bidirectional Neural Interface

Many challenges exist for high-performance bidirectional neural interface devices and the major ones may include stimulation artifact [1], bandwidth of communication protocol, limited computing power, efficacy of neuron recruitment.

While recording neuronal activity, electrodes pick up the undesired artifact waveform due to the applied stimulation and these artifacts are typically several order larger in amplitude than neuronal responses [1]. Traditional methods to deal with artifacts include analog filtering and blanking require relatively simple hardware [1]. However, limitations apply to analog filtering and blanking does not preserve neural signals during the blank period. Considering the drawbacks of conventional methods, advancement needs to be made to provide better signal quality without interrupting on-going stimulation.

The effectiveness of the close-loop control mechanism in bidirectional neural interfaces relies on processing recorded neural response in real-time. Methods to process neural responses to provide feedback control signal can be rough divided into 2 categories. For one, recorded neural responses are transmitted to back-end device where signal is processed to generate feedback control which gets sent back to the stimulator through either wire or wireless communications. Performing computation off the implanted neural interface provides sufficient computing resource to accommodate complicated inference algorithm. However, the convenience in wire communication and bandwidth of wireless communication protocol becomes the bottleneck. For another, processing recorded neural responses within the implanted neural interface device is highly desirable as it removes the need of an associated back-end device which enables the free movement of the experimental subjects or patients. The limitation of such paradigm lies in the trade-off between the computing power, memory space, and the size, power consumption of the whole implanted system.

Neural stimulation protocols have been actively developing over the years. For invasive neural interfaces, electrical signals are often modulated in shape, frequency, and amplitudes. In recent years, biomimetic stimulation protocol with random frequency components gains more attention due to the effectiveness in recruiting surrounding neurons [2]. To leverage such benefit, clever algorithms need to be developed to avoid the need for extra computational power and large storage space to generate and store the precise waveform.

1.3. Thesis Outline

The rest of this thesis is formatted in the following order. Chapter 2 provides the necessary background knowledge related to the development of the system. Chapter 3 to 6 discusses the development of a wireless single channel stimulation and recording system with real-time artifact cancellation capabilities. The discussion in these chapters includes the overview of system architecture, hardware, and printed circuit board (PCB) design, the firmware implementation, the design of graphical user interface, and algorithm to remove stimulation artifact. Chapter 7 discusses the in-vitro and in-vivo testing result of the prototype with the 2nd iteration PCB. Finally, Chapter 8 concludes the work, envisions and expansion of the current prototype to a close-loop neural interface where artifact cancellation is performed on chip, and forecasts potential challenges of building an adaptive close-loop neural interface with deployment of machine learning models on-chip based on this work.

Chapter 2 : Background Knowledge

2.1. Overview of CC13x2/CC26x2RSIP Microcontroller

CC13x2/CC26x2 family microcontrollers (MCU) are used as core controllers in the system. The hardware overview of CC1352R and CC2652RSIP are shown in Figure 2-1 and 2-2 below.

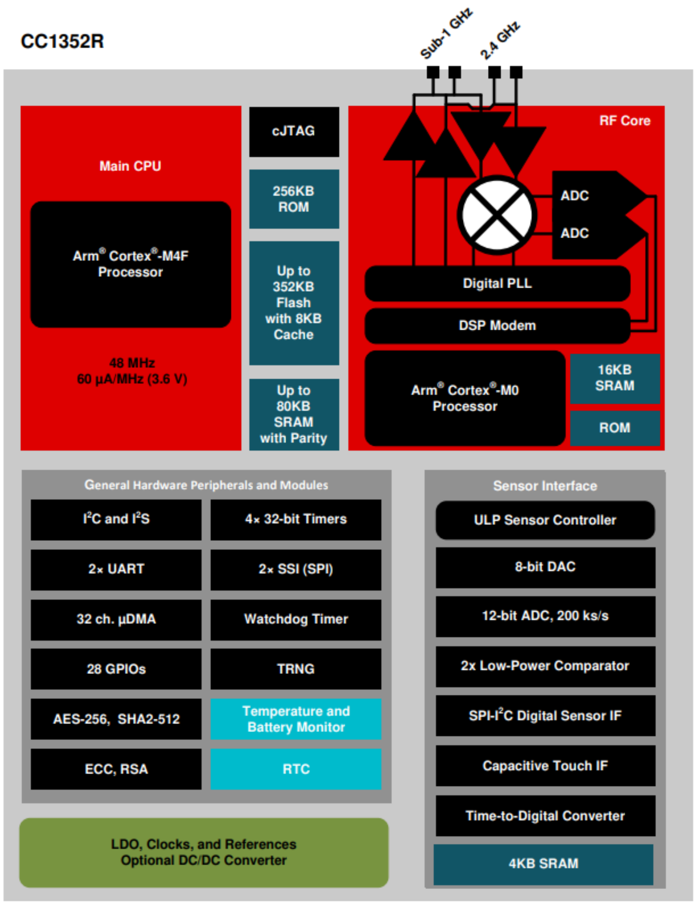


Figure 2-1: CC1352R Hardware Overview [3]

CC1352R and CC2652RSIP are low-power wireless MCUs with Bluetooth Low Energy (BLE) capability, on-chip support of SPI and UART. Both MCUs have a 48-MHz Arm Cortex-M4 processor in parallel with a 24-MHz autonomous ultra-low power sensor controller to interface analog-to-digital converter (ADC) without interrupting tasks in main CPU. Additionally, hardware accelerator in CC1352R/CC2652RSIP significantly reduces runtime of multiplication and division operation. Both MCUs have single-clock-cycle multiplication instruction and require only 2-to-12 clock cycles for division instruction. Compared to CC1352R, CC2652RSIP integrates the passive components and crystal oscillator in the package, which further reduces the overall system footprint and eases the development process. Provided features mentioned above, CC1352R and CC2652RSIP are great candidates for the core controllers for the proposed system.

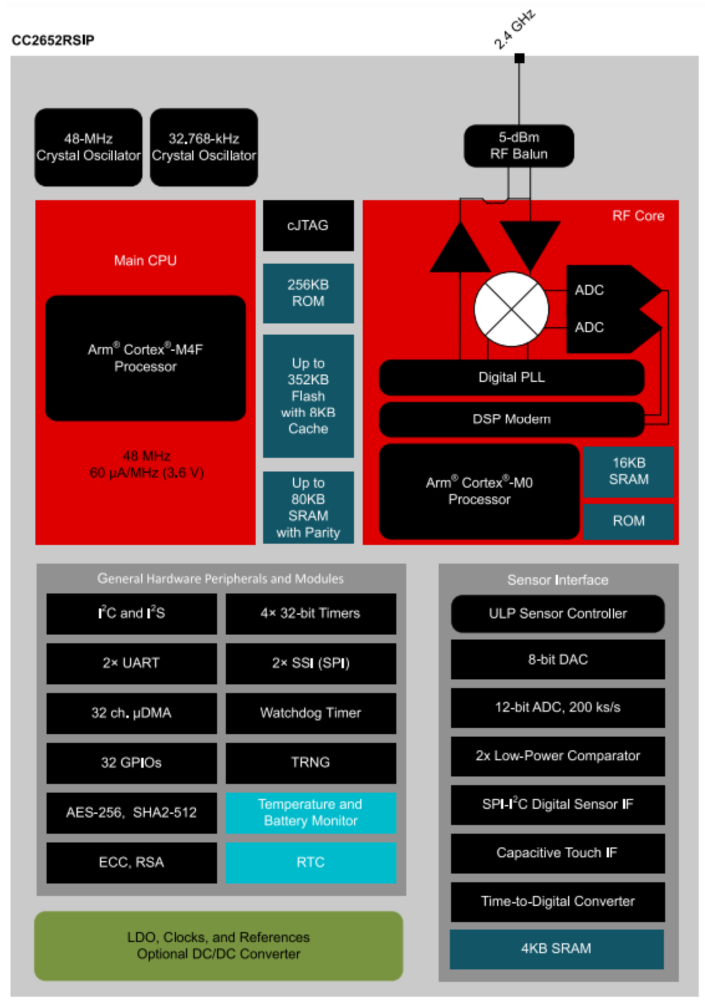


Figure 2-2: CC2652RSIP Hardware Overview [4]

2.2. Bluetooth Low Energy (BLE)

2.2.1. Overview of Bluetooth Low Energy (BLE)

Bluetooth Low Energy (BLE) is a low-power wireless communication protocol introduced in 2001. While sharing similar features with Bluetooth Classic, BLE consumes much less power compared with Bluetooth Classic and its data layer structure are more suitable for the purpose of our application [5]. With appropriate configurations, data transmission rate of BLE can go up to approximately 1250Kbit/s which is sufficient for single-channel neural recordings. The following discussion provides the prerequisites to understand the firmware design of the system.

2.2.2. BLE Physical Layer (PHY)

BLE uses 2.4GHz in ISM band which is the same frequency band used by Bluetooth Classic and WI-FI [5]. The advertising band starts from 2402MHz to 2480MHz [5]. The entire band is divided into 40 channels of 1MHz wide and separated by 2MHz. The arrangement of channels in the frequency is shown in Figure 2-3 [5].

Channels 37, 38, and 39 are reserved as advertising channel to exchange advertisement packet which contains device-specific information to establish connections between central and peripheral device. The placement of the advertising channels deliberately avoids the overlap with WI-FI bands and other sources [5]. The separation of 3 advertising channels avoids interferences from WI-FI, Bluetooth Classic, Microwaves, etc. BLE radio transmits under a modulation scheme with 4 options for Bluetooth 5.0: LE 1M, LE 2M, LE coded with S = 2, LE coded with S = 8. Speed-range tradeoff for different modulations is summarized in table 2-1 below. Notice that additional support is needed for Bluetooth 5.0 to operate under coded modulation.

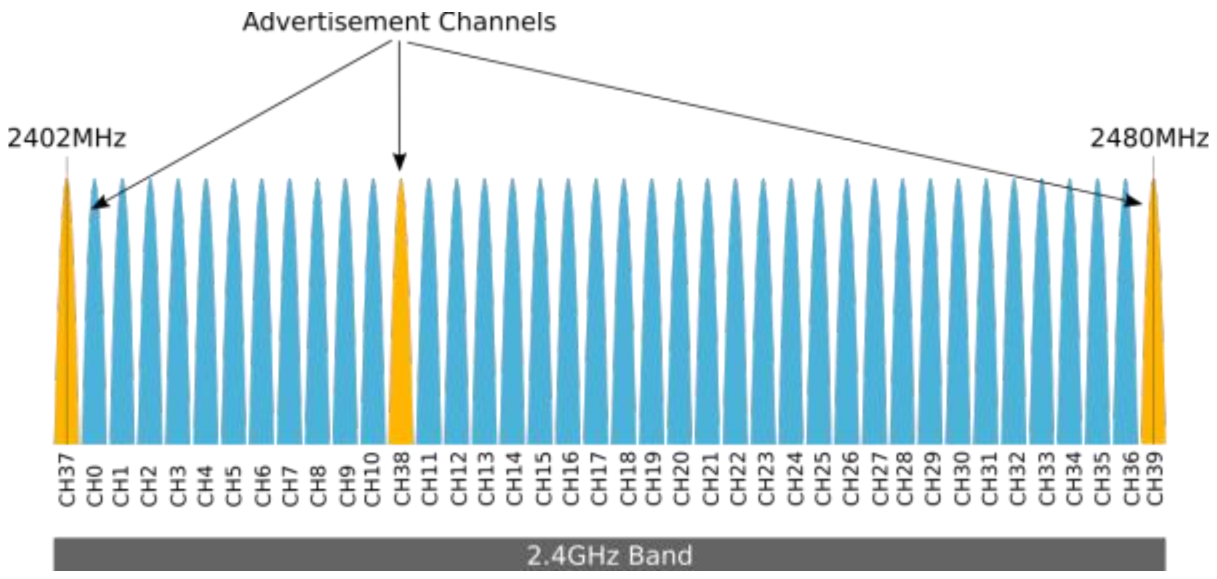


Figure 2-3: Frequency Band of Bluetooth Low Energy [5]

Table 2-1: BLE Modulation Modes [6]

	Theoretical Data Rate	**Range Multiplier (approx.)**	**Bluetooth 5.0 Support**
LE 1M	1 Mbit/s	1	Yes
LE 2M	2 Mbit/s	0.8	Yes
Coded S=2	500 Kbit/s	2	Optional
Coded S = 8	125 Kbit/s	4	Optional

2.2.3. Connection

Connection is established between the central and peripheral devices after the exchange of connectable advertisement packet. After the connection established, the central device is responsible for managing the connection and approving the requested connection parameters from the peripheral device. During a connection event, the central and peripheral devices send packets to each other in turns to exchange information until all data has been exchanged or the maximum connection interval has reached. A connection event will be terminated if the central device does receive a return packet from the peripheral [6]. The maximum connection interval can be requested by the peripheral ranging from 7.5ms to 4s [6].

2.2.4. Generic Attribute Profile (GATT)

Generic Attribute Profile (GATT) defines the data transfer procedures and formats in a structural way. The structure of GATT can be summarized in Figure 2-4 below. Service in BLE serves as a container of logically related data item. A BLE peripheral profile can have multiple services and one service contains zero or more characteristics [7]. Data item in service is called characteristic which contains multiple properties used as identification and definition of data format [7]. The description of each property in characteristic is listed as the following:

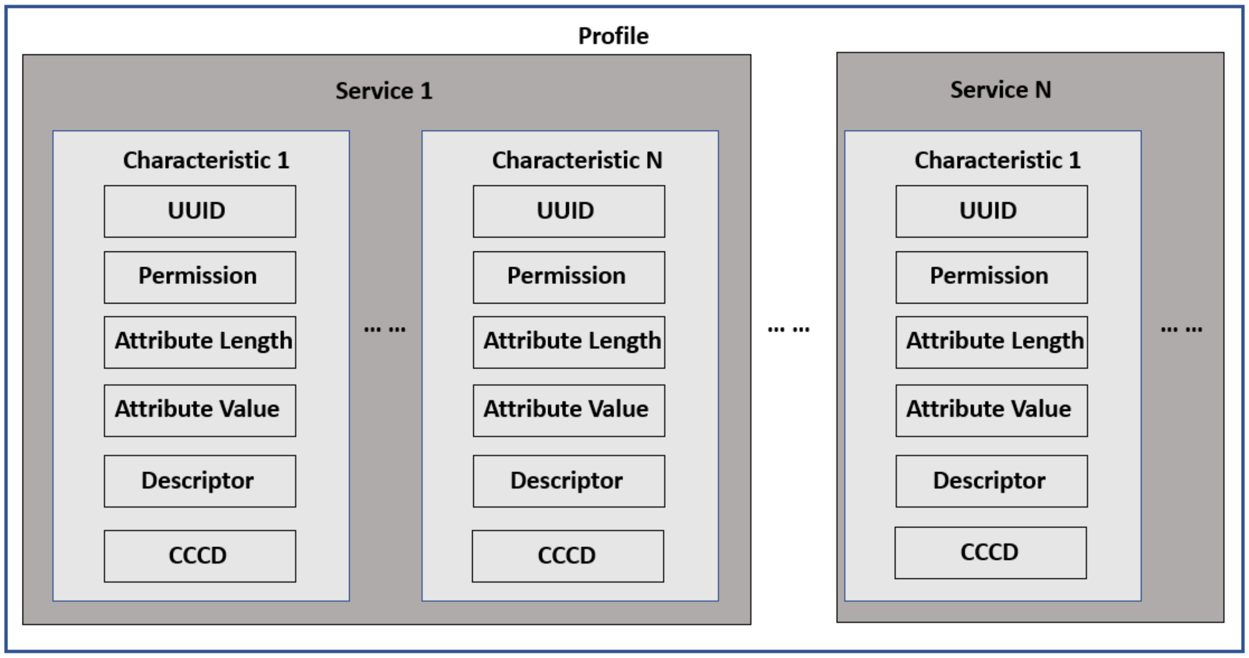


Figure 2-4: Structure of GATT

1. UUID: UUID is a 16-bit unique identifier for each attribute in GATT server, which makes the characteristic attribute addressable. In theory, UUID ranges from 0x0001 to 0xFFFF so that one GATT service can accommodate up to 65535 characteristics [7]. In practice, a service may contain up to tens of characteristics [7].
2. Permission: Access permission defines whether the characteristic attribute can be written or read (or both) by the central device. The permission field has 4 types of configurations [7].
    a. None: The attribute cannot be accessed.
    b. Readable: The attribute can be read by the central device.
    c. Writable: Central device can write to the attribute.
    d. Readable and Writable: The Attribute can be read and written by the central device.

   Additionally, peripheral can request the central device to provide authenticated key to access the specified attribute [7].
3. Attribute length: Determines the length of the attribute value. The maximum length allowed is 512 bytes [7].
4. Attribute value: Attribute value stores the actual content of the characteristic attribute. There is no data type restriction for attribute value.
5. Descriptor: This is a user-readable description of the characteristic in UTF-8 string. An example of descriptor would be “Room Temperature (°F)” [7].
6. CCCD: CCCD is the abbreviation of Client Characteristic Configuration Descriptor. CCCD grants the permission for a client to stream data to the central device rather than initiating read commands continuously from the central device [7]. CCCD allows two

styles of data streaming: notification and indication. Compared to notification, indication sends data continuously without requesting for a return packet from the central device to indicate a successful receipt.

2.3. Serial Peripheral Interface (SPI)

CC1352R/CC2652RSIP equips with serial peripheral interface (SPI) with dedicated SPI master up to 6MHz clock speed. The hardware SPI support facilitates the interface of external high-performance sigma-delta ADC. The following discussion covers the background of SPI communication protocol.

SPI is a synchronous, full deplex main-subnode-based (or master-slave-based) interface [8]. The protocol is available in 3-wire and 4-wire modes. The following discussion will be based on the popular 4-wire format. The interface between main and subnode is shown in Figure 2-5.

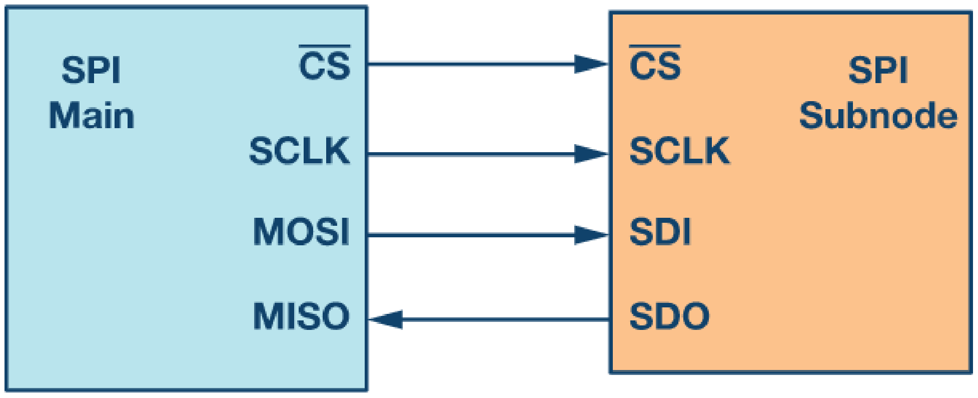


Figure 2-5: SPI Configuration with Main/Master and a Subnode/Slave [8]

There are 4 signal lines between the main and subnode device and the functionality of each is described below:

- CS: Chip-select pin selects subnode to communication. Typically, chip-select is an active-low signal.
- SCLK: Serial clock is the clock signal provided by the SPI main device to synchronize the two-way communication.
- MOSI: Master-out-slave-in is abbreviated as MOSI, meaning the data is sent from the master/main to the slave/subnode. The corresponding pin on the slave/subnode is called SDI (serial-data-in).
- MISO: Master-in-slave-out is abbreviated as MISO, meaning the data is sent from the slave/subnode to master/main. The corresponding pin on the slave/subnode is called SDO (serial-data-out).

Figure 2-6 demonstrates one communication frame of 8-bit SPI protocol. The communication begins when the chip-select pin is pulled low, shown in the green dashed line. Main/Master device determines whether to sample/shift data at rising or falling edge [8]. Figure 2-6 shows an example of sampling at rising edge and shifting at falling edge.

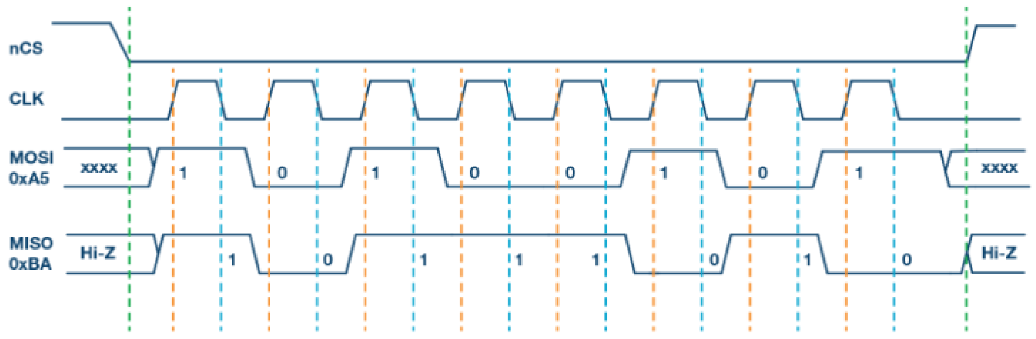


Figure 2-6: One Frame of 8-bit SPI with Data Sampled at Rising Edge and Shifted at Falling Edge [8]

The sampling edge is indicated by the orange dashed line, meaning the logical value in MOSI line at rising edge of SCLK is read by the subnode [8]. The shifting edge is indicated by the blue dashed line [8], meaning the logical value in MISO line at falling edge of SCLK is read by the main. Additionally, one main device can interface with multiple subnodes in the configuration shown in Figure 2-7 below. In this scenario, an individual chip-select pin from the main is needed to interface each subnode. The main device can only communicate to one subnode at a time.

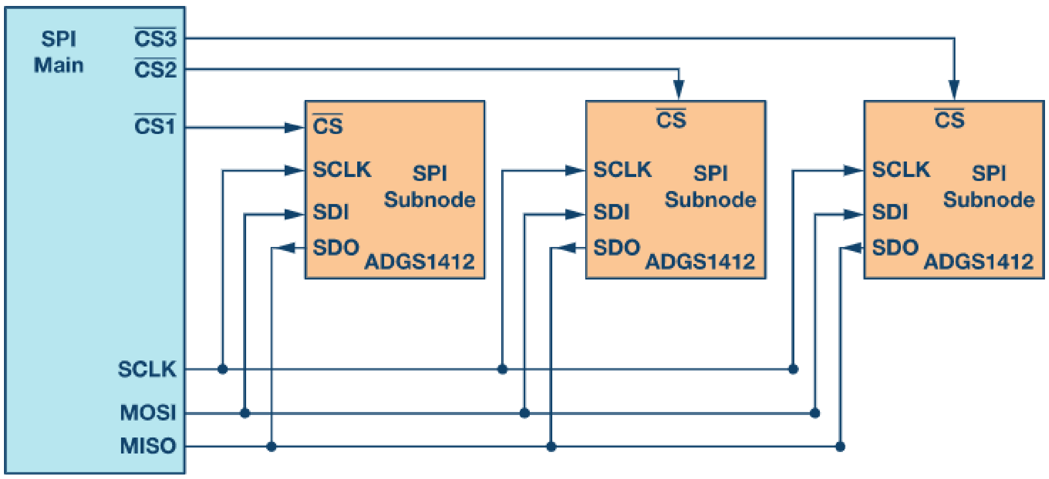


Figure 2-7: Multi-subnode SPI Configuration [8]

2.4. Universal Asynchronous Receiver/Transmitter (UART)

Universal Asynchronous Receiver/Transmitter, or UART, is a dedicated hardware for serial communication, which requires only two wires between devices as shown in Figure 2-8. TX pin refers to the transmitter pin and RX pin refers to the receiver pin. UART can be configured in simplex, half-duplex, or full-duplex modes based on applications [9]. Data is transmitted as frames in UART, and the rest of the discussion will be focused on the formatting content of UART frames.

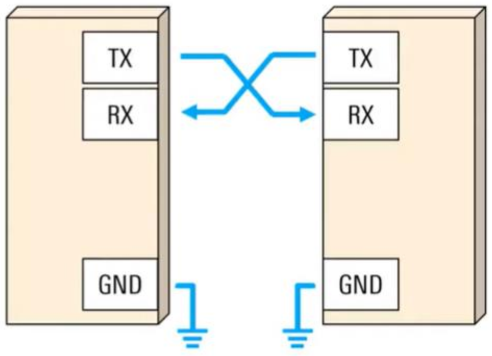


Figure 2-8: TX and RX Wires between Devices in UART [9]

Figure 2-9 illustrates a typical UART frame. As an asynchronous communication protocol, UART does not have a clock signal to synchronize the transmitter and the receiver. Therefore, the transmitter and the receiver must be configured to send data at the same speed.

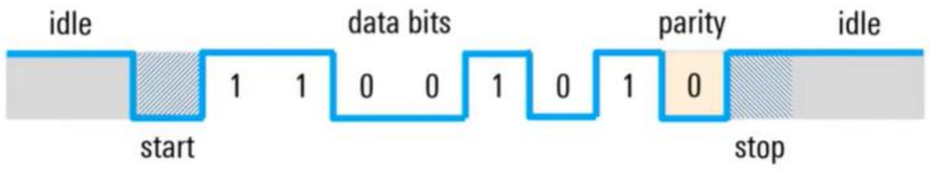


Figure 2-9: A Typical UART Frame [9]

A UART frame consists of start/stop bits, data bits, followed by an optional parity bit [9]. During the idle state, the TX/RX line is pulled high. The start bit is a transitional bit from the idle high to low to signal the start of a new frame. Likewise, a stop bit is a transitional bit back to the idle state by holding the TX/RX line high. Note that data bits in the illustration consist of 8 bits in total. In UART protocol, data bits can have from 5 to 9 bits. The optional parity bit is set based on the parity in use to error detection. For even parity, the parity bit is set so that the number of 1's in the frame is even [9]. For odd parity, the parity bit is set so that the number of 1's in the frame is odd [9].

2.5. Stimulation Artifact

2.5.1. The Stimulation Artifact Waveform

To aid the discussion of the artifact cancellation algorithm in Chapter 6, it is crucial to understand the distorted stimulation artifact waveform due to the tissue-electrode interface. When electrical current is injected into tissues, chemical changes happen in the chemical environment at the tissue-electrode interface by primarily two mechanisms [1]. First, a redistribution of ions in the chemical environment occurs to supply the current flow [1]. This is process is called non-Faradaic reaction. Secondly, electron flow between the electrode and the electrolyte to create current flow [1], which is Faradaic reaction. The combination of two types of reactions can be simply modeled by a Randall Cell model, Shown in Figure 2-10.

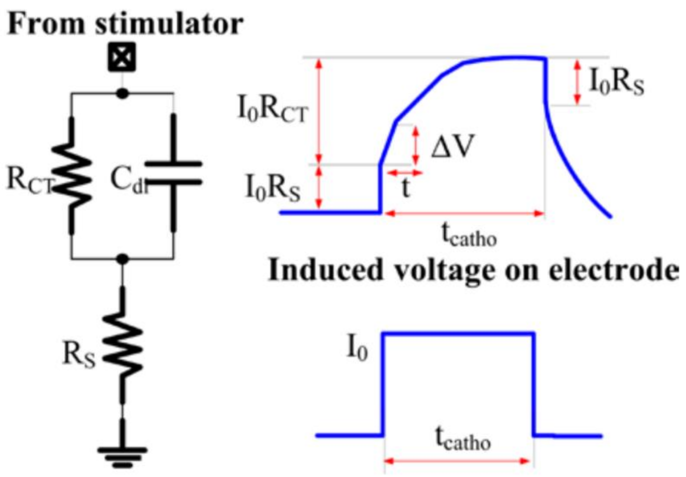


Figure 2-10: Randall Cell Model with the Induced Voltage on Electrode [10]

The non-Faradaic charge redistribution may be modeled as a simple electrical double-layer capacitor, C_{dl} [1]. The Faradaic processes can be modeled by a Faradaic impedance, approximated by a resistor R_{ct} to model the charge transfer [1]. Additionally, R_s is used to model the current flow in electrolyte medium [1]. When constant current stimulation is delivered by a square-wave pulse, non-Faradaic reaction causes the charging process in the induced voltage waveform. The time-domain waveform can be modeled by the following equation.

$$V(t) = \left[I_0 * R_{ct}\left(1 - e^{\frac{-t}{Rct*Cdl}}\right) + I_0 * R_s\right] * u(t) \text{ [10]}$$

Based on the discussions above, the scenario of a biphasic pulse train injected into tissues can be illustrated in Figure 2-11. The original square-wave pulses injected into tissue are distorted due to Faradaic and non-Faradaic process, attenuated over some distance, and collected by the recording electrode.

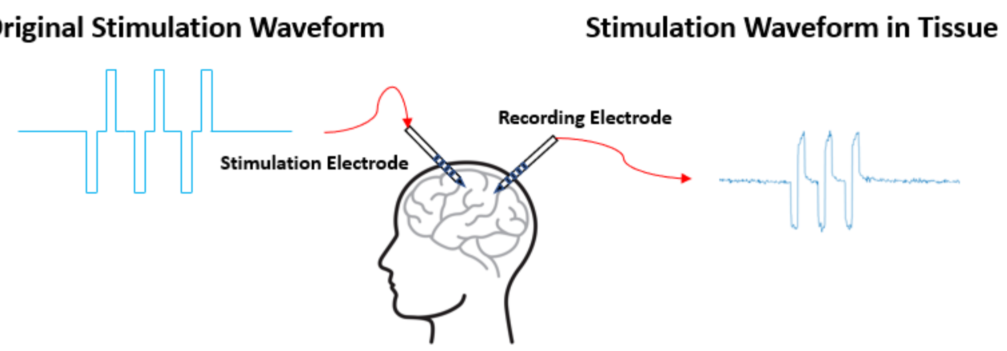


Figure 2-11: Scenario of Stimulation Artifact Collected by Recording Electrode

2.5.2. Review of Existing Artifact Cancellation Method

While recording neuronal activity, electrodes pick up the undesired artifact waveform due to the applied stimulation and these artifacts are typically several order larger in amplitude than neuronal responses, which dramatically disturbs the collect of neural signal due to potential amplifier saturation and distortion in the neural signals [1]. Different techniques used to eliminate stimulation artifact from the recording signals have been long investigated and could in roughly divided into 2 categories: front-end and backend methods.

2.5.2.1. Front-end Methods

Traditional analog filtering is typically used in scenarios of high-frequency stimulation and low-frequency neural signals. However, neural signals and stimulation artifacts are almost guaranteed to overlap in frequency spectrum. Therefore, the actual neural signals could be distorted, and the artifacts may not be suppressed sufficiently [1]. Moreover, for aperiodic and biomimetic stimulations of which the power spectral density is more spread-out, frequency-based filtering technique simply becomes impractical.

Blanking is commonly used and relatively mature technique. In this method, the amplifier is disconnected from the input signal when the stimulus is applied, avoiding the amplifier being saturated due to the high-amplitude stimulus artifact [1]. Then, the amplifier is turned back on when the recording site discharges to a certain point where the voltage amplitude would not cause the amplifier to saturate, or the amplitude of the remained voltage is comparable to the signal of interest. Blanking method is easy to apply and does not require significant amount of hardware and power. However, all signals including neural signal of interest and artifacts are completely lost in the duration that the amplifier is turned off. And there's no remedy to the loss of information.

Frequency shaping circuit can be used to adjust the gain of the amplifier so that the gain becomes smaller for lower frequency signals [2]. Given such conditions, if stimulation parameters are carefully chosen such that the stimulation pulse consists of mainly low frequency components, one can prevent the amplifier from saturation and achieve continuous recording. However, the resulting drawback of this approach is that the choices of stimulation parameters become very limited.

An approach used in commercialized device by Medtronic is to place two recording electrodes around a stimulating electrode symmetrically, shown in Figure 2-12 [1][11]. Since two recording electrodes are equally distant to the stimulating electrode, the amount of stimulus artifact experienced by two recording electrodes are the same in ideal situation [1][11]. On the contrary, the recording electrodes are not symmetrical with respect to the surrounding neurons. Thus, the neural responses recorded by two recording electrodes are not identical. Therefore, treating one of the symmetric electrodes as reference electrode can remove the stimulus artifact while preserving neural response. In practice, perfect symmetry is impossible. Therefore, significant amount of filter is required to remove the residual stimulus artifact, which causes neural responses with overlapping frequency spectrum to be removed or distorted [1]. With all these limiting factors, this method is useful in some specific applications but does not generalize well.

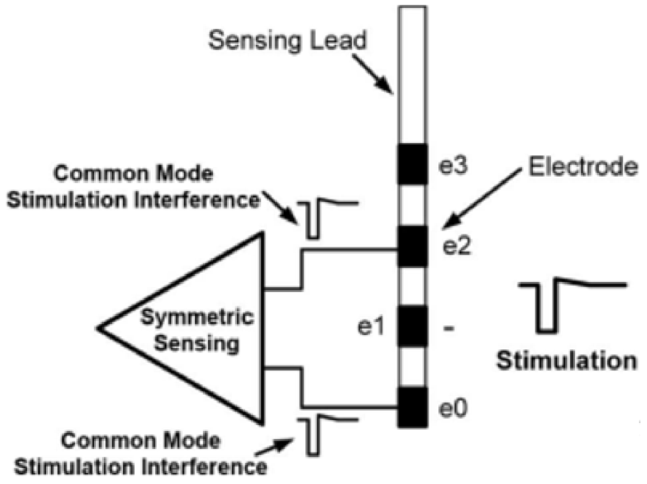

Sensing Lead
Common Mode Stimulation Interference
e3
Electrode
e2
Symmetric Sensing
e1
-
Stimulation
e0
Common Mode Stimulation Interference

Figure 2-12: Design of Symmetrical Sensing by Medtronic. Inc [11]

Except the ones mentioned above, many other front-end methods exist. Several designs introduce a hard reset mechanism to reset the amplifier so that it can quickly recover from saturation [12][13][14][15]. Template subtraction in hardware scheme to remove stimulation artifact is proposed in several works [16][17][18]. While these works seem to be promising, the results suffer from relatively low signal-to-noise ratio (SNR) [12]. [19] reports a clever way to avoid amplifier saturation by subtracting a fixed voltage level in front-end so that the resulting voltage is within the range of the amplifier. However, while the amplifier subtraction is avoided successfully, the remaining artifact is relatively large in amplitude and contaminates the meaningful neural signals. Front-end circuit methods to cancel stimulation artifact mentioned above suffer from the problem that they are not generalized well to all applications [1]. Additionally, many of the existing proposed front-end methods lack verifications by in-vitro and in-vivo experiments.

2.5.2.2. Back-end Methods

Back-end methods targets the recorded digitized signals and aims to extract useful information buried in large-amplitude stimulus artifact. Therefore, recorded data used in back-end methods is required to be saturation free. Back-end methods can be further divided into 3 categories: data reconstruction, template subtraction, component decomposition.

Data reconstruction removes data points contaminated by stimulus artifact and replace them by interpolated or approximated values [11]. A naïve method simply samples and hold the last artifact free sample before the stimulation starts to execute until the amplitude of the artifact becomes lower than certain threshold. Such method is simple to apply and only consumes a small amount of computing resource, which comes with the cost of significant distortion. To reduce the amount of distortion, linear interpolation, cubic spline interpolation, and Gaussian estimation can be applied to data points between the start and the end of the artifact [11]. The complexity of the application varies based on the techniques used to generate values within the window of stimulus artifact. In general, data reconstruction is easy to apply but not as reliable when the signal of interest is high frequency [11]. Therefore, interpolation is more suitable for LPF and ECoG recordings [11]. Additionally, action potentials within the stimulus window tends to be discarded so that not much information with in the artifact window can be revealed.

Template subtraction was initially proposed as a post processing method and require larger dynamic range to accommodate large stimulus artifacts without losing the underlying neural activities [12]. Template used for subtraction may be formed from averaging the multiple periods of artifact or fitting critical points to a predefined function [12]. However, both ways suffer from under-sampling and tiny shifting in stimulus artifact due to timing inaccuracy. More sophisticated methods to align the stimulus artifact waveform have been investigated. For example, one can up-sample, shift, and down-sample the artifacts from different periods so that the critical points of each contributing component align together before averaging [12]. However, algorithm for such method is rather demanding, making it different for real-time applications. Additionally, adaptive filtering can be applied to recordings from the adjacent electrodes to form the subtraction template [12]. All methods mentioned in the category of template subtraction require some time for the template to converge and will need to update throughout the recording duration. Therefore, artifact cancellation quality in this scheme is time varying [12].

Component decomposition decomposes the input waveform into multiple components and reconstruct the waveform without components that contributing to the artifact [12]. Ensemble empirical mode decomposition (EMD) and independent component decomposition (ICD) are commonly used and offer great accuracy [12]. However, both methods are computation demanding and difficult to deploy to real-time application in firmware.

Chapter 3 : Overview of the System

3.1. System Architecture

The development of the system is a collaboration between Yu Nong and Yanpeng Chen. The design of single-channel stimulation and recording system with real-time artifact cancellation capability consists of 3 major components: an implanted/remote device, a communication adapter, and a backend device (usually laptop/computer or smart phone). Figure 3-1 shows a block diagram representation of the system structure.

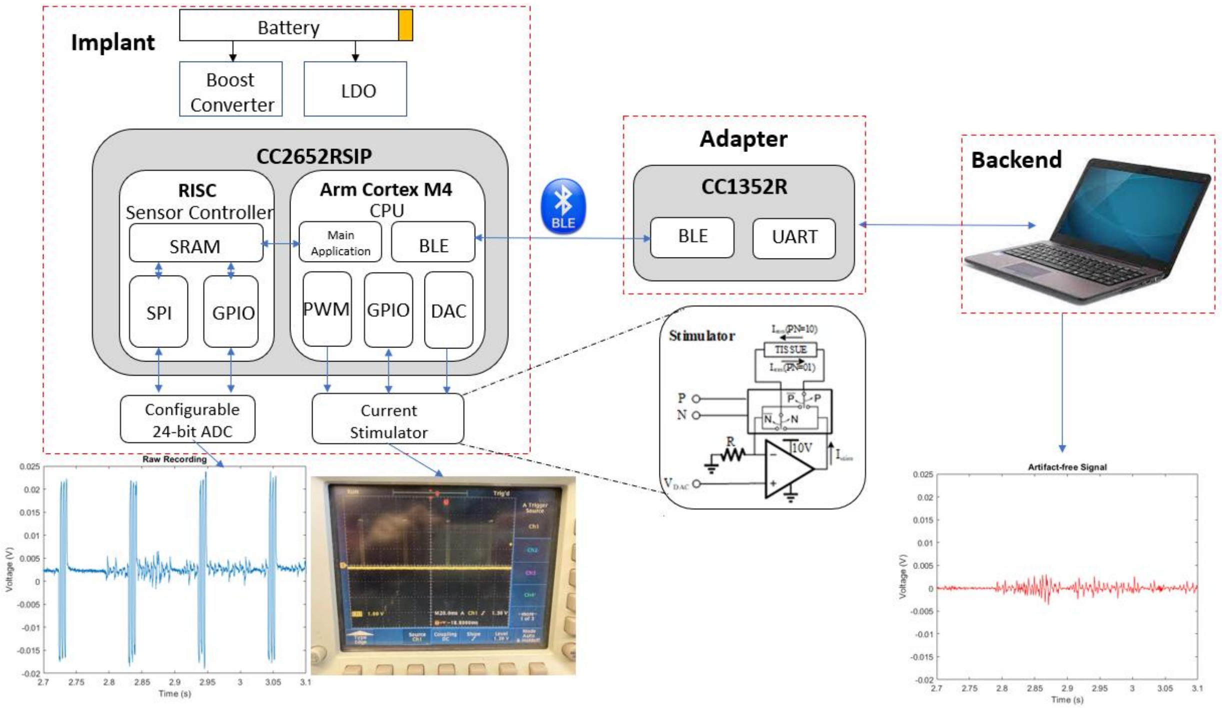


Figure 3-1: System Architecture Diagram

CC2652RSIP and CC1352R are used as the core controller for the implant and adapter, respectively. The implant/remote module and adapter exchange information through Bluetooth Low Energy (BLE) to offload recorded neural signal and adjust stimulation parameters. The adapter serves as a bridge between the implant and the backend device to improve the compatibility of the system and resolve the limitation of BLE streaming speed with laptop/computer. A typical experimental setup of the system is shown in Figure 3-2

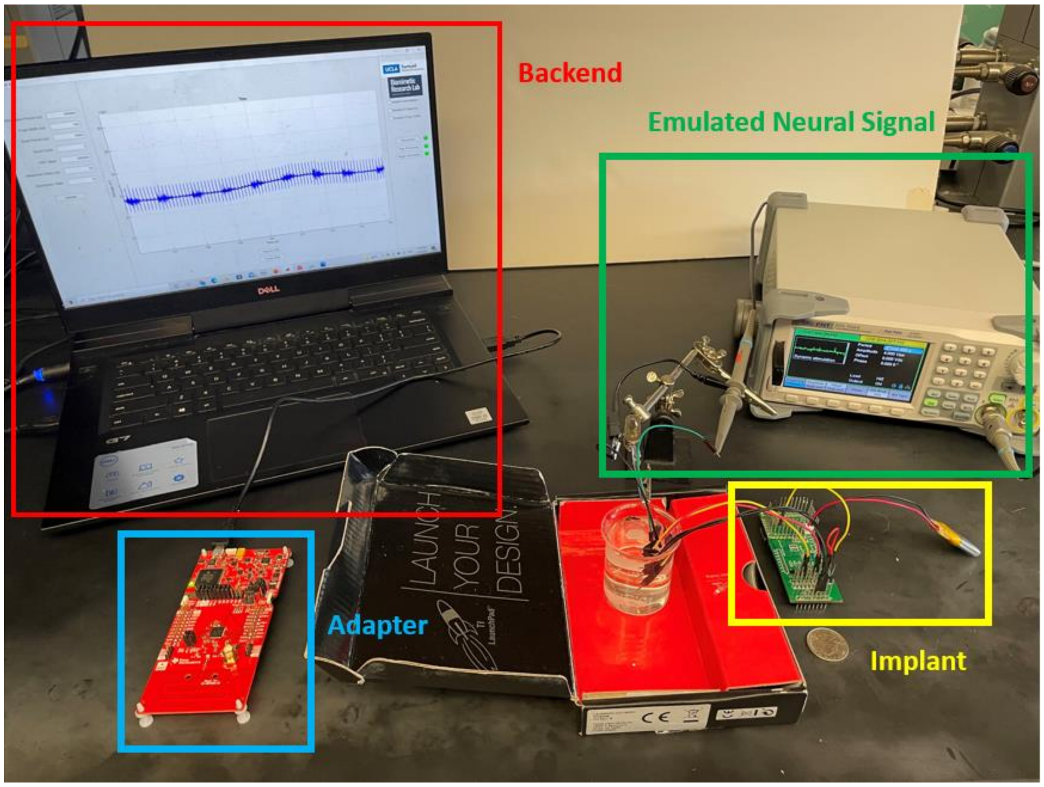


Figure 3-2: Experimental System Configuration

The signal flow during normal operation is the following. Recorded neural signal is sent to the adapter from the implant through BLE, buffered in adapter, and streamed to the backend device through universal asynchronous receiver-transmitter (UART) for data storage and real-time signal processing. Commands to adjust stimulation parameters is instantiated in the backend device and sent to the adapter through UART, then forwarded to the implant wirelessly through BLE.

3.2. Stimulation Protocol

The current version of the system can provide biphasic and monophasic mode, periodic and aperiodic Poisson-distributed mode, fixed and Poisson-distributed rando pulse width mode current burst stimulation. Configurable parameters include stimulation period, burst period, burst count, pulse width, stimulation amplitude, inter-phase delay, and asymmetric ratio.

Figure 3-3 below illustrates the periodic stimulation with respect to all configurable parameters. The cathodic phase is configured to be the leading phase in biphasic stimulation protocol of the system. Firmware can be changed to accommodate the need of anodic phase as the leading phasic in biphasic stimulation protocol. However, it has been demonstrated that cathodic stimulation is more efficient to elicit neural responses than anodic stimulation. Therefore, the firmware in the system is default to generate biphasic stimulation with cathodic leading phase. The example shown in Figure 3-3 illustrates the scenario of a periodic burst train stimulation with 3 biphasic pulses in each burst train. A traditional periodic stimulation protocol with no burst trains can be easily implemented by setting the parameter burst count to 1. Typically, the purpose of the second phase in a biphasic stimulation is to neutralize the extra charge introduced into the organism by the leading phase. Therefore, the pulse width of both phases is typically set to equal. However, the system provides the functionality to adjust the ratio of two phases so that 1). Cathodic and anodic monophasic stimulation can be achieved by setting the asymmetric ratio to 0 or a large number (typically greater 75). 2). Asymmetric biphasic stimulation can be achieved to further investigate different stimulation protocol.

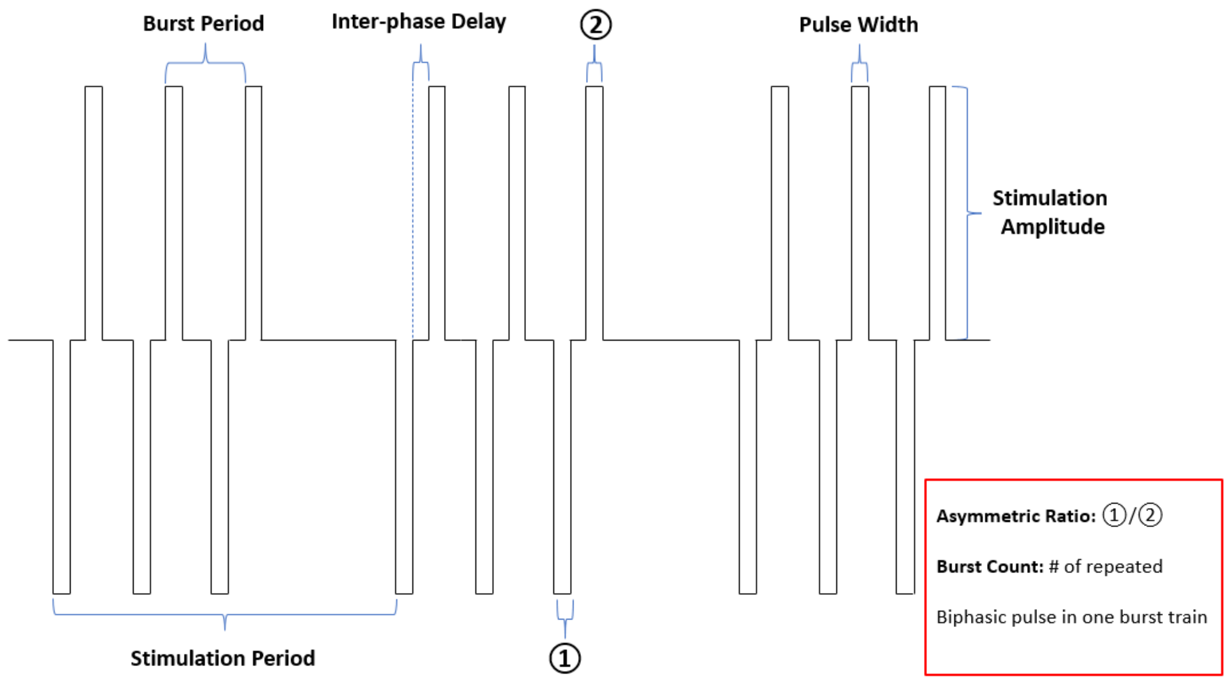


Figure 3-3: Periodic (Fixed Frequency, Fixed Pulse Width) Stimulation Protocol

The system can accommodate aperiodic randomized stimulation where the period between adjacent burst trains is Poisson distributed with λ equals the stimulation period parameter. Figure 3-4 illustrates an example of aperiodic randomized stimulation with burst trains of 3 biphasic pulses. The same scheme can be applied to traditional stimulation without burst trains.

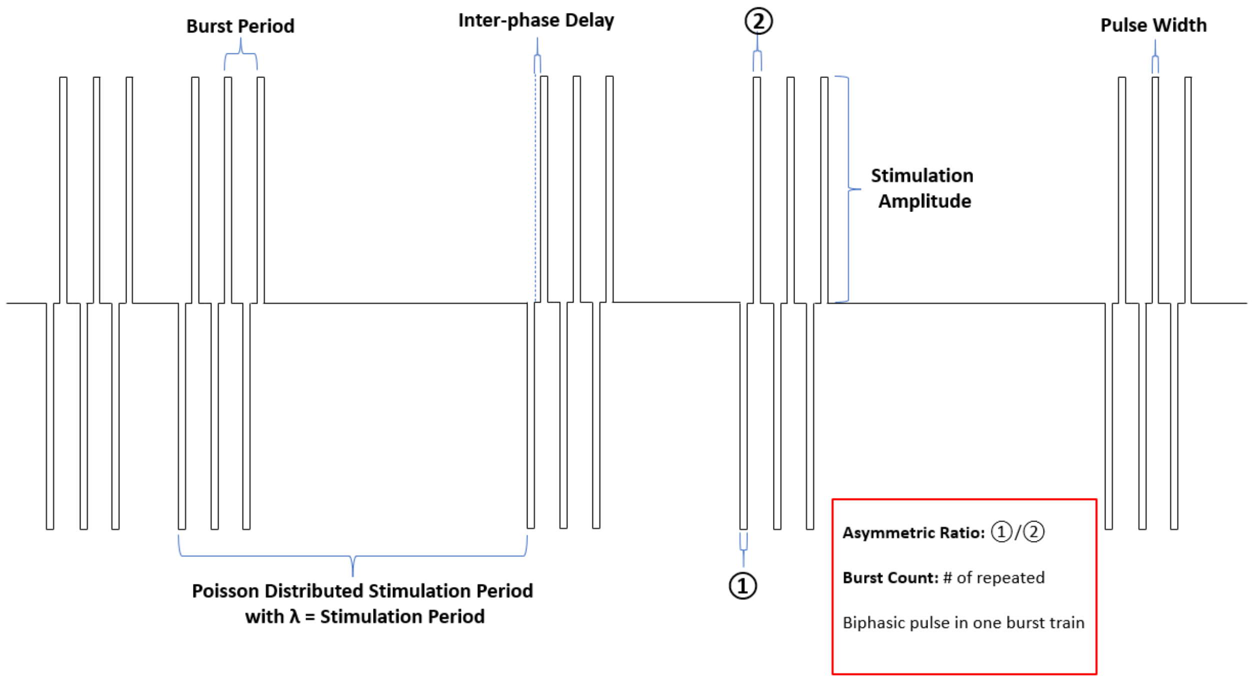

Burst Period
Inter-phase Delay
②
Pulse Width
Stimulation Amplitude
Poisson Distributed Stimulation Period with λ = Stimulation Period
①
Asymmetric Ratio: ①/②
Burst Count: # of repeated
Biphasic pulse in one burst train

Figure 3-4: Aperiodic (Poisson Distributed Frequency, Fixed Pulse Width) Stimulation Protocol

A different scheme of randomization is available in periodic stimulation where the stimulation pulse width is Poisson distributed with λ equal the parameter pulse width and the stimulation period is fixed. In the random frequency mode, the burst stimulation is no longer supported and burst count is fixed to 1. Figure 3-5 illustrates an example of Poisson distributed frequency mode. The asymmetric ratio can be adjusted in this mode as well to provide more flexibility in the stimulation waveform. While the influence of randomized pulse with on neural response is not clear, this system serves as a convenient tool for further investigation.

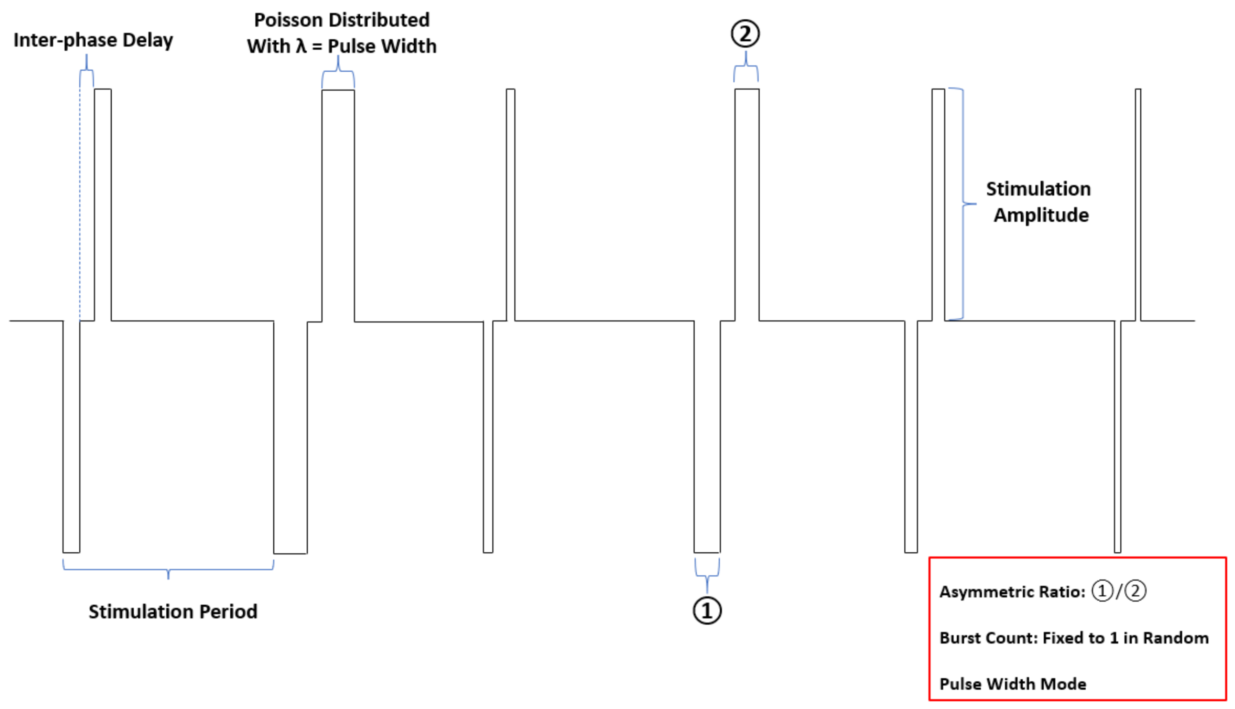


Figure 3-5: Periodic (Fixed Frequency, Poisson Distributed Pulse Width) Stimulation Protocol

Additionally, random frequency mode and random pulse width mode can operate simultaneously where stimulation period and pulse width are Poisson distributed. A sample stimulation waveform is shown in Figure 3-6. Similar to random pulse width mode, burst train stimulation is no longer supported in this mode and burst count is fixed to 1. All other parameters retain the same flexibility and are allowed to be modified in real-time.

3.3. System Specification

The specification of the current system is summarized in Table 3-1 below. Additionally, implicit constraints exist for the system due to the definition of the stimulation protocol.

Referring to Figure 3-3, the duration of the entire burst train needs to be shorter than stimulation period. Therefore, an implicit constraint is the following:

$$\mathit{Burst\ Count} * \mathit{Burst\ Period} \leq \mathit{Stimulation\ Period}$$

Table 3-1: System Specification

Parameters/Conditions	Range/Limit
Stimulation Frequency	<=1000Hz
Pulse Width	>=100µs
Burst Period	>=100µs
Stimulator Compliance Voltage	10V
ADC # of Channels	1 (max 4)
Sampling Frequency	15.625kHz (max 31.25kHz)
Resolution	24-bit

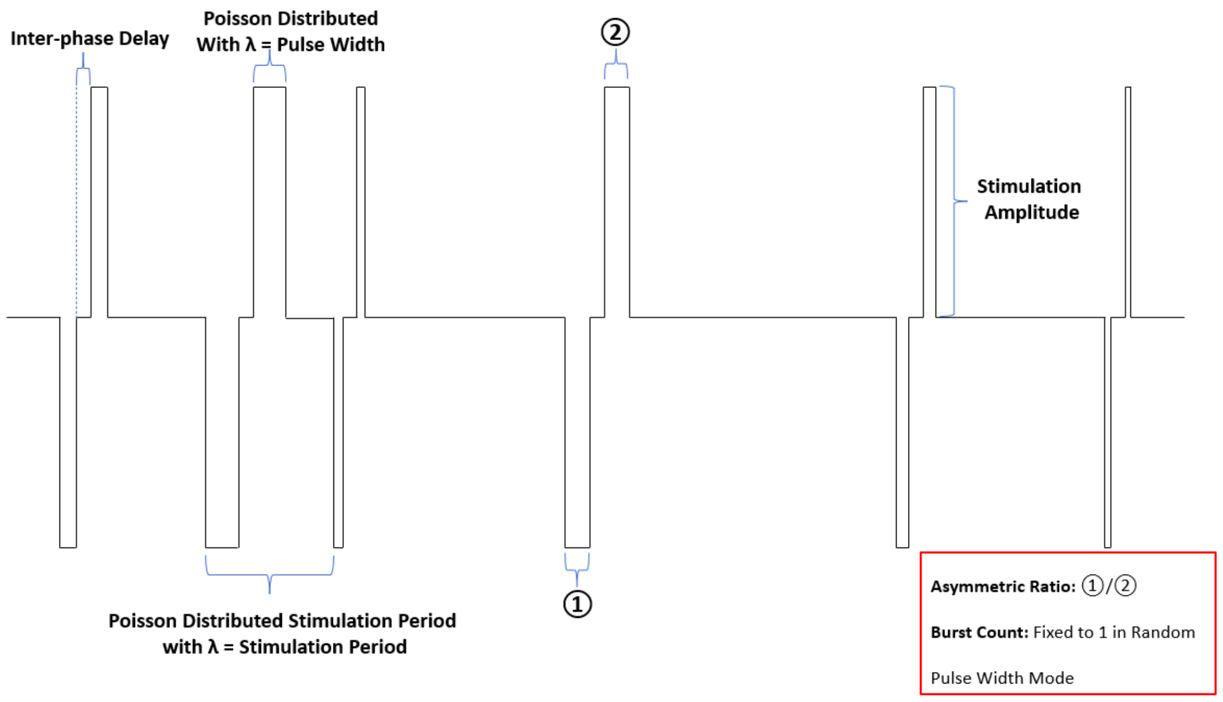

Inter-phase Delay
Poisson Distributed With λ = Pulse Width
②
Stimulation Amplitude
Poisson Distributed Stimulation Period with λ = Stimulation Period
①
Asymmetric Ratio: ①/②
Burst Count: Fixed to 1 in Random Pulse Width Mode

Figure 3-6: Aperiodic (Poisson Distributed Frequency, Poisson Distributed Pulse Width) Stimulation Protocol

Chapter 4 : Hardware Design

4.1. Power Management

The power management of the system consists of 4 parts: battery/power supply, wireless charger, boost converter, low dropout. The system can be working with a battery or power supply ranging from 3.3 V to 4.5 V. However, since a wall power supply introduce undesired 60 Hz noise, a battery is preferred. Currently, a 600 mAh 4 V LiPo battery is used for experiment and testing, which can last more than 40 hours given the maximum current consumption of the system is approximately 15 mA. A wireless charger which consists of a charging IC LTC4124 and a coil. Note that no ground copper plate is allowed below the coil, which needs to be considered when design PCB with the minimum possible area. A TPS7A0233PDQNR low dropout (LDO) IC is used to provide stable 3.3 V supply for MCU, ADC, and pins that need to be constantly pulled high. TPS7A0233PDQNR consumes 25 nA current and the maximum output current is 200 mA. Additionally, This LDO is available in X2SON package with 1.00 mm × 1.00 mm, which provides significant advantage to reduce overall PCB area. A boost converter LTC3459EDC#TRMPBF consuming 10 µA quiescent current is used to provide 10 V supply to the single pole double throw switch to control the polarity of the stimulation current. This boost converter is available in a footprint of 2.00 mm × 2.00 mm, which provides convenience to shrink down the size of the PCB.

4.2. Analog to Digital Converter (ADC)

3 different ADC have been considered including ADS131A02, AD4114BCPZ, and ADS131M04 in the first and second iteration of the system. ADS131M04 is used in the current

implementation. AD4114BCPZ has a large dynamic range to capture large-amplitude stimulation artifact for real-time artifact-removal post-processing algorithm. Therefore, the 2nd iteration PCB was designed with both ADS131M04 and AD4114BCPZ for comparison. However, due to the recent global chip shortage, testing was not carried out on AD4114BCPZ. ADS131A02 was used in the 1st iteration PCB. The impedance of this ADC is around 100kΩ, which could be prone to large signal distortion when recording bio-signals. Also, the relatively large footprint (5.00 mm × 5.00 mm) and need for more GPIOs to interface could be potential roadblock of system miniaturization.

ADS131M04 sigma-delta ADC is chosen to digitize analog neural signals for the following considerations. 1). ADS131M04 is available in WQFN package which has a footprint of 3.00 mm × 3.00 mm. Such small dimension provides advantages to miniaturize the PCB area and the overall size of the remote system. 2). ADS131M04 has programmable gain up to 128, which eliminates the need of an additional front-end amplifier, which also contributes to the reduction of the overall PCB area. 3). Three operations mode are available, and the low power provides good tradeoffs between sampling rate, noise performance, and input impedance. At the desired sampling rate and input gain, noise level is 3.63 μV_{RMS} and the input impedance is 30MΩ. 4. Maximum four differential recording channels are available. While the project currently targets a single-channel stimulation and recording device, the redundancy in BLE bandwidth does allow the extension to a multi-channel system given the availability of ADC channels. 5. The selected ADC has 24-bit resolution.

One caution on this selected ADC is that ADS131M04 requires a clock signal at 8.192MHz by default. However, supplying a clock signal at this specific frequency with a crystal oscillator introduces extra system footprint. A resolution in the system is used the system clock of CC2652RSIP to supply a clock signal at 8MHz as an alternative. In this case, the resulting sampling rate becomes slightly lower than using 8.192MHz clock signal.

4.3. Stimulator

The Design of stimulation to delivery electrical stimulation is shown in Figure 4-1 below. A voltage-controlled current source is built with a low-power operational amplifier LT1637 with polarity control enabled by a single pole double throw (SPDT) switch ADG5436. The input voltage of the amplifier V_{DAC} is provided by the digital-to-analog converter on the microcontroller CC2652RSIP. The current limiting resistor R limit the maximum current stimulation that can be delivered by the stimulator. In the current implementation, the current limiting resistor is set to 500Ω, and the corresponding largest supported stimulation current is 2mA. If a larger stimulation current is needed, a smaller current limiting resistor can be used to replace the 500Ω resistor currently in used. The polarity of ADG5436 SPDT switch is controlled by 2 GPIOs from CC2652 MCU to generate biphasic current stimulation pulse.

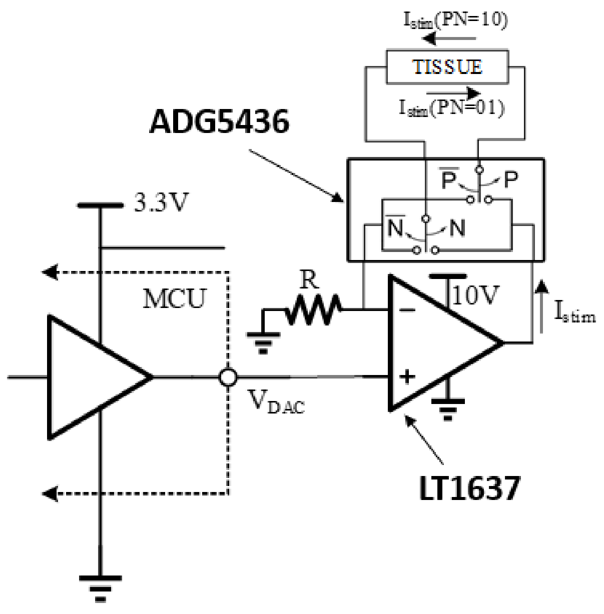


Figure 4-1: Stimulator Design

The design of the stimulator circuit is verified through simulation in LTSpice. The simulated circuit built in LTSpice is shown in Figure 4-2 below. Simulation is run to generate biphasic pulses with 100µs pulse width and 1ms period to validate the control scheme and the stimulation results are shown in Figures 4-3 and 4-4. Also, it is crucial to verify the rise-time of induced voltage as a long rise-time could destroy the integrity of constant current stimulation. Additionally, notice that a voltage sweep is applied to the non-inverting input of LT1637 to confirm the stable operation of the stimulator over a wide range of output current.

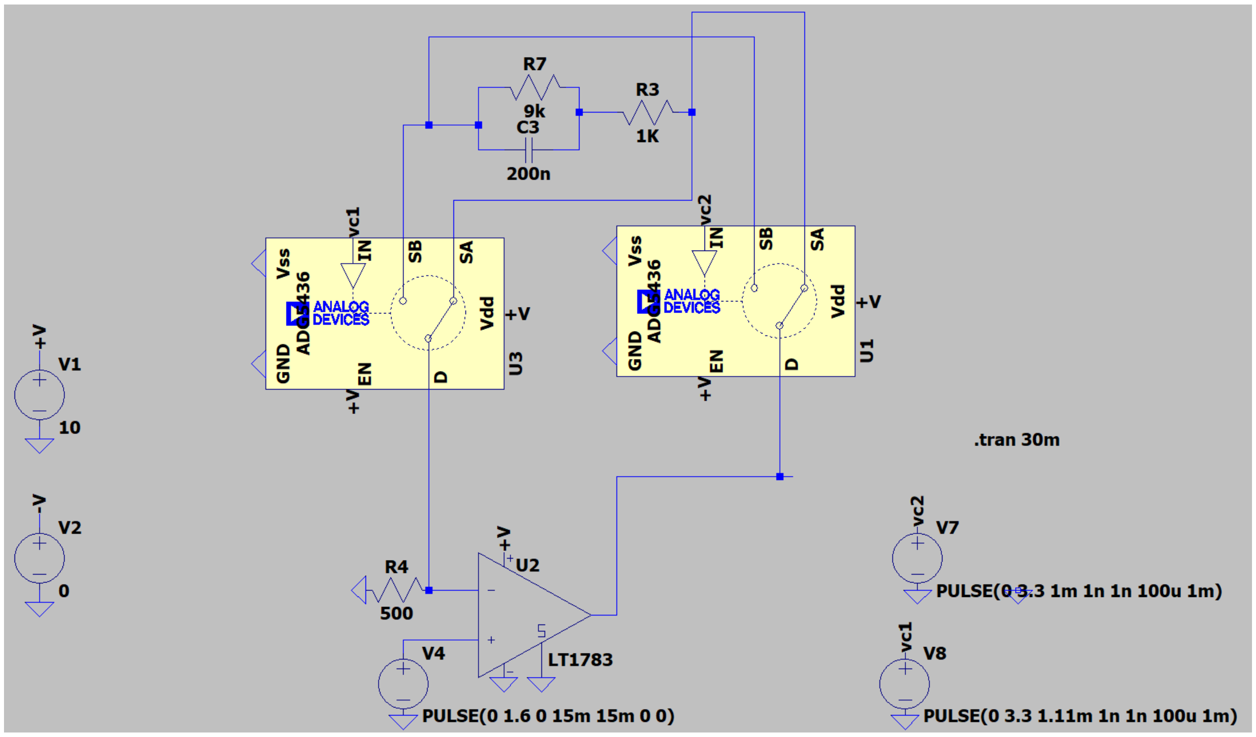

R7
9k
C3
200n
R3
1K
Vss
GND
IN
SB
SA
Vdd
EN
D
vc1
vc2
+V
-V
U3
U1
ADG436
V1
10
V2
0
R4
500
U2
LT1783
V4
PULSE(0 1.6 0 15m 15m 0 0)
.tran 30m
V7
PULSE(0 3.3 1m 1n 1n 100u 1m)
V8
PULSE(0 3.3 1.11m 1n 1n 100u 1m)

Figure 4-2: Simulated Circuit in LTSpice for Design Validation

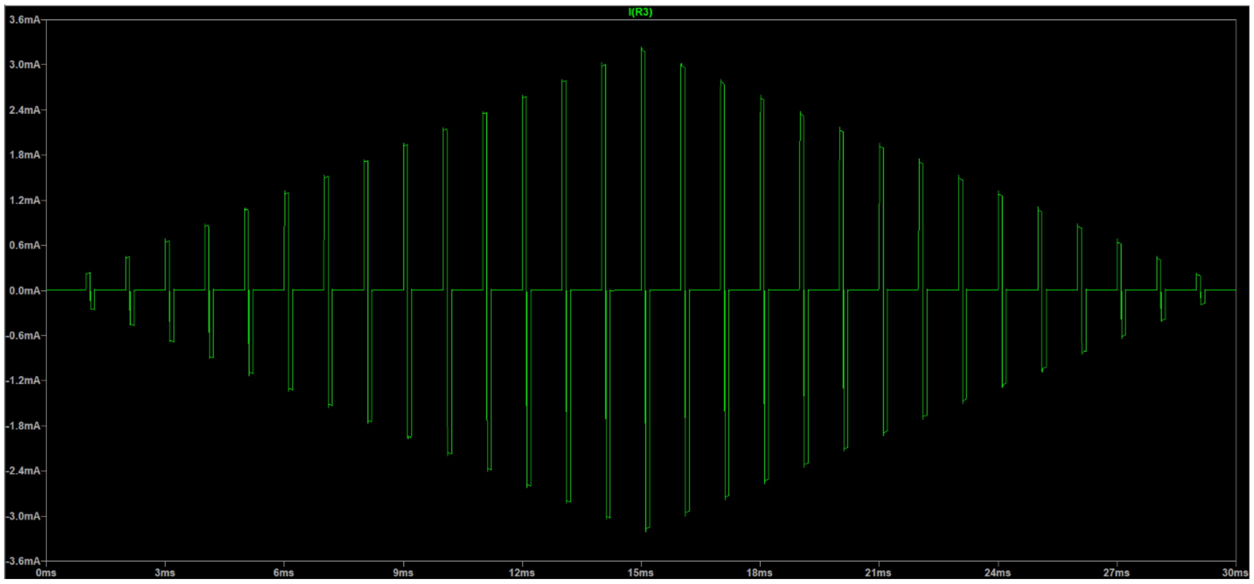


Figure 4-3: Simulation Result

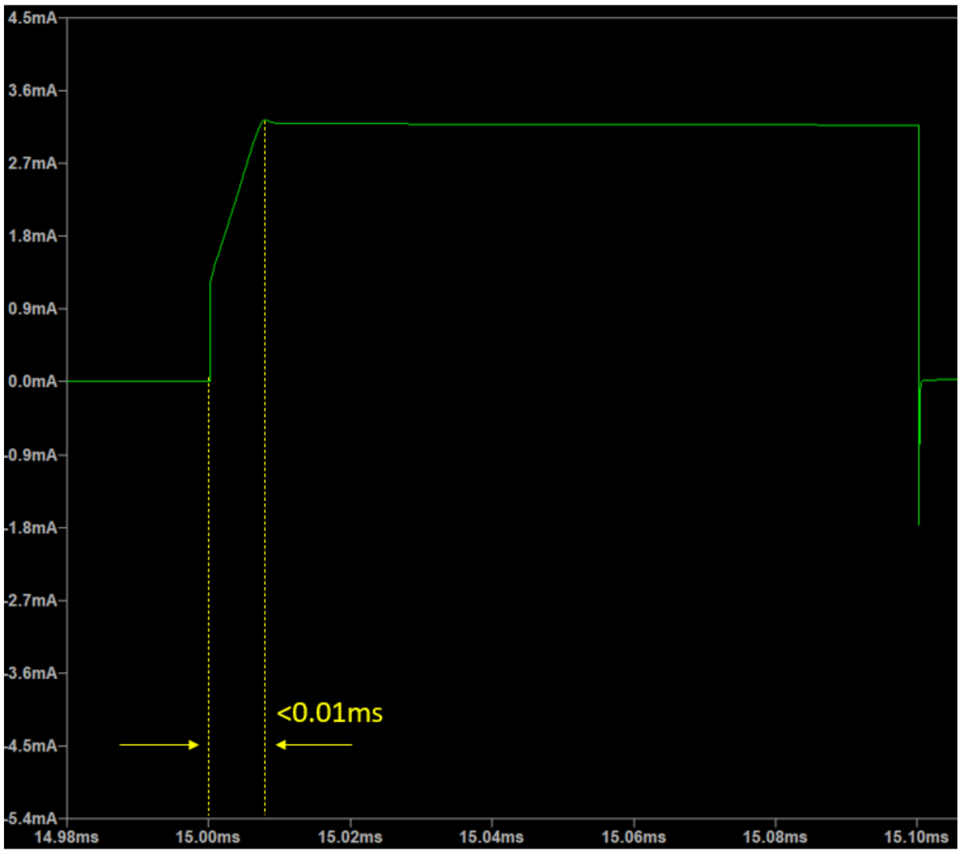


Figure 4-4: Rise-time of the Stimulator with 100us Pulse Width

4.4. Printed Circuit Board (PCB) Design

In this system, a CC1352R1 or similar launchpads that have BLE capability can be used as an adapter between the implant and backend since no size restriction exists on the adapter. Therefore, PCB design only needs to be carried out on the implanted/remote device to create a functional prototype. The development of the implant is devised in 3 steps. In step 1, the 1st iteration of the PCB is layout aiming to validate the system design without an on-board MCU and the emphasis on overall PCB area. In step 2, with some issues of discovered from the testing in step 1, the system design is modified, and the 2nd iteration PCB is layout incorporating an on-board MCU with some effort to reduce the overall PCB area. In step 3, finalize the system design and layout a multi-layer final version miniaturized PCB with the targeted dimension around 2.00 cm×1.75 cm.

The design file and the manufactured 1st iteration PCB are shown in Figure 4-5 and Figure 4-6 below. Modular design is applied for the convenience of incremental testing and avoid possible failure leading to a complete breakdown of the system. The design of the 1st-iteration PCB consists of 5 modules: power management, recorder, stimulator, wireless charger, and blanking circuitry. Note that the wireless charger and blanking circuitry modules are designed by Yanpeng Chen. Different modules are completely isolated including the ground copper plane. Reserved headers across different modules can be used to connect the ground plane of subsystems when needed. To ease the design process, a CC1352R1 launchpad is used to interface each subsystem without using an on-board MCU. The blanking circuitry was used for artifact cancellation in the original design. However, it is replaced by the real-time signal processing method proposed later. The board area of 1st iteration PCB is not well minimized purposefully as the motivation for this PCB is mainly to validate the circuit design and detect possible system failure.

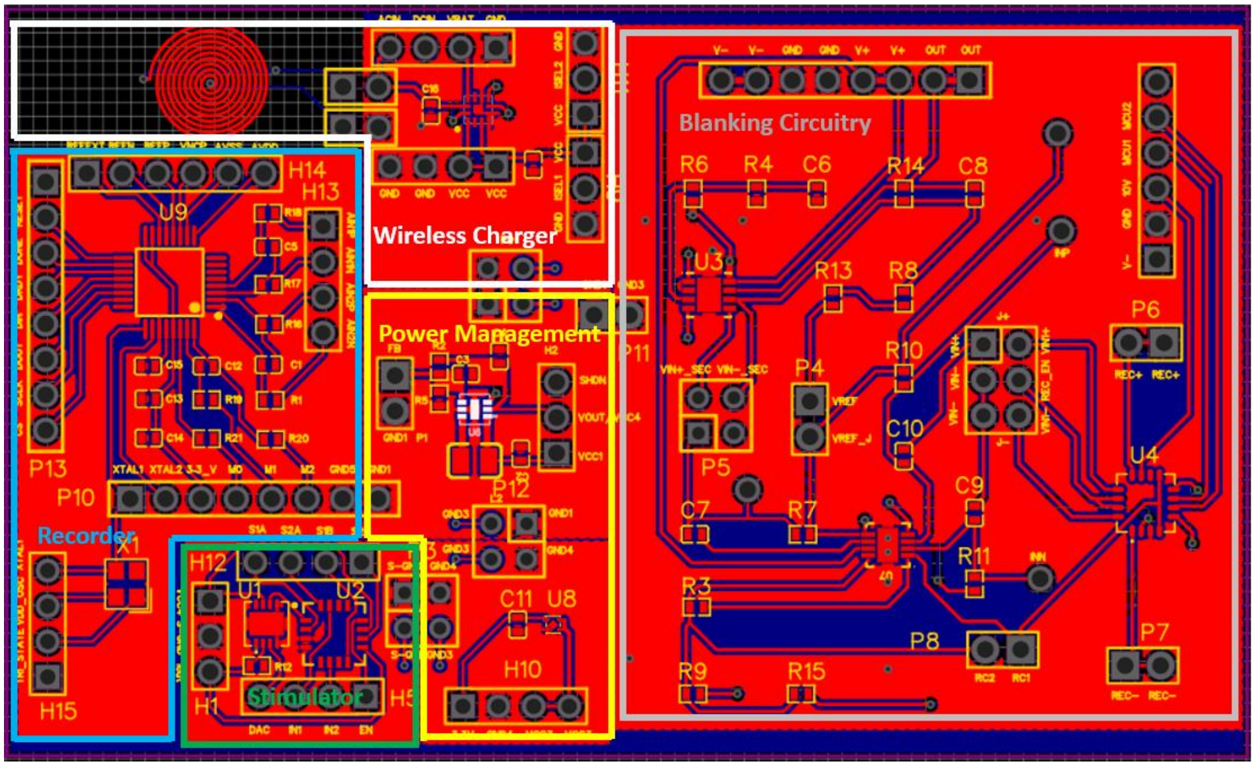


Figure 4-5: 1st Iteration PCB Drawing

The testing procedures can be summarized as the following:

1. Verify the operation of LDO and boost converter using a power supply then with a battery.
2. Verify the operation of Stimulator with a power supply then with the power management module powered by a power supply.
3. Verify the operation of recorder with a power supply then wutg the power management module powered by a power supply.
4. Verify the simultaneous operation of stimulator and recorder with a power supply then with the power management module powered by a power supply.

5. Verify the simultaneous operation of stimulator and recorder with the power management module powered by a battery.

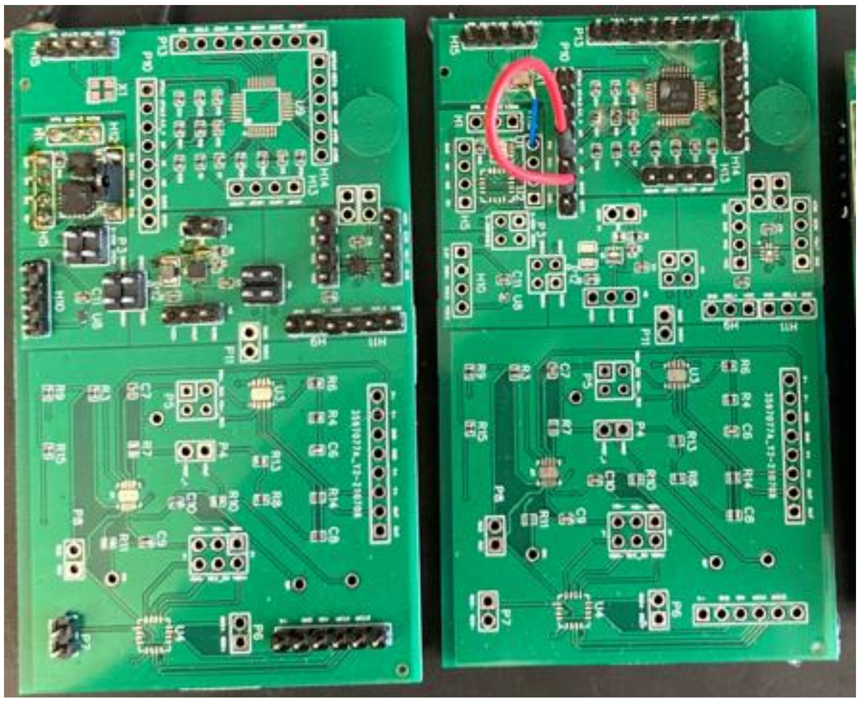


Figure 4-6: 1st Iteration PCB

The design file and the manufactured 2nd iteration PCB are shown in Figure 4-7 and 4-8 below. In the revised design, subsystems and the ground plane are no longer isolated like it was in 1st iteration PCB. Rather, they are interconnected and placed close to each other to somewhat reduce the overall PCB area, which is a transitional stage to the final system. The size of the PCB is reduced to 8.00 cm×3.00 cm.

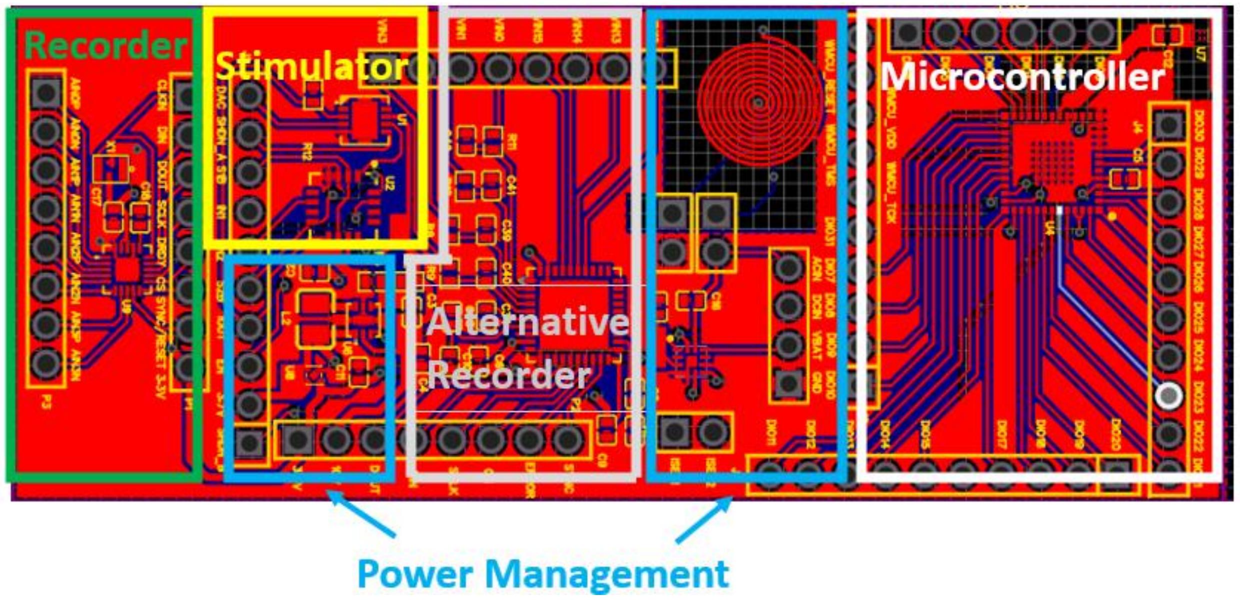


Figure 4-7: Drawing of 2nd Iteration PCB

In the original plan, a CC1352R MCU is used as the controller for the remote system. However, additional resistors, capacitors, and clock source are required to drive CC1352R, which introduces additional PCB area. A new product CC2652RSIP from Texas Instrument. Inc is system-in-package microcontroller chip with integrated DCDC components, balun, and crystal oscillators. Such features provide significant convenience to minimize the overall size of the implanted/remote system. Additionally, CC1352 and CC2652 belong to the same family of which the source codes are mostly compatible. Thus, CC2652RSIP is used in the revised design to replace CC1352R. In terms of the recorder, as mentioned previously, the 2nd iteration PCB was designed to compare 2 ADCs (ADS131M04 and AD4114BCPZ) to replace ADS131A02 in 1st iteration PCB. The designs of both ADS131M04 and AD4114BCPZ are incorporated into PCB as separate modules. However, due to chip shortage, AD4114BCPZ was out of stock and the current system operates with ADS131M04. Also, the design of blanking circuitry in 1st iteration is removed and artifact cancellation would be done in a real-time post-processing algorithm in MATLAB which will be discussed in detail in section 6.2. Ultimately, such algorithm would be migrated and executed on MCU. By removing the original blanking circuitry, the size of the PCB can be further reduced.

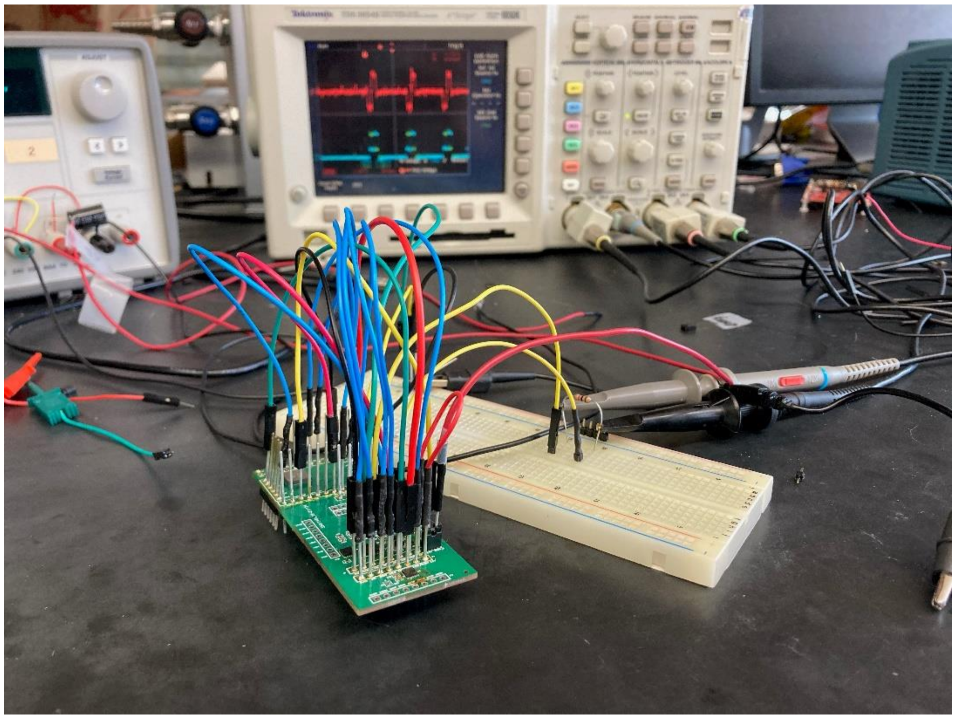


Figure 4-8: 2nd Iteration PCB

The 2nd iteration PCB was tested following the same procedures. The setup of 2nd iteration PCB after the testing procedures is shown in Figure 4-9. Furthermore, jumper wires on the top are mostly removed for the convenience of animal testing. Instead, the necessary connections are replaced by wires soldered at the bottom of the board to avoid the risk of loose jumper wires being plugged out accidentally, which is highly possible in future chronic animal experiments.

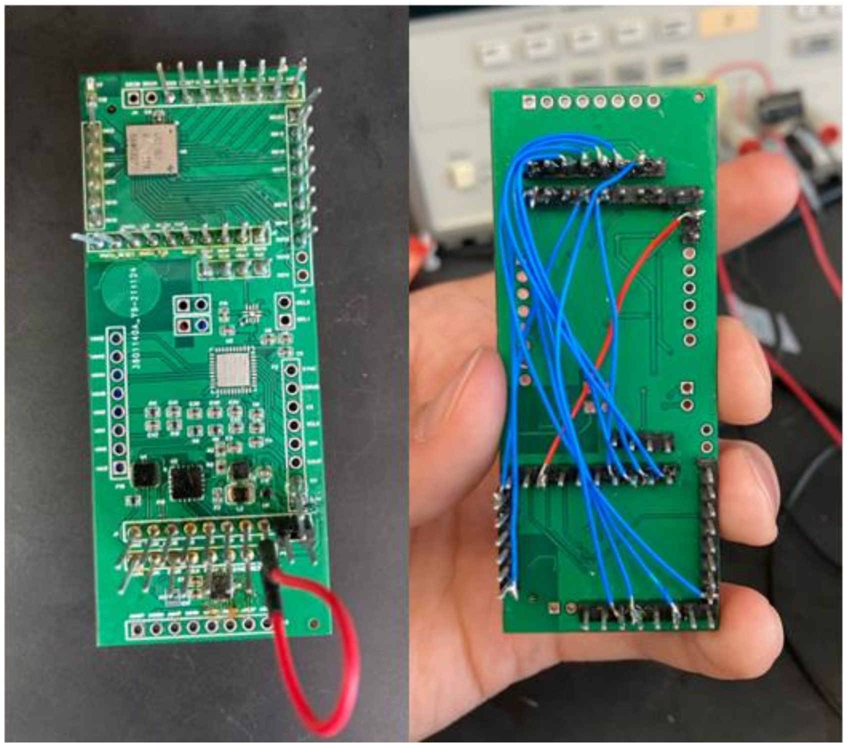


Figure 4-9: 2nd Iteration PCB with Jumper Wires Removed

4.5. Adapter

Adapter is a major component of the system that connects the remote system and the backend. As most commercially available laptops in the recent years are equipped with Bluetooth module, in the original plan, the system consists of implanted/remote device and backend, and they directly communicate through Bluetooth Low Energy. In testing, it was noticed that the maximum data streaming rate between the remote device and a backend with Windows operating system is approximately 144Kbit/s which is much less than the maximum speed of BLE (720Kbits/s) under 1 Megabit Physical Layer (1M PHY) due to the limitation of connection interval. While Bluetooth Low Energy (BLE) connection interval ranges from 7.5 ms to 4s in theory, device-specific limitation also exists. For laptop with Windows operating system, the maximum connection interval of BLE is usually 20ms. Such narrow range limits the maximum BLE data streaming rate between the remote system and the backend. Furthermore, many laptops with Windows operating system does not support 2 Megabit Physical Layer (2M PHY) which could potentially lift the transmission rate even more (up to about 1400 Kbits/s). Such limitation prevents the potential extend of the system to a multi-channel stimulation and recording system. Aside from the capability of BLE, the adapter needs to have UART capability to forward recorded data and receive command from backend.

Due to the aforementioned limitations, an adapter that supports BLE speed over the data generation rate of single-channel recording is needed to avoid data loss. By default, the remote device samples neural signals at 15.625kHz and retains 16 noise-free bits out of total 24-bit resolution to reduce the occupation of bandwidth. Therefore, the minimum data rate required would be:

$$Minimum\ Data\ Rate = 15625\ samples/s * 16\ bits = \ 250Kbits/s$$

Additionally, a larger bandwidth allowed by the adapter should be consider for extendibility of the system into a multi-channel system in the future. Thus, an adapter with 2M PHY is highly desired to ensure the potential to extend the system to a 4-channel system. Moreover, the convenience of development needs to be considered. Since no strict size restriction exists for the adapter, a commercially available ready-to-use product that can be programmed would be an ideal option.

With all considerations above, a CC1352R1 launchpad, shown in Figure 4-10 below, is used as the adapter of the system. In addition to meeting all requirements above, firmware support for CC1352 and CC2652 series can be used interchangeably. Currently, a CC1352R1 ready-to-use launchpad can simply be programmed without the need of additional hardware to produce a functional prototype. In the future, a CC1352R MCU chip can be used to develop a smaller adapter to improve user-friendliness.

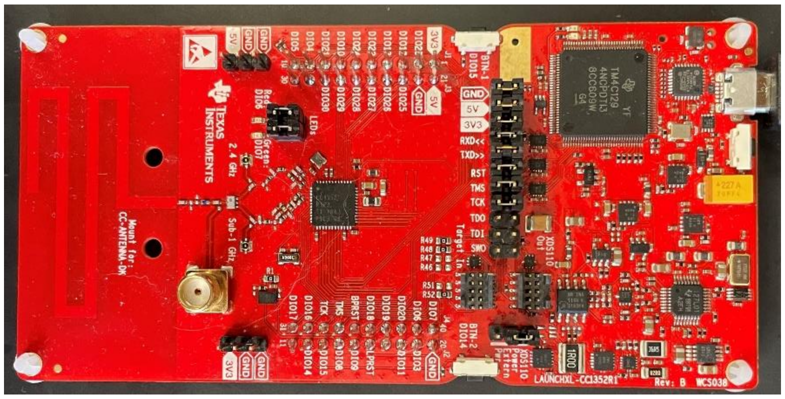


Figure 4-10: CC1352R1 Launchpad

Chapter 5 : Firmware Design

5.1. Firmware Design of the Implanted/Remote Device

5.1.1. Overview of Firmware Design of Implanted/Remote Device

The remote device operates as a peripheral device in BLE communication protocol. Proper configuration of BLE stack is crucial to meet the target design requirements. Important configurations of BLE stack include the following:

1. Physical Layer (PHY): 1M PHY
2. Address Mode: Public address
3. Minimum Connection Interval: 25ms
4. Maximum Connection Interval: 25ms
5. Data Length Extension: Yes

Note that both minimum and maximum connection interval are set to 25ms. In general, devices in BLE protocol send data with a variable number of bytes during one connection event within the configured minimum and maximum connection interval. With high BLE data rate and multiple tasks operating in parallel, connection events with unpredictable duration may create data loss in some scenarios. Therefore, such practice is implemented to ensure the predictable behavior of the system and guarantee zero buffer overflow. The setup of Generic Attribute Profile (GATT) determines the firmware design. Aside from default services, a custom service is built with 15 characteristics to control the operation and stimulation parameter over-the-air. Table 5-1 below summarizes the universally unique identifier (UUID), attribute value length, descriptions, and the associated command type of all 15 characteristics. Refer to section 3.2 for descriptions of stimulation parameters in the table.

Table 5-1: Descriptions of Characteristics in Custom Service

UUID	**Attribute Value Length**	**Description**	**Control or Parameter Update Command**
0x1235	1	Data length extension enable	Control
0x1236	1	PHY layer selection	Control
0x1237	1	Notification enable and recording data after enabled	Control
0x1238	1	Stimulation enable	Control
0x1239	3	Stimulation period (μs)	Parameter update
0x123A	2	Pulse width (μs)	Parameter update
0x123B	2	Burst period (μs)	Parameter update
0x123C	1	Burst count	Parameter update
0x123D	1	Update temporarily stored parameters	Control
0x123E	3	DAC value (I_{out} = DAC value / 500 μs)	Parameter update
0x123F	1	Inter-phase delay (μs)	Parameter update
0x1240	1	Asymmetric ratio, the ratio between the pulse width of the first phase over second phase. (Actual ratio = asymmetric ratio/10)	Parameter update
0x1241	1	Random pulse width enable	Control
0x1242	1	Random frequency enable	Control
0x1243	1	Recording ADC enable	Control

Table 3-1: Descriptions of Characteristics in Custom Service

Attributes used to represent stimulation parameters in Table 3-1 do not follow the normal binary rule in digital memory. At the beginning of development, a smartphone is used to connect to the remote device using LightBlue. While it is convenient to interface the remote device and change stimulation parameters in this way, user-input values in LightBlue are default in hexadecimal without pronounced annotation, which is likely to cause confusion. For example, a user intending to adjust the burst period to 5000μs could result in 0x5000 μs stimulation period without an intentional conversion from decimal to hexadecimal. Figure 5-1 below shows the user interface of LightBlue, in which the write command allows hexadecimal numbers by default. Therefore, a custom rule is applied in the attribute values so that the user can input the desired value in decimal form to avoid confusion and the inconvenience of manual conversion. The data format conversion procedure based on a custom rule can be demonstrated in Figure 5-2 below.

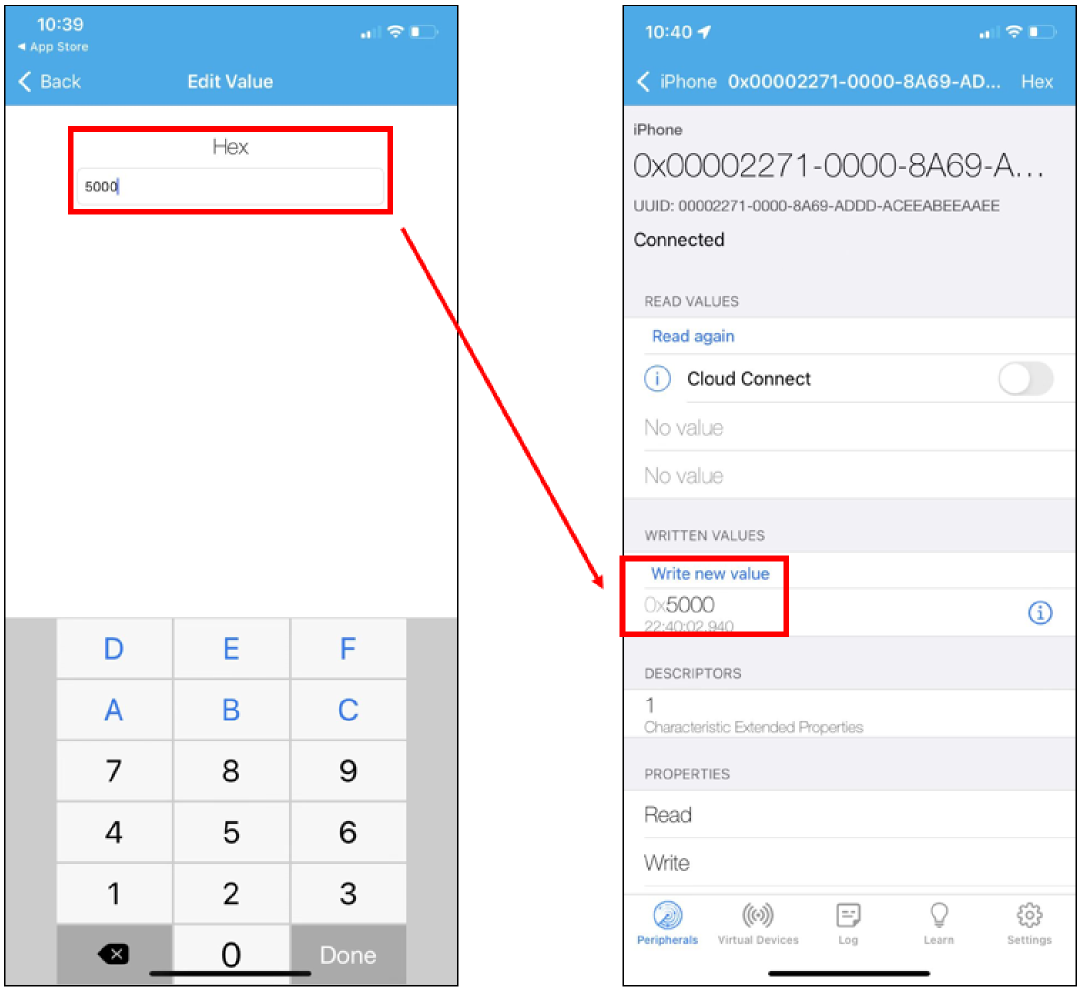


Figure 5-1: User Interface of LightBlue

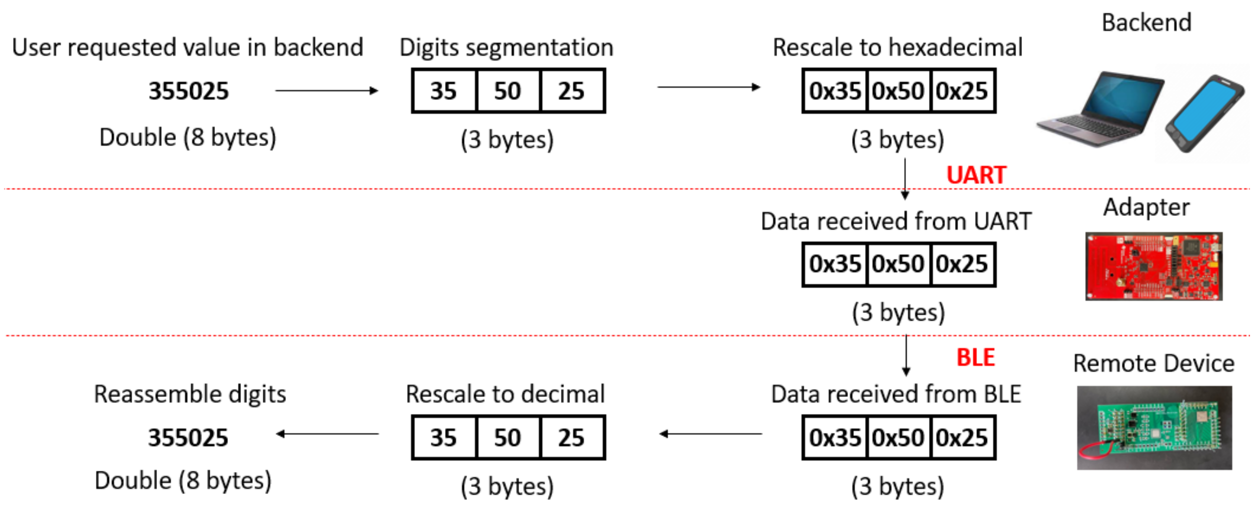


Figure 5-2: Data Conversion Procedure

User-input value of stimulation parameters in double-precision floating-point format is segmented per 2 digits, rescaled every 2 digits to ostensibly the same in hexadecimal in backend, which is then sent to the remote device through adapter. Upon the receipt of attribute value, the remote device reassembles 3 bytes of 2-digit numbers back to the original double-precision floating-point format to configure the stimulator. The algorithms below demonstrate the conversion between the desired value and the intermediate 3-byte format for communication in pseudocode.

$\boldsymbol{Algorithm}\ \mathbf{1}\ Pseudo\ Code\ for\ Conversion\ to\ 3-byte\ format$

```
**double** parameter = user-input value in backend;

**int** attribute_len; // length of the corresponding attribute value

**uint8** converted_value[3] = {0x00, 0x00, 0x00};

**string** param_str = num2str(parameter);
```

```
if (attribute_len == 3)
    converted_value[0] = str2int(param_str[0]*16) + str2int(param_str[1]);
    converted_value[1] = str2int(param_str[2]*16) + str2int(param_str[3]);
    converted_value[2] = str2int(param_str[4]*16) + str2int(param_str[5]);
else if (attribute_len == 2)
    converted_value[0] = str2int(param_str[0]*16) + str2int(param_str[1]);
    converted_value[1] = str2int(param_str[2]*16) + str2int(param_str[3]);
else if (attribute_len == 1)
    converted_value[0] = str2int(param_str[0]*16) + str2int(param_str[1]);
end if
```

$\boldsymbol{Algorithm}\ \mathbf{2}$ $Pseudo\ Code\ for\ the\ Reassembly\ to\ Double-Precision\ Format$

```
double parameter; // Stimulation parameter in decimal
int attribute_len; // length of the corresponding attribute value;
uint8 attribute_value[3]; // Attribute values received from BLE
if (attribute_len == 3)
    parameter = (attribute_value[0]/16*10+attribute[0]%16)*10000 +
    (attribute_value[1]/16*10+attribute[1]%16)*100 +
    (attribute_value[2]/16*10+attribute[2]%16);
```

```
else if (attribute_len == 2)
    parameter = (attribute_value[0]/16*10+attribute[0]%16)*100 +
    (attribute_value[1]/16*10+attribute[1]%16)*1;
else if (attribute_len == 1)
    parameter = attribute_value[0]/16*10+attribute[0]%16;
end if
```

An overview of the firmware workflow of the implanted/remote device is shown in Figure 5-3 below. A real-time operating system designed by Texas Instrument (TI-RTOS) is used to manage multiple tasks and achieve real-time operation on a single-core MCU. Three tasks created in TI-RTOS including stimulation task, BLE task, and recording task operate concurrently with the support of TI-RTOS. Stimulation task manages the amplitude, polarity, and timing of current stimulation by controlling the polarity of SPDT switch and the non-inverting input of the amplifier. Recording task monitors ADC readings buffered in SRAM and sends buffered data to adapter as BLE notifications in time to avoid buffer overflow. BLE task maintains the wireless connection to the adapter, changes operation status and modify stimulation parameters upon the receipt of parameter update command. The priorities of three tasks in descending order are the following: recording task, BLE task, stimulation task. The order of priorities is set to guarantee no data loss will occur during the operation with potentially acceptable micro-second level delay in stimulation waveform.

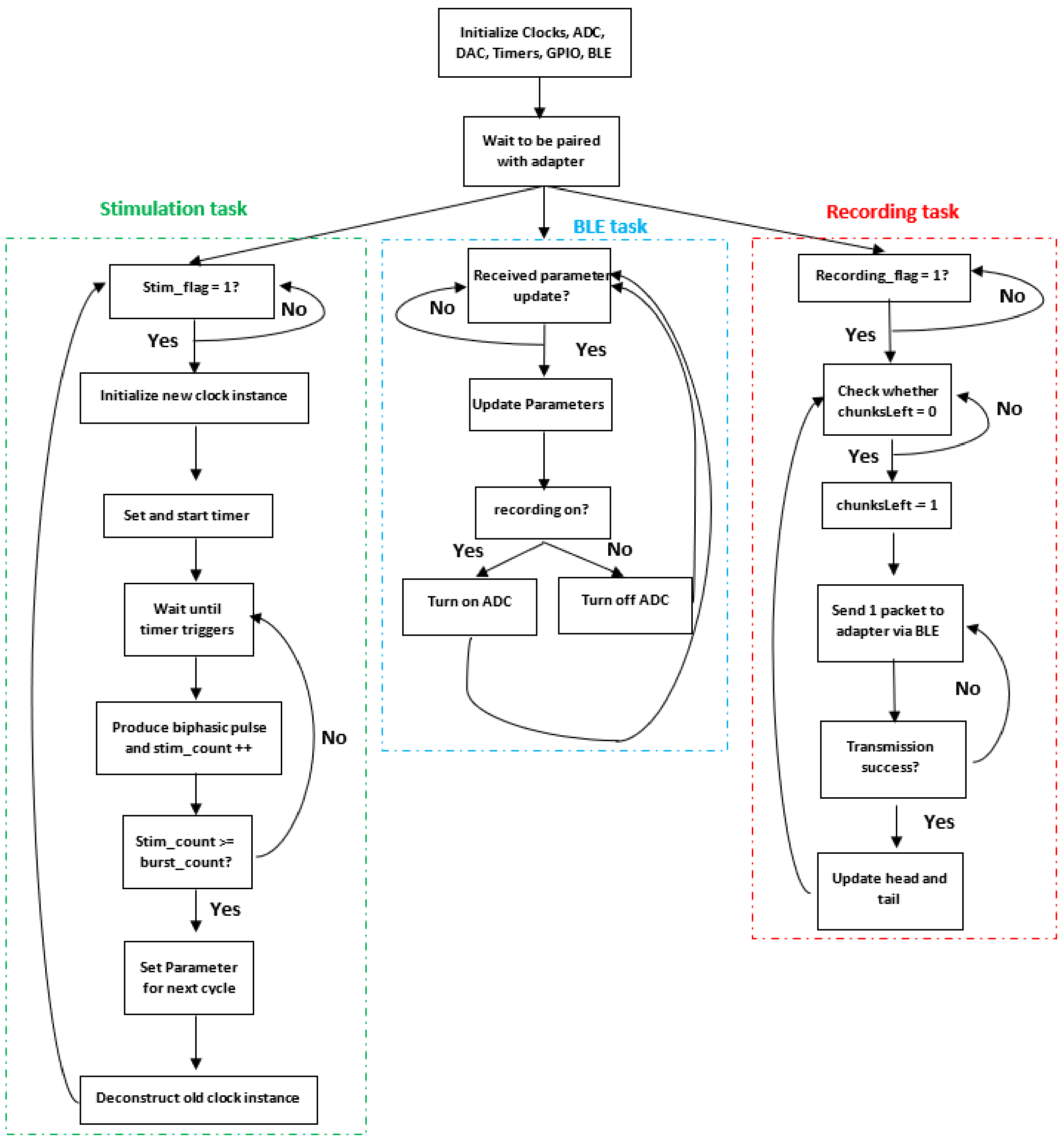

Initialize Clocks, ADC, DAC, Timers, GPIO, BLE
Wait to be paired with adapter
Stimulation task
BLE task
Recording task
Stim_flag = 1?
No
Yes
Initialize new clock instance
Set and start timer
Wait until timer triggers
Produce biphasic pulse and stim_count ++
Stim_count >= burst_count?
No
Yes
Set Parameter for next cycle
Deconstruct old clock instance
Received parameter update?
No
Yes
Update Parameters
recording on?
Yes
No
Turn on ADC
Turn off ADC
Recording_flag = 1?
No
Yes
Check whether chunksLeft = 0
No
Yes
chunksLeft = 1
Send 1 packet to adapter via BLE
No
Transmission success?
Yes
Update head and tail

Figure 5-3: Firmware Flowchart of Implanted/Remote Device

5.1.2. Stimulation Task

The stimulation task recruits 2 GPIOs, 1 DAC, 1 timer, and 1 TI-RTOS clock instance to manage the amplitude and timing of current stimulation. When stimulation is enabled, a TI-RTOS clock instance is initialized to be triggered when the next burst train is about to begin. When this clock instance is triggered, a timer is initialized to measure the duration until the next biphasic pulse in a burst train should be produced. When this timer triggers, 1 biphasic pulse is produced by setting DAC to the corresponding voltage level and switching the polarity of SPDT appropriately. The pulse width is created by a CPU delay function written in assembly code. Then the timer restarts with the same counter value for the next biphasic pulse. The reinitialization of timer is repeated until the user-defined number of biphasic pulses has been executed. Finally, the program sets the stimulation parameters for the next burst train and deconstructs the clock instance when 1 complete burst train is finished. The timing of clock instance, timer, and CPU delay in periodic stimulation protocol is illustrated in Figure 5-4 below.

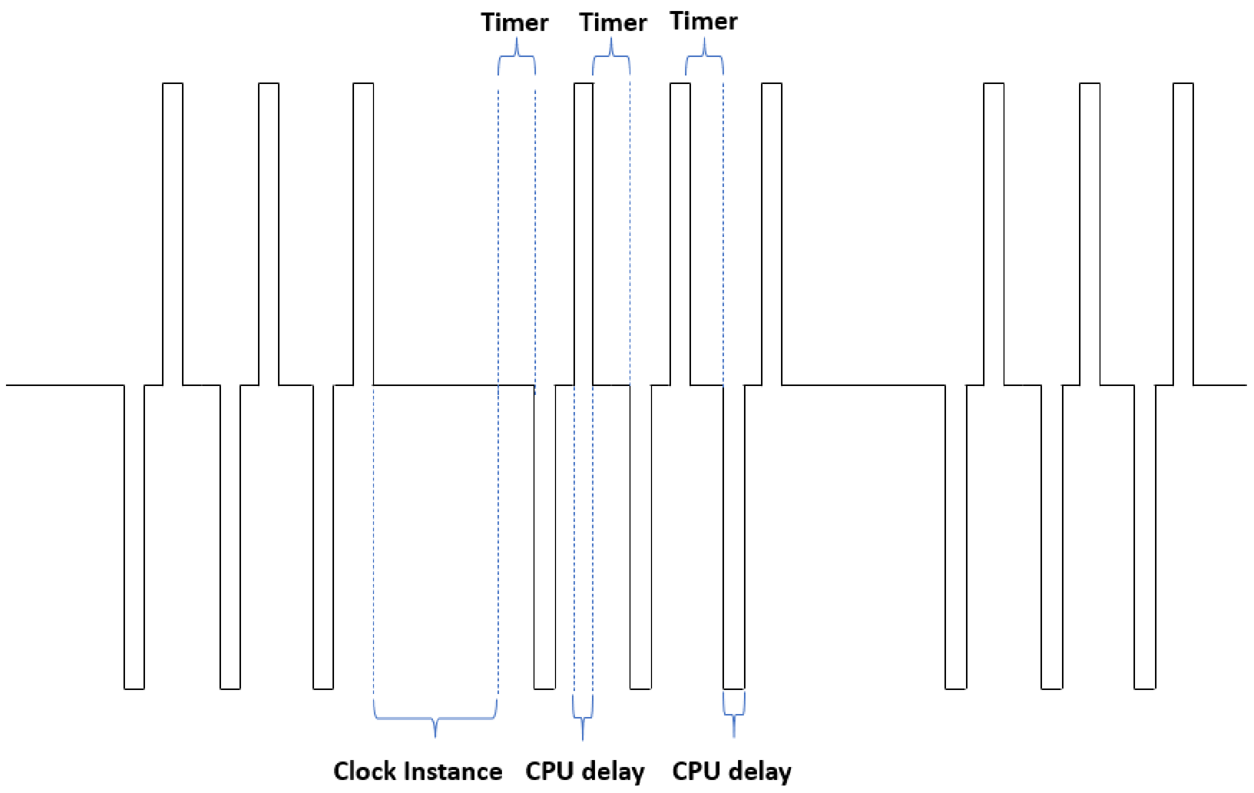


Figure 5-4: Timing of Clock Instance, Timer, and CPU Delay with Periodic Stimulation Protocol

In the aperiodic case, at the end of each burst train, the randomized parameter is calculated and the clock instance, Timer, and CPU delay for the next burst train will be configured accordingly. An example for the timing of clock instance, timer, and CPU delay in aperiodic stimulation protocol is illustrated in Figure 5-5 below.

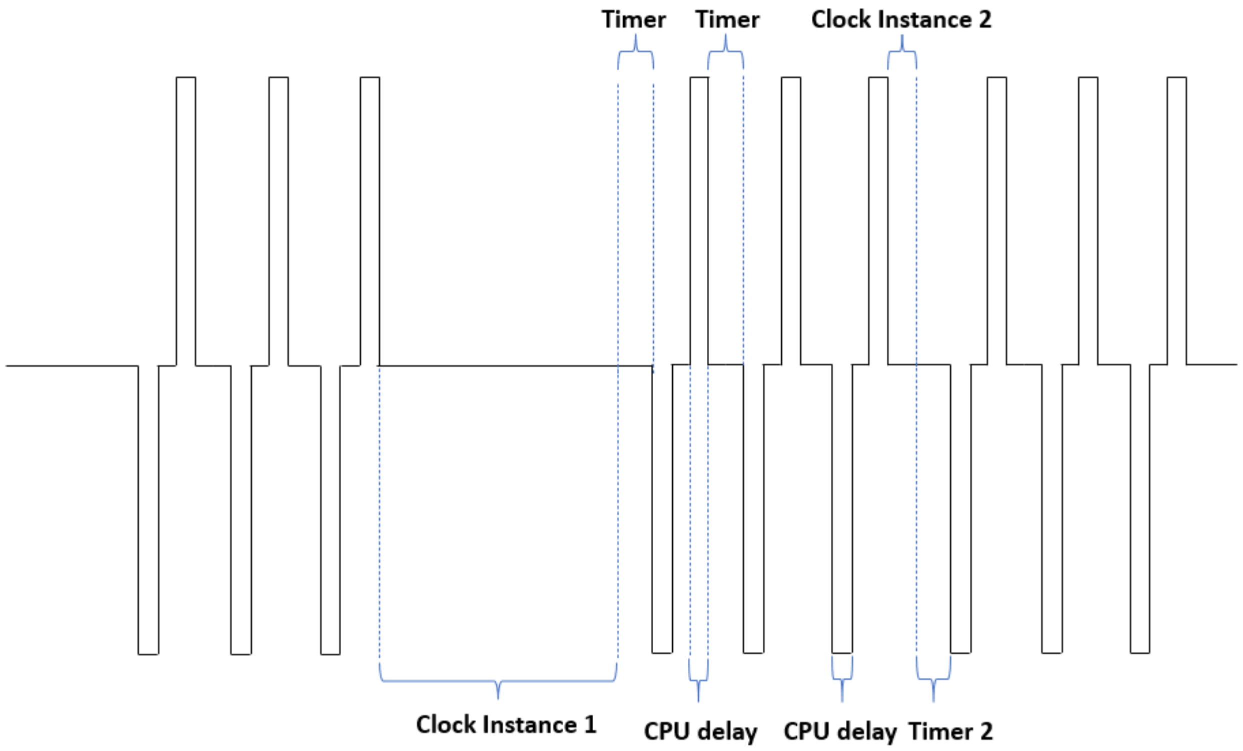


Figure 5-5: Timing of Clock Instance, Timer, and CPU Delay with Aperiodic Stimulation Protocol

The algorithm to generate the above timing event and produce current stimulation is shown in the pseudocode below.

$\boldsymbol{Algorithm}$ **3** *Pseudo Code of Stimulation Task*

```
int burst_count; // User-defined number of pulses in a burst train
int stim_count; // Number of pulses executed in a burst train

int dac_value; // DAC value related to the corresponding DAC voltage

while (1)

    while (stimulation on)

        Initialize a clock instance based on stimulation parameters;

        Wait until the clock instance triggers;

        Initialize a timer based on stimulation parameters;

        while(stim_count <= burst_count)

            if (timer triggers)

                DAC_setVoltage(dac_value);

                Turn on negative phase of SPDT switch;

                CPU delays;

                Turn on negative phase of SPDT switch;

                DAC_setVoltage(0);

            if end

        while end

        if  (aperiodic mode)

            Calculate and update the randomized parameters for the next burst train;
        end if
        Deconstruct timer and clock instance;
    while end
while end
```

5.1.3. Recording Task

The recording task is a collaboration between the ARM Cortex-M4 main CPU and a RISC (Reduced Instruction Set Computer) sensor controller. For one, the sensor controller configures the recording ADC, read and store sampled data into a buffer in a 4-KB SRAM (Static random-access memory) that can be accessed by both the sensor controller and the main CPU. For another, the main CPU monitors the volume of buffered data and dispatches buffered data to the adapter through BLE in a timely manner to avoid buffer overflow. The pseudocode below describes the workflow of sensor controller in the recording task.

***Algorıthm* 4** *Pseudo Code for the Workflow of Sensor Controller*

```
Reset ADC;
Set CS (chip select) pin to high and start SPI communication;
Send op-code to set command format to 16-bit;
Configure sampling frequency;
Set the ADC to low-power mode;
int response; // Used to read command response from ADC
int samples[1464]; // Used to buffer ADC samples
int head = 0, tail = 0; // Keep track of the unsent data in buffer
while (response != 0x3333)
    write op-code to set gain to 16;
    response = read_response();
end while
set interrupt data-ready GPIO to trigger ADC read event;
while (recording on)
    if (data-ready interrupt triggers)
        samples[head] = read_response();
        if (head == 1463) // Wrap buffer around when the end of the buffer is reached
            head = 0;
        end if
    end if
end while
```

Figure 5-6 below illustrates how incoming ADC sampled data is accumulated in the buffer in SRAM along with the increment of variable head to keep track of the volume of buffered data. Blocks in green represents buffered data that awaits to be forwarded to adapter.

Each block in the buffer contains 244 bytes of data and 12 blocks contain 2928 bytes in total. Buffer block size is chosen based on the length of BLE data packet, which can carry maximum 249 bytes. However, in the 2nd iteration PCB, the small chip antenna for wireless communication soldered by vendor was broken and hand soldering was performed to replace the broken antenna. Due to the limitation of hand soldering, it was noticed that the last 2 to 3 bytes of a 249-byte BLE packet tend to contain error. Therefore, a buffer block size of 244 is used to accommodate the actual size of BLE packet used in practice. In this buffer, every new neural signal sample is stored at the address indexed by the current variable head which is incremented by 2 since only 16 noise-free bits out of 24-bit resolution is buffered and the least significant 8 bits are discarded.

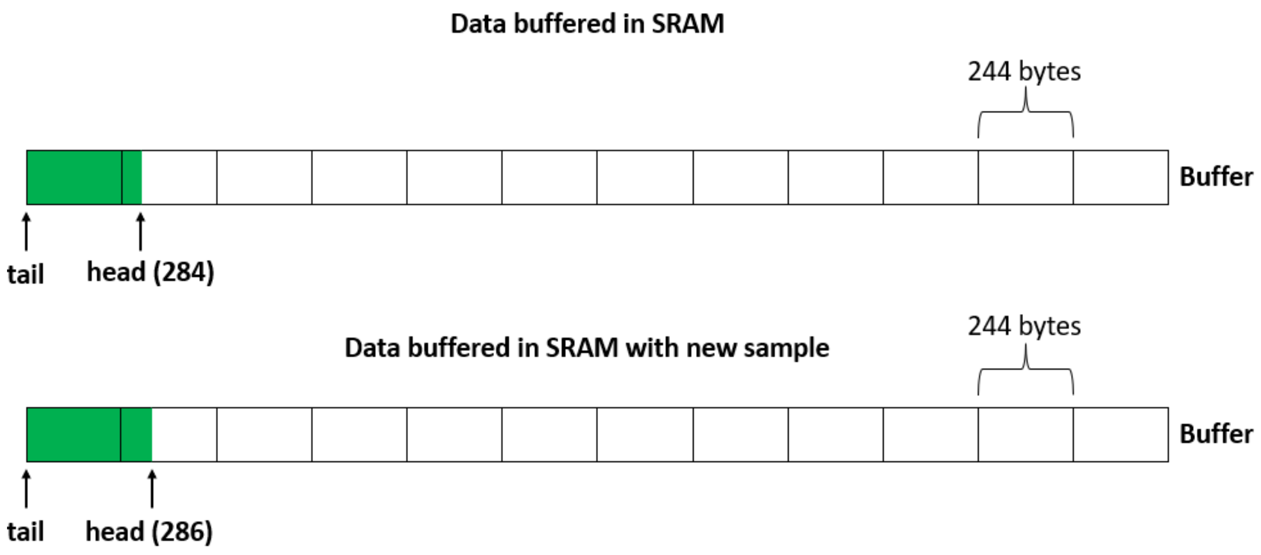


Figure 5-6: Illustration of Data Accumulates into Buffer in SRAM

The workflow of the main CPU to work with the sensor controller to dispatch buffered data is described in the pseudocode below.

***Algorithm* 5** *Pseudo Code for the Workflow of Main CPU in Recording Task*

```
// Main CPU can access variables initialized in SRAM (head, tail, samples)
int chunksLeft = 0;
attHandleValueNoti_t noti; // Handle for send BLE notification
while (recording on)
    allocate memory and configure handle class object for BLE notification;
    notif.len = 249; // Set length for BLE notification handle
    int sampleCount = head – tail;
    chunksLeft = int(sampleCount / 122);
    while (chunksLeft --)
        for (int n = 0; n < 122; ++n)
            noti.pValue[2*n] = samples[tail] >> 8; // Load the first byte
            notif.pValue[2*n+1] = samples[tail] & 0x00ff; // Load the second byte
            tail = tail + 1;
            if (tail >= 1464) // Set tail to 0 if the end of buffer is reached
                tail = 0;
            end if
            while (1) // Send BLE notification until the communication is successful
                status = send_notification();
                if (status == SUCCESS)
                    break;
                end if
            end while
        end for
    end while
end while
```

Figure 5-7 below illustrates the scenario of the buffer when 244 bytes of samples are sent to the adapter by the main CPU along with the increment of tail to keep track of unsent buffered samples. When the counter condition is met, the first 244 bytes of unsent buffered data would be transferred to adapter through BLE, followed by the variable tail incremented by 244.

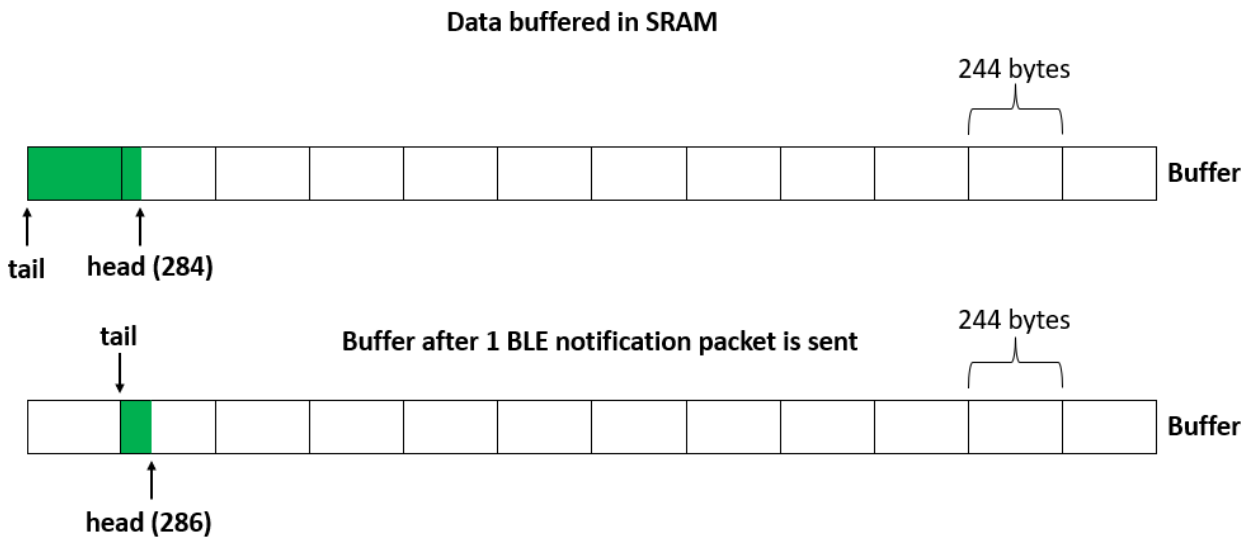


Figure 5-7: Illustration of Buffer in SRAM after 1 BLE Notification Packet is Sent

5.1.4. BLE Task

In addition to maintaining BLE connection and dispatching buffered data to the adapter, BLE task monitors the incoming operation and parameter update command and buffers the new parameters, waits to execute stimulation based on new parameters until the receipt of update command. Upon the receipt of new stimulation parameters, BLE task decodes and reassembles the incoming BLE notification packets back to the double-precision floating-point format, described in Figure 5-2 and algorithm 2. The management of different operation and stimulation parameter of the implanted/remote device is based on GATT and the assignments of characteristic attributes are summarized in table 3-1. The format of operation and parameter update command will be discussed in the next chapter.

5.2. Firmware Design of the Adapter

5.2.1. Overview of Firmware Design of Adapter

An overview of the firmware workflow of the adapter is shown in Figure 5-8 below. Similar to the firmware design of the remote device, TI-RTOS is used to manage multiple tasks and guarantee the real-time operation. Three tasks operate concurrently in the adapter: data forward task, BLE speed-drop prevention task, and stimulation parameter update task. Data forward task tracks receives and buffers the recorded neural signal recording from the remote device through BLE. Once the buffer is filled up to a certain level, buffered data will be forwarded to the backend device through UART for long-term storage, real-time visualization, and real-time post-processing. Parameter update task monitors stimulation parameter update command from the backend device via UART, which would be transferred to remote device via BLE. The format of the parameter update command would be discussed later in detail. A BLE speed-drop prevention task is added to the system to avoid potential BLE notification speed drop due to the overload of SRAM of the RF core. This task keeps track of the operation duration of BLE notification and restart BLE notification every 40 seconds with a small pause of 25 millisecond to accommodate the transient of turn-on and turn-off operation. Periodicity to restart BLE notification is chosen experimentally without generating data loss. The priorities of three tasks in descending order are the following: BLE speed-drop prevent task, data forward task, parameter update task. Such setting prioritizes data transmission in the adapter to avoid data loss.

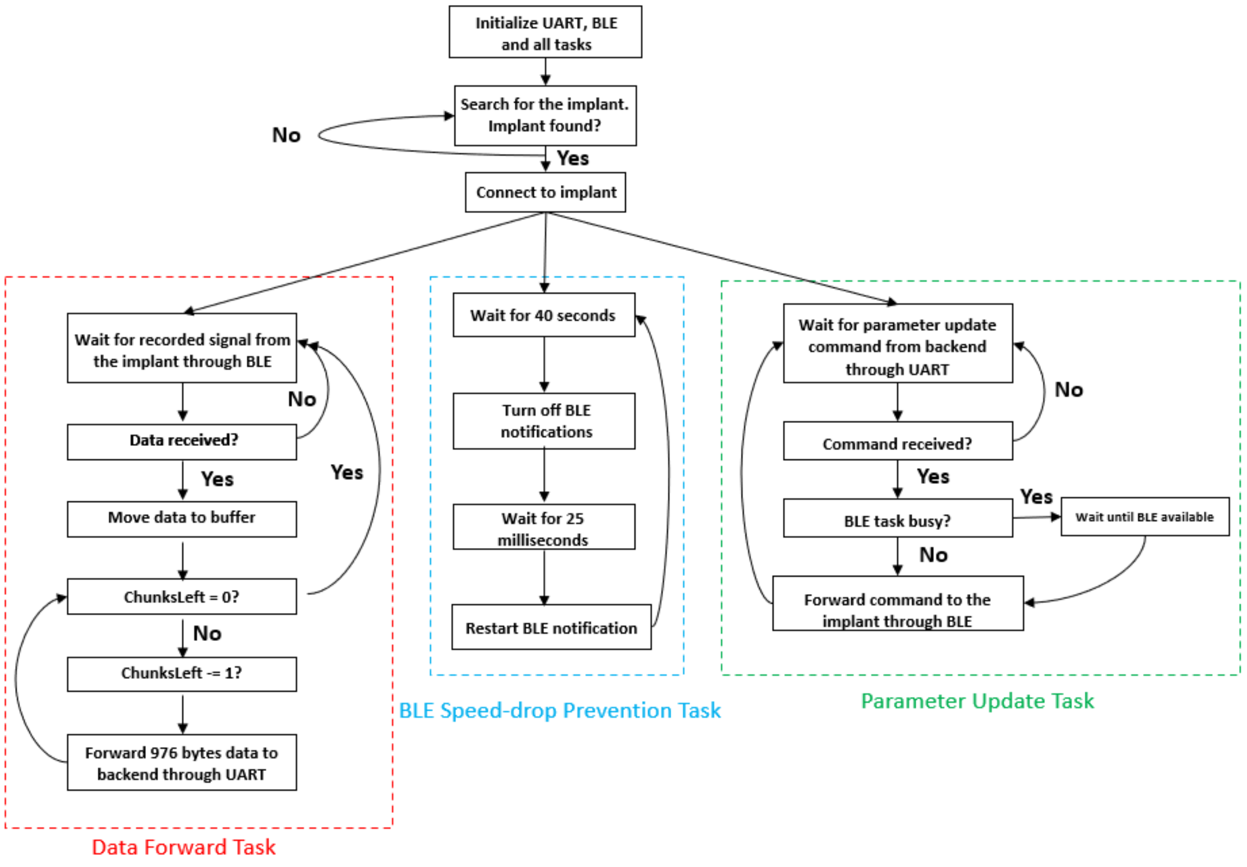

Initialize UART, BLE and all tasks
Search for the implant. Implant found?
No
Yes
Connect to implant
Wait for recorded signal from the implant through BLE
No
Data received?
Yes
Yes
Move data to buffer
ChunksLeft = 0?
No
ChunksLeft -= 1?
Forward 976 bytes data to backend through UART
Data Forward Task
Wait for 40 seconds
Turn off BLE notifications
Wait for 25 milliseconds
Restart BLE notification
BLE Speed-drop Prevention Task
Wait for parameter update command from backend through UART
No
Command received?
Yes
BLE task busy?
Yes
Wait until BLE available
No
Forward command to the implant through BLE
Parameter Update Task

Figure 5-8: Firmware Flowchart of Adapter

The details of firmware in adapter will be discussed in the following. When the adapter powers on, UART, BLE, and 3 tasks mentioned above are initialized. The adapter serves as a central device in BLE protocol and determines to either accept or reject the requested connection parameters from the peripheral devices. PHY is set to 1M to allow stronger signal strength for better signal penetration ability. Baud rate of UART is set to 961200 to provide sufficient redundancy to transmit buffered data in time without data overflow. The size of the buffer is set to 3904 bytes, which is 16 transmission units of BLE notification. Data forward task sends received data when the buffer is filled up to 1/4 of the total size, i.e., 976 bytes, which is 4 transmission units of BLE notification. The stack size for data forward task is set to 10240 bytes. The stack size is set to 1024 bytes for the other 2 tasks.

After initialization, the adapter starts searching for peripheral device, i.e., implanted/remote device to establish connection. As mentioned in section 5.1.1, the implant uses public address mode and has a global fixed address. The adapter will continuously search for peripheral with this global fixed address and establish connection when it is available.

5.2.2. Data Forward Task

Pseudocode describes the operation of data forward task is shown below.

***Algorithm* 5** *Pseudo Code for Data Forward Task in Adapter*

```
counter = 0;
while (recording on)
    Data forward task sleeps for 10 ms; // See consideration of 10ms sleep below
    counter = int ((head – tail)/976);
    while (counter != 0){
        UART_write(buffered_data[tail:tail+976]);
        tail = tail + 976;
        counter --;
        if (tail > 3904)
            tail = 0;
        end if
    end while
end while
```

Data forward task is set to a 10ms sleep when there is not enough data to forward in the buffer. During 10ms sleep period, about 160 samples (320 bytes) are accumulated in the buffer. Data forward task forwards recorded data when the buffer has more than 976 bytes of data. A sleep period of 10ms guarantees data is sent out faster than it enters buffer. The variables head and tail are the indices used to locate unsent data in the buffer. Note that in the pseudocode above, only variable tail is updated. The variable head is updated by BLE stack continuously when new data is received through BLE notification. Pseudocode shown below describes how the variable tail is updated along with the incoming buffered data in BLE stack.

***Algorithm* 6** *Pseudo Code for Data Buffering in BLE Stack*

```
while (1)
```

```
    if (A new BLE notification packet is received)
        memcpy(buffer_data[head], BLE_notification_packet, 244);
        head = head + 244;
        if (head >= 3904)
            head = 0;
        end if
    end if
end while
```

Figure 5-9 below illustrates how incoming BLE notification packet is buffered in BLE stack along with the update of variable head. Blocks in green represents buffered data that awaits to be forwarded to backend. Each block in the buffer contains 244 bytes of data and 16 blocks contain 3904 bytes in total. New BLE notification packet received by the adapter is stored at the address indexed by the current variable head which is incremented by 244 accordingly afterwards.

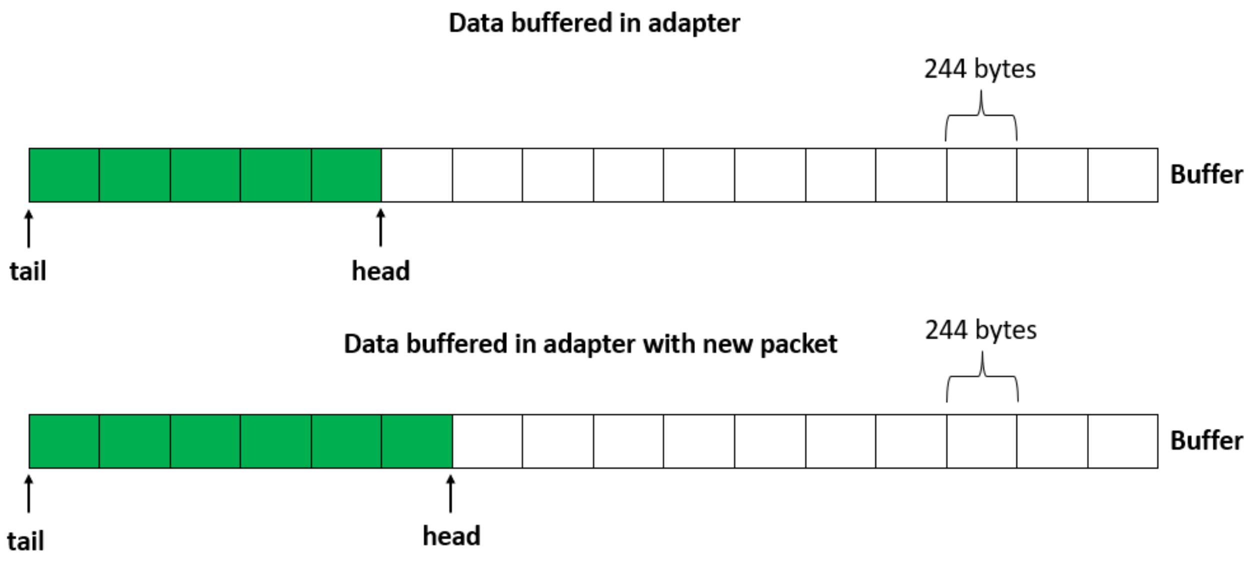


Figure 5-9: Illustration of Data Accumulates in Buffer in Adapter

Figure 5-10 below illustrates how the variables tail keeps track of buffered data in data forward task. When the counter condition is met, the first 976 bytes of buffered data would be transferred to backend through UART, followed by the variable tail incremented by 976.

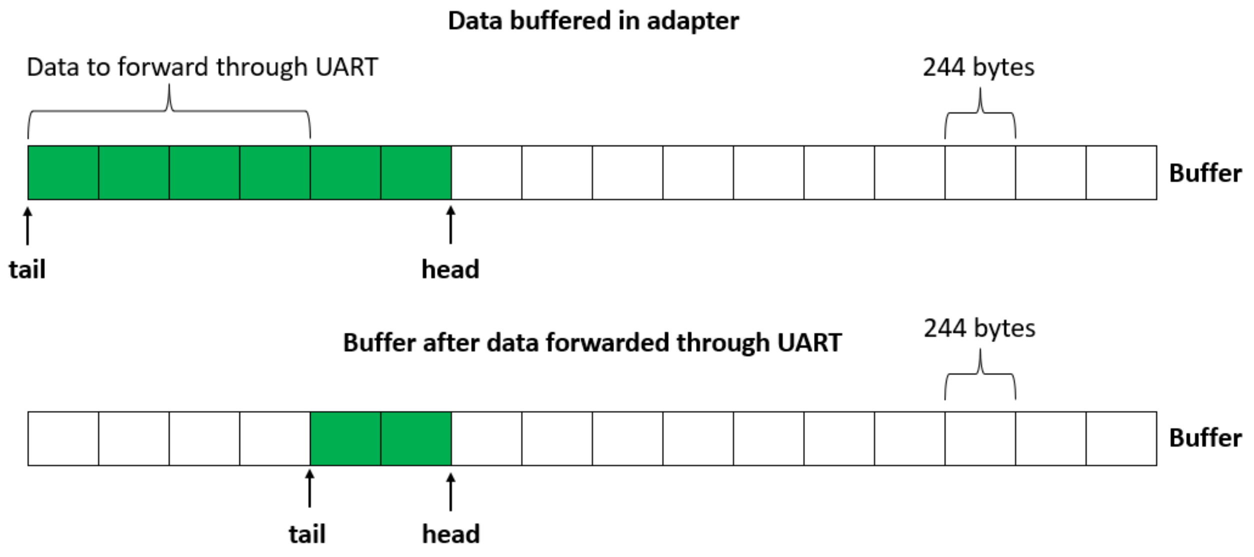


Figure 5-10: Illustration of Buffer When Data is Forwarded to Backend

When a new BLE notification is received by the adapter, the last 5 bytes would be dropped as they are trivial data for the purpose of avoiding bit errors at the end of a packet discussed in Chapter 5.1.3, and the first 244 bytes would be moved to a buffer of which the total size is 3904 bytes. As recorded data transferred through BLE gets continuously buffered in adapter, the data forward task monitors the accumulation of data by integer division between the number of bytes in the buffer and 976, the result of which gets stored in a counter. If the value of counter is not zero, the adapter forwards 976 bytes to the backend through UART and decrements the counter by 1. Such process continuous repeatedly until the recording turns off by the user.

5.2.3. BLE Speed-drop Prevention Task

BLE speed-drop prevention task simply turns off BLE notification every 40 seconds and restarts BLE notification after a 25ms delay. In experiments, it was noticed that continuous BLE notification over 2 minutes will result in speed drop. Also, restarting BLE notification less than 15ms after the termination is not successful due to the transience between on and off state. A short delay allows BLE notification to fully rest in off state. Therefore, 40 seconds on time and 25ms delay are chosen in practice. 25ms delay would not cause data loss due to buffer overflow with 15.625kHz sampling frequency and 16 noise-free bit data format, which is proved in the calculation below.

$$Data\ generated\ in\ 25\ milliseconds = \frac{Samping\ Frequency * \#\ bits}{8\ bits} * 0.025$$

$$= \frac{15625\ \ * 16}{8} * 0.025 = 781.25\ bytes < 3904\ bytes\ (buffer\ size)$$

5.2.4. Parameter Update Task

Parameter update task stores the parameter update command in a class object with a custom format. The received command is temporarily stored in the adapter in a custom format illustrated in Figure 5-11 below.

Command ID	**Data Length**	**data_array[0]**	**data_array[1]**	**data_array[2]**

Figure 5-11: UART Command Data Format.

Each block presents 1 byte. The description of each field is the following:

1. Command ID: Determined by the type of parameter to update. Every configurable parameter is associated with an ID from 0 to 14. This is the same as the BLE characteristic ID when the parameter is sent through BLE write command. Refer to table 3-3-1 for the corresponding parameter to command ID.
2. Data Length: Data length indicates the length of the attribute value of the associated command ID. Note that the data length byte is not included in the raw command received through UART. It is added to the class object based on the associated command ID for the convenience for initiating a BLE write command.

3. data_array: data_array is an unsigned 8-bit integer array. Following the same format of attribute value defined in the Section 5.2, data_array can represent attribute values from 0 to 999999 by segmenting the attribute value into 3 bytes. For instance, an attribute value of 501230 is represented as [0x50, 0x12, 0x30]. For attributes with length smaller than 3 bytes, zeros are padded to the right side of data_array. For instance, an attribute value of 2580 is represented as [0x25, 0x80, 0x00].

The pseudocode below describes the workflow to transfers a received parameter update command from UART to the remote device through BLE.

***Algorithm* 3** *PseudoCode for Parameter Update Task*

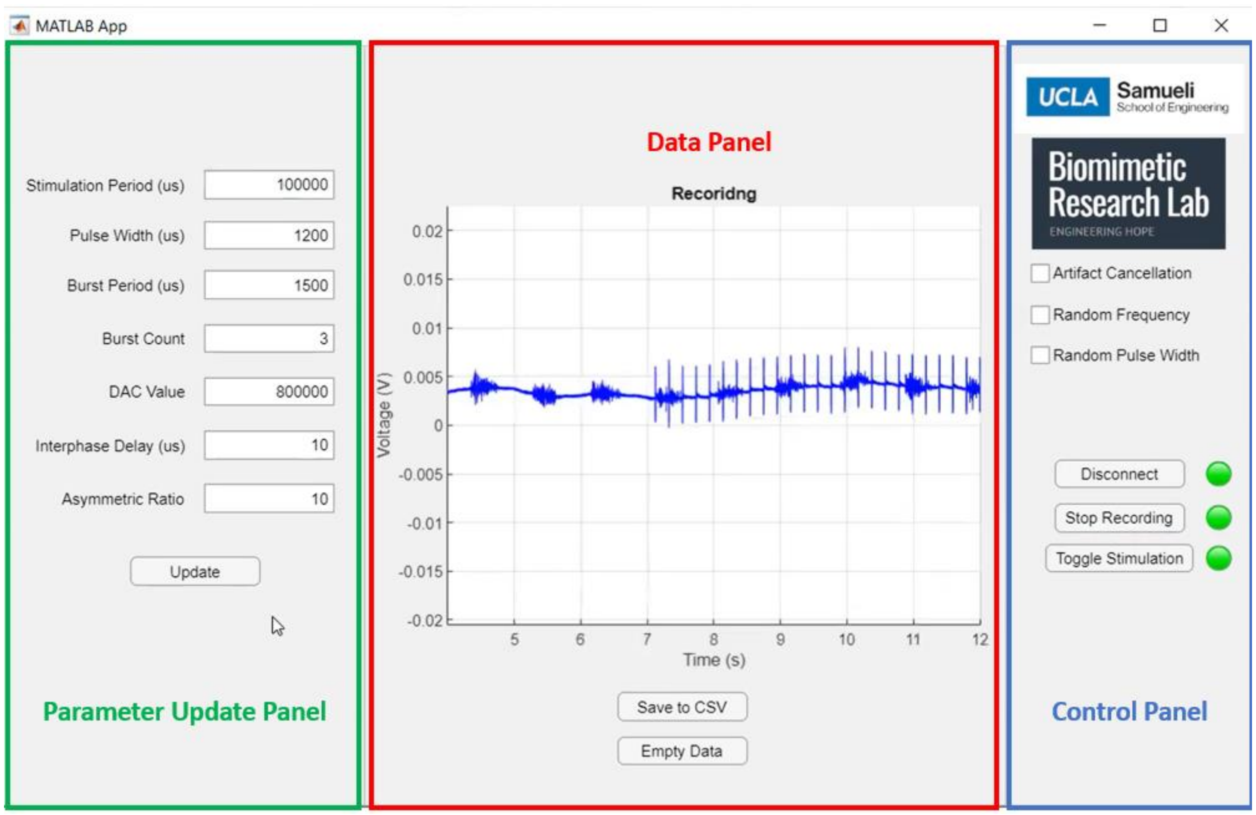
```
typedef struct {
    uint8_t commandID;
    uint8_t len;
    uin8_t pData[3];
}
uint8_t UARTCommand[4] = {0, 0, 0, 0};
while (1)
    Parameter update task sleeps for 10 ms;
    Read 4 bytes data through UART and store in UARTCommand array;
    if (UARTCommand[0] != 20)
        commandHandle.commandID = UARTCommand[0];
        memcpy(&commandHandle.pData, &UARTCommand[1], 3);
        switch (UARTCommand[0])
            commandHandle.len = the associated BLE characteristic attribute length;
            if (UARTCommand[0] == 2)  // Characteristic ID to toggle throughput
                Toggle BLE throughput;
            end if
        end switch
        if (commandHandle.commandID != 2) // Characteristic ID to toggle throughput
            Send BLE write request to update the specific characteristic attribute;
        end if
        UARTCommand[0] = 20;
end while
```

When parameter update task wakes up from a 10ms sleep, adapter attempts to read 4 bytes of data from UART if available. If the attempt succeeded, the attribute length associated to the command ID is determined, followed by transferring all parameter update information to a commandHandle class object. Note that a command ID of 2 is an exception such that BLE throughout is toggle immediately at this step. Next, a BLE write request is initiated based on the commandHandle class object. Finally, the first byte of UARTCommand array is set to a trivial number that is different from all functional command ID, 20 in our implementation, to indicate the completion of a parameter update command to avoid repeated update.

Chapter 6 : Software Design

6.1. Design of GUI (Graphical User Interface)

The GUI used to control the operation of the system, visualize, process, and store the recorded neural signal is built using the App Designer in MATLAB. The interface of the application is shown in Figure 6-1 below.



Stimulation Period (us)	100000
Pulse Width (us)	1200
Burst Period (us)	1500
Burst Count	3
DAC Value	800000
Interphase Delay (us)	10
Asymmetric Ratio	10

☐ Artifact Cancellation
☐ Random Frequency
☐ Random Pulse Width

Figure 6-1: Graphical User Interface

The interface is divided into three panels to improve user experience: parameter update panel, data panel, and control panel. The left panel is used to control stimulation parameters. An update command is sent to the remote device through adapter only when user clicks the update button to enable bulk update. The middle panel displays the recorded signal (or artifact-free signal when artifact cancellation is enabled) in real-time. At the end of a recording session, user can choose to save the recorded data into a .csv file and delete recorded data to start a new recording session. The right panel provides control of the operation of the system. Three buttons provide the major control of the system. Connect/disconnect button initiates/terminates the connection between adapter and backend. Start/stop recording button enables/disables the on/off state of the recording ADC and BLE notification. Toggle stimulation button switches the on/off state of the stimulator. 3 checkboxes on the top determine the operation mode. Artifact cancellation checkbox enables/disables the real-time artifact cancellation capability. Random frequency checkbox enables/disables Poisson-distributed modulation of inter-burst-train periods. Random pulse width checkbox enables/disables Poisson-distributed modulation of stimulation pulse width.

The GUI establishes communication with the adapter through UART after user enabling the connection by connect/disconnect button. When the two-way communication is enabled, the program initializes a serial port to send control and parameter update commands. Additionally, discussed in section 5.1.1, samples of neural signals are sent to backend as a stream of 8-bit numbers. The backend program also reassembles the 8-bit stream to decode ADC conversion data. Since data in ADS131M04 is given in binary two's complement format and the remote device retains 16 noise-free bits out of 24-bit resolution, the equation below can be used to calculate the least significant bit (LSB). The full scale reference of ADS131M04 is 2.4V (-1.2V to 1.2V).

$$1\,LSB = \left({}^{Full\ Scale}\!/\!_{Gain}\right)/2^{16} \ = +FSR/2^{15} \ \ [3]$$

Based on the equation above, ideal output code from negative full-scale reference (-FSR) to positive full-scale reference (+FSR) can be summarized in Table 6-1 below.

Table 6-1: ADC Conversion with Output Code

Input Signal	**Ideal Output Code**
>= FSR $(2^{15} - 1)/ 2^{15}$	0x7FFF
FSR / 2^{15}	0x0001
0	0x0000
- FSR / 2^{15}	0xFFFF
<= -FSR	0x8000

Using rules described above, bit stream from the adapter is decoded to the corresponding voltage level of input signal by UART callback function, which triggers when the amount of data buffered by UART is equivalent to 1-second interval of ADC samples, i.e., 15625 ADC samples at 15.625 kHz sampling frequency. The callback function operates differently depending on whether the artifact cancellation capability is enabled. When artifact cancellation is disabled, the pseudocode below describes the workflow to offload and display incoming data in real-time.

***Algorithm* 8** *Pseudo Code to Store and Display Neural Data*

```
data = []; // Array to store decoded ADC recordings
while (artifact cancellation disabled)
    if (buffered data size in UART > 31250 bytes)
        data_temp = read_UART(31250, ‘uint8’); // Read 31250 bytes in UART buffer
        adcConversion = zeros(1, 15625);
        **for** i = 1:15625
            temp = uint8(data_temp[(i-1)*2+1:i*2]); // Take 2 bytes that form 1 sample
            adcConversion[i] = typecast(temp, “uint16”); // Decode 1 sample by combining 2 bytes of 8-bit number
            **if** (adcConversion[i] > 32768)
                adcConversion[i] = adcConversion[i] – 65536; // Account for binary two’s complement format
            **end if**
            adcConversion = (2.4/gain)/$2^{16}$*adcConversion;// Convert to input voltage
            data = [data adcConversion]; // Append the most recent frame of decoded input voltage to the current recording session
    **end if**
**end while**
```

When artifact cancellation is enabled, real-time algorithm to remove stimulation artifact is added to Algorithm 8. Template-subtraction-based algorithm is applied to the decoded input voltage, which will be discussed in Section 6.2 below.

UART communication between backend and adapter is also responsible for delivering control and parameter update commands to adapter. When edit fields, buttons, and check boxes are utilized, their corresponding callback function write control or parameter update command as bit stream through UART to alter the behavior of the remote device. The format of control and parameter update command initiated from backend is illustrated in Figure 6-2 below.

Command ID	data_array[0]	data_array[1]	data_array[2]

Figure 6-2: Format of Commands in Backend

Each block presents 1 byte. The description of each field is the following:

1. Command ID: Determined by the type of parameter to update. This is the same as the command ID field in Section 5.2.4.
2. data_array: data_array is an unsigned 8-bit integer array and follows the same rule defined in Section 5.2.4. Additionally, details of data_array is determined by the type of commands. For control command, data_array[1] and data_array[2] are set to zero. data_array[0] is set to binary true and false depending on the desired on/off state. Refer to Figure 5-2 for the workflow of parameter update command and algorithm 1 for the data format.

6.2. Method to Remove Stimulus Artifact in Real-time

The real-time artifact cancellation capability of the system is realized by a template-subtraction-based algorithm. Such an algorithm takes advantages of the similar voltage response from periodic stimulation to form an averaged template which is subtracted from the raw input signal with appropriate timing. Figure 6-3 illustrates the conception templating to deal withstimulation artifact in periodic stimulation where the periodic stimulus responses can be overlaid to form a template. The left panel shows a segment of recorded neural signal with stimulation artifact. The right panel shows ten periods of stimulation artifact from the left panel overlaid together. It is noticeable that ten adjacent artifact waveforms are similar in shape and amplitude. Such behavior is beneficial to establish a artifact template which can be subtracted from the raw input signal to remove stimulus artifact.

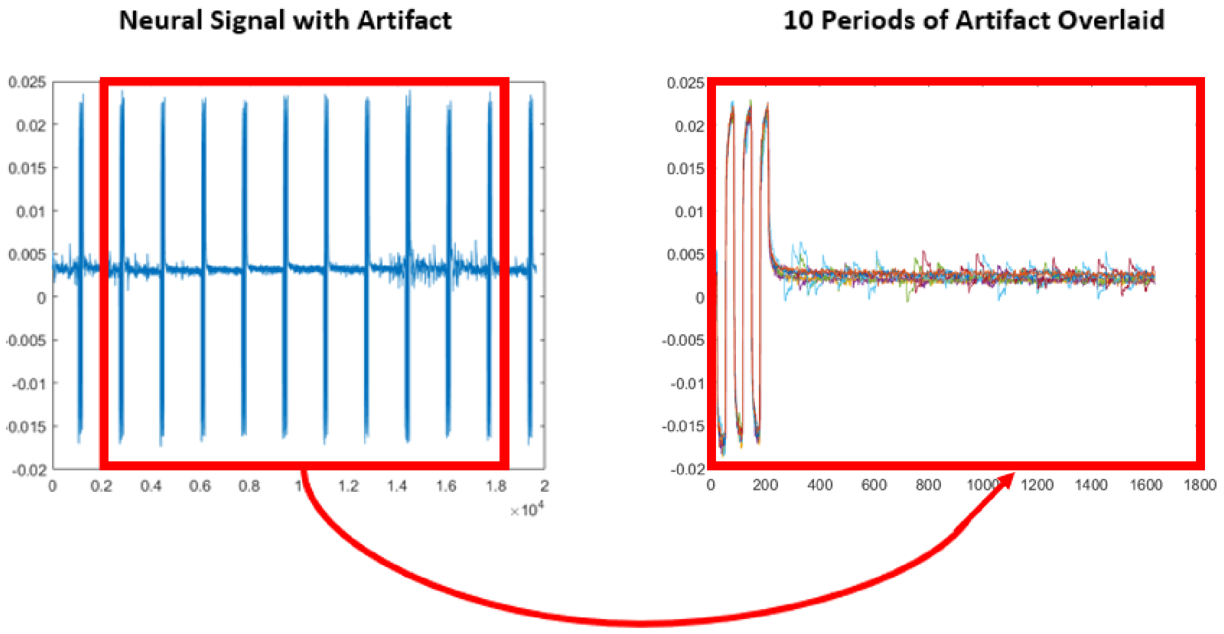


Figure 6-3: Concept of Template Averaging

The workflow of the algorithm is discussed in the following:

1. As mentioned in Section 6.1, UART callback function reads data through UART when there is equivalently 1-second interval of buffered data. Therefore, artifact cancellation

algorithm in the system operates on a window basis as well. Figure 6-4 demonstrates a window of raw input signal with stimulus artifact present. Note that the raw input signal experiences noticeable DC drifting, which introduces significant errors when forming and subtracting artifact template. To mitigate this issue, the algorithm appends the last 0.256 seconds from the previous window to the current 1-second window to form a 1.256-second window, which provides redundancy for the convenience to eliminate DC drift in step 2 and 3. The window size 1.256 is empirically determined and will be explained later.

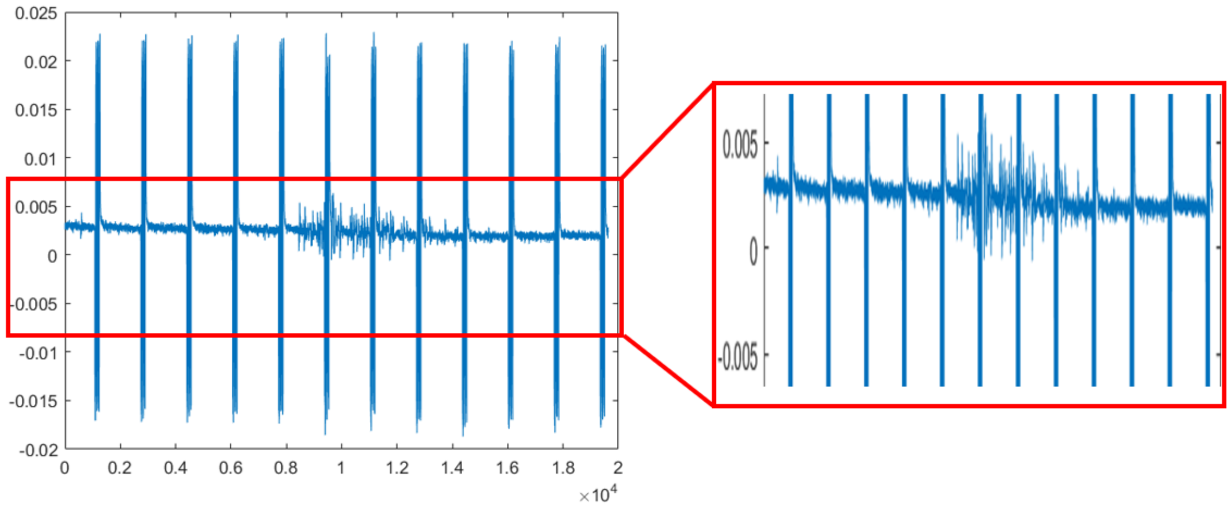


Figure 6-4: Window with Redundancy for DC Drifting

2. With 1.256-second window in step 1, a 2nd-order high-pass Infinite Impulse Response (IIR) filter with cutoff frequency at 5Hz is used to remove low-frequency DC drifting. Figure 6-5 below shows the resulting waveform after filtering. Notice that IIR filter does not have linear phase and causes phase distortion. The resulting signal after passing

through IIR filter is noticeably distorted in the front of the window. This situation is undesired and needs to be accounted for.

3. As mentioned previously, the data panel of GUI displays the signal at a 1-second interval while the initial window is 1.256-second with 0.256-second redundancy from the previous window. Window size of 1.256 second is determined empirically. During experiments, the IIR filter mentioned previously introduces distortion in less than 0.2 second of data. A redundancy of 0.256 second performs well to remove those distortion in practice. With this pre-allocated redundancy, simple truncation can be done to remove the first 0.256 seconds of samples containing significant distortion. The resulting 1-second DC-free signal is shown in Figure 6-6 below.

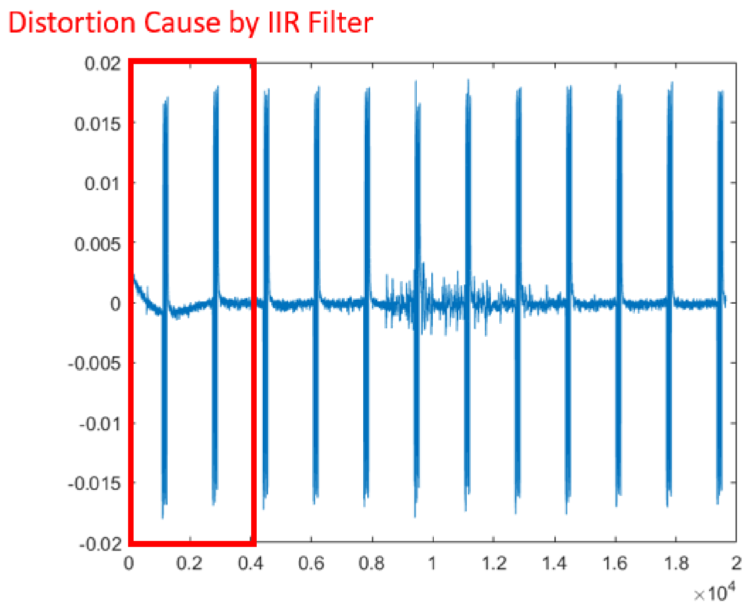


Figure 6-5: Resulting Waveform from IIR High-pass Filtering

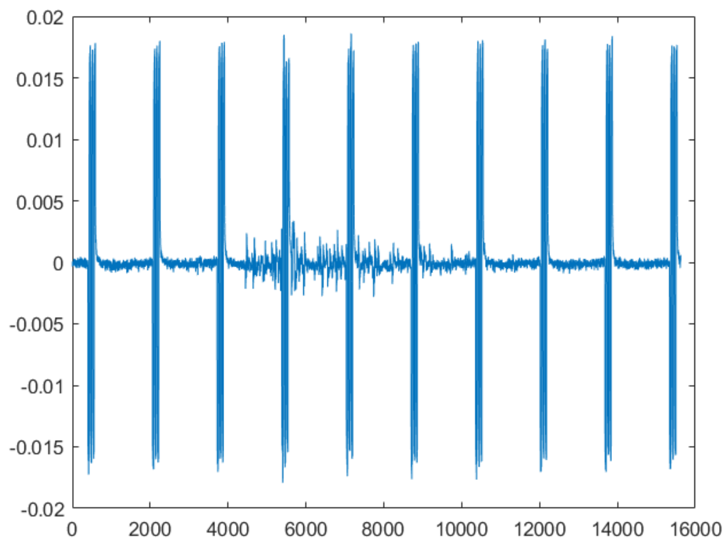


Figure 6-6: DC-free Signal after Filtering and Truncation

4. After a DC-free 1-second window is obtained, location of the starting point of stimulus artifacts needs to be determined using signal processing method due to the lack of trigger signal through wireless communication. In general, the amplitude of stimulus artifacts is much larger than that of meaningful neural signal. Such property can be leveraged to design a magnitude-based method to segment stimulus artifact into different periods. The histogram of a typical 1-second windowed DC-free signal is shown in Figure 6-7.

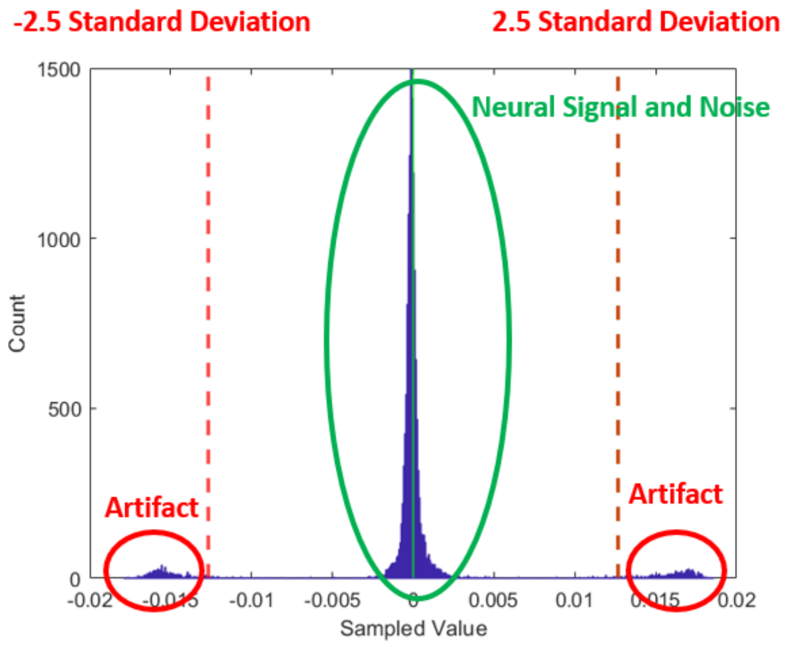


Figure 6-7: Distribution of DC-free Signal in Figure 6-6

Clearly, large-amplitude samples originated from stimulus artifact are greater than 2.5 standard deviation from the mean of all samples. Hence, a threshold calculated within the current windowed DC-free signal can be applied to detect the starting point of a stimulation pulse train. Using such observation, the proposed algorithm selects the first point that is greater than the established threshold in a stimulation period as the anchor point to locate the beginning of a stimulation period. Ideally, an adaptive threshold based on stimulation parameters (amplitude, frequency, pulse width, etc.) may provide more benefit to localize stimulus artifact more precisely. In practice, a threshold between 2 to 3 standard deviations regardless stimulation parameters produces decent performance. Moreover, as a part of future work, such template-subtraction-based algorithm will be migrated to on-chip execution within CC2652RSIP so that the need to localize stimulus artifact statistically is eliminated. Red dots in Figure 6-8 below indicates the anchor point to perform template averaging using the aforementioned statistical method.

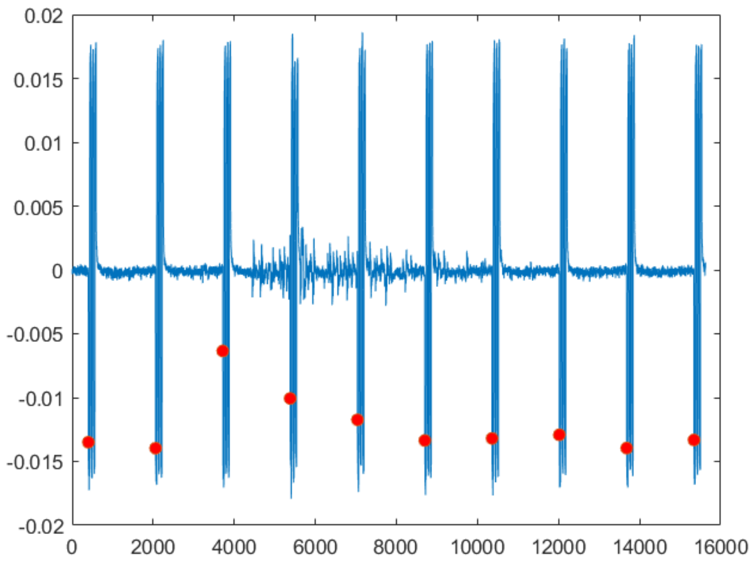


Figure 6-8: Anchor Points to Perform Template Averaging

A potential issue needs to consider when detecting anchor points. First, stimulus artifacts can appear at the edge of the 1-second interval. Thus, those artifacts may not be complete in the current window and run-time error of the algorithm could occur. Such scenario is demonstrated in Figure 6-9 below. The solution to avoid this situation is to apply the scheme of adaptive window width, i.e., shrink or extend the window width when necessary to guarantee all detected stimulus artifact are enclosed in the current window width.

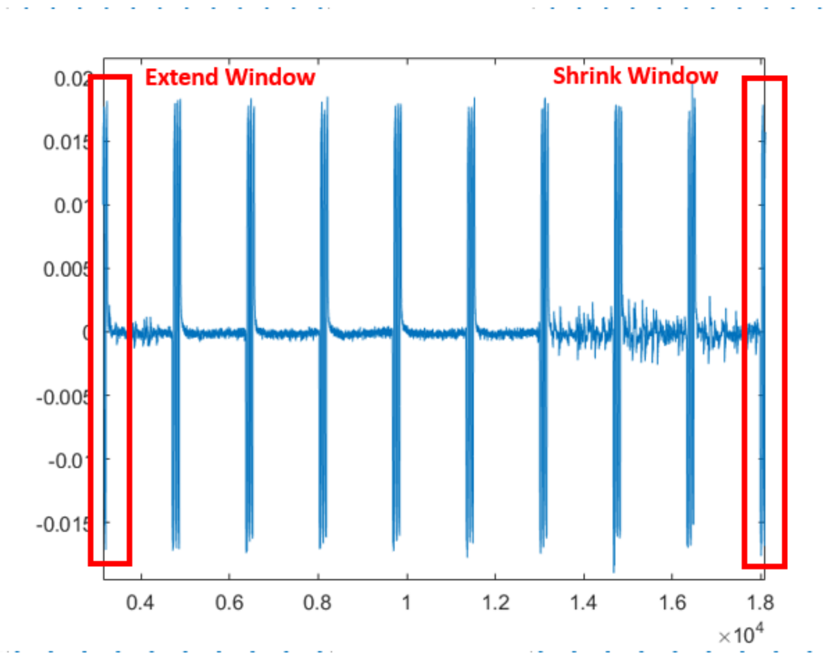


Figure 6-9: Scenario of Stimulus Artifacts Appearing at the Edge of 1-second Window

5. With the anchor points detected in step 4, stimulus artifacts are segmented into different periods. A template of the stimulus artifacts can be formed by averaging the overlaid artifact segments as shown in Figure 6-10.

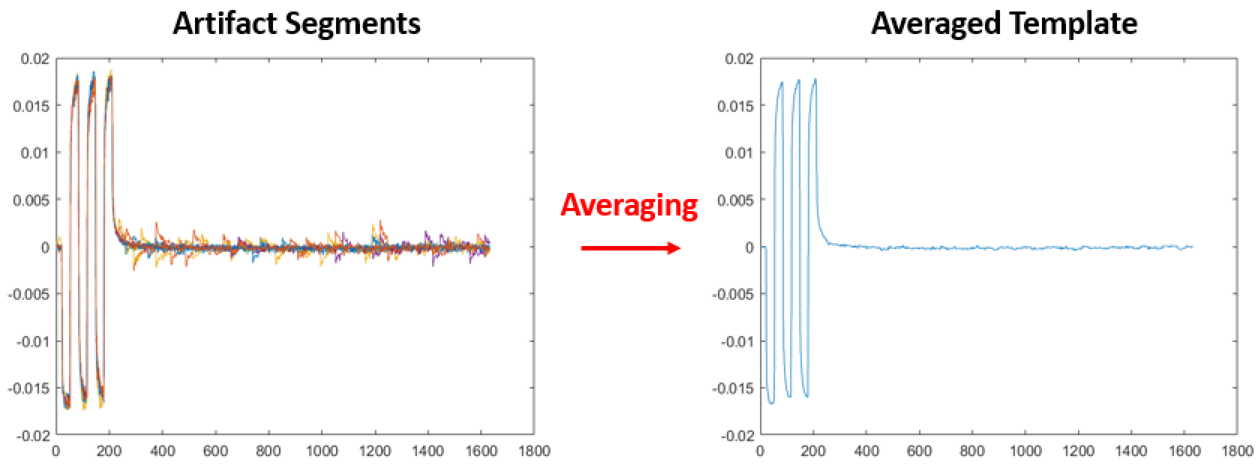


Figure 6-10: Template Averaging

The number of artifact segments used to for a template can be pre-determined or ideally, determined by the residuals resulting after subtraction. In practice, updating the template over time using 10 to 20 segments provides satisfactory performance.

6. The template obtained in step 5 is subtracted from the DC-free 1-second window, i.e., signal obtained in step 3, at each anchor point obtained in step. Template subtraction technique tends to suffer from under-sampling, misalignment of stimulation, and sample timing. A typical scenario of misalignment is shown in Figure 6-11 below where the template and artifact in DC-free signal are offset by 1 sample, which leads to exacerbated decline in performance of the algorithm. One solution applied in the algorithm is to perform subtractions referring the anchor points as well as indices adjacent to anchor points and calculate the sum of the residuals with respect to different referenced points. The referenced point with the smallest sum of residuals is selected to be the adjusted anchor point for final subtraction. The result after this process can be seen in Figure 6-12.

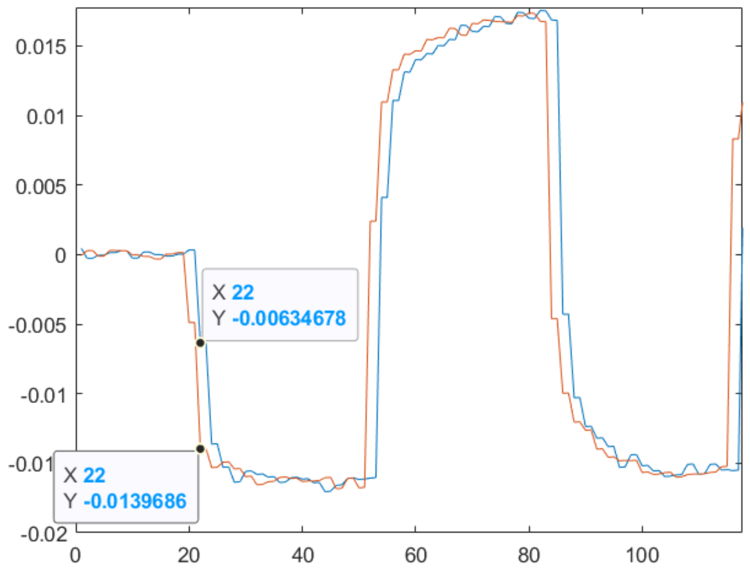


Figure 6-11: Misalignment between Artifact and Template

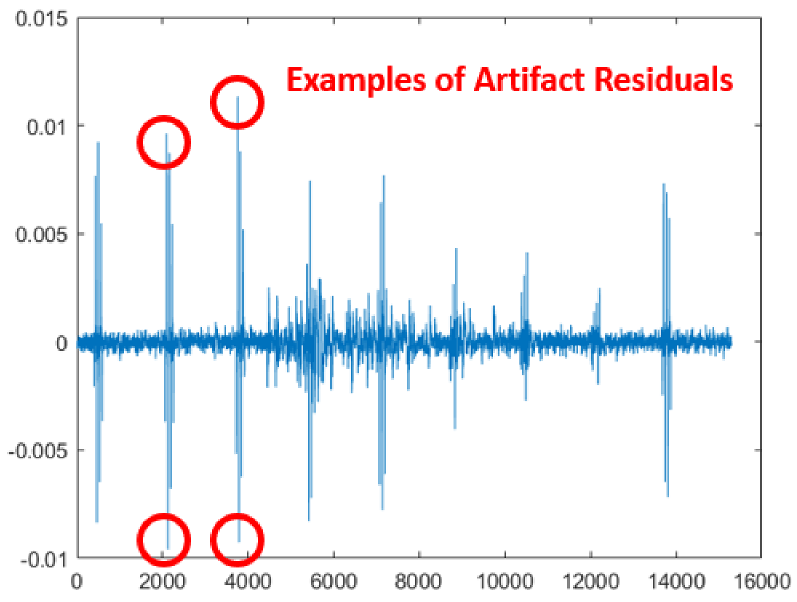


Figure 6-12: Result after Template Subtraction

7. After template subtraction with misalignment attenuation, noticeable residual artifacts exist in various amplitude. Figure 6-13 below provides a close view of a typical scenario of artifact residuals. Generally, residual artifacts appear in the form of narrow pulse with amplitude drastically higher than the adjacent neural signals or noise floor. Due to the resemblance to outlier data points, residual artifacts can be removed using a Hampel filter in combination with a bandpass filter for out-of-band noise to recover artifact-free signal. In the current implementation, Hampel filter is set to 200 adjacent points with 2 standard deviations. The resulting artifact-free signal in comparison with the raw input signal is shown in Figure 6-14. In this qualitative comparison, the post-processing signal retains the shape and phase features from the raw input signal with significant attenuation of

stimulus artifacts. More discussions with qualitative and quantitative analysis on the performance of the algorithm will be included in Chapter 7.

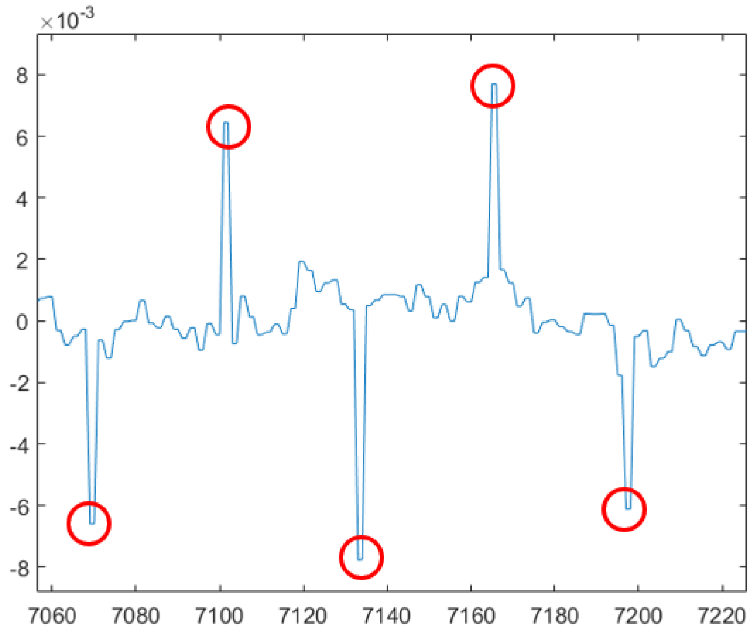


Figure 6-13: Magnified View of Typical Residual Artifacts

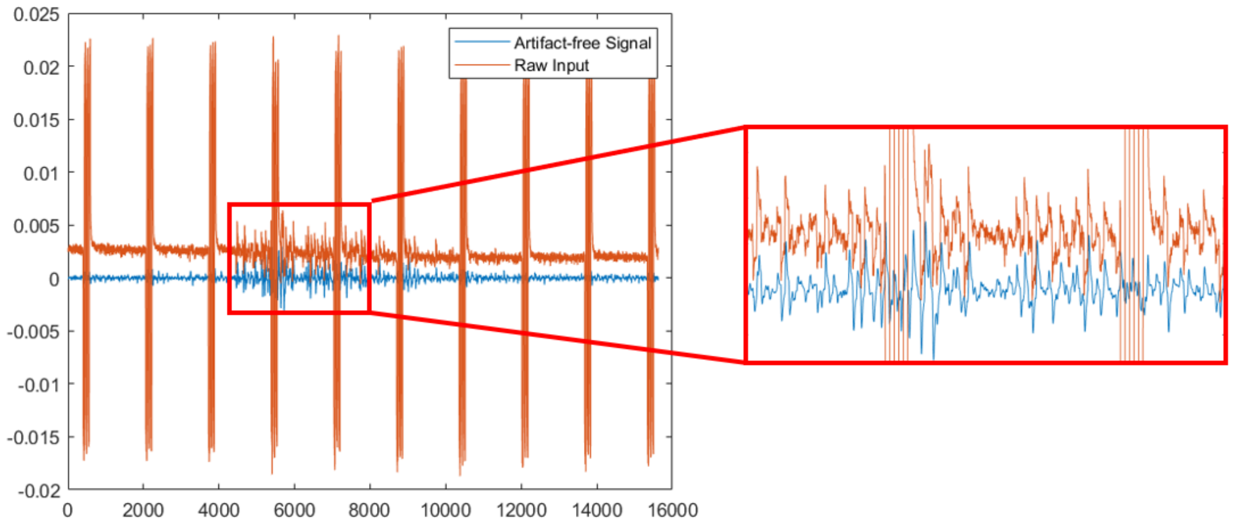


Figure 6-14: Comparison between Artifact-free Signal and Raw Input Signal

Chapter 7 : Testing of the System

7.1. Testing of the Stimulator

7.1.1. Overview of Testing Methods

The functionality of the stimulator with the associated firmware to generate regulated stimulation waveform is verified with resistive load, saline, and in-vivo experiments. While resistive load does not match the property of real issue, a resistive load provides non-capacitive voltage response to current stimulus. Such voltage response contains negligible transient, which is ideal to assess the timing accuracy and voltage amplitude of the stimulator with the associated firmware with respect to stimulation parameters. The testing of the stimulator during in-vivo experiments will be discussed in Chapter 7.4.

7.1.2. Testing with Resistive Load

The setup of testing with resistive load is shown in Figure 7-1 below. The remote device is powered by a battery and executes stimulation to two 450Ω resistors in series. An oscilloscope is used to measure the voltage response against different stimulation parameters and stimulation mode. Figures 7-1 to 7-3 below are voltage response under periodic stimulation mode with stimulation parameters summarized in Table 7-1. Stimulation current probed by oscilloscope is 0.885V/(2×450) Ω = 0.983 mA.

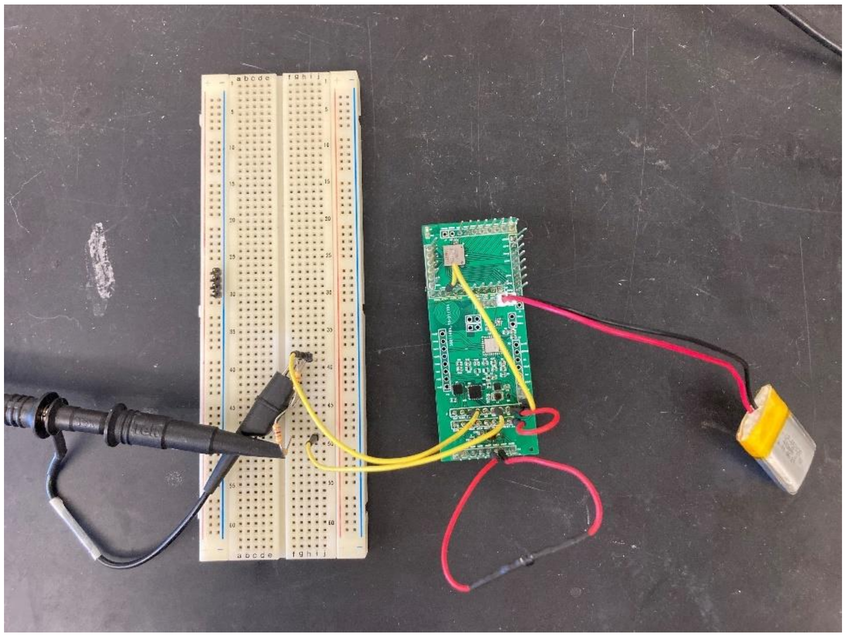


Figure 7-1: Setup of Testing with Resistive Load

Table 7-1: Stimulation Parameters of Periodic Stimulation with Resistive Load (Case 1)

Parameters	Value
Stimulation Period	100 ms
Pulse Width	100 µs
Burst Period	1 ms
Burst Count	4
Current	1 mA
Interphase Delay	10 µs
Asymmetric Ratio	1

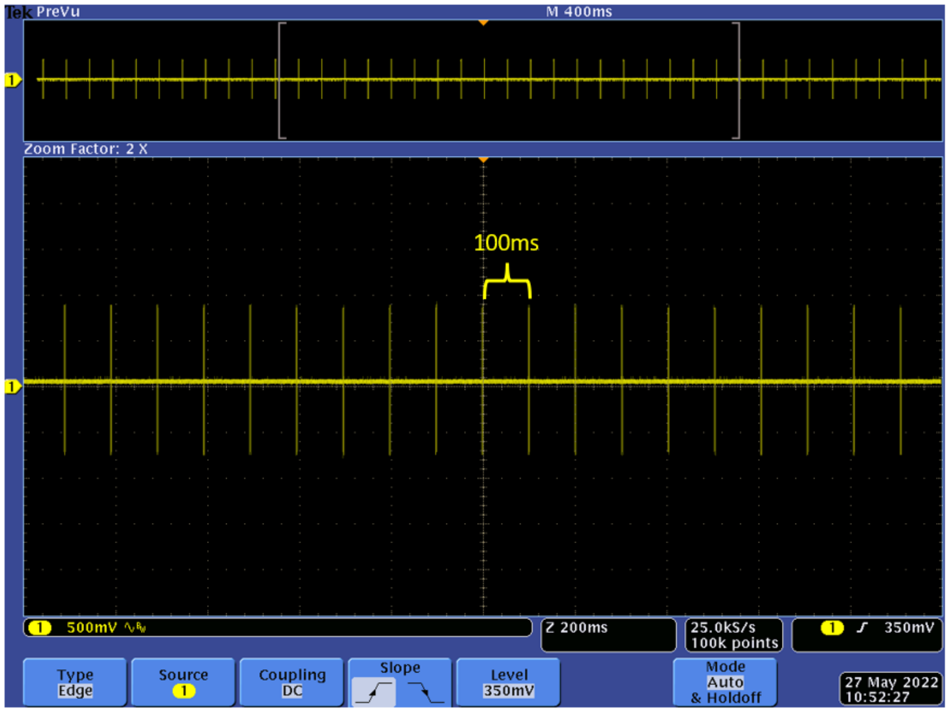
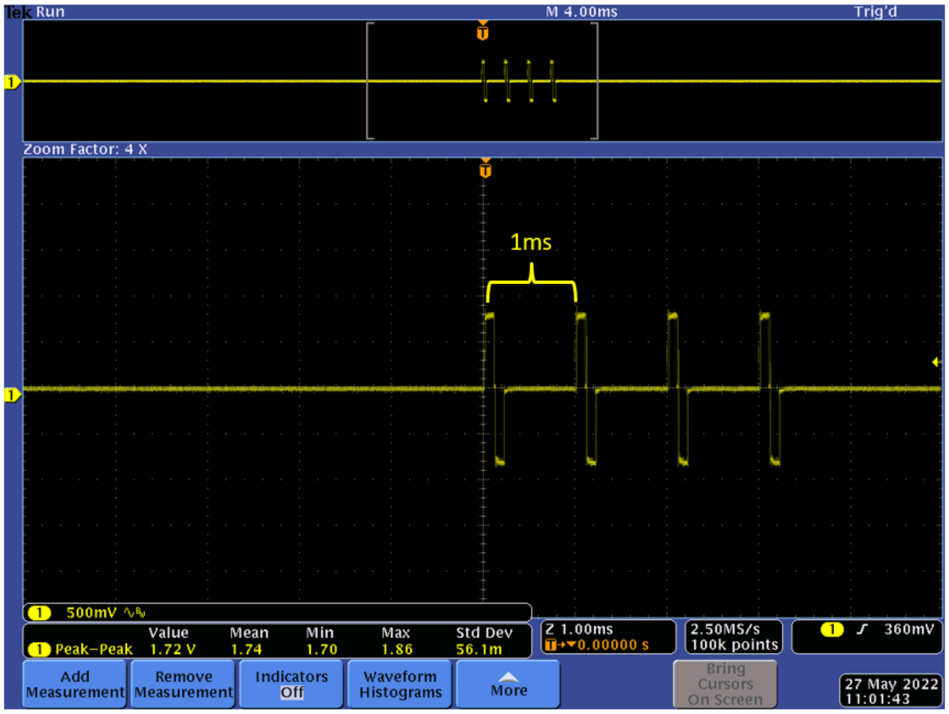
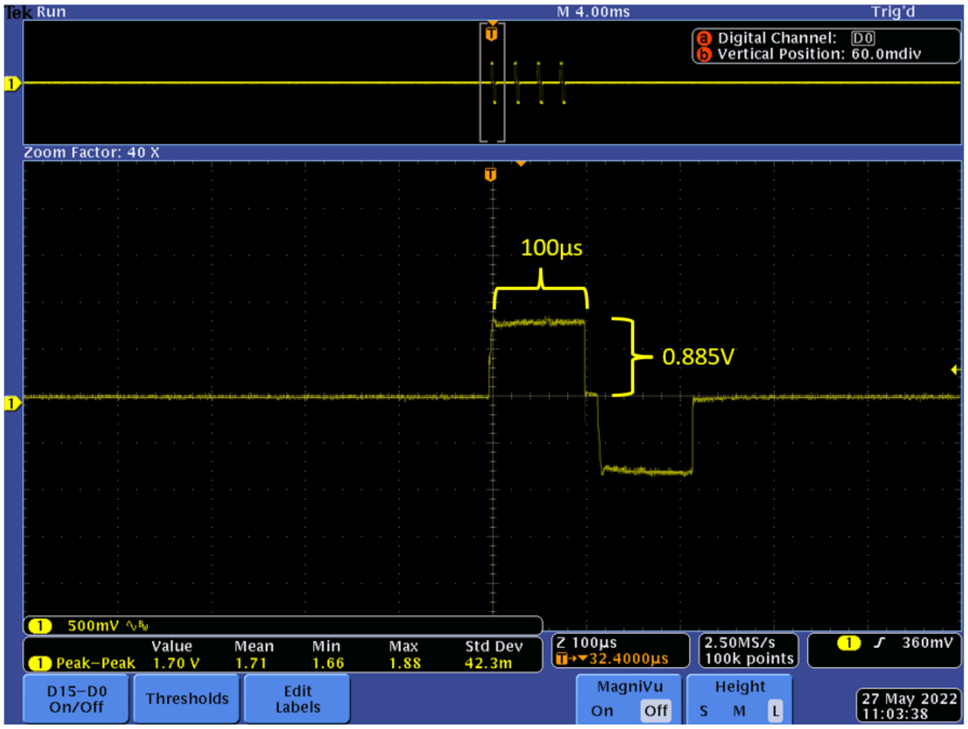


Figure 7-2: Waveform from Oscilloscope for Stimulation Parameters in Table 4-1 (a)



	Value	Mean	Min	Max	Std Dev
1 Peak–Peak	1.72 V	1.74	1.70	1.86	56.1m

Figure 7-3: Waveform from Oscilloscope for Stimulation Parameters in Table 4-1 (b)



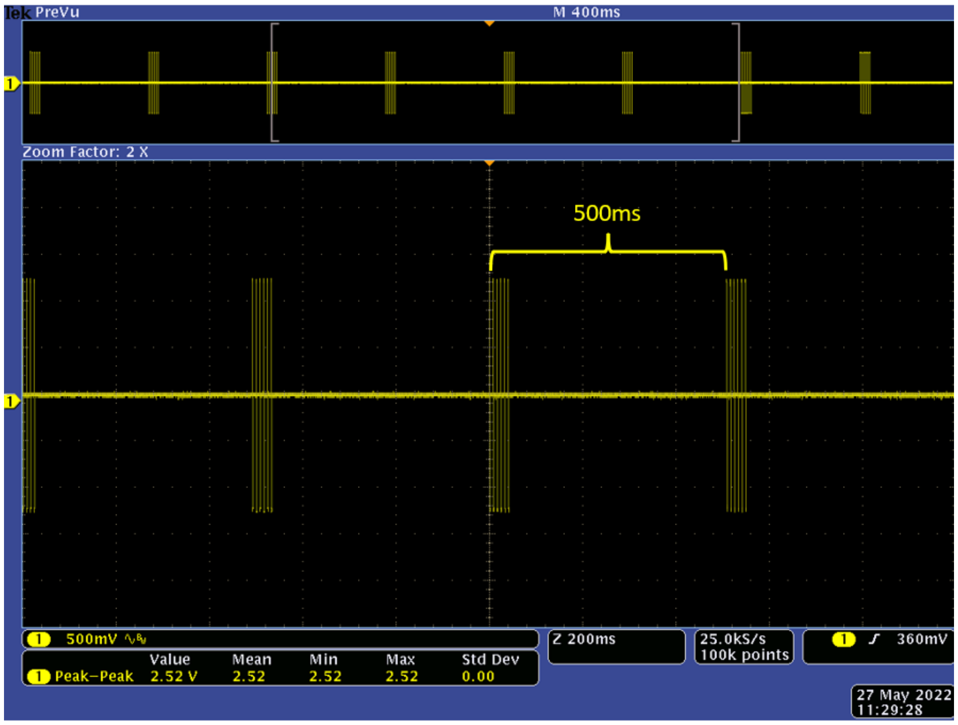
	Value	Mean	Min	Max	Std Dev
Peak–Peak	1.70 V	1.71	1.66	1.88	42.3m

Figure 7-4: Waveform from Oscilloscope for Stimulation Parameters in Table 5-1 (c)

Figures 7-5 to 7-7 below are voltage response under periodic stimulation mode with stimulation parameters summarized in table 7-2. Stimulation current probed by oscilloscope is 1.28V/(2×450) Ω = 1.43 mA.

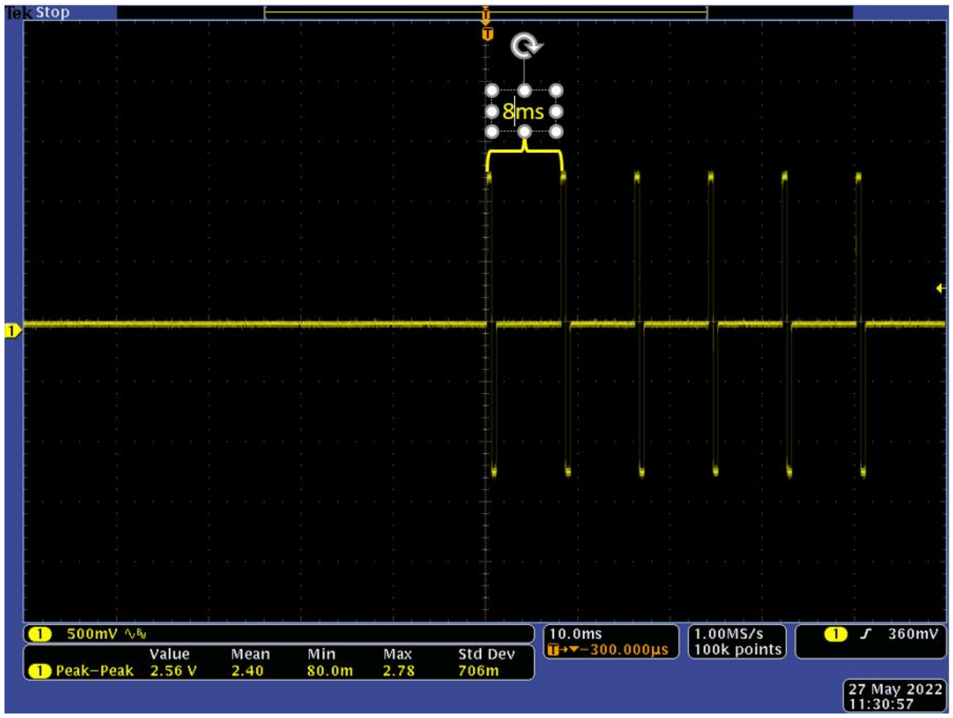
Table 7-2: Stimulation Parameters of Periodic Stimulation with Resistive Load (Case 2)

Parameters	Value
Stimulation Period	500 ms
Pulse Width	500 µs
Burst Period	8 ms
Burst Count	6
Current	1.5 mA
Interphase Delay	10 µs
Asymmetric Ratio	1



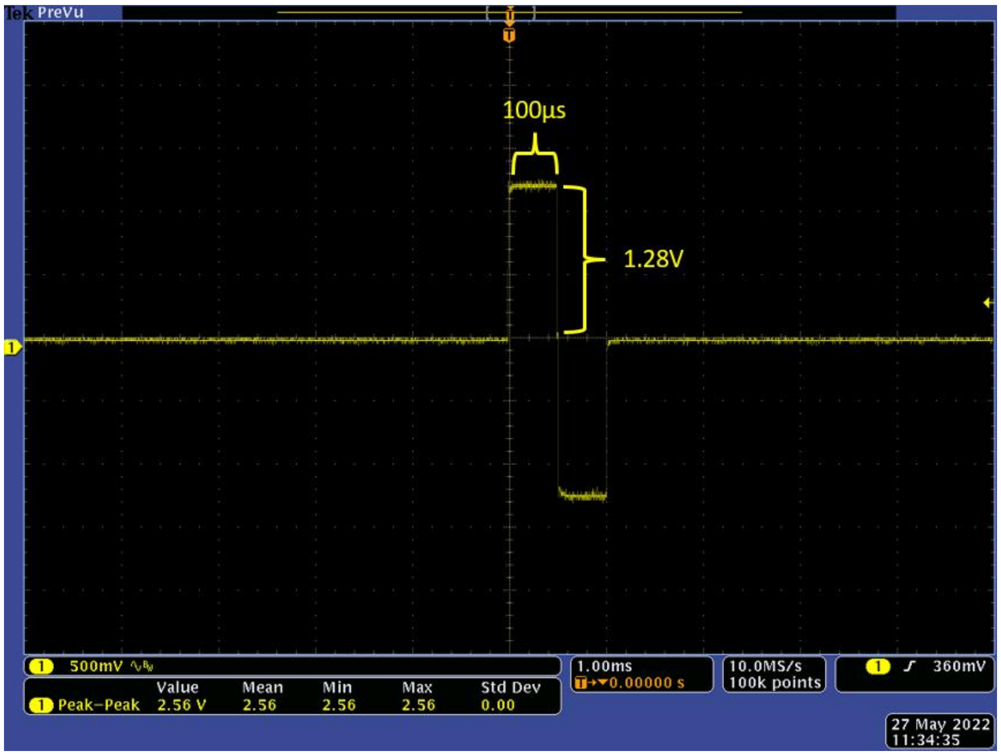
```
        Value   Mean  Min   Max   Std Dev
Peak-Peak  2.52 V  2.52  2.52  2.52  0.00
```

Figure 7-5: Waveform from Oscilloscope for Stimulation Parameters in Table 4-2 (a)



```
        Value   Mean  Min    Max   Std Dev
Peak-Peak  2.56 V  2.40  80.0m  2.78  706m
```

Figure 7-6: Waveform from Oscilloscope for Stimulation Parameters in Table 4-2 (b)



```
   Value   Mean  Min   Max   Std Dev
1 Peak-Peak 2.56 V  2.56  2.56  2.56  0.00
```

Figure 7-7: Waveform from Oscilloscope for Stimulation Parameters in Table 4-2 (c)

With resistive load, testing is also performed with aperiodic stimulation protocol. Figure 7-8 shows the stimulation waveform in random frequency mode with stimulation parameters in table 7-3. Figure 7-9 shows the stimulation waveform in random pulse width mode with stimulation parameters in table 7-4. Figure 7-10 shows the stimulation waveform with both random frequency and random pulse width mode enabled with stimulation parameters in table 7-5.

Table 7-3: Stimulation Parameters of Random Frequency Stimulation with Resistive Load

Parameters	Value
Stimulation Period	100 ms
Pulse Width	100 µs
Burst Period	1 ms
Burst Count	4
Current	1 mA
Interphase Delay	10 µs
Asymmetric Ratio	1

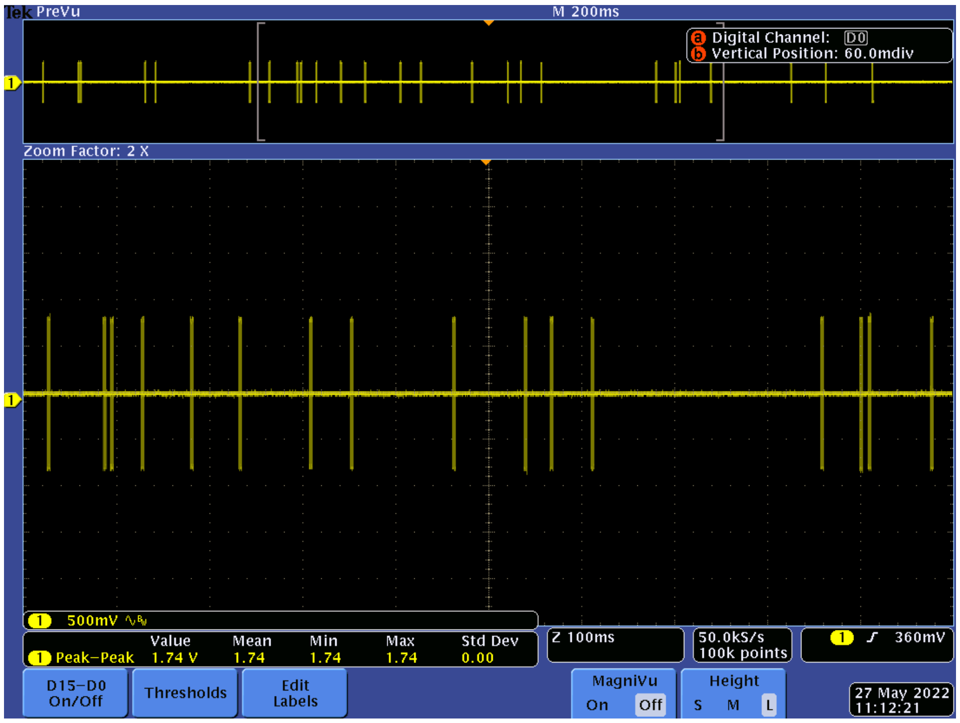


Figure 7-8: Waveform from Oscilloscope for Stimulation Parameters in Table 4-4 under Random Frequency Mode

Table 7-4: Stimulation Parameters of Random Pulse Width Stimulation with Resistive Load

Parameters	Value
Stimulation Period	5 ms
Pulse Width	300 µs
Burst Period	1 ms
Burst Count	1
Current	1 mA
Interphase Delay	10 µs
Asymmetric Ratio	1

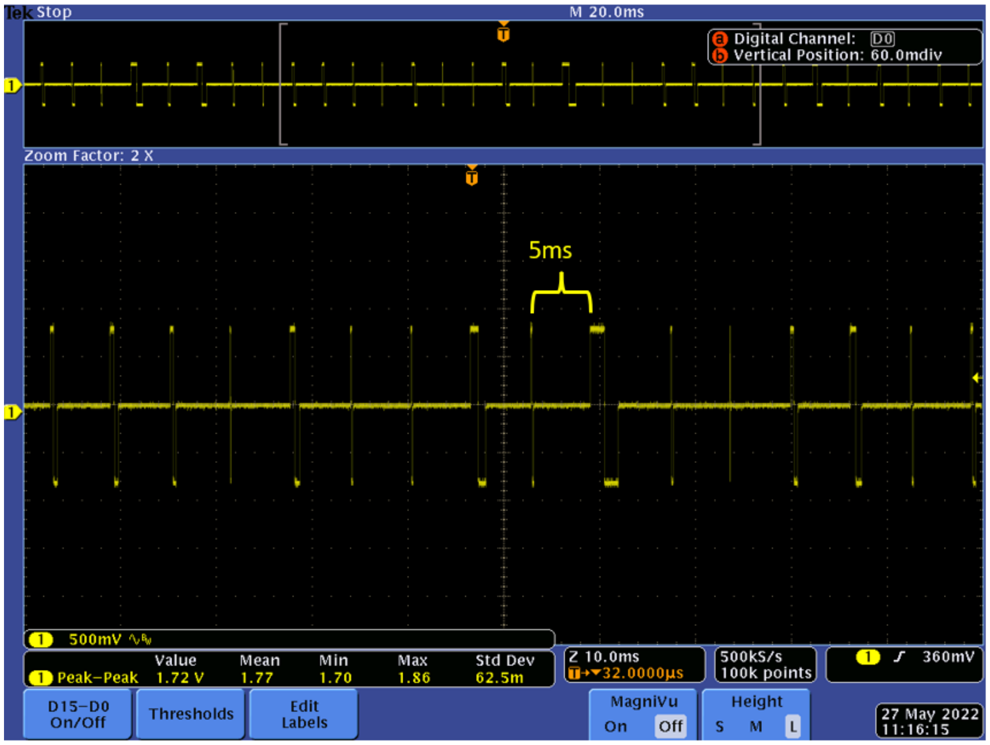


Figure 7-9: Waveform from Oscilloscope for Stimulation Parameters in Table 4-5 under Random Pulse Width Mode

Table 7-5: Stimulation Parameters of Random Pulse Width Stimulation with Resistive Load

Parameters	Value
Stimulation Period	5 ms
Pulse Width	500 µs
Burst Period	1 ms
Burst Count	1
Current	1.5 mA
Interphase Delay	10 µs
Asymmetric Ratio	1

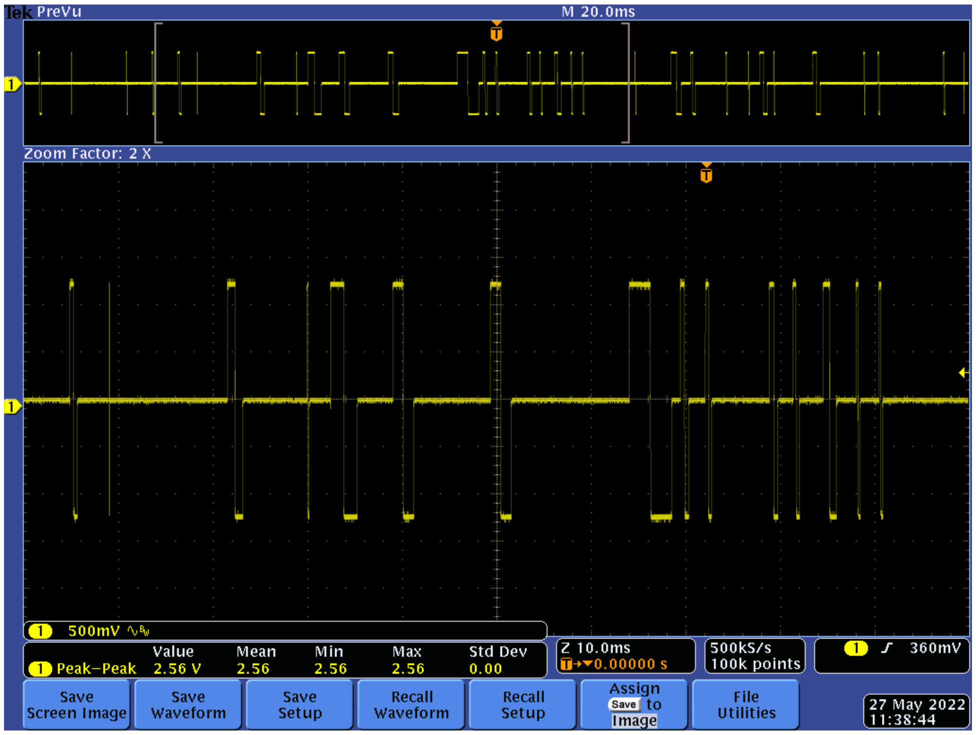


Figure 7-10: Waveform from Oscilloscope for Stimulation Parameters in Table 4-6 under both Random Frequency and Random Pulse Width Mode

Multiple sets of testing with resistive load above demonstrate the capability of the stimulator with the associated firmware to deliver accurate current stimulation waveform adhered to the specified stimulation parameters and stimulation mode.

7.1.3. Testing with Saline

The setup of testing with saline is shown in Figure 7-11 below. The remote device is powered by a battery and executes stimulation to 0.9% saline. An oscilloscope is used to measure the voltage response against different stimulation parameters and stimulation mode. Figures 7-12 to 7-14 below are voltage response under periodic stimulation mode with stimulation parameters summarized in table 7-6. Figure 7-15 is the voltage response under both random frequency and random pulse width model with stimulation parameters summarized in Table 7-7. Note that in both cases, the voltage waveform did not settle due to the narrow the pulse width and the large RC constant contributed by the electrode-saline interface. From Figure 7-14, the time constant can be estimated using the formula below:

$$V_{1,2} = V_0(1 - e^{-\frac{t}{\tau}})$$

From Figure 7-14, V_1 is approximately 0.359V when t is 200µs, V_2 is approximately 0.2V when t is 200µs. The time constant of the electrode-saline interface can be estimated to be 4.36×10^{-4}s.

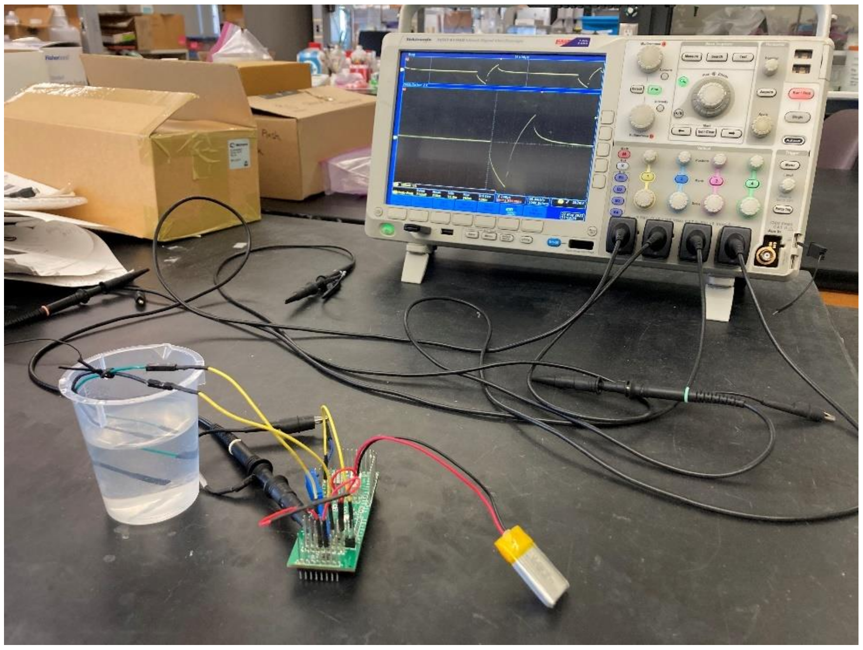
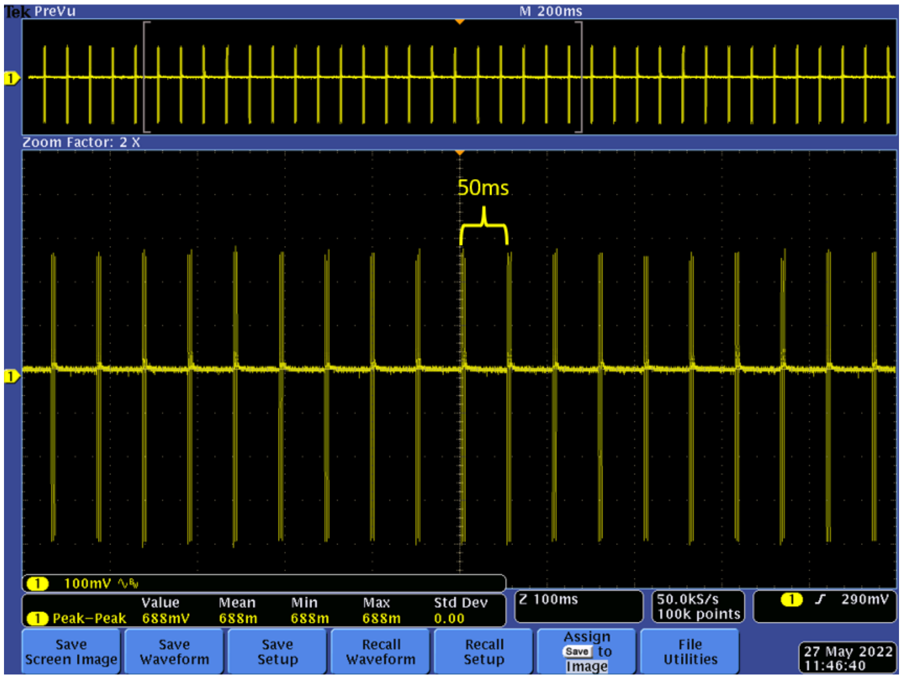


Figure 7-11: Setup of Testing with Saline

Table 7-6: Stimulation Parameters of Random Pulse Width Stimulation with Resistive Load

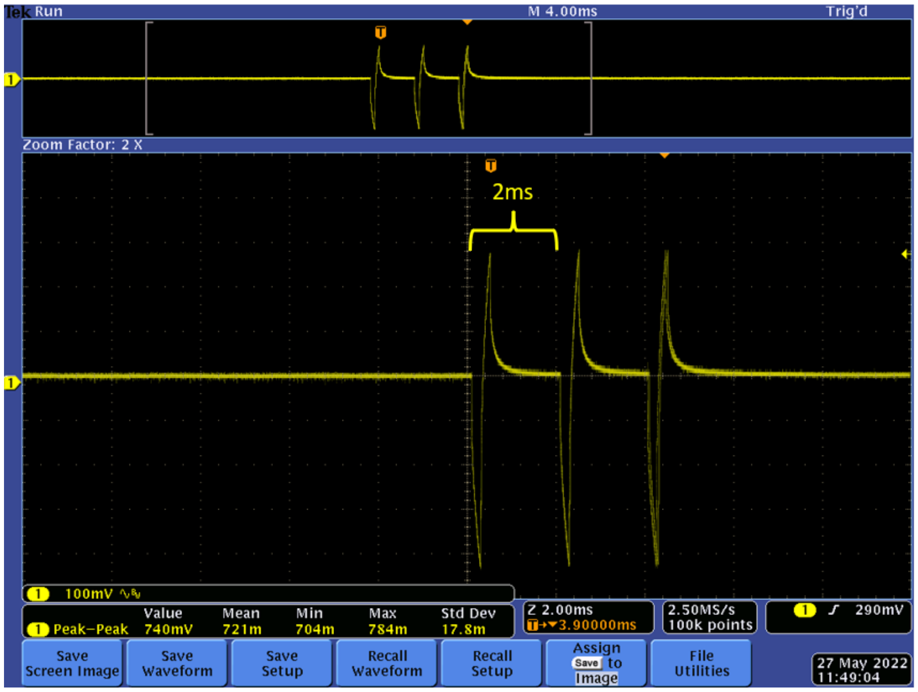
Parameters	Value
Stimulation Period	50 ms
Pulse Width	200 µs
Burst Period	2 ms
Burst Count	3

Current	1.5 mA
Interphase Delay	10 µs
Asymmetric Ratio	1



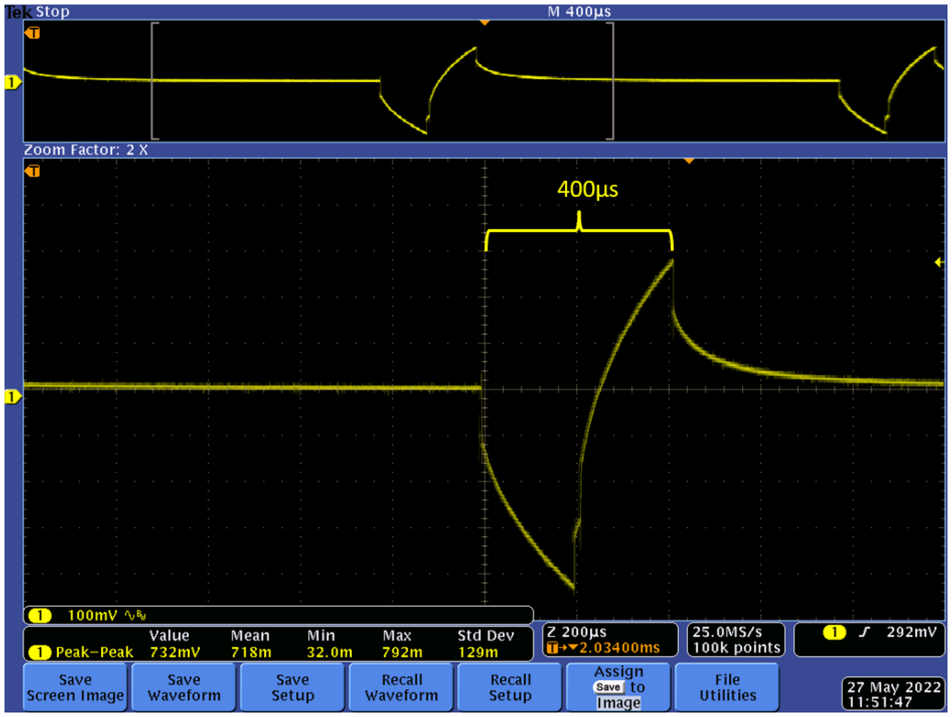
	Value	Mean	Min	Max	Std Dev
Peak−Peak	688mV	688m	688m	688m	0.00

Figure 7-12: Waveform from Oscilloscope for Stimulation Parameters in Table 4-5 under Periodic Mode in Saline (a)



	Value	Mean	Min	Max	Std Dev
1 Peak−Peak	740mV	721m	704m	784m	17.8m

Figure 7-13: Waveform from Oscilloscope for Stimulation Parameters in Table 4-7 under Periodic Mode in Saline (b)

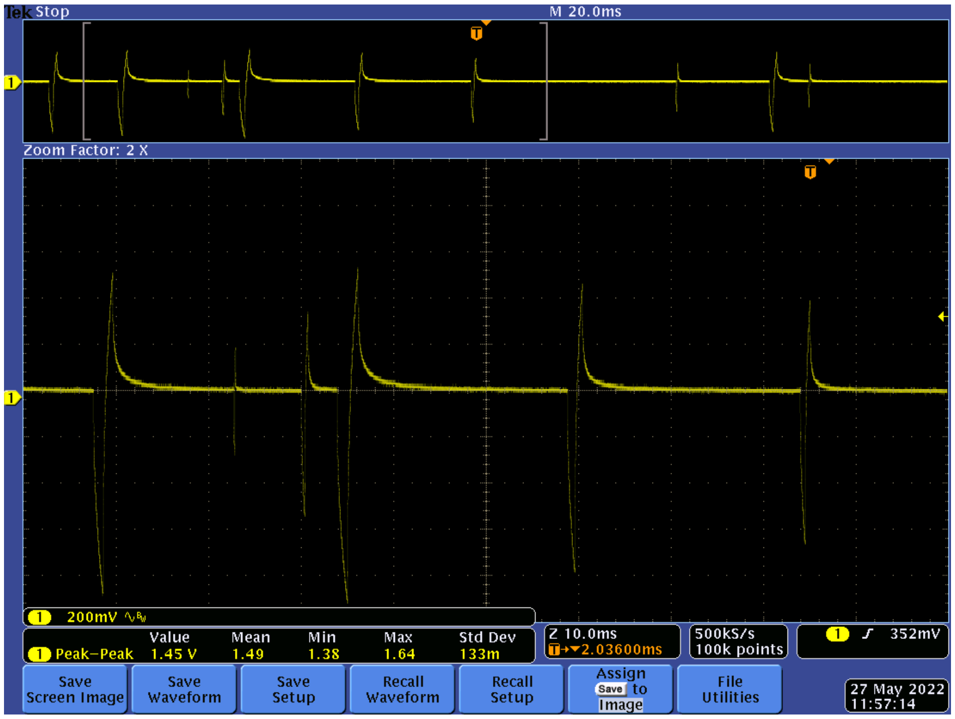


```
            Value   Mean   Min    Max    Std Dev
Peak-Peak   732mV   718m   32.0m  792m   129m
```

Figure 7-14: Waveform from Oscilloscope for Stimulation Parameters in Table 4-7 under Periodic Mode in Saline (c)

Table 7-7: Stimulation Parameters of Random Frequency and Pulse Width Stimulation with Resistive Load

Parameters	Value
Stimulation Period	50 ms
Pulse Width	200 µs
Burst Period	2 ms
Burst Count	3
Current	1.5 mA
Interphase Delay	10 µs
Asymmetric Ratio	1



	Value	Mean	Min	Max	Std Dev
1 Peak−Peak	1.45 V	1.49	1.38	1.64	133m

```
Z 10.0ms
T→▼2.03600ms
500kS/s
100k points
1 ∫ 352mV
27 May 2022
11:57:14
```

Figure 7-15: Waveform from Oscilloscope for Stimulation Parameters in Table 4-7 under Random Frequency and Random Pulse Width Mode (Saline)

Testing with 0.9% saline above demonstrates noticeable transient effect due to the saline-electrode interface. This phenomenon is observed during in-vivo experiments due to the non-Faradaic current created by the tissue-electrode interface. When using 0.9% saline to emulate real tissue environment, the stimulator and associated firmware functions properly, which provides the prerequisites for in-vivo testing.

7.2. Simulation of Real-time Artifact-cancellation Algorithm

Template-subtraction-based algorithm to remove stimulus artifact in real-time discussed in section 6.2 is first verified in simulation before the deployment to the system. The simulation also eases the process of developing GUI with intense requirement for computing resource. The procedure to validate the feasibility of the proposed algorithm is the following.

1. Raw input signal containing neural signals super-positioned with stimulus artifact is collected with the setup shown in Figure 7-16 below. A beaker containing 50g of 0.9% saline simulates real animal tissues and provides voltage response due to tissue-electrode interface. A function generator is used to inject pre-recorded electromyography (EMG) signal into saline as emulated neural signal. Simultaneously, the implanted/remote device delivers periodic current stimulation into beaker and record the voltage response from saline.

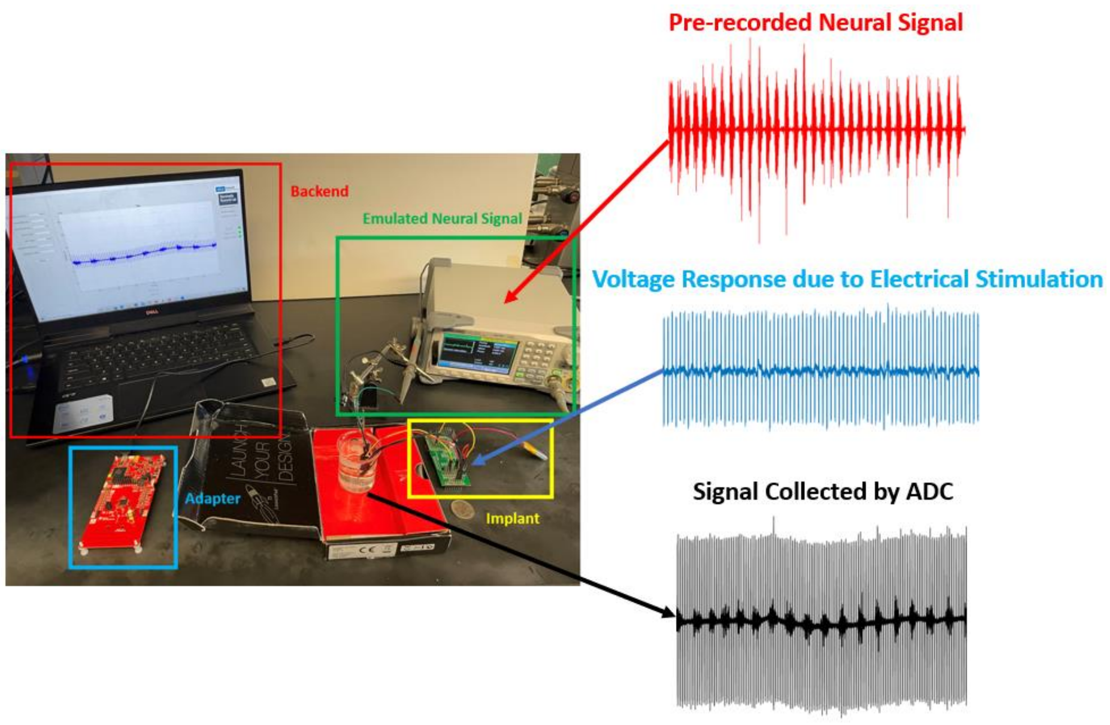

Pre-recorded Neural Signal
Backend
Emulated Neural Signal
Voltage Response due to Electrical Stimulation
Adapter
Implant
Signal Collected by ADC

Figure 7-16: Experiment Configuration to Collect Artifact-contaminated Neural Signal

2. With the collected voltage response data in step 1, the algorithm discussed in Section 6.2 is written in MATLAB script to remove stimulus artifact in the collected data in a quasi-

real-time manner. In the simulation, raw input signal is processed per 1-second interval to mimic ADC behaviors in actual implementation. Therefore, the complexity of the algorithm can be accessed to ensure the real-time feature of the entire system.

3. The performance of the algorithm is evaluated in 3 aspects: similarity in time domain, similarity in frequency domain, processing time for 1-second interval of data. Due to time constraint and the transitional nature of this validation stage, qualitative analysis is mostly conduct in quasi-real-time simulation to verify the feasibility of the algorithm. For time-domain evaluation, Figures 7-17 and 7-18 below show a comparison between the post-processing signal, DC-free input signal, and raw input signal. Features of neural signals are largely preserved in time domain and the high-amplitude stimulus artifacts are attenuated to visually close to baseline noise level.

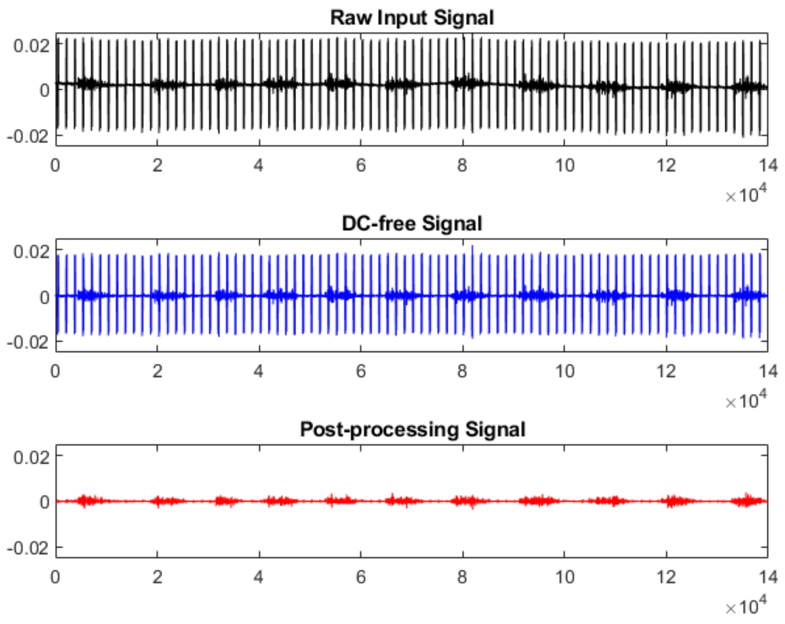


Figure 7-17: Comparison of Raw Input Signal, DC-free Input Signal, and Post-processing Signal

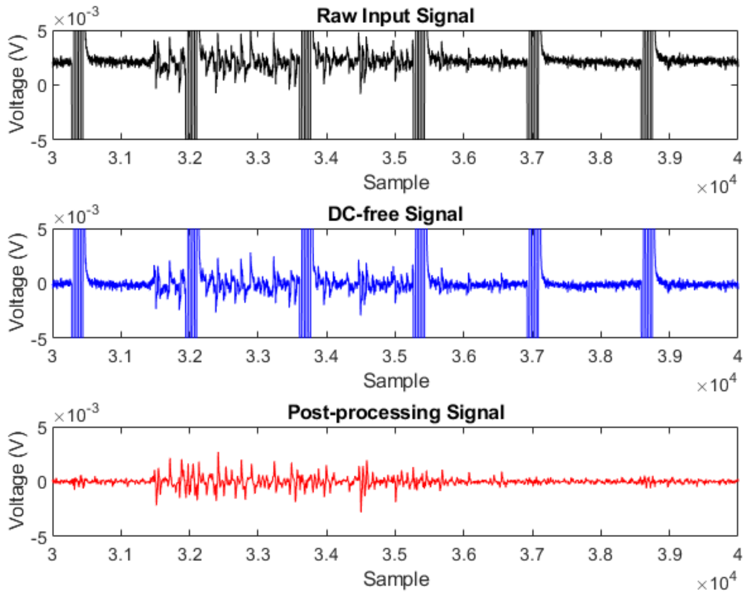


Figure 7-18: Comparison of Raw Input Signal, DC-free Input Signal, and Post-processing Signal (Magnified)

Frequency analysis compares Fast Fourier Transform (FFT) of the post-processing contaminated signal and ADC sampled emulated neural signal in saline for two reasons. First, pre-recorded neural signals in .csv does not contain DC drifting and background noise caused by the electrode, ADC, and environment. This scenario of is demonstrated in Figure 7-19 below. Secondly, function generator produces emulated neural signals using sample-and-on method. Signal generated in this way may contain additional frequency components that are not presented in the original pre-recorded signal. Therefore, the comparison of FFT spectrums between post-processing signal and emulated signal recorded saline provides a better visualization to evaluate the effectiveness of the algorithm.

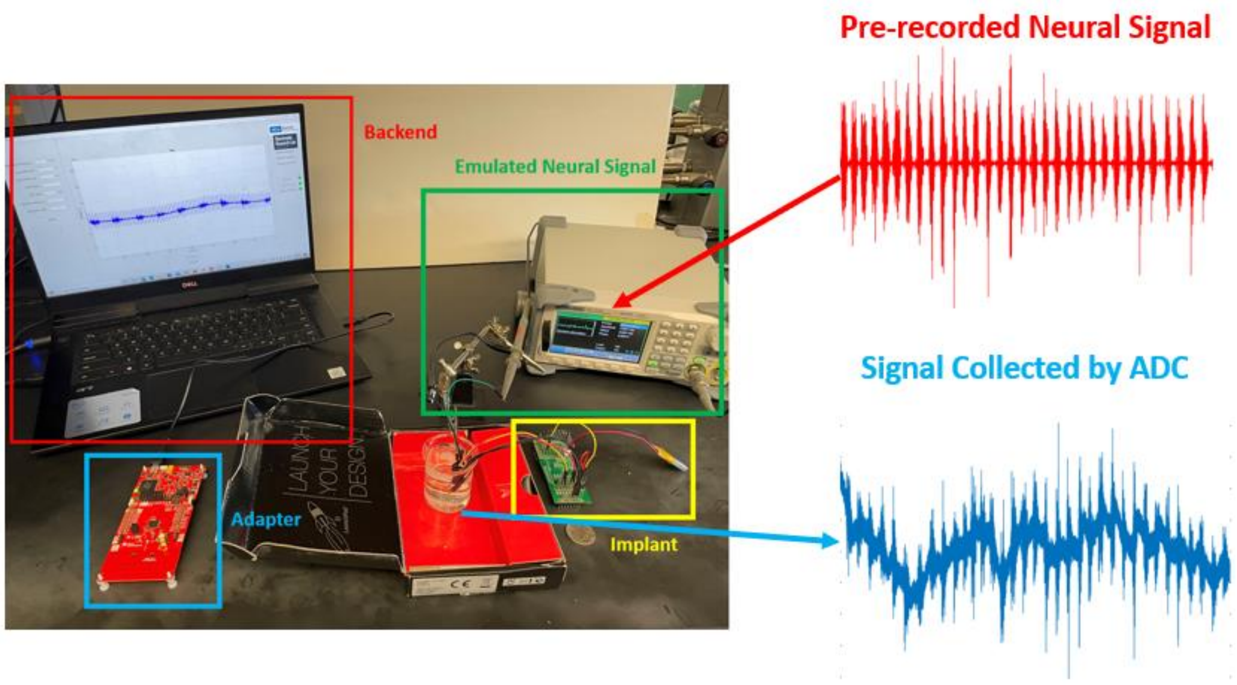


Figure 7-19: Source of Emulated Signal Recorded in Saline in Frequency Analysis

Figure 7-20 below shows FFT of the raw input signal with stimulus artifact. As mentioned in Section 2.5, stimulus artifact introduces tones that distributes over a wide frequency spectrum.

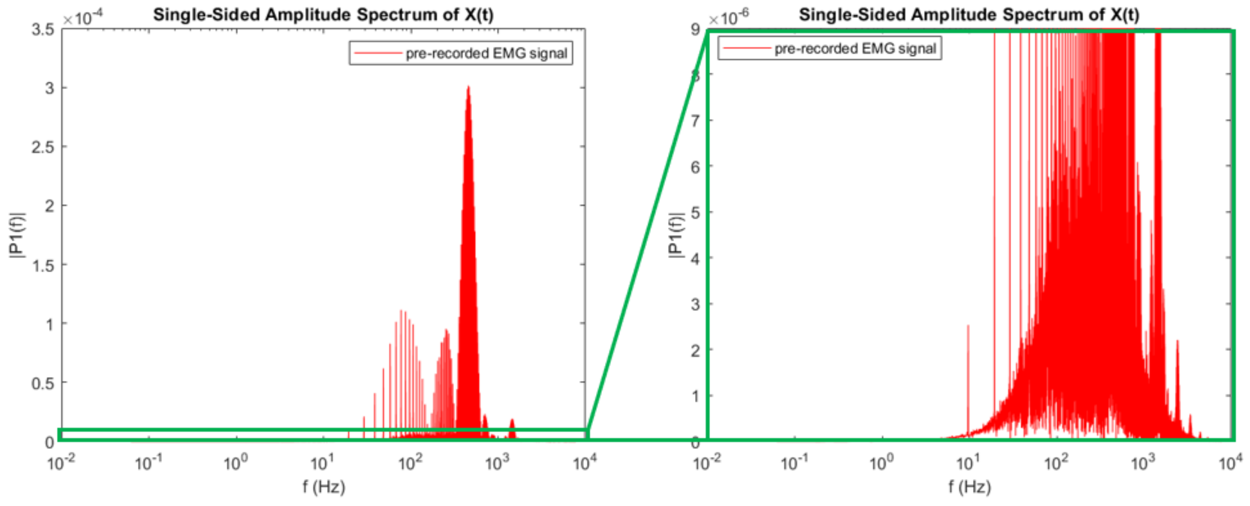
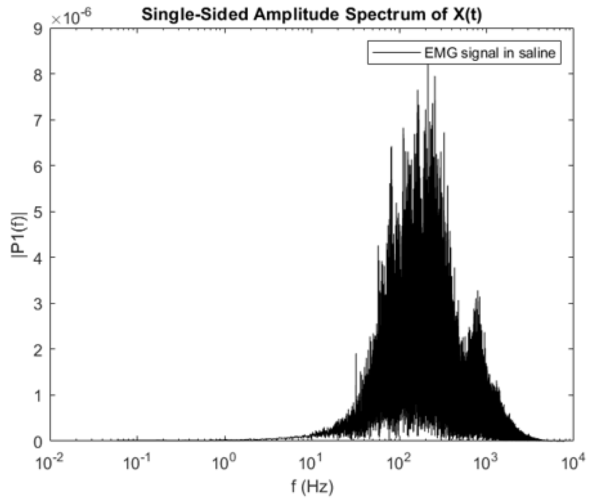


Figure 7-20: FFT of Raw Input Signal Contaminated by Stimulus Artifact

In Figure 7-21, FFT of post-processing signal largely suppresses tones created by stimulus artifact. Compared with emulated neural signals in saline with stimulation disabled, the shape and magnitude of FFT spectrum are well recovered from FFT of the raw input signal with tones mixed with signal of interest.



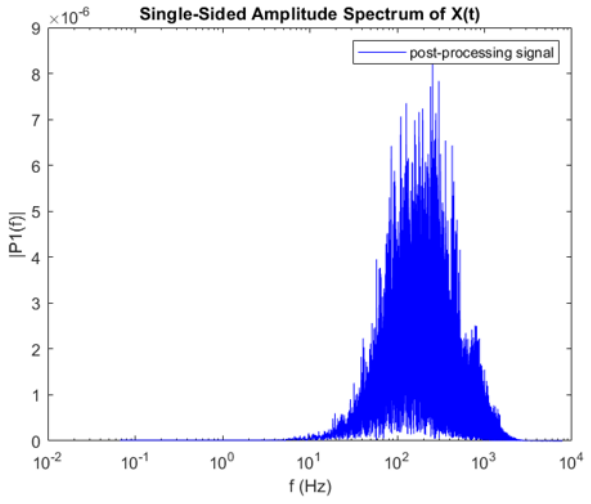
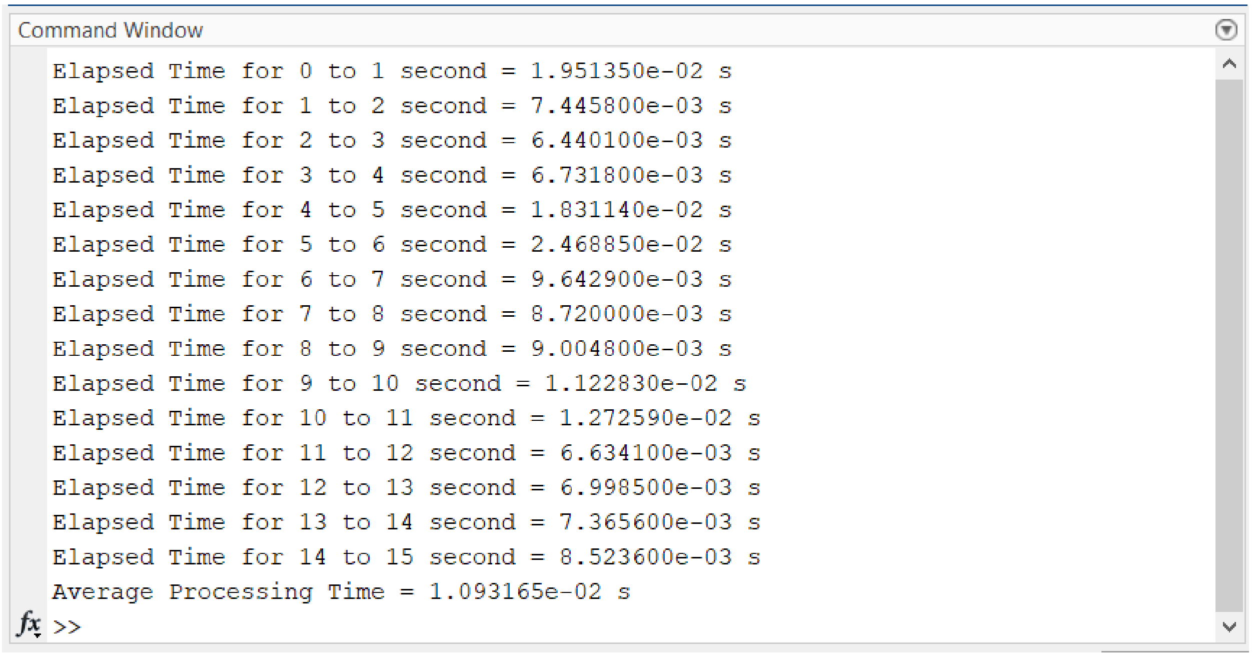


Figure 7-21: FFT of Emulated Neural Signal in Saline vs Post-processing Signal

Finally, the processing time for each 1-second interval is measured and used to produce arithmetic mean. When the algorithm is applied to GUI in real-time, delay from communication and data visualization place more stringent requirement on the maximum run-time. Furthermore, optimized version of such algorithm is planned to implement on MCU for artifact cancellation on chip as future work, which requires sufficiently efficient coding implementation. Figure 7-22 below shows the typical run-time for 1-second interval in quasi-real-time simulation. The average run-time for 1-second samples is at the level of 10ms with reasonable amount of redundancy.



```
Elapsed Time for 0 to 1 second = 1.951350e-02 s
Elapsed Time for 1 to 2 second = 7.445800e-03 s
Elapsed Time for 2 to 3 second = 6.440100e-03 s
Elapsed Time for 3 to 4 second = 6.731800e-03 s
Elapsed Time for 4 to 5 second = 1.831140e-02 s
Elapsed Time for 5 to 6 second = 2.468850e-02 s
Elapsed Time for 6 to 7 second = 9.642900e-03 s
Elapsed Time for 7 to 8 second = 8.720000e-03 s
Elapsed Time for 8 to 9 second = 9.004800e-03 s
Elapsed Time for 9 to 10 second = 1.122830e-02 s
Elapsed Time for 10 to 11 second = 1.272590e-02 s
Elapsed Time for 11 to 12 second = 6.634100e-03 s
Elapsed Time for 12 to 13 second = 6.998500e-03 s
Elapsed Time for 13 to 14 second = 7.365600e-03 s
Elapsed Time for 14 to 15 second = 8.523600e-03 s
Average Processing Time = 1.093165e-02 s
fx >>
```

Figure 7-22: Processing Time in Quasi-real-time Simulation

7.3. In-vitro Experiment to Access Artifact-cancellation Capability

With the verified quasi-real-time simulation in Chapter 7.2, the algorithm is deployed to the backend of GUI, followed by in-vitro experiments to examine the performance. The setup of the in-vitro experiment is shown in Figure 7-23. The implement of the artifact cancellation algorithm processes the incoming raw input signal by 1-second interval and displays the post-processing signal in real-time. Figure 7-24 demonstrates the post-processing signal being displayed in the data panel of GUI over time.

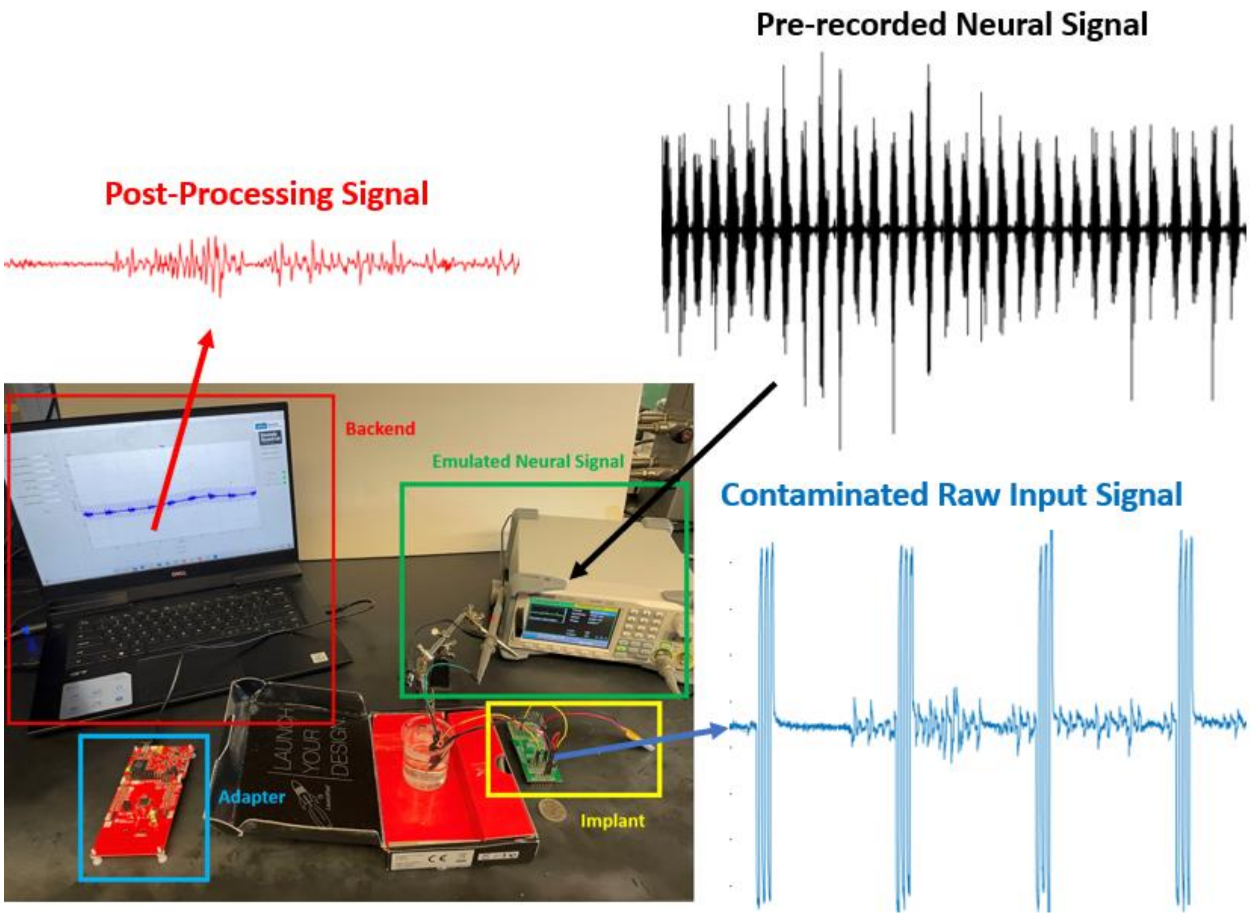


Figure 7-23: Setup and Signal Flow of In-vitro Testing

The algorithm is tested against different stimulation parameters. At the beginning of in-vitro testing, the 30 second of emulated neural signal in saline is recorded by the remote device without enabling stimulation as a baseline. For each testing case, 30 second of raw input and post-processing signal with concurrent stimulation are recorded for evaluation. The performance with respect to different stimulation parameters is evaluated under two metrics in frequency domain. For the first metric, the magnitude of largest tone in FFT of the raw input is compared to the same frequency component in FFT of the post-processing signal to calculate the maximum attenuation. Maximum attenuation can be calculated using the following equation. Index i refers to the frequency at which FFT of the raw input signal has the largest tone.

$$Maximum\ Attenuation = 20 * log_{10}\left(\frac{FFT\_raw[i]}{FFT\_post\_processing\ [i]}\right)$$

Figure 7-25 shows a sample comparison between FFT of the raw input signal and the post-processing signal with labeled largest tone.

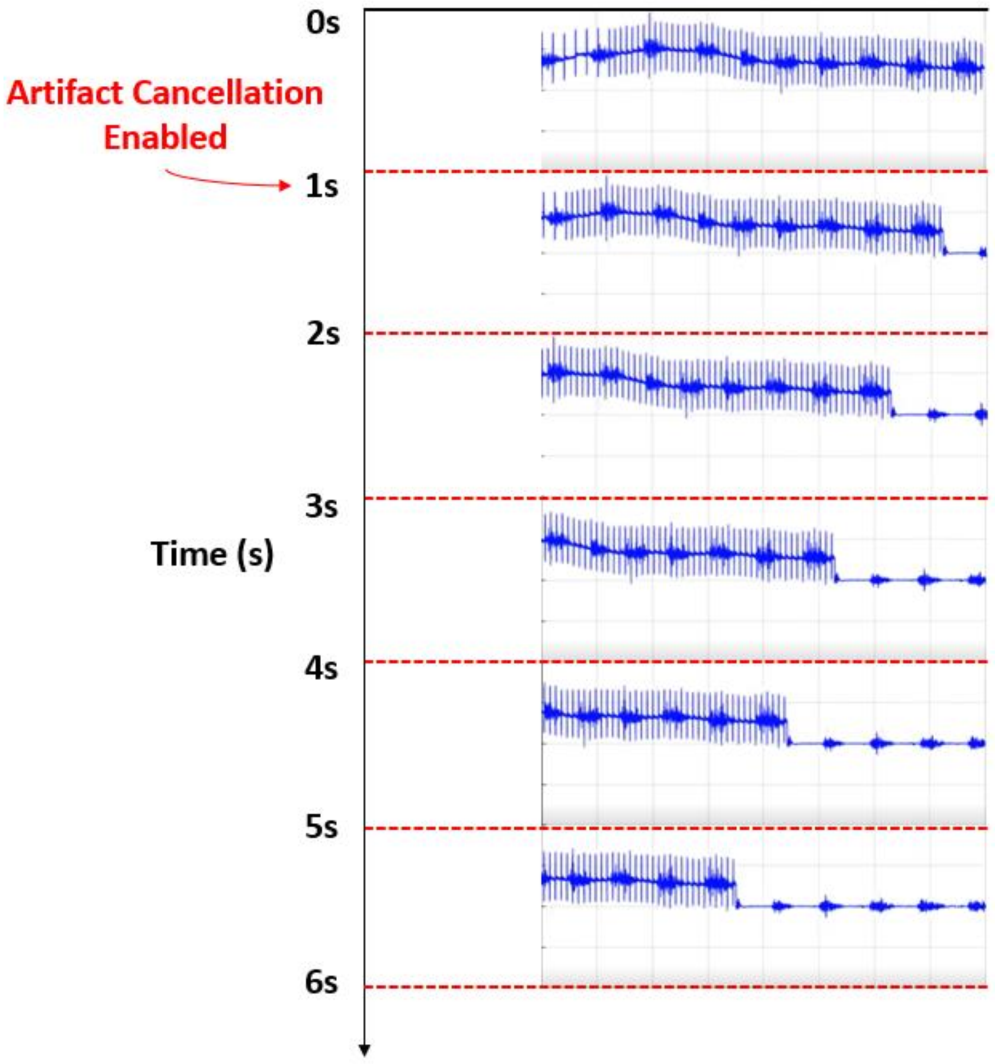


Figure 7-24: Post-processing Signal Displayed in GUI over Time

For the second metric, the normalized cross-correlation at shift of 0 is used to compare the similarity between the FFT of the baseline and raw input signal, the FFT of the baseline and post-processing signal. Zero-shift cross-correlations computed between the FFT of raw input signal and baseline, the FFT of post-processing signal and baseline provide insight of the improvement in frequency spectrum provided by the algorithm. Figure 7-26 provides a visual comparison of the FFT spectrum of emulated neural signal, raw input signal, and post-processing signal. Intuitively, the FFT of post-processing signal is more correlated to the FFT of emulated neural signal than the FFT of raw input signal is. The unwanted tones introduced by current stimulation is largely suppressed by the algorithm.

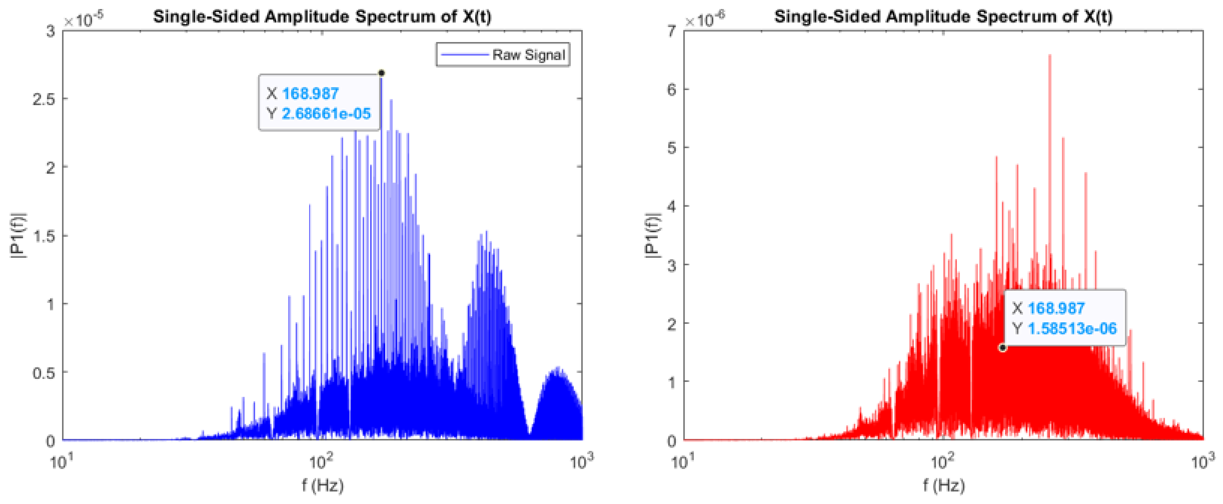
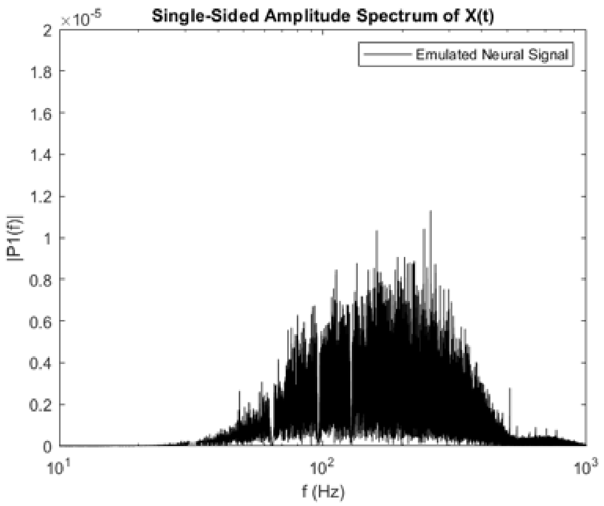
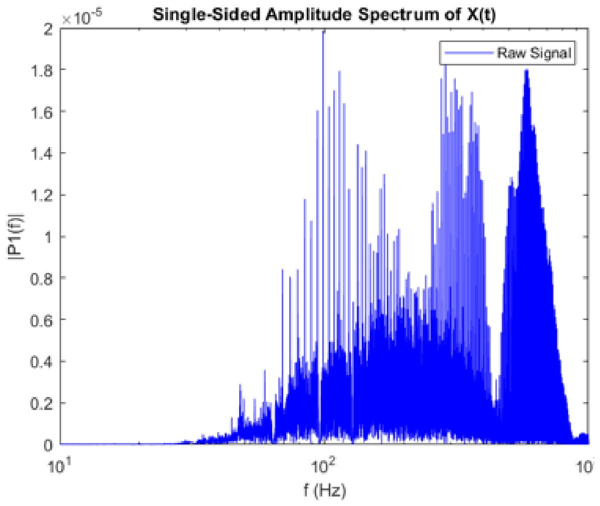


Figure 7-25: Example of Maximum Attenuation





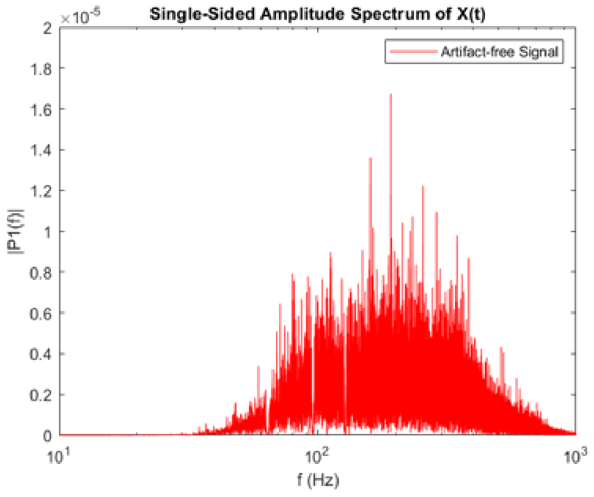


Figure 7-26: Example of FFT Comparison

Intuitively, pulse width and burst period have significant influence on the shape stimulus artifacts. Therefore, testing cases are designed to mainly assess the robustness of the algorithm against varying pulse width and burst period. Table 7-8 summarizes the stimulation parameters in 4 testing cases. Note that parameter fields that contain multiple number are the varying parameters for comparison.

Table 7-8: Stimulation Parameters for 4 Testing Cases

UUID	Case 1	Case 2	Case 3	Case 4
Stimulation Period	100ms	200ms	200ms	200ms
Stimulation Current	0.6mA	0.6mA	1mA	1mA
Burst Count	3	3	3	3
Interphase Delay	10µs	10µs	10µs	10µs
Asymmetric Ratio	1	1	1	1
Pulse Width	250µs	250µs	250µs	250µs, 500µs, 750µs, 1000µs
Burst Period	500µs, 1000µs, 1500µs, 2000µs	500µs, 1000µs, 1500µs, 2000µs	500µs, 1000µs, 1500µs, 2000µs	2000µs

Resulting metrics for 4 testing cases are summarized in Figure 7-27 to 7-30 in the manner of two bar plots per testing case. The first bar plot compares the maximum attenuate across the varying stimulation parameters. The second bar plot demonstrates the improvement of normalized cross-correlation provided by the algorithm. From the first metric, the algorithm attenuates the maximum tone by over 20 dB and the level of attenuation increases and pulse width/burst period increases. From the second metric, zero-shift cross-correlation between the raw input signal and the baseline generally decreases as pulse width/burst period increases. The red dash-line in the second bar plot marks the level of cross-correlation (0.8) that indicates a strong correlation. The zero-shift cross-correlation between the raw input signal and the baseline remains below 0.8 and even drops below 0.4 in some cases. The proposed algorithm boosts the zero-shift cross-correlation over the level of strong correlation (0.8) regardless the pulse width/burst period. This property provides notable benefit as information of neural signal is often encoded in frequency domain

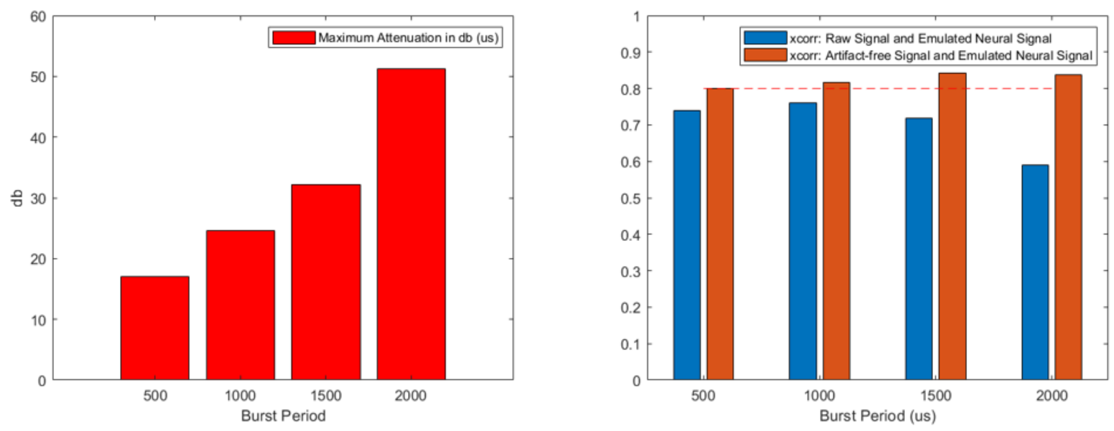


Figure 7-27: Resulting Metrics for Case 1

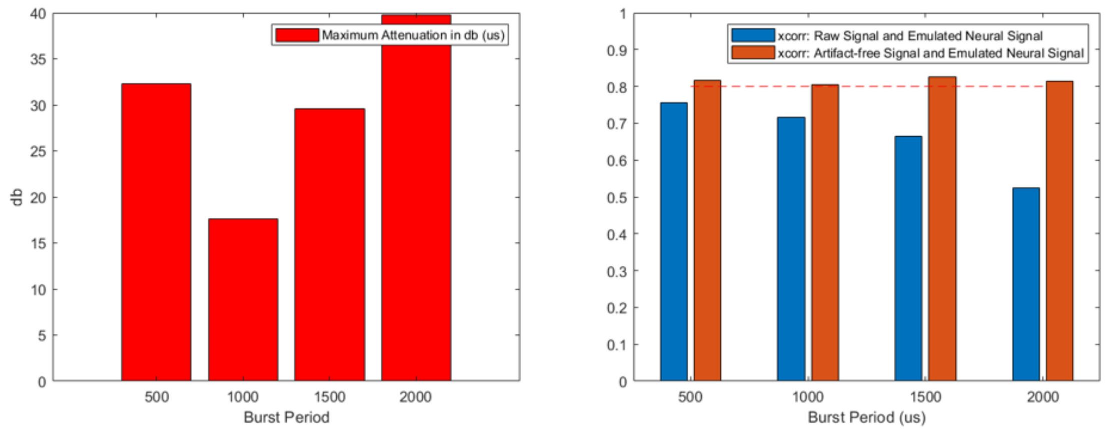


Figure 7-28: Resulting Metrics for Case 2

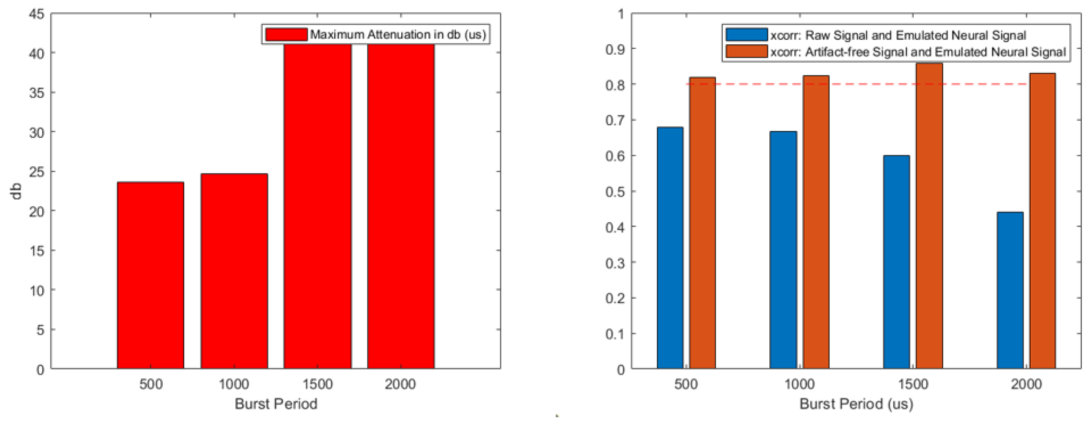


Figure 7-29: Resulting Metrics for Case 3

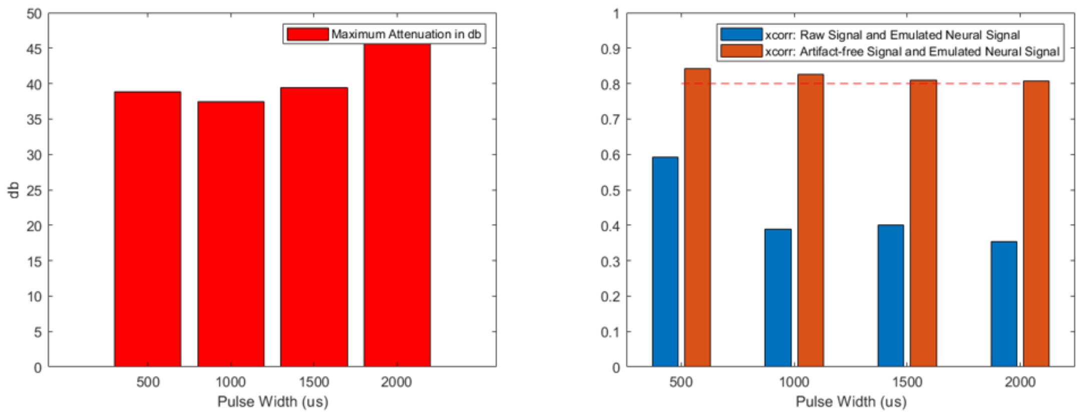


Figure 7-30: Resulting Metrics for Case 4

7.4. In-vivo Experiment

7.4.1. In-vivo Testing for Stimulator

Thanks for the opportunity provided by Dr. Million Mulugeta, an acute system testing was performed to verify functionality of the system in real animal experiment. The setup for the in-vivo testing is shown in Figure 7-31 below.

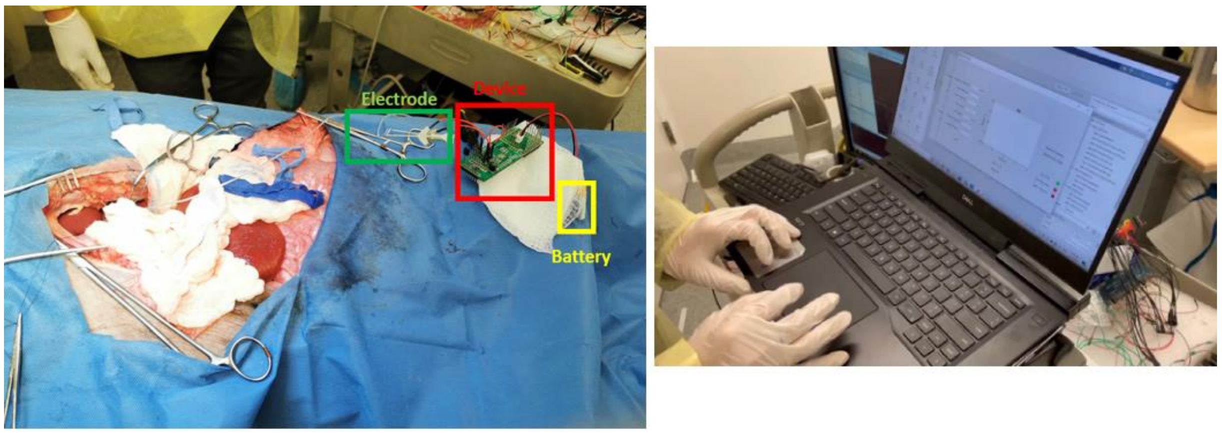


Figure 7-31: Setup of Acute Testing

During the experiment, periodic constant current stimulation is delivered to celiac branch of the Vagus Nerve of a porcine. Two sets of stimulation with the same stimulation parameters, one periodic and another one with randomized frequency, were performed. The timeline of the experiment is shown in Figure 7-32.

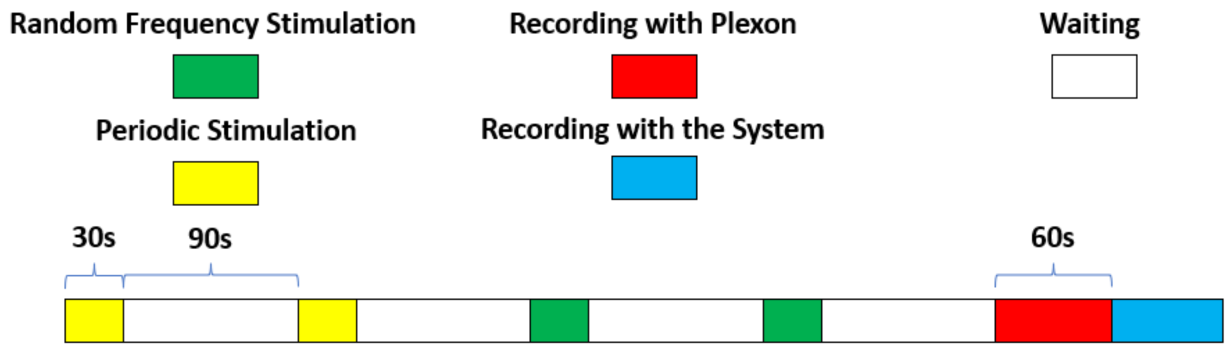


Figure 7-32: Timeline of In-vivo Experiment

The stimulation parameters are shown in table 7-9. The voltage response to such stimulation protocols is captured by oscilloscope, shown in Figure 7-33 to 7-35. In Figure 7-35, noticeable voltage transience due to Randall Cell effect is observed, which confirms the result from experiment with saline.

Table 7-9: Stimulation Parameters of SPARC Protocol

Parameters	Value
Periodic, Random Frequency/Pulse Width	Periodic
Stimulation Period	500 ms
Pulse Width	300 µs
Burst Period	9 ms
Burst Count	20
Current	1 mA
Interphase Delay	10 µs
Asymmetric Ratio	1

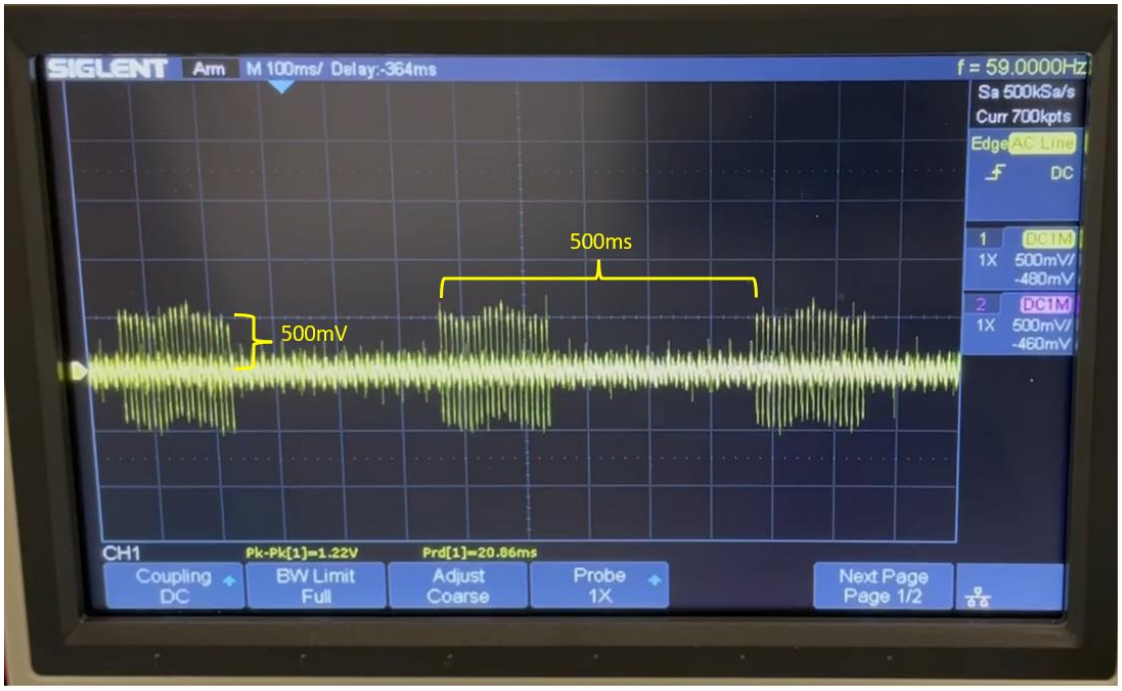


Figure 7-33: Voltage Response of Periodic Current Stimulation with Stimulation Parameters in Table 4-12 in Acute Testing

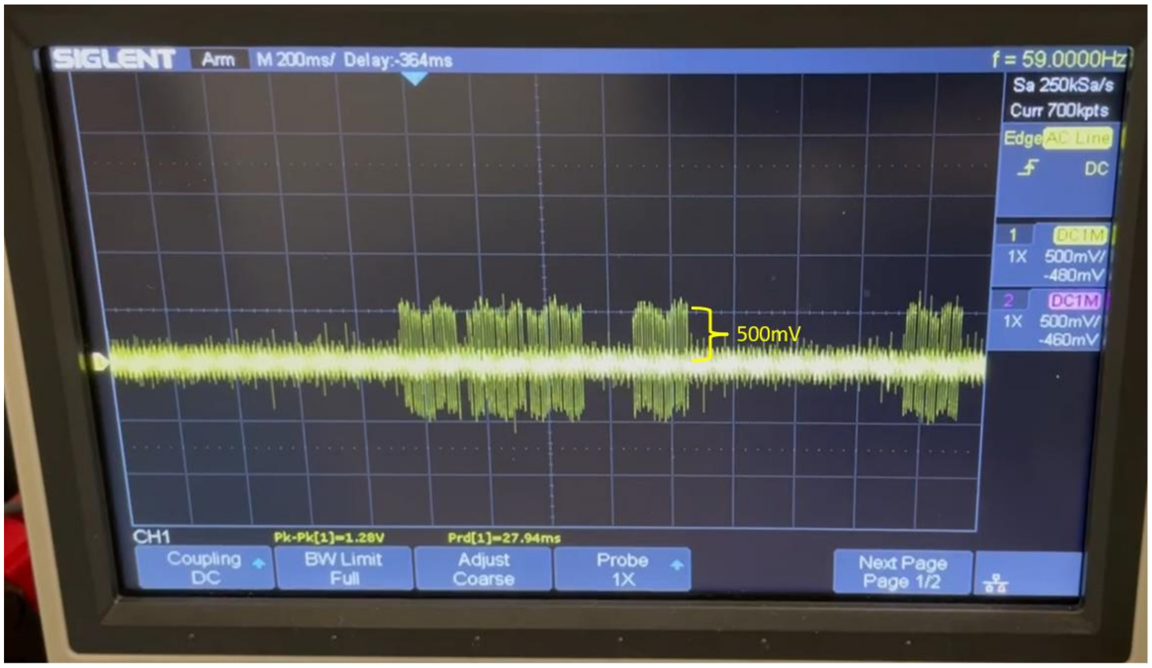


Figure 7-34: Voltage Response of Random Frequency Current Stimulation with Stimulation Parameters in Table 4-12 in Acute Testing

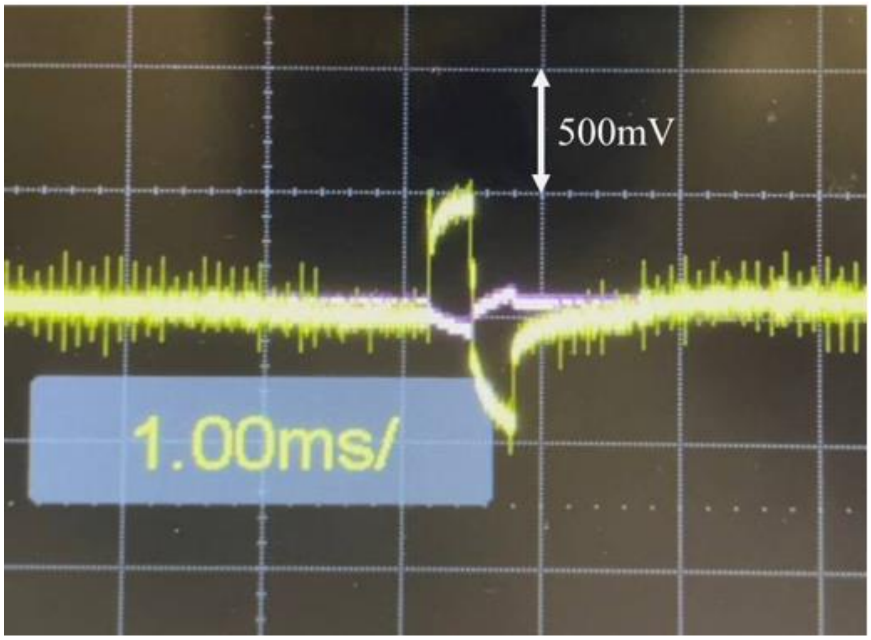


Figure 7-35: Voltage Transient due to Tissue-Electrode Interface

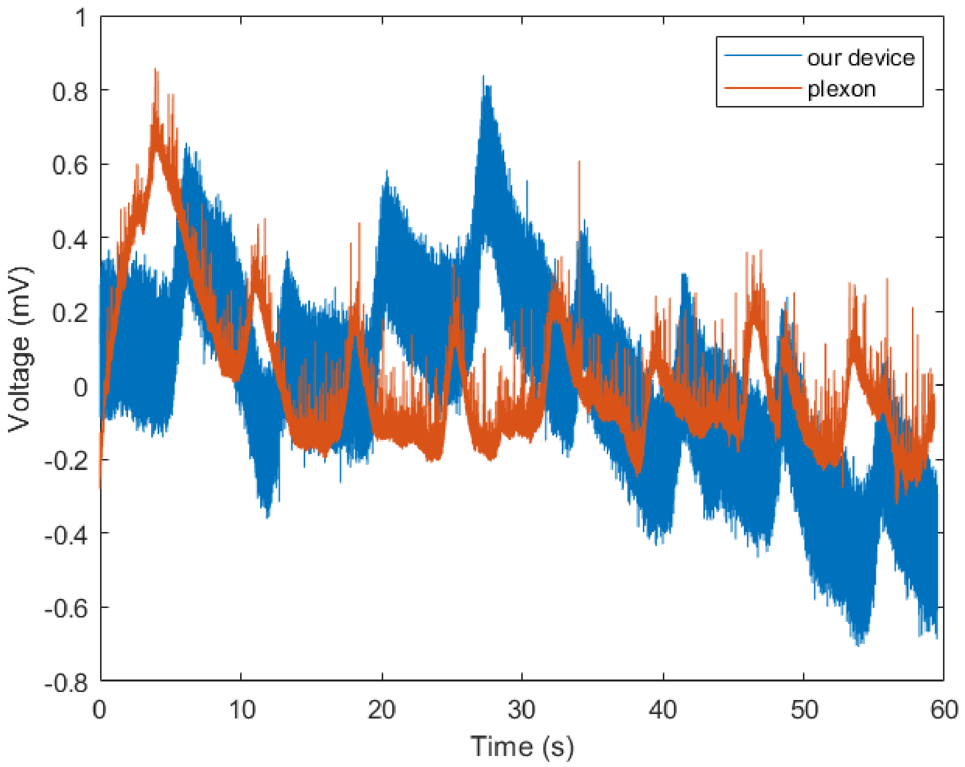


Figure 7-36: Comparison of Recordings from Two Systems

7.4.2. In-vivo Testing for Recorder

Followed by 2 sets of stimulation, recordings are taken with the system and a commercially available device (Plexon) to evaluate the noise performance of the recorder design. Figure 7-36 above shows a segment of the recording with each system. While the data acquisition with the system and Plexon are not overlapped, similar pattern due to the life-support system is observed in both recordings with similar frequency. The noise floor of the recording was at the level of 400µ$V_{p\text{-}p}$ which is 8 times higher than that of Plexon due to 2 major reasons. First, the gain of the ADC was set to 8 during the experiment. A higher gain can significantly lower the magnitude of noise-floor. Secondly, the traces connected to the input pins of ADC are not layout properly so that extra noise is introduced from the long and unbalanced copper wires to the differential recording channel. This issue of PCB layout can be seen in Figure 7-37. Channel 0 used during in-vivo testing is highlighted in green box. The traces of both differential inputs to channel 0 is extended to the far left of the ADC chip with long wires. In comparison, channel 2 highlighted in blue box has much shorter traces with similar length. This is a significant contributing factor since channel 0 used in the in-vivo experiment has the worst layout among all 4 channels. Also, the long headers connected to the recording electrodes collects excessive unwanted noise.

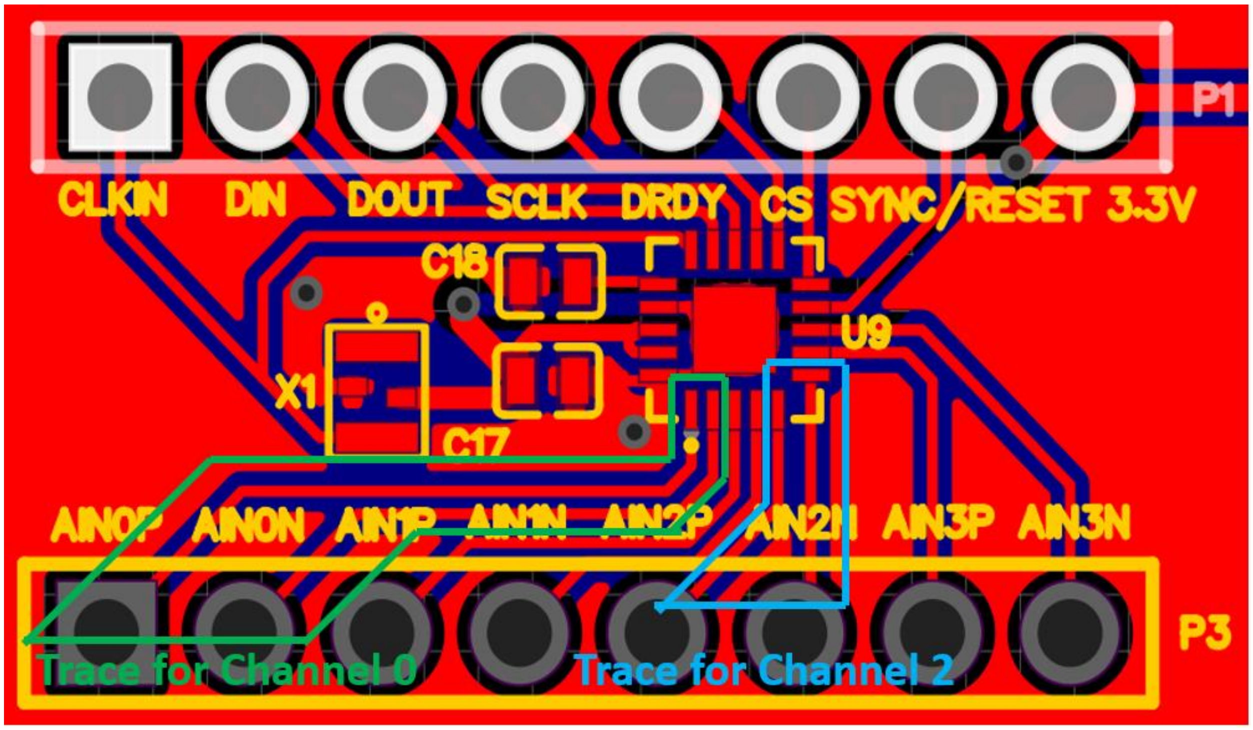


Figure 7-37: Layout of ADC

Efforts are made to reduce the noise-floor based on the result from in-vivo experiments. The gain of ADC is adjusted to 16. While noise-floor generally increases with higher sampling frequency, to better capture the high-frequency components in neural signals, no adjustment was made on the sampling rate. More importantly, channel 2 which has the shortest trace, shown in Figure 7-37 is configured to be the active instead of channel 0. In Figure 7-37, the 400µV$_{p\text{-}p}$ noise floor is contributed by both the noise introduced by PCB and the relatively large environmental noise in the surgery room in DLAM (Division of Laboratory Animal Medicine at UCLA). In a less noisy environment, the noise floor with the same setup is measured to be around 300µV$_{p\text{-}p}$. In the same environment, the ADC is configured to a gain of 16 with channel 2 enabled instead of channel 0.

Figure 7-38 below shows the noise-floor after the adjustments mentioned above. The noise-floor is decreased to around 30μV_{p-p}.

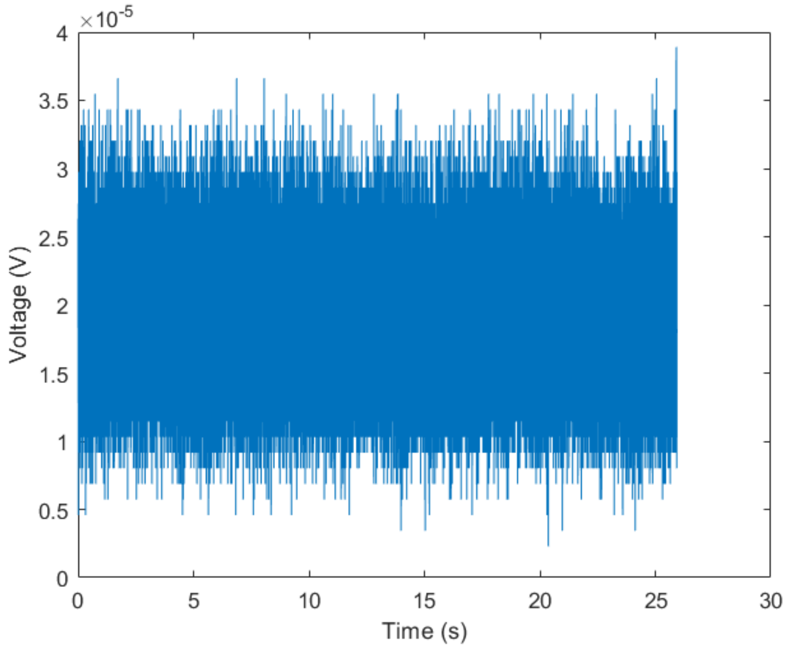


Figure 7-38: Noise Floor after Optimization

Chapter 8 : Future Work

8.1. Extension to a Multi-Channel System

Currently, the functional system supports simultaneous operation of 1 stimulating channel and 1 recording channel. The framework of the system has the potential to be extended to a multi-channel system. For recording, the current framework can accommodate up to 4 channels without modifying hardware design. Only firmware change is necessary. The main limiting factor to accommodate multiple recording channels is the insufficient bandwidth for BLE. With 2M PHY layer and data length extension, the maximum speed of BLE over time is approximately 1250 Kbit/s. Discussed in section 4.5, the minimum BLE speed requirement to avoid data loss is 256Kbit/s. When all 4 channels of ADS131M04 are enabled, the minimum required BLE speed would be 1024Kbit/s in theory. Even though 2M PHY layer with data length extension can accommodate such requirement with some redundancy, the lower signal penetration ability of 2M PHY causes concern when the final iteration of the remote device is implanted under tissue.

For stimulation, to support multiple stimulation channels, additional amplifiers and SPDT switch need to be added to PCB, which increases the overall size of the implant. Therefore, the weight and size of the implant is a limiting factor determined by the subject of experiment. Another constraint lies in the number of GPIO pins and general-purpose timers to control the stimulation waveform. Each stimulation channel consumes 3 GPIOs and 1 general-purpose timer. Consequently, the current framework can support at most 4 stimulation channels.

8.2. Artifact Cancellation on MCU

As of now, the implementation of real-time stimulus artifact cancellation relies on the computing power in backend device. This method is easy to apply without much effort to optimize the algorithm to reach the goal of real-time operation and relatively fast to produce a functional prototype. However, multiple disadvantages exist for this off-chip solution.

1. The recorded neural signal in remote device is offloaded to backend through wireless communication. Using wireless communication protocol for data exchange creates difficulty for generating a trigger signal as a cue of new periods of stimulation which is a crucial for template-subtraction-based artifact cancellation algorithm. Therefore, in current implementation, anchor points are statistically determined, which introduces potential errors.
2. Mentioned in Figure 6-8, the anchor points used to form templates are determined with the assumption that the recorded stimulus artifact are much larger than the meaningful neural signal, which does not always hold. In applications where the stimulating and recording electrodes are far apart, artifacts and signal of interests could be the same order of magnitude. In this case, the proposed algorithm fails completely.
3. As an ultimate goal, it is desired to develop a closed-loop neural interface based on the current framework. As an important feature, closed-loop neural interface should be capable of eliciting stimulation to the subject, and adjusting stimulation parameters given the recent artifact-free neural response. With the current design and algorithm, an external backend device with high-performance CPU is a prerequisite. Not only an external device with high-end CPU is cost-ineffective, but also generate additional

delay to return parameter update command which might be significant for time-sensitive neural modulations.

Limitations above can be greatly alleviated by migrating a highly optimized artifact cancellation algorithm to be executed on MCU. Due to the limiting computing power and memory space of CC2652RSIP, extra cautions are needed to directly migrate the artifact cancellation algorithm implemented in MATLAB to MCU. Two major aspects need to be considered: size of SRAM, computation time.

Discussed in Section 2.1, the maximum SRAM available in CC2652RSIP is 88KB. To remove artifact on-chip without the support of external backend device, samples of neural signals need to be stored in SRAM temporarily to construct artifact template and perform the subtraction operation. While IIR filter and other hyper-parameters occupy space of SRAM, ADC samples are the major limiting factor since thousands of samples are required to form artifact template for low-frequency stimulation. Take the low-frequency stimulation protocol in Table 4-12 as an example. Assume it takes 20ms for the compartment voltage settles. The stimulus artifact duration can be calculated as the following to estimate the minimum duration of artifact template/segment.

$$\begin{aligned} Stimulus\ Artifact\ Duration &= burst\_count\ \times burst\_period\ +\ voltage\_settle\_time \\ &= 20 \times 9\ + 20 = 200ms \end{aligned}$$

Then, the minimum number of samples for artifact template/segment would be:

$$Number\ of\ Sample\ for\ Artifact\ Segment = 0.2 \times\ Sampling\ Rate = 3200$$

As the resolution of ADS131M04 is 24-bit, a single-precision floating number which takes 4-bytes memory space, can be used to store artifact segments. Therefore, the memory space needed for each artifact segment is:

$$Bytes\ for\ Artifact\ Segment = \ Number\ of\ Sample\ for\ Artifact\ Segment \times 4 = 12800$$

Discussed in section 6.2, for the optimal performance, artifact template should be the average of 10 to 20 artifact segments. However, with 88KB SRAM, the maximum number of artifact segments CC2652RSIP can accommodate would be the following:

$$Maximum\ Number\ of\ Artifact\ Segment = floor(88KB * \frac{1024}{12800}) = \ 7$$

In terms of computation time, CC2652RSIP is equips with hardware division and single-cycle multiply instruction [3], which significantly reduces computation time when performing average calculation and filtering the input signal. In CC2652RSIP, multiplication operation takes 1 clock cycle and division operation takes 2-12 clock cycles depending on the values.

For the IIR filter used to remove DC drifting, in the current implementation in MATLAB, a 4th-order IIR high-pass filter achieves reasonably good performance. The time-main equation of a 4th-order IIR filter in recursive form is the following:

$$y[n] = \sum_{k=0}^{4} b_k x[n-k] - \sum_{k=1}^{4} a_k y[n-k]$$

In this recursive equation, 8 addition/subtraction and 9 multiplication operations are needed for each sample. Processing time to filter 1 sample and the delay between adjacent samples can be calculated as the following:

$$Runtime\ to\ Filter\ 1\ Sample = \frac{Clock\ Cycles}{CPU\ Clock\ Frequency} = \frac{(8 \times 1 + 9 \times 1)}{48000000}) = 350ns$$

$$Delay\ between\ Adjacent\ Samples = \frac{1}{Sampling\ Rate} = 62.5\mu s$$

With hardware acceleration for multiplication, runtime to filter 1 sample is significantly shorter than the delay between adjacent samples so that the real-time operation can be guaranteed.

To obtain the artifact template, significant amount of addition and division is needed. For the current implementation in MATLAB, the artifact template updates with every new artifact segment. Therefore, the computation time for the update of template and its subtraction from the raw input signal must be shorter than the stimulation period less the duration of pulse trains and voltage settle time. This condition can be illustrated in Figure 8-1 below.

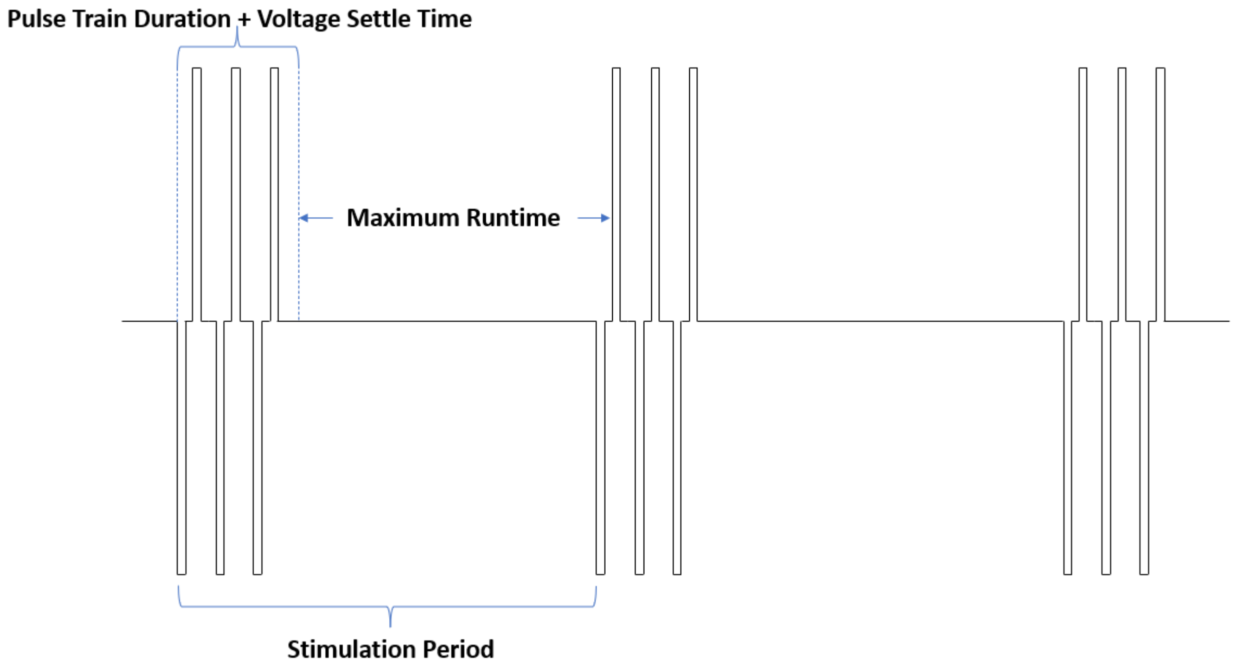


Figure 8-1: Illustration of Maximum Runtime

Again, take the stimulation protocol in table 4-12 as an example. The maximum runtime allowed is 300ms. Runtime mainly consists of for loops, summation of artifact segments, and division. Assume 7 artifact segments are used to form artifact template and 1 segment includes 3200 samples, as mentioned previously. An estimated runtime assuming division takes the maximum 12 clock cycles and 1 iteration of for loop take 20 clock cycles can be calculated as the following:

$$Runtime = For\ Loops + Summation + Division$$

$$= \frac{7 \times 3200 \times 20 + 3200 \times 7 \times 1 + 3200 \times 12}{48000000} = 10.6ms$$

Analysis above shows that the hardware accelerator for division and multiplication dramatically reduces the runtime of updating artifact template and subtracting template from raw input signal. However, the limited space of SRAM may prevent proper convergence of the artifact template due to the limited number of artifact segments used for averaging when the stimulation protocol dictates long pulse train. Therefore, other efficient matrix-operation-based method such as Kalman filter to remove stimulus artifact is worth investigating.

Reference

[1] Culaclii, S. (2019). Design of a System for Cancelling Stimulus Artifact in Multi-Channel Neural Interfaces (Order No. 27668462). Available from ProQuest Dissertations & Theses Global. (2329741339). https://www.proquest.com/dissertations-theses/design-system-cancelling-stimulus-artifact-multi/docview/2329741339/se-2?accountid=14512

[2] Culaclii, S., Wang, P. M., Taccola, G., Yang, W., Bailey, B., Chen, Y. P., ... & Liu, W. (2021). A Biomimetic, SoC-Based Neural Stimulator for Novel Arbitrary-Waveform Stimulation Protocols. *Frontiers in Neuroscience*, 943.

[3] *CC2652RSIP SimpleLink™ Multiprotocol 2.4-GHz wireless system ... - ti.com*. (n.d.). Retrieved June 2, 2022, from https://www.ti.com/lit/ds/symlink/cc2652rsip.pdf

[4] *CC1352R*. CC1352R data sheet, product information and support | TI.com. (n.d.). Retrieved June 1, 2022, from https://www.ti.com/product/CC1352R

[5] *Introduction to bluetooth low energy (BLE)*. Argenox. (n.d.). Retrieved June 1, 2022, from https://www.argenox.com/library/bluetooth-low-energy/introduction-to-bluetooth-low-energy-v4-0/

[6] Woolley, M. (2022, March 29). *Exploring bluetooth 5 -going the distance*. Bluetooth® Technology Website. Retrieved June 1, 2022, from https://www.bluetooth.com/blog/exploring-bluetooth-5-going-the-distance/

[7] Townsend, K., Cufí, C., Akiba, & Davidson, R. (n.d.). *Getting started with Bluetooth Low Energy*. O'Reilly Online Learning. Retrieved June 1, 2022, from

https://www.oreilly.com/library/view/getting-started-with/9781491900550/ch04.html#gatt_udd

[8] *Introduction to SPI interface*. Introduction to SPI Interface | Analog Devices. (n.d.). Retrieved June 1, 2022, from https://www.analog.com/en/analog-dialogue/articles/introduction-to-spi-interface.html

[9] *Understanding UART - youtube*. (n.d.). Retrieved June 2, 2022, from https://www.youtube.com/watch?v=sTHckUyxwp8

[10] Lo, Y. K., Chang, C. W., & Liu, W. (2014). Bio-impedance characterization technique with implantable neural stimulator using biphasic current stimulus. *Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Annual International Conference*, *2014*, 474–477. https://doi.org/10.1109/EMBC.2014.6943631

[11] Stanslaski, S., Afshar, P., Cong, P., Giftakis, J., Stypulkowski, P., Carlson, D., ... & Denison, T. (2012). Design and validation of a fully implantable, chronic, closed-loop neuromodulation device with concurrent sensing and stimulation. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, *20*(4), 410-421.

[12] Zhou, A., Johnson, B. C., & Muller, R. (2018). Toward true closed-loop neuromodulation: artifact-free recording during stimulation. *Current opinion in neurobiology*, *50*, 119-127.

[13] Brown, E. A., Ross, J. D., Blum, R. A., Nam, Y., Wheeler, B. C., & DeWeerth, S. P. (2008). Stimulus-artifact elimination in a multi-electrode system. IEEE transactions on biomedical circuits and systems, 2(1), 10-21.

[14] Johnson, B. C., Gambini, S., Izyumin, I., Moin, A., Zhou, A., Alexandrov, G., ... & Muller, R. (2017, June). An implantable 700μW 64-channel neuromodulation IC for simultaneous recording and stimulation with rapid artifact recovery. In 2017 Symposium on VLSI Circuits (pp. C48-C49). IEEE.

[15] Heer, F., Hafizovic, S., Ugniwenko, T., Frey, U., Franks, W., Perriard, E., ... & Hierlemann, A. (2007). Single-chip microelectronic system to interface with living cells. Biosensors and Bioelectronics, 22(11), 2546-2553.

[16] Smith, W. A., Uehlin, J. P., Perlmutter, S. I., Rudell, J. C., & Sathe, V. S. (2017, June). A scalable, highly-multiplexed delta-encoded digital feedback ECoG recording amplifier with common and differential-mode artifact suppression. In 2017 Symposium on VLSI Circuits (pp. C172-C173). IEEE.

[17] Mendrela, A. E., Cho, J., Fredenburg, J. A., Nagaraj, V., Netoff, T. I., Flynn, M. P., & Yoon, E. (2016). A bidirectional neural interface circuit with active stimulation artifact cancellation and cross-channel common-mode noise suppression. IEEE Journal of Solid-State Circuits, 51(4), 955-965.

[18] Nag, S., Sikdar, S. K., Thakor, N. V., Rao, V. R., & Sharma, D. (2015). Sensing of stimulus artifact suppressed signals from electrode interfaces. IEEE Sensors Journal, 15(7), 3734-3742.

[19] Zhou, A., Santacruz, S. R., Johnson, B. C., Alexandrov, G., Moin, A., Burghardt, F. L., ... & Muller, R. (2019). A wireless and artefact-free 128-channel neuromodulation device for closed-loop stimulation and recording in non-human primates. Nature biomedical engineering, 3(1), 15-26.