An Energy Efficient non-volatile FPGA Digital Processor for Brain Neuromodulation

by

Lijuan Xia

A thesis presented for the degree of

Doctor of Philosophy

School of Engineering
Newcastle University, UK

January 2020

Abstract

Brain stimulation technologies have the potential to provide considerable clinical benefits for people with a range of neurological disorders. Recent neuroscience studies have shown that considerable information of brain states is contained in the low frequency local field potential (lf-LFP; below 5Hz) recordings with application in real-time closed-loop neurostimulation for treating neurological disorders. Given these signals can be sampled at low sampling rate and hence provide sparse data streams, there is an opportunity to design implantable neuroprosthesis with long battery lifecycles which enables enough processing power to implement long-term, real-time closed loop control algorithms. In this thesis, a closed-loop embedded digital processor has been created for use in rodent neuroscience experiments. The first contribution of this work is to develop a mathematical analytical design approach of feedback controller for suppressing high-amplitude epileptic activity in the neuron mass model to form a better understanding of how to perform a better closed-loop stimulation to control seizures. The second contribution and the third contribution are combined to present an exploratory energy-efficient digital processor architecture built with commercial off-the-shelf non-volatile FPGAs and microcontroller for sparse data processing of brain neuromodulation. A digital hardware design of an exemplar PID control algorithm has been implemented on this proposed digital architecture. A new power computing diagram of this time-driven approach significantly reduced the power consumption which suggests that a digital combined control system of non-volatile FPGAs and microcontroller outweighs a digital control system of microcontroller with microcontroller regarding computing time cost and energy consumption supposing one microcontroller is always required. Taken together, this digital energy-efficient processor architecture gives important insights and viewpoints for the further advancements of neuroprosthesis for brain neurostimulation to achieve lower power consumption for sparse sampling data rate.

Acknowledgements

First, I would like to express my wholehearted thanks to my primary supervisor Dr. Patrick Degenaar throughout the past five years' research in the United Kingdom. He is a marvelous researcher and supervisor. I greatly appreciate the time, efforts and dedication of the wonderful supervision you have given to me. Thanks for being a strong advocate of my work and keeping my best interest in your mind. It has truly been a rewarding experience to work in your world-class collaborative neuroprosthesis lab where I got my first taste of research.

I would like to acknowledge my co-supervisors Professor Andrew Jackson and Dr. Graeme Chester. I cannot find the right words to complement their support and continuous help for me. Much of what I learned about neuroscientific knowledge came from the many engaging meetings and discussions I had with Professor Andrew Jackson over my second year PhD. Many thanks also go to my postgraduate panel Dr. Alex Bystrov for serving as my PhD annual report panel and providing invaluable feedback.

I'd also like to thank Dr. Geoff Merrett and Dr. Rishad Shafik to be part of my PhD thesis examiner committee. Your time and dedication to review this thesis are deeply appreciated by me.

My time at Newcastle was absolutely enriched by all the talented researchers I interacted with on a daily basis. Thank you with all of my heart to all the past and present colleagues in the Neuroprosthesis lab and the microsystem group, especially Dr.Ahmed Soltan, Dr. Musa Al-Yaman, Dr. Graeme Coapes, Dr. Jun Luo, Dr. Austin Ogweno, Ms.Rachael Savidis, Dr. Reza Ramezani, Dr. Fahimeh Dehkhoda and Mr.Dimitrios Firfilionis.

I would also like to thank all the lovely and friendly staff in the School of Electrical and Electronic Engineering of Newcastle University, especially those technicians who helped me with making the experimental test of proposed PCBs featured in this thesis possible.

Finally, I'd like to thank all my Newcastle gals who have supported me all the way throughout the past five years outside the lab environment. You girls are literally a ray of sunshine in my research life outside the lab environments. Helen Kenny, thanks for keeping me fit on our morning runs and after-work runs. Hang Xiang, I couldn't ask for a better friend who I can form a brain storm with on weekly basis.

My gratitude also goes to my family in Shanghai, who have encouraged and supported me to finish my PhD work. I am more than grateful that I chose to spend my dynamic young time (22-27 years) getting closer to the pursuit of truth of science and philosophy.

Hard journey, but I am healthy, happy, more present and grateful than ever.

Thank you from the bottom of my ever-expanding heart for all your support.

And so, Done and Done!

- *Gratitude for their great support*

Dedication

To my daddy (夏卫东),

Thank you for all your love, effort and continuous support. You are the best daddy.

To my mummy （陈芳）,

Thank you for giving me birth, you are the only and most lovely mommy I could have.

To my brother（夏晨峰）,

Thank you for bringing laugh to my life and keeping me company, my dearest brother.

List of Achievements

<u>Published papers:</u>

- *IEEE International conference papers*

<u>1. Lijuan Xia</u>, Ahmed Soltan, Junwen Luo, Andrew Jackson, Graeme Chester and Patrick Degenaar, "Closed-loop Proportion-Derivative Control for Suppressing Epileptic Seizure in a Neural Mass Model," submitted to IEEE International Symposium on Circuits and Systems (IEEE ISCAS2019).

<u>2. Lijuan Xia</u>, Junwen Luo, Ahmed Soltan, Andrew Jackson, Graeme Chester and Patrick Degenaar, "A Rodent Control System for Closed-loop Optogenetic Stimulation to Suppress Seizures," published in the 18th IEEE Nano Conference (IEEE Nano2018), 2018, 1-4

<u>3. Lijuan Xia</u>, Ahmed Soltan, Junwen Luo, Graeme Chester and Patrick Degenaar, "*A Flash-FPGA based Rodent Control System for Closed-loop Optogenetic Control of Epilepsy*," published in the *51th International Conference of the IEEE International Symposium on Circuits and Systems (IEEE ISCAS2018)*, 2018, 1-5.

<u>4. Lijuan Xia</u>, Nabeel Fattah, Ahmed Soltan, Andrew Jackson, Graeme Chester and Patrick Degenaar, *"A Low Power Flash-FPGA based Brain Implant Micro-System of PID"*, published in the 39th *International Engineering in Medicine and Biology Conference (IEEE EMBC2017)*, 2017, pp. 8006-8009

5. Ahmed Soltan, **<u>Lijuan Xia</u>,** Andrew Jackson, Graeme Chester and Patrick Degenaar, "Fractional Order PID System for Suppressing Epileptic System", published in 4th IEEE International Conference on Applied System Innovation(IEEE ICASI 2018)

<u>Under Review Manuscript:</u>

- *IEEE Journal Paper*

I. **<u>Lijuan Xia</u>**, Ahmed Soltan, Junwen Luo, Andrew Jackson, Graeme Chester and Patrick Degenaar, " *An Energy-efficient Reconfigurable Closed-loop Processor for Sparse Data Processing of Brain Neuromodulation*," Submitted to IEEE Transactions on Biomedical Circuits and Systems. (preparing for resubmission)

<u>PhD Research Awards</u>

I. Doctoral Training Awards Conference, School of Engineering, Newcastle University, 23/24 April 2017, best poster awards and best presentation awards.
II. Doctoral Training Awards Conference, School of Engineering, Newcastle University, 23/24 April 2018, best presentation awards.
III. Newcastle University Doctoral Training Awards (2014-2018).
IV. Newcastle University Oversea Research Scholarship (2014-2017).

International and local Symposium

I. School of Electrical and Electronic Engineering, Newcastle University, Annual Research Conference 23/24 January 2015, poster.
II. School of Electrical and Electronic Engineering, Newcastle University, Annual Research Conference, 21/22 January 2016, (Oral Presentation – 12 minutes).
III. School of Electrical and Electronic Engineering, Newcastle University, Annual Research Conference, 21/22 January 2017, (Oral Presentation – 12 minutes).
IV. Lecture Presentation in The International Symposium on Circuits and Systems (ISCAS2018, Italy, Florence) , (Oral Presentation – 15 minutes).
V. Poster Presentation in The IEEE Nano conference (IEEE nano 2018, Ireland, Cork)

- *Group presentations*

28 lab presentations to the biomedical lab group.

List of Acronyms

Ψ	Magnetic Flux
μ	Permeability
µLED	Micro Light-Emitting Diode
AP	Action Potential
ADR	Average Detection Rate
ALU	Arithmetic Logic Unit
AM	Amplitude Modulated
AED	Anti-epileptic Drugs
ASK	Amplitude Shift Keying
BGA	Ball Grid Array
BLE	Bluetooth Low Energy
BPSK/QPSK	Binary/Quadrature Phase Shift Keying
CANDO	Controlling Abnormal Network Dynamics Using Optogenetics
CCK	Complementary Code Keying
CCS	Current Controlled Stimulation
ChR2	Channel-Rhodopsin 2
COFDM	Coded Orthogonal Frequency Division Multiplexing
CRC	Cyclic Redundancy Check
DBS	Deep Brain Stimulation
DC	Direct Current
DFT	Discrete Fourier Transform
DWT	Discrete Wavelet Transform
DSP	Digital Signal Processing
EEG	Electroencephalogram
EI	Epileptogenicity Index
FIR	Finite Impulse Response
FDA	Food and Drug Administration
FFT	Fast Fourier Transform
FPGA	Field Programmable Gate Array
FP	False Positive
FN	False Negative
IC	Integrated Circuit
ICA	Independent Component Analysis

TP	True Positive
TN	True Negative
JPEG	Joint Photographic Experts Group
JTAG	Joint Test Action Group
KDS	Kinetis® Design Studio
LFP	Local Field Potential
MCU	Microcontroller
M-QAM	M-Ary Quadrature Amplitude Modulation
MSE	Mean-Square Error
N	Number of Turns
NMM	Neural Mass Model
NI	National Instrument
NORD	National Organization for Rare Disorders
OCD	Obsessive Compulsive Disorder
Optrode	Optical Probes
PCA	Principle Component Analysis
PCB	Printing Circuit Board
PD	Proportional Derivative
PID	Proportional Integral Derivative
PCB	Printing Circuit Board
PCB	Printing Circuit Board
PID	Proportional Integral Derivative
SAR	Specific Absorption Rate
SLA	Stereo lithography Apparatus
STL	Stereo Lithography
SoC	System on Chip
UPS	Uninterruptible Power Supply
UWB	Ultra-Wideband
VHSIC	Very High-Speed Integrated Circuit
VHDL	VHSIC Hardware Description Language
VNS	Vagus Nerve Stimulation
USEA	Utah Slanted Electrode Array
WHO	World Health Organization
WPT	Wireless Power Transfer

Relative contribution

This table shows the joined efforts between me and our teammates who are involved in Control Abnormal Neuron Dynamics by Optogenetics (CANDO) projects from 2014 to 2018. They contributed to this PhD research projects either by providing research insights or by technical assistance.

Component	**Type**	**Correspondents**	**Comments**
FPGA PCB Board Design	Hardware	Miss Lijuan Xia	Altium Designer
MCU PCB Board Design	Hardware	Miss Lijuan Xia	Altium Designer
FPGA FSM Master Implementation	Software	Miss Lijuan Xia and Mr. Dimitrios Firfilionis (jointly)	VHDL (Microsemi Libero) C++ (Kinetis Design Studio)
FPGA FSM Slave Implementation	Software	Miss Lijuan Xia and Mr. Dimitrios Firfilionis (jointly)	VHDL (Microsemi Libero) C++ (Kinetis Design Studio)
PID Algorithm Implementation	Software	Miss Lijuan Xia	VHDL (Microsemi Libero) C++ (Kinetis Design Studio) Matlab

Optical Algorithm Implementation	Software	Miss Lijuan Xia and Dr. Patrick Degenaar (jointly)	VHDL (Microsemi Libero) C++ (Kinetis Design Studio)
Black Box Design and Fabrication	Hardware	Miss Lijuan Xia and Mr. Jeffrey Warren (jointly)	Technician
GUI Design for communicating with Black Box	GUI	Miss Lijuan Xia	Matlab
Experimental Testing	Experimental Testing	Miss Lijuan Xia	**1.Power Supply:** Keysight E3648A Dual Output DC Power Supply **2.Current Multimeter:** Keysight 34460A Digit Multimeter **3.Oscilloscope:** Agilent Technologies MSO-X 4034A Mixed signal oscilloscope **4.Waveform Generator:** Keysight 33500B Series Waveform Generator **5.Power Analyzer:** Agilent Technologies N6705B DC Power Analyzer

Contents

Chapter 1. Introduction

1.1 Motivation

In recent decades, millions of people have been affected by epileptic seizures, and this number continues to rise [1] [2]. With accelerating progress in responsive neurostimulation research, there has been a growing interest in developing closed-loop implantable neurostimulators aiming to treat drug-resistant epilepsy. Most existing implantable devices apply sensor probes to record brain activities (EEG, LFP, MCG etc.) [3] [4]. Moreover, these neurostimulators' control units have been designed to interface with recording sensor probes to deliver stimulation inside the brain to help control epileptic neurons [5] [6]. One of the most successful commercial devices to date, the Neuropace responsive stimulator, attempts to analyse brain recordings in order to detect seizure patterns prior to seizure intervention by delivering electrical charges. This technology is defined as supervised open loop stimulation [7]. The weakness of the supervised open loop stimulation strategy is that seizure detection accuracy has a direct effect on the stimulation performance. In other words, any false alarms generated by the seizure detection algorithms will lead to false stimulations which can cause potential security issues [8]. Hence, the open research question arises: how to determine real-time closed-loop control algorithms to help suppress seizures, and how to implement the proposed control algorithms into battery-powered hardware devices with flexible reprogrammability for epilepsy treatment research.

However, most published closed-loop controller design work are mathematical modeling-based [9], [10], [11]. And hence it is difficult to map those algorithms in wearable or implantable hardware devices with a reasonable power consumption. Even though engineers are capable of attemping to map complicated and sophisticated supervised open loop algorithms and closed-loop control algorithms in implantable/wearable devices, it will lead to a certain level of power-hungry design with limited reprogrammability [12], [13]. This can cause a series of issues for implantable devices, as most implantable neurostimulators are typically required to operate on the limited power budget of a wearable battery and desire reprogrammability during the neuroscience experiment tests. These results demonstrate that a simple closed-loop reprogrammable control algorithm is needed for hardware designs in the closed-loop neuroprosthesis to help treat epilepsy to further benefit biological experiments.

In this thesis, in order to help answer the open question, we shall study the feasibility of this next generation low-power neurostimulators' implementation to deliver closed-loop stimulation for controlling seizures. We will focus on the theoretical exploration of a feasible closed-loop control algorithm to suppress seizures, and propose the feasibility of mapping the proposed algorithm into a digital controller with limited power consumption.

1.2 Current commercial neurostimulator review

The first FDA clinically-proven implantable device for epilepsy treatment is the Vagus nerve stimulation device (VNS) [14]. A VNS device is a small electrical device similar to a pacemaker which is placed under the skin of the chest for delivering bursts of electricity. The electrical charge is sent to a probe in the Vagus nerve which can help change the electrical signals in the brain. A VNS device operates on a wearable battery and develops open-loop stimulus by delivering a fixed frequency electrical pulse. However, the VNS device cannot cure epilepsy fundamentally, and can only help make epilepsy symptoms less severe and less frequent. Hence, the most recent widely reported therapy is responsive neurostimulation (RNS)[15]. The RNS system is a device or stimulator which is surgically placed on the bone covering the brain. The stimulator delivers small pulses of stimulation to the implanted brain area whenever abnormal brain activity is detected by seizure detection algorithms. The RNS system has been clinically proven to reduce seizures and improve patients' life quality in some cases. RNS was approved by the U.S. Food and Drug Administration (FDA) in 2013.

In recent years, there have been efforts to advance the current existing open loop brain stimulation towards a closed-loop brain control system which can deliver therapeutic modulation as well as providing fixed frequency stimulations [16]. The main efforts of the closed-loop control systems can fall into two categories: low power hardware architecture and closed-loop control algorithms to suppress seizures.

From a hardware perspective, neuroengineers have made extraordinary strides in integrating sophisticated specific integrated circuits consisting of basic electronic elements such as transistors, resistors and capacitors onto tiny silicon chips for implant surgery in a low power, low thermal, and small sized manner[17] [18] [19]. Recent progress in neuroprosthesis research has led neuroengineers and neuroscientists to record brain activity of electroencephalograms (EEG) [20] and local field potential (LFP) [18], analyse the recordings and deliver subsequent treatments in real-time by means of closed-loop control systems [21]. Implantable silicon chips have been reported to perform electrical recordings, multichannel recordings with activity extraction, electrical stimulation and optical stimulation [22].

From an algorithm modelling perspective, closed-loop controllers for suppressing epileptic seizures have been proven to be a promising strategy for suppressing seizures. Proportional algorithms have been used to control seizure amplitude in rats. Integral control is employed to provide feedback for the charge-balanced suppression of seizures [23]. Differential control models are used to eliminate activity in a theoretical math model of human cortical and electrical activity [24]. However, there are two weakness to the above algorithm work. One is that for closed-loop PID based algorithm work, the control parameters are suggested or picked up based on the designers' experience with a 'trial and error' approach. Secondly, most of the algorithms are theoretical modelling-based and do not have a feasible hardware implementation for applying them in controlled rodent neuroscience experiments.

To conclude, the most common way to understand how neuron network works is to do electrical physiological recordings and picking up the electrical signals generated by the neurons. Recent neuroscience efforts report that considerable information about the brain's state is contained in low-frequency local field potential recordings in real-time closed loop neurostimulation for neurological disorders [25]. Given that these recording signals can be sampled at low rates thus providing a sparse data stream, there is an opportunity for bioengineers to design implantable neuroprosthetics with long battery lifecycles and sufficient processing power to implement a long-term, real-time and closed loop control algorithm. The key objective of this research project is to explore the next generation of embedded optogenetic-optoelectronic brain implants for application in controlled rodent neuroscience experiments. Hence once a device is created, it can be used to explore and generate stimulus for the modulation of in-vitro epileptic activity using closed-loop stimulation in rodent brain slices.

1.3 Contribution and organisations

This thesis concerns how to build the next-generation of energy-efficient digital processors for low-sampling-rate data processing to deliver neurostimulation to targeted neuron networks. The major contributions of this thesis are listed as follows:

1. ***The closed-loop PD control framework in brains:*** The first advancement that we have proposed is an analytical approach to closed-loop Proportional-Derivative (PD) control that can be applied to determine the stimulation parameters for suppressing high-amplitude epileptic activity in a neural mass model. This allows us to explore the relationship between the model parameters of inducing seizures and the PD feedback controller parameters of stabilising seizures which helps develop a better understanding of how best to suppress epileptic seizure activity by applying closed-loop stimulation. This computational modelling work parallels the *in vitro* closed-loop optogenetic stimulation experiments.

2. ***The comparison efforts between different digital platforms (MCUs and FPGAs).*** A comparison study of non-volatile FPGAs was conducted and shows some extinct properties compared to other digital platforms (MCU, GPU, DSP etc.). Hence, another contribution is the feasibility study of flash-based FPGAs for this application by comparing FIR filter implementations on a microcontroller and an FPGA.

3. ***The energy efficient digital processor design:*** A rodent wearable digital processor has been built using a commercial off-the-shelf non-volatile FPGA and microcontroller platform for low-sampling-rate data processing. Taking advantage of the distinct flash freeze mode of non-volatile FPGAs, a co-processor MCU can be programmed to send a pulse to a non-volatile FPGA to enable entering and exiting an ultra-low power flash freeze (sleep) model to save on energy consumption (8.032uA). A new power computing diagram based on non-volatile FPGAs and microcontroller architecture have been integrated onto a 2.5cm x 2.5cm PCB board.

1.4 Thesis outline

Following the thesis motivation is constructed as follows:

Chapter 2: **<u>Medical background and literature review.</u>** This chapter reviews the relevant background and state-of-the-art optogenetic implants for brain neurostimulation. Firstly, the medical background of epilepsy seizures is introduced, followed by a review of epilepsy treatments. The focus then moves onto reviewing and comparing the current progress of neuroprosthesis for seizure control. Finally, the hardware implementation publications of neuroprosthesis are justified.

Chapter 3: **<u>Closed-loop control of the brain</u>**. This chapter presents an analytical approach to closed-loop PID controls to determine stimulation parameters inside the stabilisation area for suppressing high amplitude epileptic seizure activity generated by a neural mass model. The model suggests that the PID control algorithm with appropriate PID parameter settings within the stabilisation area can help control high amplitude epileptic activity generated from a neural mass model. This chapter then details the possible feasible design of optimised hardware implementation of the proposed PID control algorithm.

Chapter 4: **<u>System implementation.</u>** This chapter depicts the system implementation of the proposed algorithm developed in chapter 3. Different digital hardware platforms have been compared from different prospects, then an optimized PID control algorithm has been presented in chapter 4.

Chapter 5: **<u>Energy efficient digital processor case study.</u>** This chapter details a low power digital processor built by non-volatile FPGAs and MCU chips for this low-sampling-rate data processing. Then an energy-efficient reconfigurable closed-loop processor has been employed to interface with an implantable device to carry electrical recordings and optogenetic stimulation for brain neurostimulation.

Chapter 6: **<u>Conclusion.</u>** Chapter 6 summarizes the main work of the thesis and concludes the contribution of the thesis. Future work is also presented in this chapter.

Chapter 2. Medical background and literature review

2.1 Chapter overview

In recent decades, responsive neurostimulation has been recognized as potential great potential alternative to help treat drug-resistant epilepsy (DRE). The key challenge of how to build a highly portable and reliable integrated neural interface with more responsive control algorithms for seizure suppression is still an open question. In other words, how can we best design intelligent algorithms for seizure suppression and map those proposed control algorithms into battery-powered hardware platforms before scientists can apply this promising technology for further commercialization. Mathematicians and neuroscientists are working on designing more intelligent responsive neurostimulation algorithms to deliver stimulation strategies to achieve better experimental results. Electrical engineers are currently working towards designing lower power consumption electrical systems to deliver stimulations within a battery powered device. In recent years, there have been joint efforts between mathematicians and neuroscientists to investigate different control algorithms to deliver neurostimulation to treat epilepsy. This includes supervised open loop stimulation algorithms and closed-loop stimulation algorithms. Supervised open loop stimulation can detect seizures before they happen or even predict seizure patterns based on real-time recordings [26]. Once seizure patterns are detected or predicted, this responsive neurostimulation system can deliver an electrical stimulation or other stimulation method to stop the seizures. Closed-loop control algorithms refer to continuous closed-loop control in order to stabilise network dynamics and prevent the development of seizures [27]. From a hardware perspective, engineers are working on system-level designs of miniaturised, low-power neural interface implementation of supervised open loop algorithms and closed-loop control algorithms to generate real time stimulation for seizure suppression [28]. However, implementing a highly portable and reliable integrated neural interface is still an open question. This chapter is designed to provide a systematic review of neurostimulation methods for seizure suppression from an algorithm perspective and hardware perspective. Hence, we shall explain where the research opportunities lie in the field of closed-loop digital controllers to implement closed-loop algorithms.

Section 2.2 will review the neurostimulation in detail, and section 2.3 will provide a systematic review of hardware. Section 2.4 will conclude this chapter and offer an outlook for the following chapters.

2.2 Overview of current neurostimulation methodology

Epileptic seizures can be defined as a neurodisorder disease inside the brain characterised by an enduring predisposition to having seizures [29]. There are about 30 different epileptic seizure syndromes which can be subdivided into three main categories: spreading seizures, widespread seizures and focal seizures. Focal onset seizures refer to partial seizures, meaning a seizure only happens in one area of the brain. In this project, we are mainly interested in studying neurostimulation strategies to help treat focal onset seizures. In the rest of this thesis, we will refer to focal onset seizures by using the term “seizures”. Figure 2.1 shows a 10-second pre-recording dataset plot of seizures onsets in the brain cortex of a rodent.

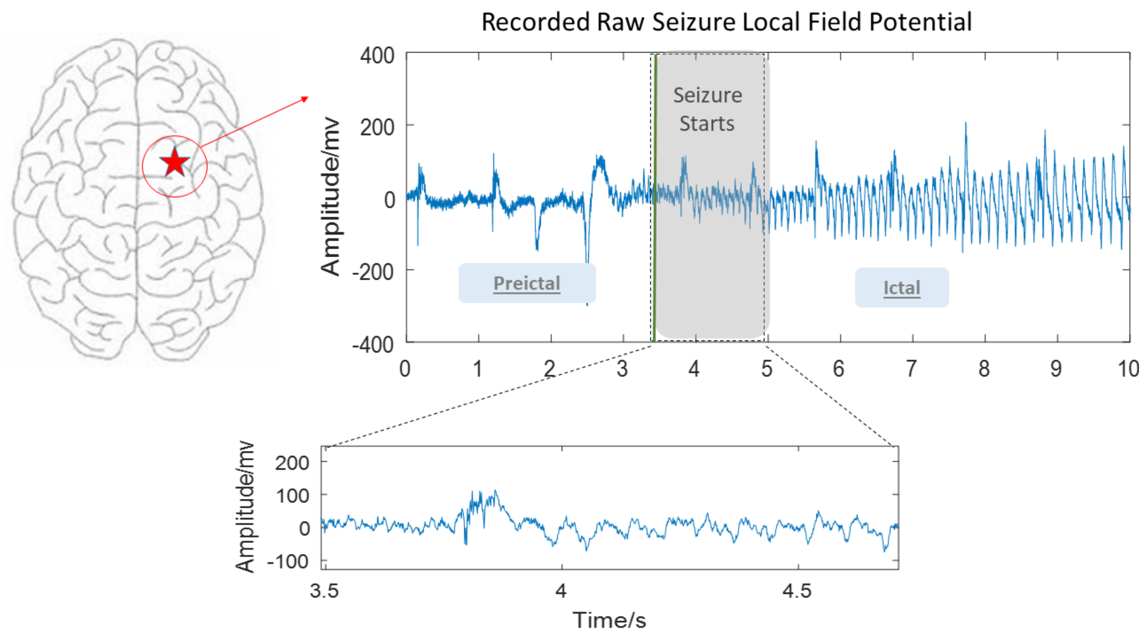


Figure 2.1: Seizure onset local field potential recordings from a rat’s cortex lasting 10 seconds. The dataset was provided by Professor Andrew Trevelyan from the Institute of Neuroscience at Newcastle University, recorded in Trevelyan’s lab in 2015.

For most epilepsy patients, seizure treatment starts with medication. For drug-resistant epilepsy patients, a combined analysis (EEG, computerized tomography, magnetic resonance imaging etc.) is used to diagnose the specific seizure type for patients and where is the potential seizure onset area [30]. An accurate diagnosis of a patient's seizure conditions gives patients the best opportunity for effective treatment.

Early stage seizure control can be achieved by applying appropriate stimulation technologies to seizure onset networks using implantable microelectronics. Researchers have made promising progress with three commercially-produced biomedical neurostimulation devices for epilepsy treatments: Deep Brain Stimulation (DBS) [31], Vagus Nerve Stimulation (VNS) [32], and Responsive Neurostimulation System (RNS)[33].

I. Vagus Nerve Stimulation (VNS)

VNS can be defined as a medical treatment process that involves the implantation of a battery-powered device underneath the skin of a patients' chest, which delivers electrical stimulation to the Vagus nerve. A second small incision is made in the neck to attach two tiny wires to the Vagus nerve. The wires are threaded invisibly up the neck from the device to the Vagus nerve. Bursts of electricity are sent via the wire to the Vagus nerve. For epilepsy treatment, these electrical pulses are delivered to the Vagus nerve affecting where seizures are assumed to start in the brain and may help to prevent abnormal electrical activity.

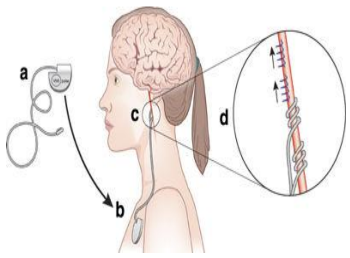
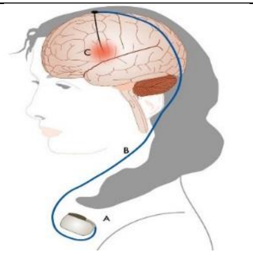
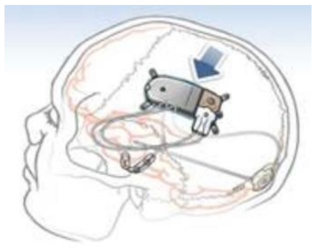
II. Deep Brain Stimulation (DBS)

DBS is a neurosurgical procedure which involves the surgical implantation of an invasive device into the brain for delivering electrical stimulation into a targeted area of the brain. Bursts of electricity are sent along wires which can help to prevent seizures by changing electrical signals in the brain.

III. Responsive Neurostimulator System (RNS)

RNS is also the world's first implanted neurostimulator for epilepsy approved by the US Food and Drug Administration (FDA) for clinical use. The main RNS system is a small implantable device that is adjustable and reversible. It is tailored to different patient cases regarding where it is placed and how it is used. RNS mainly involves a device (stimulator) placed inside the skull. Tiny wires and leads are placed on the seizure focal onset area for delivering stimulus. The main RNS devices analyse brain activity patterns to detect seizure patterns before they happen and deliver an electrical charge or drug to stop seizures.

Table 1 : A general overview of VNS, DBS, and RNS [34]

Categories	Figure Demonstration	Hardware Structure
VNS	 typical 50% improvement on 75% of patients	First approved: 1994 Stimulus target: vagus nerve Stimulus strength: 0.25-2mA Stimulus frequency: 30Hz Duty cycle: 30second on, 5 mins off
DBS	 85% (of total cohort of 40 children) saw reductions, some up to 100% with an overall 78% reduction	Stimulus target: thalamus Stimulus strength: 2mA Stimulus frequency: 130Hz Duty cycle: 90-450 us on,
RNS	 data not shown yet	First approved: 2013 (pre market) Recording target: cortical surface above seizure focus Stimulus target: seizure focus Stimulus strength: 3mA (0.5-12mA) Stimulus frequency: 100/200Hz Duty cycle: 169 us on, 100 ms off

Tables 1 compares the three main epilepsy treatment methods. Besides AED treatment and dietary treatment, the main treatments for epilepsy can be generally divided into the following categories, specifically the commercial biomedical neurostimulation devices DBS, VNS and RNS can be described as follows:

I. Deep Brain Stimulation (DBS)

DBS is a neurosurgical procedure which involves the implantation of an invasive device by surgery into the brain for delivering electrical stimulation into the targeted area of the brain. DBS can be also seen as an alternative supplementary treatment where a part of the brain can be stimulated to stop symptoms of the condition. Bursts of electricity are sent along wires which can help to prevent seizures by changing electrical signals in the brain. The DBS approach can also be employed to control a variety of debilitating neurological diseases (essential tremor, Parkinson's disease, obsessive compulsive disorder, epilepsy etc.).

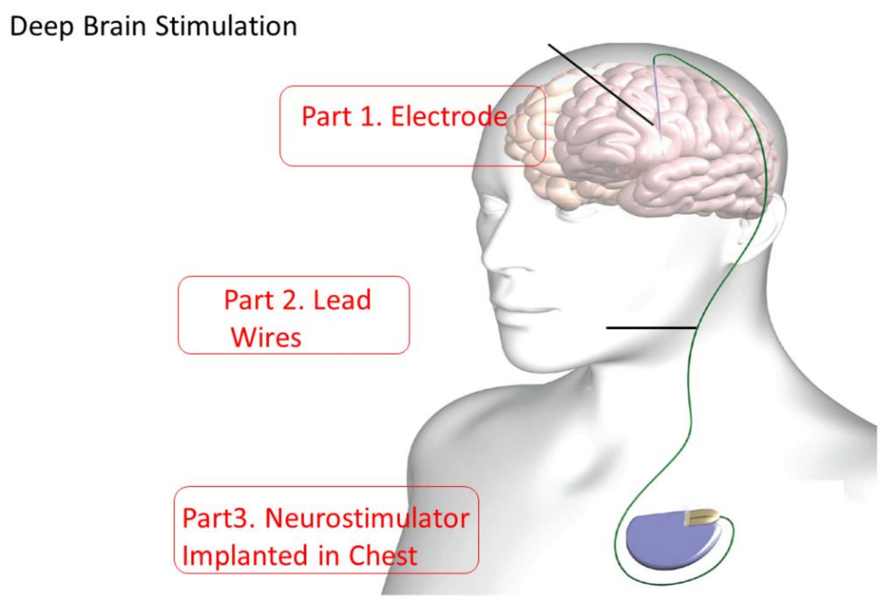


Figure 2.2: It shows a simplified schematic of a DBS system implanted in the body of a human being from Massachusetts General Hospital's website. It contains an electrode, a lead and a generator. The electrode is implanted inside the brain and the generator is implanted in the chest. Brain surgery is required for DBS systems. All parts of the device are subdermal.

For DBS treatment, brain surgery is necessary for fitting the device into patient's brain. Figures 2 -2 demonstrate the procedures involved in deep brain stimulation surgery:

1) Part 1: ***the electrode***:
   A thin, insulated electrode is put through small openings (incisions) on top of the skull for reaching the epileptic seizure onset area to deliver electrical stimulus to the target site [35].

2) Part 2: ***the wire***:

   An insulated wire is passed under the skin of the head, neck and shoulder to connect the electrode to the neurostimulator [36].

3) Part 3: ***the implanted pulse generator***:

   A battery-powered neurostimulator (pulse generator) is placed under the skin near the collarbone to send off electrical pulses to the brain that interfere with the neural activity at the target site [37].

Possible side effects of DBS approach include the following:

1. Wound infection after the operation
2. Complications if the device malfunctions
3. Complications after the surgery, such as bleeding in the brain
4. An increase in symptoms of depression and anxiety

Efficacy of DBS:

Recent trials show a modest improvement in seizure reduction, about a 15% reduction of seizure onset frequency [38].

II. Vagus Nerve Stimulation (VNS)

VNS can be defined as a medical treatment process that involves the implantation of a device which delivers electrical stimulation to the Vagus nerve. It can be viewed as a supplementary treatment alternative for treating intractable epilepsy. In addition, VNS can also be used to treat drug-resistant depression which does not respond to typical depression therapies. Nowadays, scientists and researchers are investigating applying VNS as a potential supplement treatment for a variety of conditions (multiple sclerosis, headaches, pain and Alzheimer's disease).

The Vagus is the tenth cranial nerve and arises from the medulla that carries both afferent and efferent fibres. The afferent Vagus fibres connect to the nucleus of the solitary tract which connects to the central nervous system. In conventional Vagus nerve stimulation approaches, the general procedure is to put a small generator similar to a matchbox-size pacemaker under the skin below the left collarbone. A second small incision is made in the neck to attach two tiny wires to the Vagus nerve. The wires are threaded internally up the neck from the device to the Vagus nerve. Bursts of electricity are sent via the wire to the Vagus nerve. For epilepsy treatment, these electrical pulses are delivered to the Vagus nerve affecting where seizures are assumed to start in the brain, and can help to prevent abnormal electrical activity. This device then sends impulses to the brain to prevent the electrical activity which causes seizures.

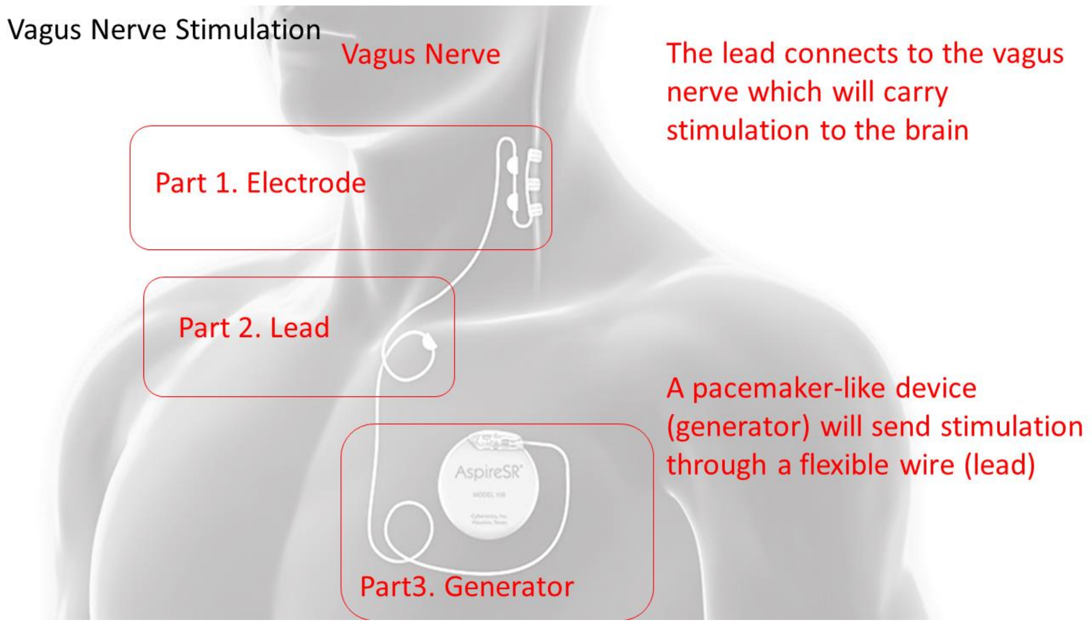

Vagus Nerve Stimulation
Vagus Nerve
The lead connects to the vagus nerve which will carry stimulation to the brain
Part 1. Electrode
Part 2. Lead
A pacemaker-like device (generator) will send stimulation through a flexible wire (lead)
AspireSR
Part3. Generator

Figure 2.3: It shows a simplified schematic description of VNS systems implanted in a human body, as shown on a VNS Therapy Website. It contains an electrode, a lead and a generator. The electrode is implanted in the Vagus nerve and the generator is implanted in the chest. AspireSR is the first and only VNS therapy that provides responsive stimulation. Brain surgery is not required for VNS treatment.

For VNS treatment, brain surgery is not necessary. There are two major electrical devices in the VNS therapy shown in Figure 2.3:

1) Part 1: ***the lead wire***

   The lead wires from the generator are tunnelled up through a patients' neck and wrapped around the left Vagus nerve for delivering electrical stimulus to the Vagus nerve [39].

2) Part 2: ***the generator***

   An implantable electrical device (similar to pacemaker) is implanted under the skin below the left collarbone [40].

Possible problems caused by the VNS system are as follows:

1. Coughing
2. Headaches
3. Neck pain
4. Sore throat
5. Difficulty breathing

Efficiency of VNS:

Recent publications show that VNS offers a mean seizure reduction of 28% for patients, with 23% of patients having reduction in seizure frequency over 50%.

III. Responsive Neurostimulator System (RNS)

Responsive Neurostimulator System, also known as RNS therapy, is the world's first implanted neurostimulator for epilepsy treatment to be approved by the US Food and Drug Administration (FDA) for clinical use. The main RNS system is comprised of a small implantable device that is adjustable and reversible. It is tailored to different patient cases regarding where it is placed and how it is used. The RNS device is similar to a heart pacemaker, but instead it can monitor brain waves and respond to seizure-like brain activity. RNS mainly involves a device (stimulator) placed under the skull. Tiny wires and leads are placed on a seizure focal onset area for delivering stimulus. Based on the RNS control algorithms, the main RNS devices generate small pulses or bursts of stimulation to the brain when abnormal brain activity is detected. The systems can help to stop seizures before the actual seizures spread [41]. The RNS procedure is reversible and can be turned off or taken out if it does not work.

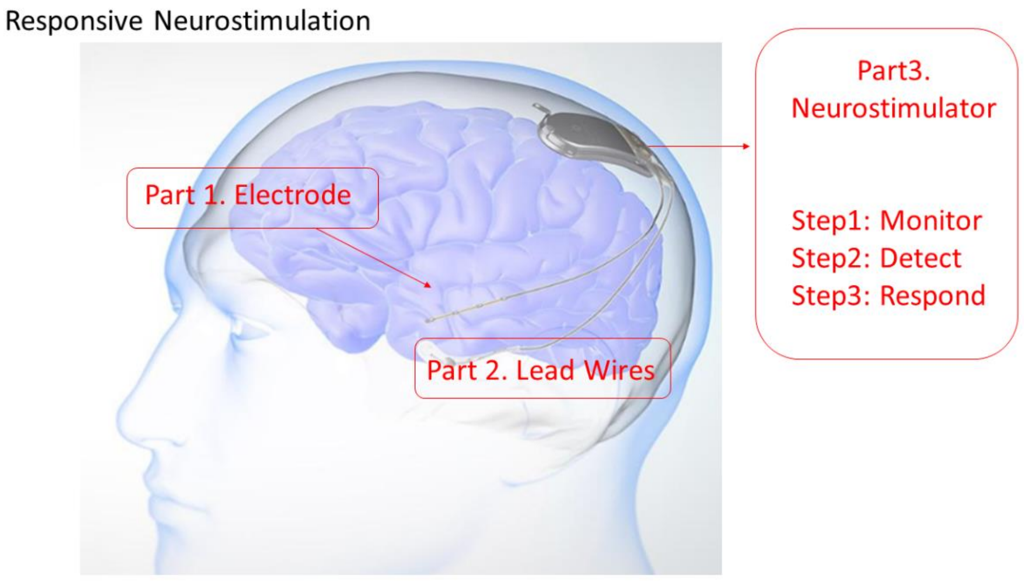


Figure 2.4: A simplified schematic description of RNS systems implanted in a human body, as demonstrated on the NeuroPace RNS® website. It displays an electrode, a lead and a generator. An electrode and a generator are implanted inside the skull [42]. Brain surgery is required for RNS systems.

The main RNS system is composed of three parts:

1) Part 1: ***the electrode***

   The electrode lead is placed under the skull.

2) Part 2: ***the wire***

   The tiny wire is fed under the skin.

3) Part 3: ***the stimulator***

   The stimulator is placed in one or two places on top of the skull where the epilepsy activity might occur according to different patients.

The RNS system continuously monitors brain activity and aims to generate electrical stimulus when seizure patterns are detected. The main procedures shown in Figure 2.4 are listed as follows:

1. Step 1: ***Monitor***

   The RNS system records brain activity in real-time.

2. Step 2: ***Detect***

The RNS system is programmed to recognise unusual or abnormal electrical activity which might lead to seizures.

3. Step 3: ***Response***

   Once abnormal brain activity is recognised, the system will respond to the neuron network by delivering pulses of electrical stimulation. The objective is to help bring the brain's electrical activity back to its normal state.

The challenges of RNS are:

1. Identifying patients who would benefit from the RNS system.
2. Identifying how best to treat these patients with the optimal stimulation strategy.

Efficacy of RNS:

Recent clinical report results show that the median frequency reduction was 56%, and the mean reduction was 43% - 100%. Observation of this study suggests that automated seizure detection positively affects electrographic seizure activity. However, this study is still the preliminary phase of trials [43] .

The specification of three responsive neurostimulation systems for controlling epileptic seizures are listed in the following table 2.2. A comparison between DBS, VNS and RNS has also been presented from both clinical and engineering perspectives. Furthermore, a more general comparison of two different categories of epilepsy treatment (brain resection surgery and neurostimulation devices) is presented in table 2.3.

Table 2.2 illustrates that brain responsive stimulation can provide great hope for patients with medically resistant epilepsy for reducing their seizures. The main challenge for biomedical engineers is how to use integrated circuit technology to create safe, robust and smaller responsive closed-loop electrical devices (such as neural interfaces, brain implants, brain machine interfaces etc.).

- **Robust:** Robust refers to low noise, low power consumption, reliable, high performance and high-security in general [44].
- **Smart:** Smart refers to how a device can listen to neurons, understand neurons and extract information accordingly, then transmit the data out and simultaneously stimulate the neurons by feedback control algorithms [45].

A comparison between our neurostimulation systems and other existing neurostimulation systems has been listed in table 2.4 from recording, stimulation perspectives.

Table 2.2: Comparison of Closed-loop Control Systems in our work.

Treatment	Neural Recording	Optogenetic Stimulation	Details
VNS	Yes	Yes	Open Loop Stimulation of Vagus Nerves
DBS	Yes	No	Open Loop Stimulation of Focal Onset Neurons
RNS	Yes	No	Electrical recording for seizure onset area, open loop electrical stimulation near seizure onset area.
Our Work	Yes	Yes	Continuous closed-loop ontogenetic intervention for ongoing discharges of characteristic of seizures

2.3 Proposed next generation neuroprosthesis

Epilepsy has been widely recognized as an induction in normal brain activity under various trigger conditions in neural networks, in rats and humans [46] [47] . References [48] describe several neural mass network models for studying dynamic mechanisms of neocortical focal seizures from different perspectives of computational modelling and system theory . Publication [49] demonstrates that abnormal values of the external input can generate high amplitude epileptic activity in the Jansen's neural mass model (the Jansen's NMM). Closed-loop controllers have been reported to connect stimulation input with correspondingly-generated local field potential to achieve local suppression of epileptic activity in neural networks [11] [50] . Over the last decade, researchers have made extraordinary progress in the development of PID type controllers to stabilize various epileptic seizure activities in neural mass models and brain tissue in the field of control engineering. Wang et al [51] proposes a proportional-integral controller to generate a real-time feedback to help stabilize the high-amplitude epileptic signal generated by the Jansen’s neural mass model.

Chapter 3 applies a proportional-derivative controller to provide feedback for suppressing high amplitude epileptic activity in the Jansen's NMM. The objective of chapter 4 is to implement a physical PD controller incorporating with the ASIC neural interface to suppress epileptic seizures in real neuroscience experiences by tuning proper gain parameters. In chapter 5, we also compared the PD algorithm optimized for microcontroller and FPGA architecture implementation and the total power consumption compared over respective wake-up and sleep processing cycles. The described result is not immediately obvious. We used one of the most highly efficient microcontroller currently available for this task which uses the 28nm technology node. In contrast, the nvFPGA available to us uses the 90nm technology node. Also, recursive functions and ring buffers are more easily implemented on a general-purpose processor than a FPGA. Nevertheless, we demonstrated the proof-of-concept that the nvFPGA is more energy-efficient for low-level closed-loop processing than microcontrollers.

Chapter 3. Closed-loop control in the brain models

3.1 Chapter overview

The final section of the previous chapter reviewed recently published efforts in neuron mass computational models. In this chapter, a closed-loop computational modelling study will be presented to describe the model-dependent feedback modulation of epileptic activity with stimulation intervention. This mathematical work mainly focuses on computational modelling that parallels in vitro closed-loop optogenetic stimulation experiments, shown in figure 3.1.

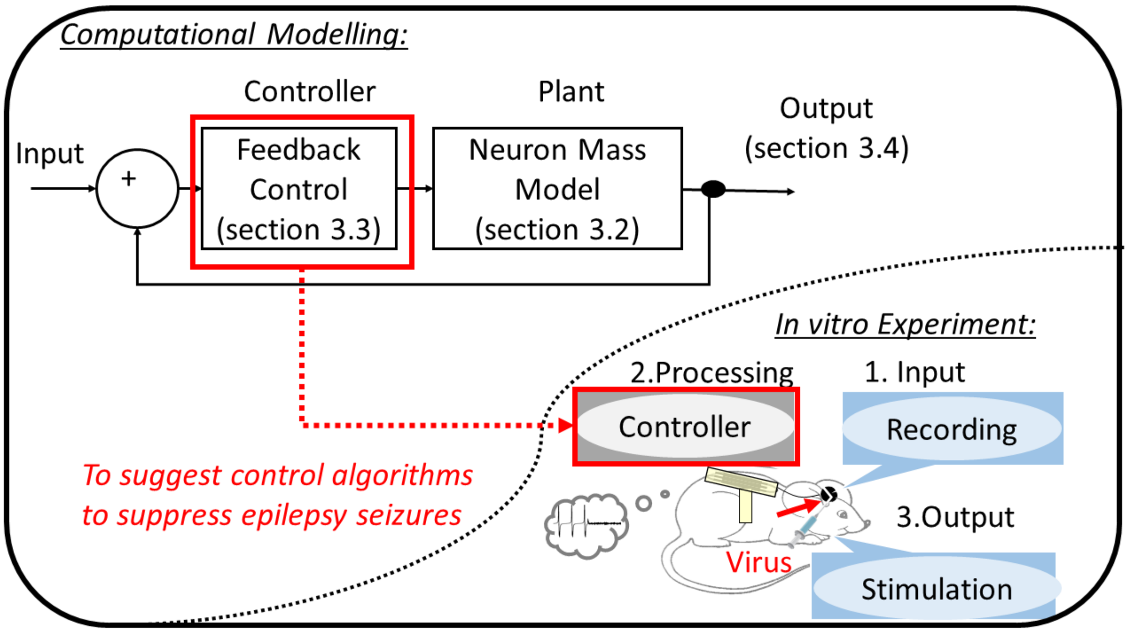


Figure 3.1: High-level schematic diagram of closed-loop control of brain activities from the perspective of computational modelling and in-vitro experiments. In computational modelling blocks, the plant is the neural mass model for describing the experimentally observed ongoing seizure-like brain activities. The controller is used to suggest potential control algorithms interacting with neuron mass models to describe the output of closed-loop control in the brain.

The overall goal of both this modelling work and in-vitro experiments is to demonstrate that closed-loop stimulation with biologically plausible parameterisation settings can alter ongoing epileptic activity in vitro. The purpose of the modelling work in this chapter is to explain the experimental observations of how the pathological electrical recording activities are controlled through stimulation by setting up proper control parameters [52].

In the following sections of this chapter, an abstract neural mass model will be introduced in section 3.2, followed by the methodology of closed-loop feedback control to intervene in neural mass model activity in section 3.3. Section 3.4 describes the simulation results and analysis of how pathological activity is altered through stimulation by altering the control algorithm parameters.

3.2 Modelling epileptiform activity

3.2.1 Jansen's neural mass model

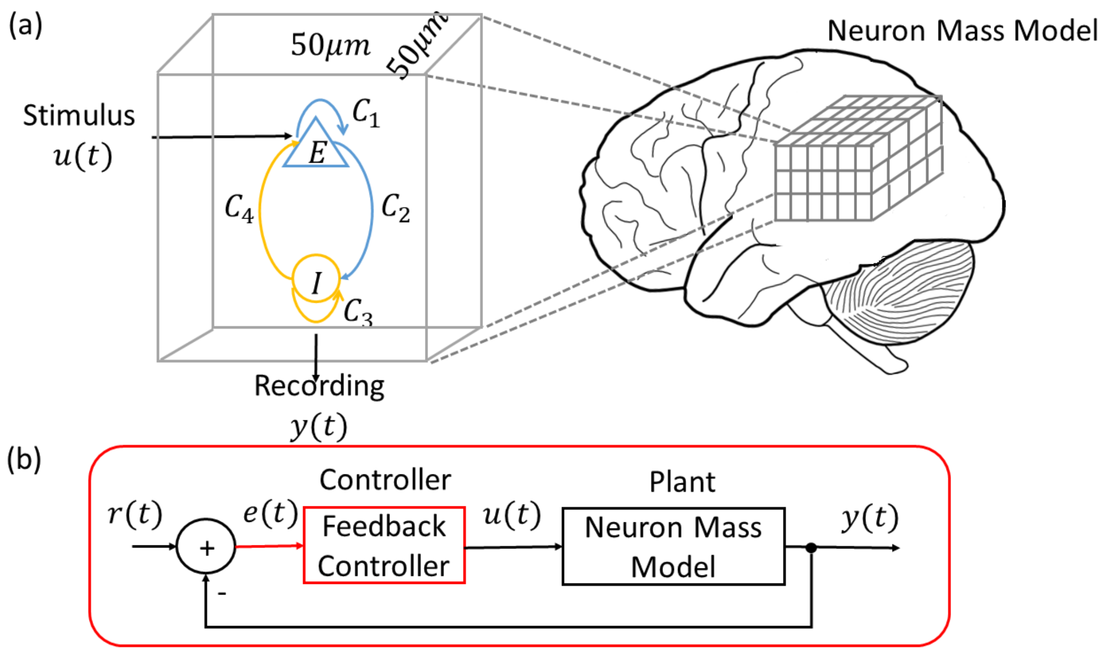


Figure 3.2: A simplistic schematic and block diagram of the neural mass model. (a) An approximation of all mini-columns to be 50μm 50 μm in size where 'E' and 'I' mean excitatory and inhibitory subpopulations which can be modelled by a Wilson-Cowan E-I unit [48]. (b) A closed-loop block diagram of an exemplar feedback controller feeding to the Jansen's NMM.*

Epilepsy has been widely recognised as the induction in normal brain activity under various trigger conditions in neuron networks, from rats to humans. Several neuron mass network models have been studied using dynamic neurological mechanisms to generate neocortical focal seizures similar to the experimentally observed ongoing seizure signals [53]. Jansen’s neuron mass model is featured at the interaction of the interlinked excitatory and inhibitory feedback loops and synaptic connection intensities [54]. Previously published work has described that abnormal values of external input and the connection intensities between excitatory and inhibitory populations can generate high-amplitude oscillations in Jansen's neural mass model (Jansen's NMM) which represent seizure-like epileptic signals [55], [56]. The most recent reported bifurcation studies found that the imbalance of the excitatory population and inhibitory population connectivity in Jansen’s NMM will generate high amplitude epileptic seizure signals [46]. Hence, we have applied Jansen’s NMM as a test bench to different closed-loop controllers to tune appropriate biologically plausible parameterisations for delivering feedback as stimulation to intervene epileptic signals in the Jansen’s NMM (For simplicity, Jansen’s NMM will be adopted in the reminder of this chapter).

In this chapter, Jansen’s NMM is proposed to describe the experimentally observed ongoing electrical recording of targeted neuron network. To be specific, we used a simplified Jansen neural mass model to describe the experimentally observed ongoing activity including epileptiform spikes and discharges. Neural population models show the activity of neuronal tissue in terms of average activity (either firing activity or field potential) of populations of neurons. Generally, principal excitatory neurons are grouped into one population, and inhibitory neurons are grouped into another population [57], [58]. Overall, this results in a system of two ordinary differential equations describing neuron network activity. Such an abstract approach means that we disregard spatial variations in activity. This is justified, as we observed less spatial variation in the dynamics across different channels. Such an approach has been shown to be sufficient to capture the most epileptiform dynamics observations in vitro and in vivo. Thus, a simple neural population model is the most cost-effective choice for a computational model which does not require an explicit assumption and is mathematically easy to work with.

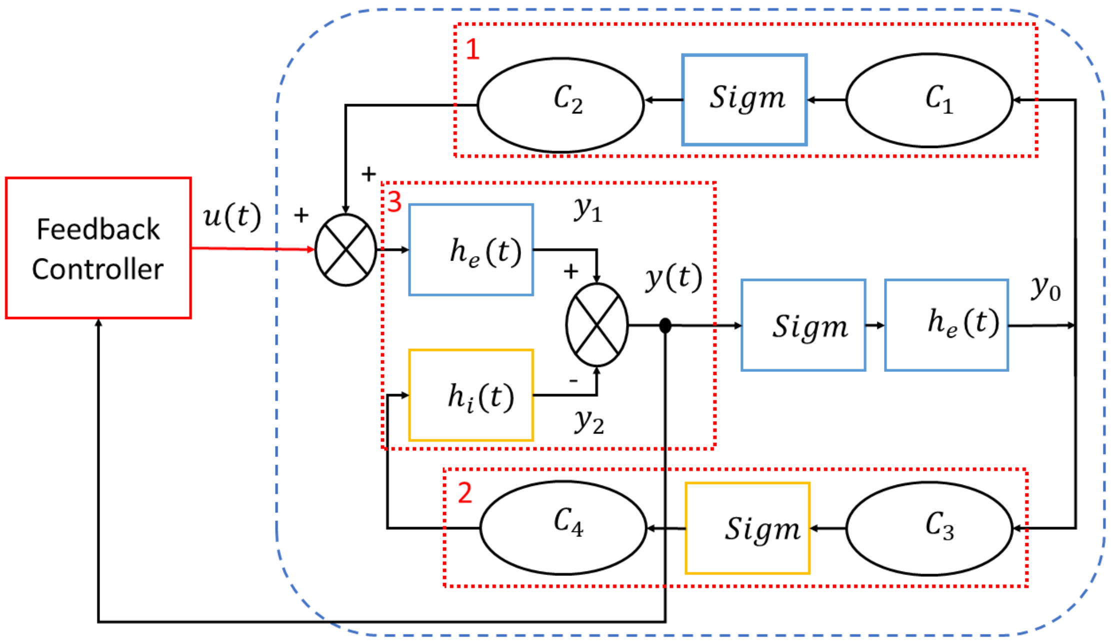


Figure 3.3: Block diagram of Jansen's NMM in which blue and yellow blocks mathematically detail excitatory and inhibitory subpopulations of the neural mass model. This also presents an approximation of all mini-columns to be 50µm 50 µm in size where 'E' and 'I' signify excitatory and inhibitory subpopulations in cortical tissue.*

In figure 3.3, a simplified Jansen's NMM has been detailed as a neurophysiologically-inspired mathematical model by a population of 'feed-forward' pyramidal neurons, receiving inhibitory and excitatory feedback from local interneurons.

Figure 3.3 describes each of the neuron populations as two blocks of 'E' and 'I' which represent excitatory and inhibitory subpopulations. Figure 3.3 divides Jansen's NMM into three interacting subpopulations:

1) Subpopulation 1 represents excitatory feedback subpopulations,
2) Subpopulation 2 represents inhibitory feedback subpopulations,
3) Subpopulation 3 is the main subpopulation.

In subpopulation 1, C1 and C2 represent the average numbers of synaptic contacts in the excitatory feedback loop, while in subpopulation 2, C3 and C4 are the average numbers of synaptic contacts in the inhibitory feedback loop. Excitatory synaptic dynamic function $h_e(t)$ and inhibitory synaptic dynamic function $h_i(t)$ linear systems transform the average postsynaptic membrane potential. $h_e(t)$ and $h_i(t)$ are defined as follows in equation (3.1) and equation (3.2).

$$h_e(t) = \begin{cases} Aate^{-at} & t > 0 \\ 0 & t < 0 \end{cases} \quad (3.1)$$

$$h_i(t) = \begin{cases} Bbte^{-bt} & t > 0 \\ 0 & t < 0 \end{cases} \quad (3.2)$$

In equation (3.1) and equation (3.2), A and B describe the maximum amplitude of excitatory and inhibitory population, while a and b are the lumped representation of the sum of the reciprocal of the time constant of passive membrane and all other spatially distributed delays in the dendritic network. As linear systems of $h_e(t)$ and $h_i(t)$ convert axonal pulses to postsynaptic potential, the impulse response of $h_e(t)$ and $h_i(t)$ are shaped to resemble an excitatory postsynaptic potential (EPSP) and an inhibitory postsynaptic potential (IPSP) respectively. The input to these linear systems is pulse density, which enables us to mimic the integrating action of the soma. In addition, $p(t)$ is modelled by Gaussian noise as the input for triggering Jansen's NMM while $y(t)$ is the output of Jansen's NMM which can be interpreted as local field potential of the NMM. $Sigm$ function in equation (3.3) describes the average membrane potential of a population of neurons into an average pulse density of action potentials fired by the neurons.

$$sigm(v) = \frac{2e_0}{1 + e^{r(v_0 - 1)}} \quad (3.3)$$

Each postsynaptic potential (PSP) of subpopulation 1 and subpopulation 2 labelled in Figure 3.3 can be modelled by two differential equations as follows:

$$\frac{d^2y}{dt^2} = Aax(t) - 2a\frac{dy}{dt} - a^2y(t) \quad (3.4)$$

Equation (3.4) can be rewritten as:

$$\frac{dy}{dt} = z(t) \quad (3.5)$$

$$\frac{dz}{dt} = Aax(t) - 2a\frac{dz}{dt} - a^2y(t) \quad (3.6)$$

Where can be rewritten as $x(t)$ and $y(t)$ are the input and output signal respectively. Hence, six different equations are derived from equation (3.5) and equation (3.6) as following:

$$\frac{dy_0}{dt} = y_3(t) \tag{3.7}$$

$$\frac{dy_3}{dt} = AaSigm\big(y_1(t) - y_1(t)\big) - 2ay_3(t) - a^2y_0(t) \tag{3.8}$$

$$\frac{dy_1}{dt} = y_4(t) \tag{3.9}$$

$$\frac{dy_4}{dt} = Aa\{p(t) + C_2Sigm(C_1y_0(t))\} - 2ay_4(t) - a^2y_1(t) \tag{3.10}$$

$$\frac{dy_2}{dt} = y_5(t) \tag{3.11}$$

$$\frac{dy_5}{dt} = Bb\{C_4Sigm[C_3y_0(t)]\,\} - 2ay_5(t) - b^2y_2(t) \tag{3.12}$$

Where y_0 , y_1 , y_2 are the output of three postsynaptic potential blocks (subpopulation 1, subpopulation 2 and subpopulation 3). The three pairs of differential equations (equation (3.7) and equation (3.8), equation (3.9) and equation (3.10), equation (3.11) and equation (3.12)) are solved by applying an integration method of the Fehlberg fourth-fifth order Runge-Kutta method [59]. Table 2.1 shows the neural mass model parameters.

3.2.2 Model parameter choice

The choice of different parameters will determine different output of the Jansen Neural Mass Model . The connectivity constants; C_1, C_2, C_3 and C_4 are proportional to the average number of synapses between the pyramidal cells and the excitatory feedback elements. The connectivity constants C_1, C_2, C_3 and C_4 are proportional to the average number of synapses between the pyramidal cells and the inhibitory feedback elements. C_1, C_2, C_3 and C_4, and can be detailed as follows:

1. C_1 is the synapses number which is generated by the feed-forward neurons to the excitatory neurons feedback loop.
2. C_2 is proportional to the synapses number which is made by the excitatory feedback loop to the feedforward neurons' dendrites.
3. C_3 stands for the synapses number which is generated by the feed-forward neurons to the inhibitory feedback loop dendrites.
4. C_4 represents the synapses number which is generated by the inhibitory feedback loop to the feedforward neuron dendrites.

[] did a study of the visual cortex pyramidal cell which suggests that

$$C_1 + C_3 = C_2 + C_2' + C_4 \tag{3.13}$$

[60] reported that in a mouse's somato-motor cortex, a pyramidal cell axon would make 87% of its synapses and 13% on shafts. Therefore White [61] observed that a synapses made a spine onto an excitatory cell, but a synapse on a shaft is equally likely to be on an excitatory or an inhibitory cell. Hence, about 6.5% of the synapses made by a pyramidal cell are inhibitory, therefore:

$$\frac{C_3}{C_1 + C_3} = 6.5/100 \tag{3.14}$$

In another publication, Liu [62] claimed that 80% of the synapses were of the excitatory type, which is made on a pyramidal cell dendrite in a cat's motor cortex, hence:

$$\frac{(C_2 + C_2')}{(C_2 + C_2') + C_4} = 0.8 \tag{3.15}$$

The main excitatory feedback loop is composed of pyramidal cells, as most excitatory cells in the visual cortex are pyramidal cells. Considering the excitatory cells population is homogeneous in synapse patterns, the synapses number made by the feedforward neuron of a cortical column on the excitatory feedback loop should be the same as the synapses number made by the excitatory feedback loop on the feedforward neurons. This leads to:

$$C_1 = C_2 + C_2' \tag{3.16}$$

According to [61], 20% of asymmetrical synapses of the excitatory type in layer IV of the cortex are formed by thalamo-cortical terminals:

$$C_1 = C_2 + C_2' = 0.2 => \; C_2' = C_2/4 \tag{3.17}$$

Substituting () in (), which can be shown as:

$$\frac{C_2}{C_1} = 0.8 \tag{3.18}$$

From () and (), we get:

$$C_3 = C_4 \tag{3.19}$$

The synapses on the excitatory and inhibitory feedback loop are very ambiguous. Substituting (3.16) and (3.19) in (3.15) yields $C_1 = 4C_3$. To be specific, the relationship between C1 and C3 may vary due to the different biological materials in the synapses counts. Let's assume that $C_1 = 4C_3$ which will generate:

$$C_1 = \frac{C_2}{0.8} = 4C_3 = 4C_4 \tag{3.20}$$

(3.20) allows us to represent C_1, C_2, C_3 and C_4 with one constant C

$$C_1 = C \tag{3.21}$$

$$C_2 = 0.8C \tag{3.22}$$

$$C_3 = 0.25C \tag{3.23}$$

$$C_4 = 0.25C \tag{3.24}$$

The variable C will vary under different physiological constraints, as it presents different synaptic phenomena in different biological applications. One of the applications is a neurotransmitter depletion which is very common and will generate drastic consequences.

The A and B parameters of the PSP functions are proportional to the output magnitude of the PSD block. [54] proposed that $A = 3.25\ mV$ and $B = 22\ mV$. There is another publication which modified the amplitude of the PSPs based on certain neural properties. Therefore, A and B could be modified with a degree of freedom. Moreover, the A and B parameters of the PSP blocks are inversely proportional to the PSP duration, which are less likely to vary over relative short periods. [49] suggests that $a = 100s^{-1}$ and $b =$

$50s^{-1}$. Table 3.1 details the parameter choosing for modelling simulation in the Jansen's NMM.

Table 3.1: Physiological biological plausible parameters interpretation, and standard values of the parameter in Jansen's neural mass model.

Parameters	**Description**	**Standard value**
He	Average gain of excitatory synaptic	3.25 mV
Hi	Average gain of inhibitory synaptic	22 mV
τe	Synaptic time constant for excitatory subpopulation	0.0108 s
τi	Synaptic time constant for inhibitory subpopulation	0.02 s
C1, C2	Synaptic contacts in excitatory feedback loop	C1=135, C2=0.8*135
C3, C4	Synaptic contacts in inhibitory feedback loop	C3=0.25*135, C4=0*135
v0, e0, r	Non-linear sigmoid function	v0=6 mV, e0=2.5, r =0.56mV^{-1}

3.3 Closed-loop feedback control

3.3.1 Proportional-derivative control of neuron mass model (PD-NMM)

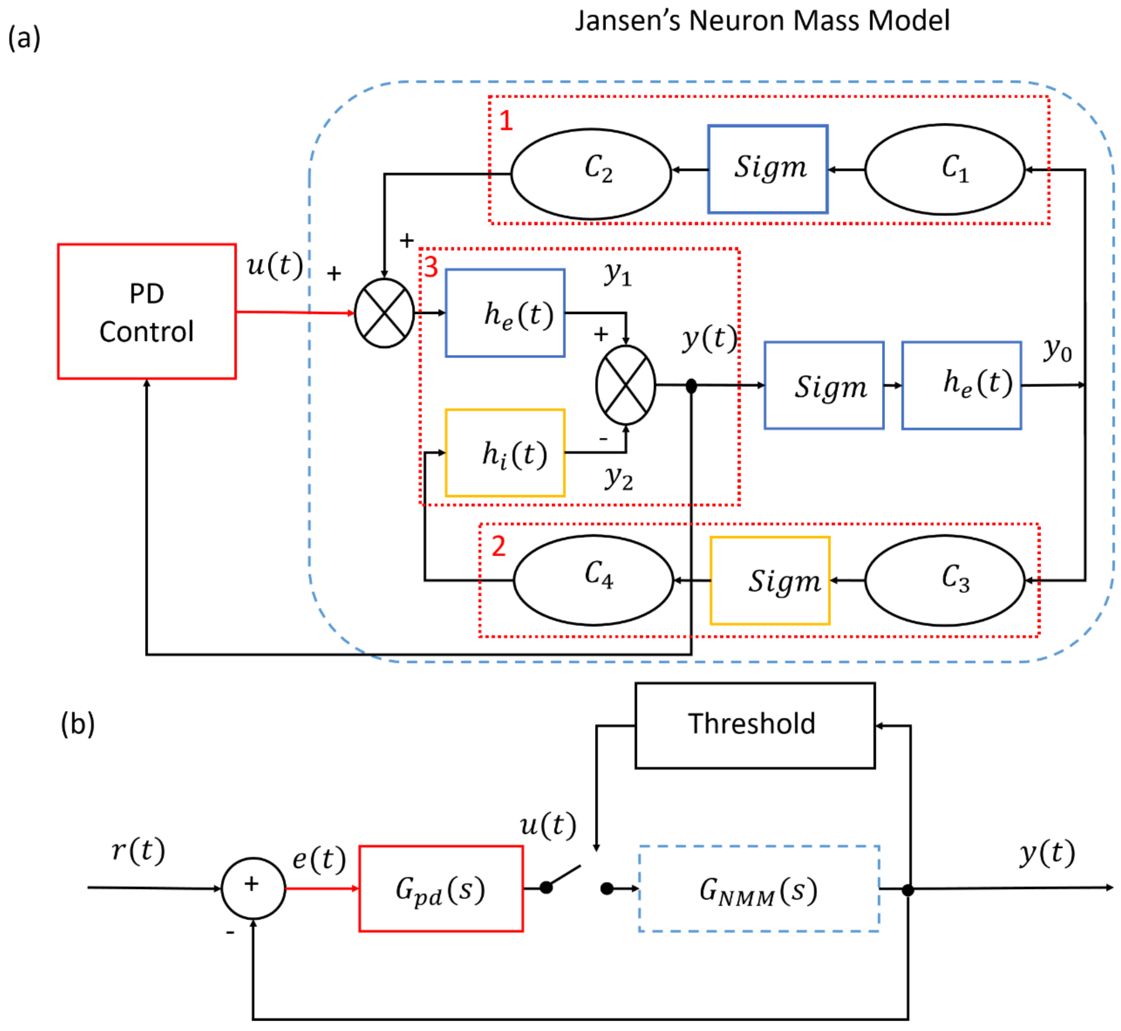


Figure 3.4: The closed-loop scheme of a PD-NMM control scheme. (a) Block diagram of the PD-based controller feeding into the Jansen's Neuron Mass Model. (b) The high-level simplified equivalent form of PD-NMM control scheme. $\mathbf{G_{pd}(s)}$ is the Laplace transform of a PD controller to describe the transfer function of PD control, and $\mathbf{G_{NMM}(s)}$ is the Laplace transform of Jansen's Neuron Mass Model to show the transfer function of Jansen's Neuron Mass Model. $\mathbf{r(t)}$ is the desired output of Jansen's Neuron Mass Model. $\mathbf{e(t)}$ is the error signal of the closed-loop control scheme. $\mathbf{u(t)}$is the output of the PD controller. PD-NMM is used to describe the closed-loop proportional-derivative control of the neural mass model.

In this section, we aim to investigate the feedback control theory to develop different types of controllers to generate closed-loop stimulation feeding into neuron mass models for altering ongoing epileptiform activity. Primarily, the proportional-derivative controller is designed to provide stimulation for suppressing high amplitude epileptic activity generated by Jansen's NMM shown in figure 3.4. Graphical stability methodology has been adopted to produce an analytical design approach for choosing proportional and derivative gain parameters to stabilise high amplitude activity of Jansen’s NMM. Therefore, the analytical design approach of this closed-loop system makes the closed-loop PD feedback control independent of a specific neuron model, which can also be applied to control methodology studies of other promising neuron models in the future. Furthermore, we intervene the feedback to the neuron mass network model to study the potential experimental observations on how the pathological activity is altered through stimulation by altering different control algorithms. Finally, we suggest an optimised hardware architecture for closed-loop algorithm implementation in custom-designed hardware in chapter 4.

Epileptic activity in a neural mass model can be categorised as high amplitude limit cycle oscillation born in Hopf bifurcation [57], which indicates that the fixed point of Jansen’s NMM will lose its stability. In figure 3.4, the designed closed-loop controller has been proposed to provide feedback stimulations to stabilise the unstable fixed point of a neural mass model for preventing the generation of Hopf bifurcation to suppress high amplitude epileptic activity. Figure 3.4 shows the interaction between the PD controller and Jansen’s NMM, where $u(t)$ is the output of the PD controller (stimulation signals), while $y(t)$ is the output of Jansen’s NMM model (local field potential). In order to define the stabilisation area of proportional-derivative gain parameters, a graphical stability analysis method can be applied by using the following four steps:

Derivation of the characteristic equation of closed-loop PD-NMM Model

<u>Step 1: Derive Laplace Transform of Jansen's NMM</u>

$$G_{NMM}(s) = \frac{H_e(s)}{1 + H_e(s)Ks^2[H_i(s)C_3C_4 - H_e(s)C_1C_2]} \quad (3.13)$$

<u>Step 2: Derive Laplace Transform of PD Controller</u>

$$G_{pd}(s) = K_p + K_d s \quad (3.14)$$

<u>Step 3: Derive the characteristic equation of PD-Jansen's NMM closed-loop control system shown in figure 3.4 (b)</u>

$$\Delta(s) = 1 + G_{pd}(s)G_{NMM}(s) = 0 \quad (3.15)$$

(The derivation details of characteristic equation

1. $r(t) = 0, \quad =>$

2. $G_{pd}(s)G_{NMM}(s) = \frac{U(s)}{E(s)} * \frac{Y(s)}{U(s)} = \frac{Y(s)}{E(s)}, \quad =>$

3. $R(s) - Y(s) = E(s), \quad =>$

4. $1 + G_{pd}(s)G_{NMM}(s) = 0$)

<u>Step 4: Make the variable substitution: $s = j\omega$</u>

$$\Delta(j\omega) = 1 + G_{pd}(j\omega)G_{NMM}(j\omega) = 0 \quad (3.16)$$

The characteristic equation of PD-Jansen's NMM closed-loop control system shown in equation (3.16) defines the stability space boundary of the PD-Jansen's NMM feedback control system.

Given that $G_{NMM}(j\omega)$ is a complex function, we can rewrite $G_{NMM}(j\omega)$ as $|G_{NMM}(j\omega)| = \sqrt{\delta_{I_{NMM}}^2(\omega) + \delta_{R_{NMM}}^2(\omega)}$, the characteristic equation of PD-Jansen's NMM control system, as:

$$K_p = \frac{-\delta_{R_{NMM}}(\omega)}{\delta_{I_{NMM}}^2(\omega) + \delta_{R_{NMM}}^2(\omega)} \tag{3.17}$$

$$K_d = \frac{\delta_{R_{NMM}}(\omega)}{\omega[\delta_{I_{NMM}}^2(\omega) + \delta_{R_{NMM}}^2(\omega)]} \tag{3.18}$$

Where $|G_{NMM}(j\omega)| = \sqrt{\delta_{I_{NMM}}^2(\omega) + \delta_{R_{NMM}}^2(\omega)}$.

3.3.2 Proportional-integral control of neural mass model (PI-NMM)

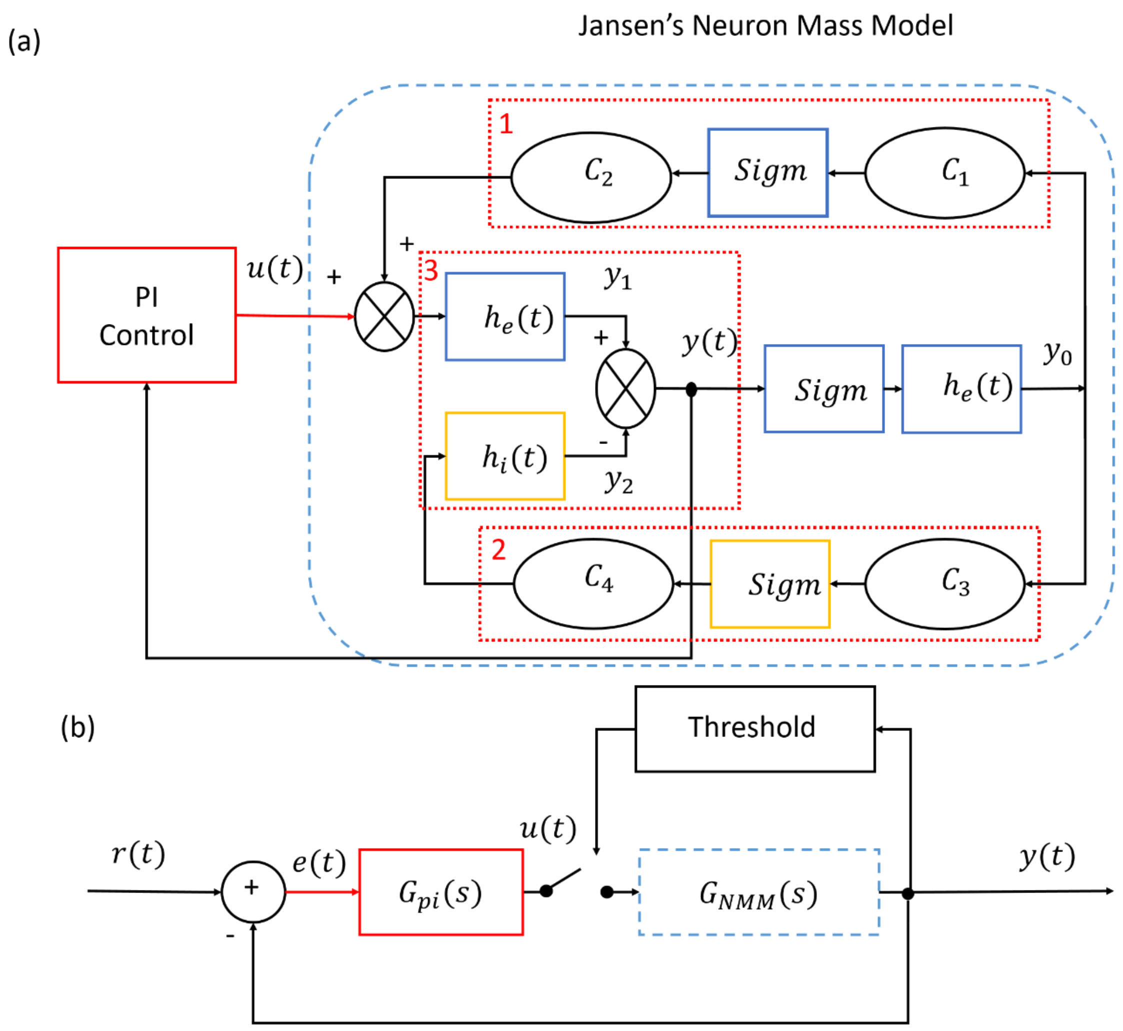


Figure 3.5: The closed-loop scheme of the PI-NMM control scheme. (a) Block diagram of the PI-based controller feeding into Jansen's Neuron Mass Model. (b) The high-level simplified equivalent form of the PI-NMM control scheme. $\boldsymbol{G_{pi}(s)}$ *is the Laplace transform of the PI controller to describe the transfer function of the PI controller, and* $\boldsymbol{G_{NMM}(s)}$ *is the Laplace transform of Jansen's Neuron Mass Model to show its transfer function.* $\boldsymbol{r(t)}$ *is the desired output of Jansen's Neuron Mass Model.* $\boldsymbol{e(t)}$ *is the error signal of the closed-loop control scheme.* $\boldsymbol{u(t)}$*is the output of the PI controller. PI-NMM is used to describe the closed-loop proportional-integral control of Jansen's Neuron Mass Model.*

For comparison, a PI controller is studied to provide the stimulus feed into a neural mass model for simulation in this section. Figure 3.5 shows a closed-loop scheme of proportional-integral control of Jansen's neural mass model. The derivation of the characteristic equation of a closed-loop PI-NMM model is shown as follows:

Derivation of characteristic equation of closed-loop PI-NMM model

<u>Step 1: Derive Laplace Transform of Jansen's NMM</u>

$$G_{NMM}(s) = \frac{H_e(s)}{1 + H_e(s)Ks^2[H_i(s)C_3C_4 - H_e(s)C_1C_2]} \tag{3.19}$$

<u>Step 2: Derive Laplace Transform of PI Controller</u>

$$G_{pi}(s) = K_p + \frac{K_i}{s} \tag{3.20}$$

<u>Step 3: Derive the characteristic equation of PI-Jansen's NMM closed-loop control system shown in figure 3.5(b)</u>

$$\Delta(s) = 1 + G_{pi}(s)G_{NMM}(s) = 0 \tag{3.21}$$

The derivation details of characteristic equation:

1. $r(t) = 0, \quad =>$

2. $G_{pi}(s)G_{NMM}(s) = \frac{U(s)}{E(s)} * \frac{Y(s)}{U(s)} = \frac{Y(s)}{E(s)}, \quad =>$

3. $R(s) - Y(s) = E(s), \quad =>$

4. $1 + G_{pi}(s)G_{NMM}(s) = 0$)

<u>Step 4:</u> Make the variable substitution: $s = j\omega$

$$\Delta(j\omega) = 1 + G_{pi}(j\omega)G_{NMM}(j\omega) = 0 \tag{3.22}$$

The characteristic equation of PI-Jansen's NMM closed-loop control system shown in equation (3-22) defines the stability space boundary of the PI- NMM feedback control system. Supposing $G_{NMM}(j\omega)$ is a complex function, $G_{NMM}(j\omega)$ as $|G_{NMM}(j\omega)| =$

$\sqrt{\delta_{I_{NMM}}^2(\omega)+\delta_{R_{NMM}}^2(\omega)}$, the characteristic equation of PI- NMM control system can be rewritten as:

$$\begin{cases} K_p = \dfrac{-\delta_{I_{NMM}}(\omega)\omega}{\delta_{I_{NMM}}^2(\omega)+\delta_{R_{NMM}}^2(\omega)} & (3.23) \\ K_i = \dfrac{-\delta_{R_{NMM}}(\omega)}{\omega[\delta_{I_{NMM}}^2(\omega)+\delta_{R_{NMM}}^2(\omega)]} & (3.24) \end{cases}$$

Where $|G_{NMM}(j\omega)| = \sqrt{\delta_{I_{NMM}}^2(\omega)+\delta_{R_{NMM}}^2(\omega)}$.

3.4 Results and analysis

3.4.1 Closed-loop PD-NMM simulation

- *PD-NMM stabilisation area*

Epileptic activity is caused by the imbalance of the excitation and inhabitation of neuronal population in neural mass models. In computation modelling work, it can be recognised as being caused by extremely large excitatory parameters H_e or small inhibitory parameters H_i respectively. Hence, the goal of this section is to discuss the effect of the two parameters H_e and H_i on the stabilising region of the proposed PD controller and PI controller.

Simulations have been demonstrated for plotting the stabilization relationship of PD gain parameters k_p and k_d with respect to two cases:

- ***Hyper-excitation scenario :*** $H_e = 5,7,9$
- ***Low inhibition scenario***: $H_i = 15,17,19$

Figures 3.6 (a) and 3-6 (b) show the effect of excitatory parameter H_e and inhibitory parameter H_i on the stabilisation area plot of the proposed PD control of the Jansen's neural mass model according to equations (3.17) and (3.18). Stabilising regions of the PD-NMM controller for abnormal values of H_e and H_i are also highlighted in blue and orange in figures 3.6 (a) and3-6 (b) respectively. The relationship of k_p and k_d is dependent on the specific neural model. Hence the whole analytical design can also be applied to other neural models as well.

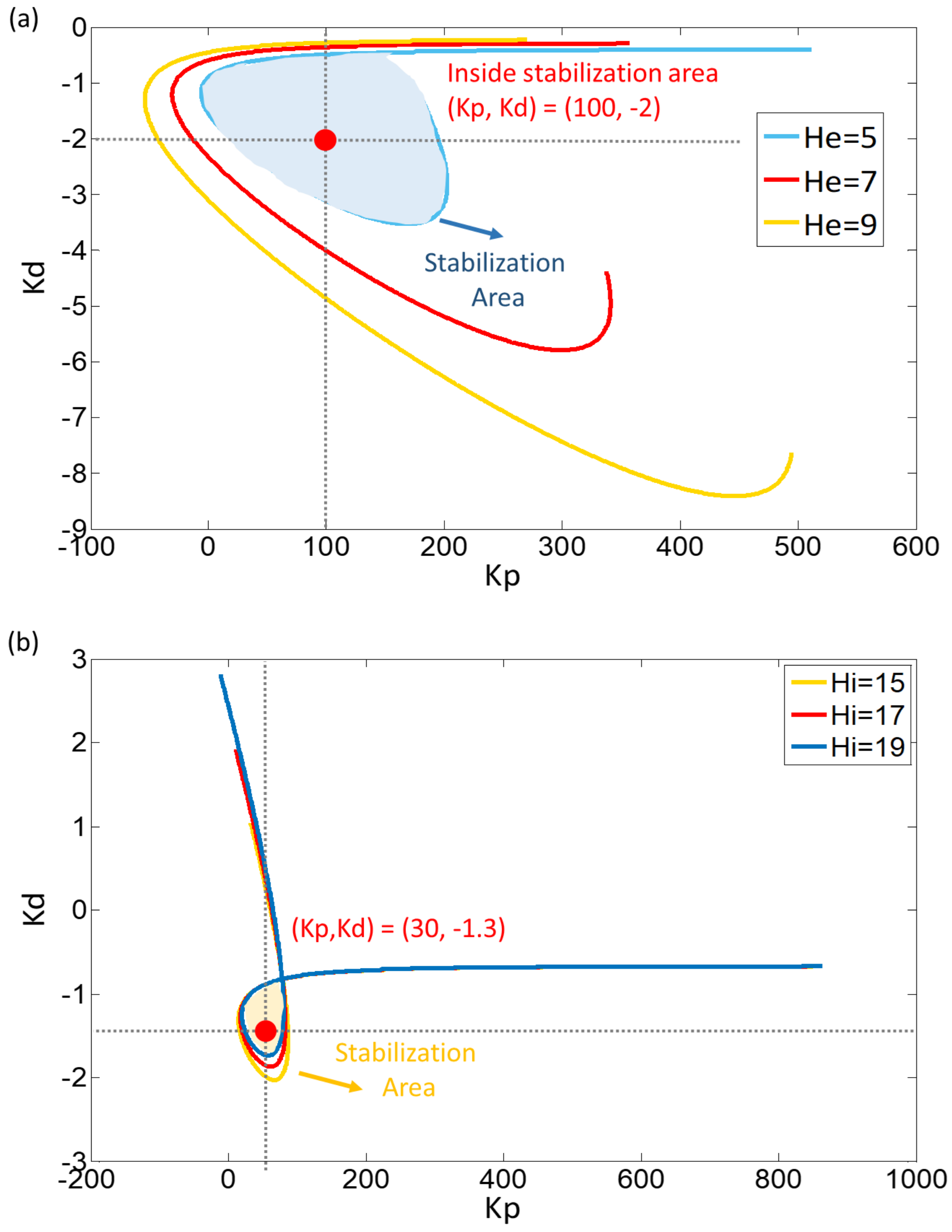


Figure 3.6: The effect of excitatory parameters H_e and inhibitory parameters H_i on changing the stabilising area of K_p and K_d within the PD controller. (a) The stabilisation area of the PD controller differs from H_e =5, 7, 9 respectively. (b) The stabilisation area of the PD controller differs from H_i =15, 17, 19 respectively.

- *Real-time simulation results*

According to the stabilisation area in figure 3.6, we simulated the following two cases:

- ***<u>Hyper-excitation scenario:</u>*** $H_e = 7$, $H_i = 22$
- ***<u>Low inhibition scenario</u>***: $H_e = 3.25$ and $H_i = 17$

Specific PD control gain parameters are picked up from the corresponding stabilisation areas highlighted in figure 3.6(a) and figure 3.6(b) for further simulation which will be detailed in this section.

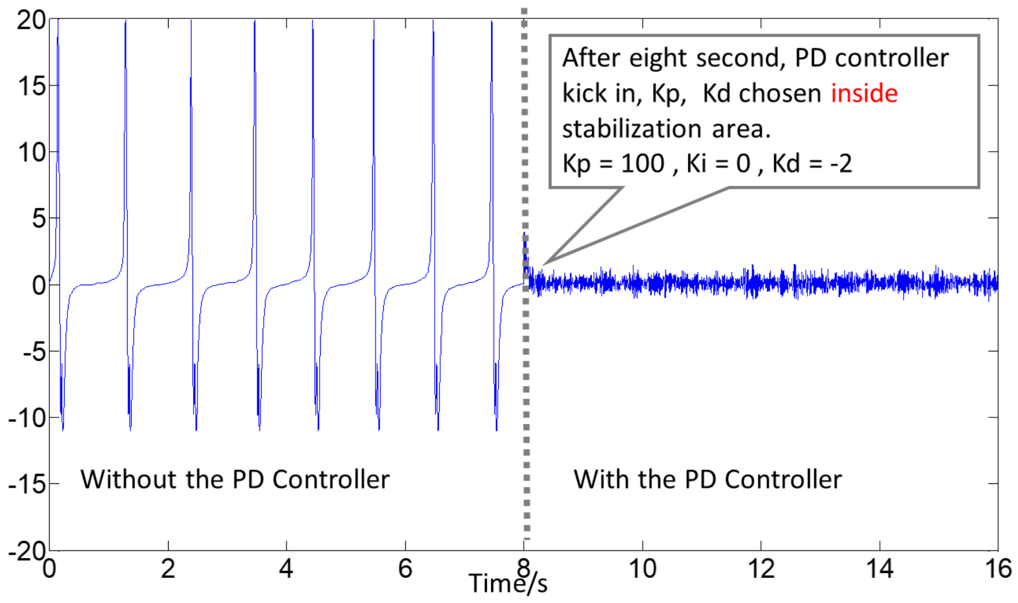
- ***<u>Hyper-excitation scenario</u>***

When H_e is set as 7 and H_i is set 22, Jansen's NMM shows a hyper excitation scenario which can generate high amplitude output. The high amplitude signals resemble high amplitude epileptic seizure-like oscillations which are also plotted in the first eight seconds of figure 3.7. After eight seconds, the feedback generated by the proposed PD controller with the chosen PD gain is intervened into the real-time Jansen's neuron mass model for further simulation.

In figure 3.7(a), $\mathrm{K_p} = 100, \mathrm{K_i} = 0, \mathrm{K_d} = -2$ have been picked up specifically from the stabilisation area highlighted in figure 3.6(a) to provide feedback stimulation from hyper-excitation simulations in Jansen's NMM ($H_e = 7$ and $H_i = 22$).

In comparison, $\mathrm{K_p} = 100, \mathrm{K_i} = 0, \mathrm{K_d} = -8$ are outside the stabilisation area which can be found in figure 3.6(b). $\mathrm{K_p} = 100, \mathrm{K_i} = 0, \mathrm{K_d} = -8$ are used for simulation, shown in figure 3.7(b). Since the parameters $\mathrm{K_p} = 100, \mathrm{K_i} = 0, \mathrm{K_d} = -8$ are picked outside the stabilised area, it does not help to suppress the high amplitude signal generated in Jansen's NMM model, and even worse, it oscillates the NMM model to make more hyper-exaction.

(a)



(b)

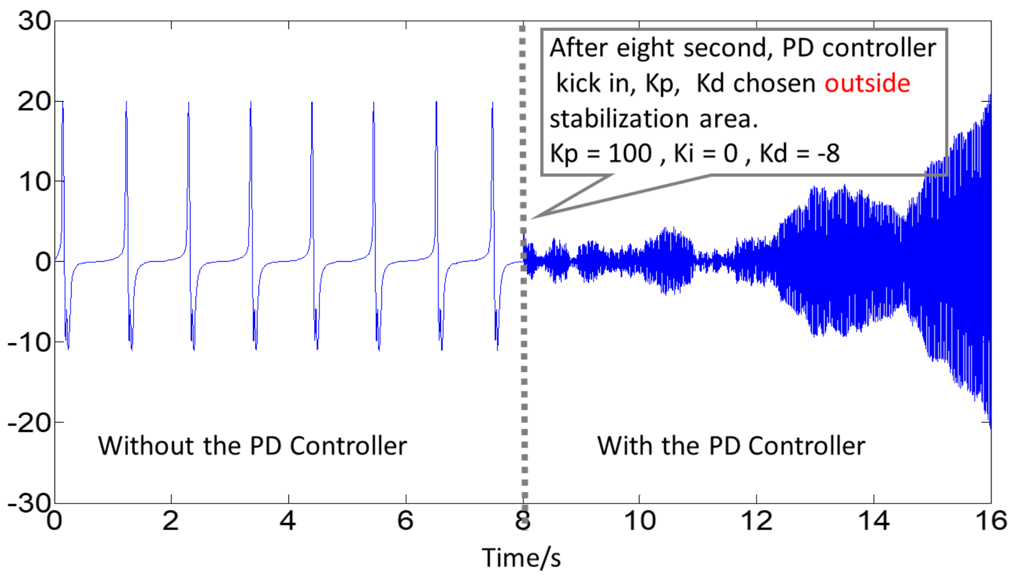


Figure 3.7: In the hyper-excitation scenario ($H_e = 7$, $H_i = 22$), the comparison of output of Jansen's NMM for the first eight second simulation without the PD controller, and the second eight second simulation with stimulation feedback from the PD controller. (a) PD control gain set up is chosen inside the stabilisation area ($K_p = 100, K_i = 0, K_d = -2$). (b) A PD control gain set up is chosen outside stabilisation area ($K_p = 100, K_i = 0, K_d = -8$).

- ***<u>Low inhibition scenario</u>***

In figure 3.8, $\mathrm{K_p} = 30,\ \mathrm{K_i} = 0, \mathrm{K_d} = \ -1.3$ have been chosen from figure 3.6(b) as the PD gain parameters for providing feedback stimulation for a low inhabitation neural mass model simulation of ($H_e = 3.25$ and $H_i = 17$). The above two experiment sets prove that the PD controller provides stimulation feedback to intervene with Jansen's NMM to suppress high amplitude epileptic seizures. It can be seen that the output of Jansen's NMM was high amplitude activity, which has been clearly demonstrated in the first 8 seconds, then under the intervention of PD controller feedback, the seizure network has been stabilised into low amplitude activity as a comparison. The graphical design of the stability analysis method has been applied to choose PD controller gain parameters for suppressing seizures in Jansen's NMM. Therefore, in this specific neural mass model simulation, high amplitude epileptic activity has been successfully suppressed by applying a closed-loop PD controller to deliver feedback stimulation with a proper PD gain parameters setup.

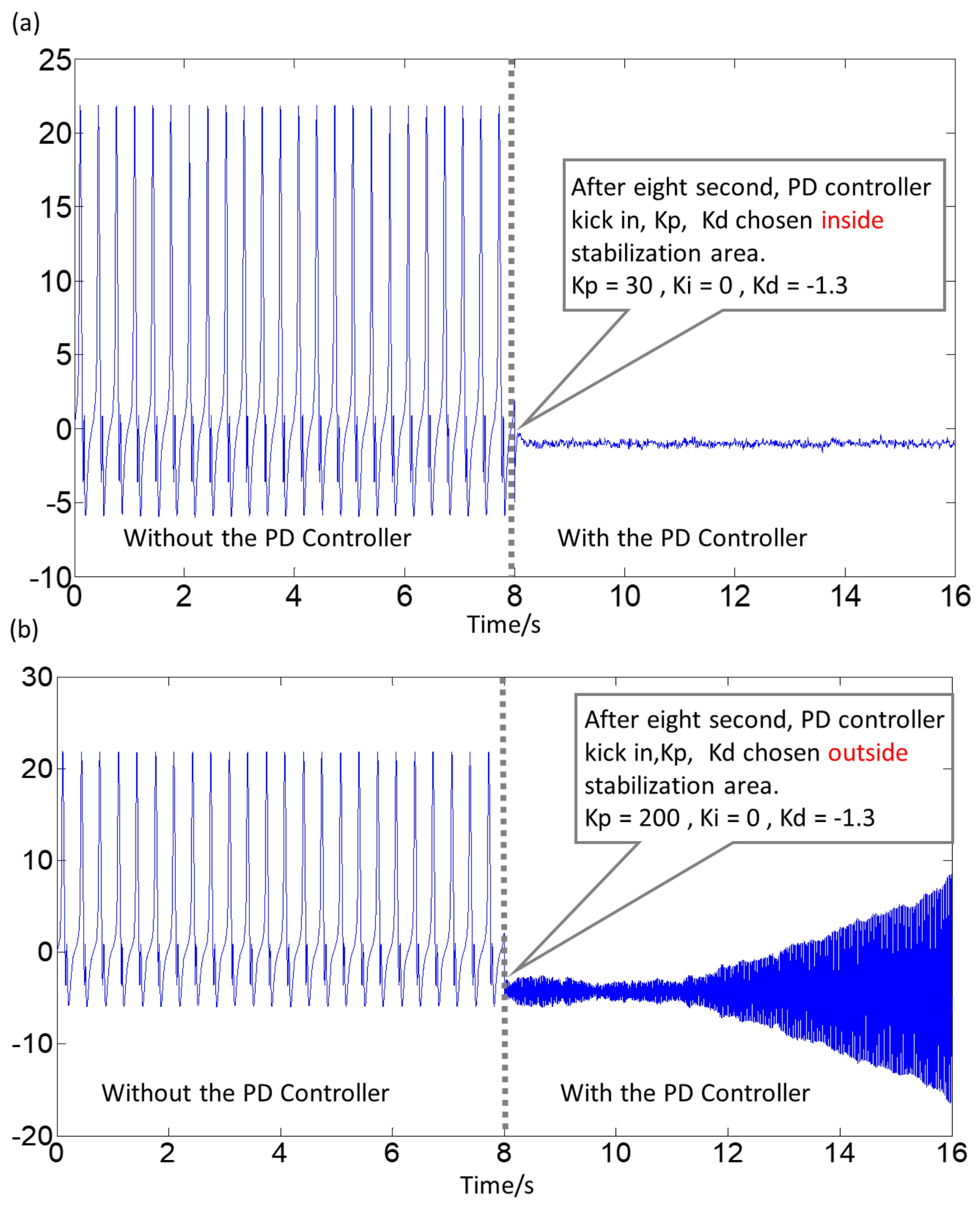


Figure 3.8: Under the low inhabitation scenario ($\mathrm{H_e} = 3.25$, $\mathrm{H_i} = 17$), the comparison of output of Jansen's NMM for the first eight-second simulation without the PD controller, and the second eight-second simulation with stimulation feedback from the PD controller. (a) The PD control gain set up is chosen inside the stabilisation area ($K_p = 30, K_i = 0, K_d = -1.3$). (b) A PD control gain set up is chosen outside the stabilisation area ($K_p = 200, K_i = 0, K_d = -1.3$).

3.4.2 Closed-loop PI-NMM simulation

- *PI-NMM stabilisation area*

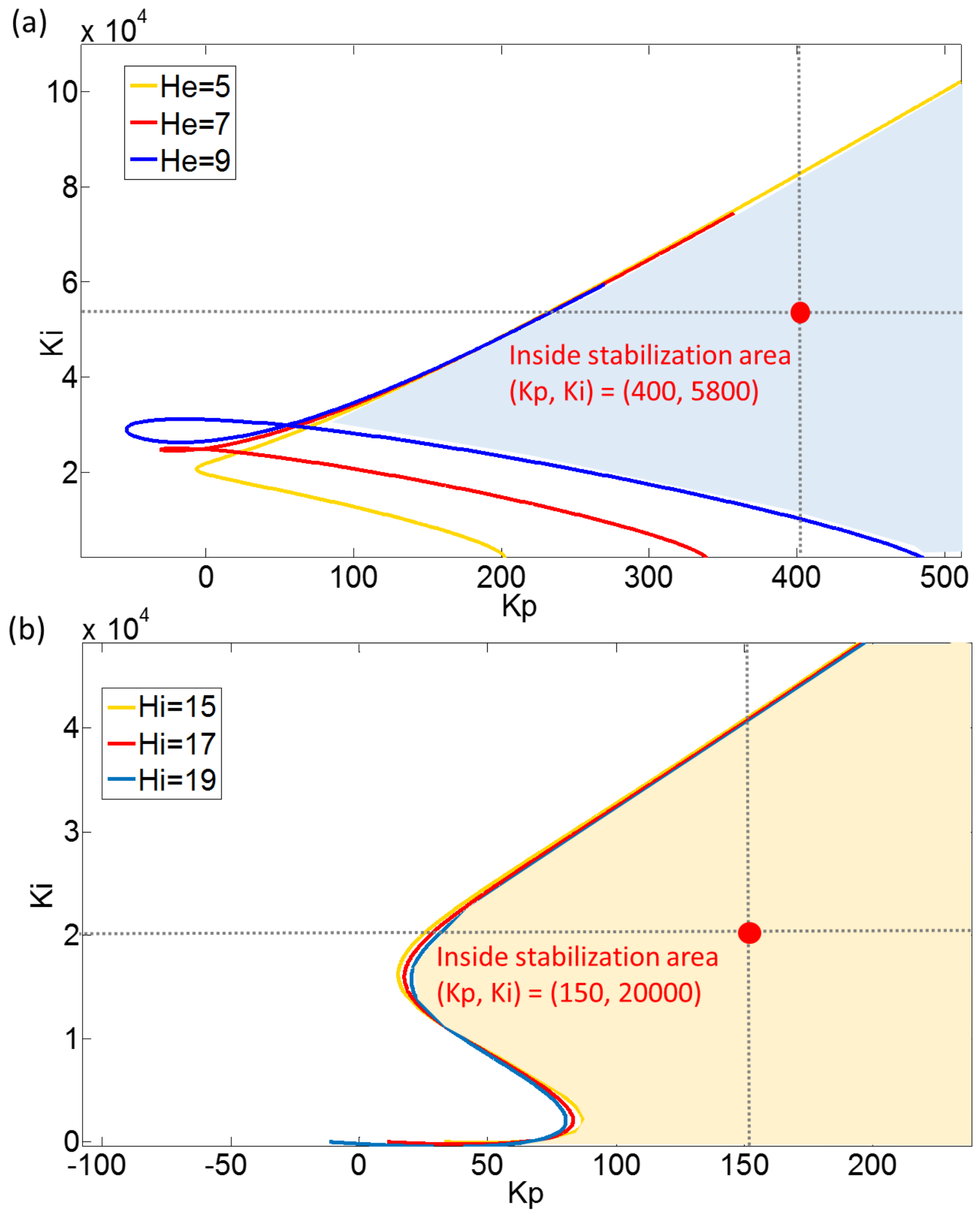


Figure 3.9: the efficiency of excitatory parameters H_e and the inhibitory parameters H_i on changing the stabilising area within the PI controller. (a) The stabilisation area of the PI Controller differs with H_e =5, 7, 9 respectively. (b) The stabilisation area of the PI controller differs with H_i =15, 17, 19 respectively.

Figure 3.9(a) and figure 3.9(b) show the effect of excitatory parameters H_e and H_i of the proposed PI control of Jansen's neural mass model according to equation (3.23) and equation (3.24). Stabilising regions between K_p and K_i of the PI-NMM controller for abnormal values of H_e and H_i scenarios are also plotted in figure 3.9 (a) and figure 3.9 (b) respectively.

- *Real-time simulation results*

For performing real time closed-loop simulations, in the first eight seconds the neural mass model generates high amplitude epileptic like signals shown in figure 3.10(a) and figure 3.10(b). After eight seconds, we chose a set of the PI controller gain parameters set up for hyper-excitation simulations and low-inhabitation simulations.

For the hyper-excitation scenario, in figure 3.10(a), $K_p = 400, K_i = 5800$ and $K_d = 0$ are picked up inside the stabilisation area as the PI controller gain set up. Figure 3.10(a) shows how the neural mass model changes after the PI controller intervenes. Moreover, $K_p = 400, K_i = 10000$ and $K_d = 0$, which are outside the stabilisation area, are applied to provide stimulation to Jansen's NMM in figure 3.10(b). It can be observed that epileptic seizures can be suppressed by proper PI gain choice, i.e. those which are located inside stabilisation area in hyper-excitation scenario.

For the low-inhabitation scenario, in figure 3.9(b), $K_p = 150, K_i = 20000$ and $K_d = 0$ are chosen inside the stabilisation area for the PI controller gain set up. Figure 3.11(a) demonstrates how Jansen's NMM behaves after the PI controller gain set up. For comparison, $K_p = -50, K_i = 20000$ and $K_d = 0$ are picked up as gain parameters outside the stabilisation area for simulations in figure 3.11(b). It can be seen that epileptic seizures are controlled by proper PI gain choice, i.e. those which are located inside stabilisation area in the low-inhabitation scenario.

- ***<u>Hyper-excitation scenario</u>***

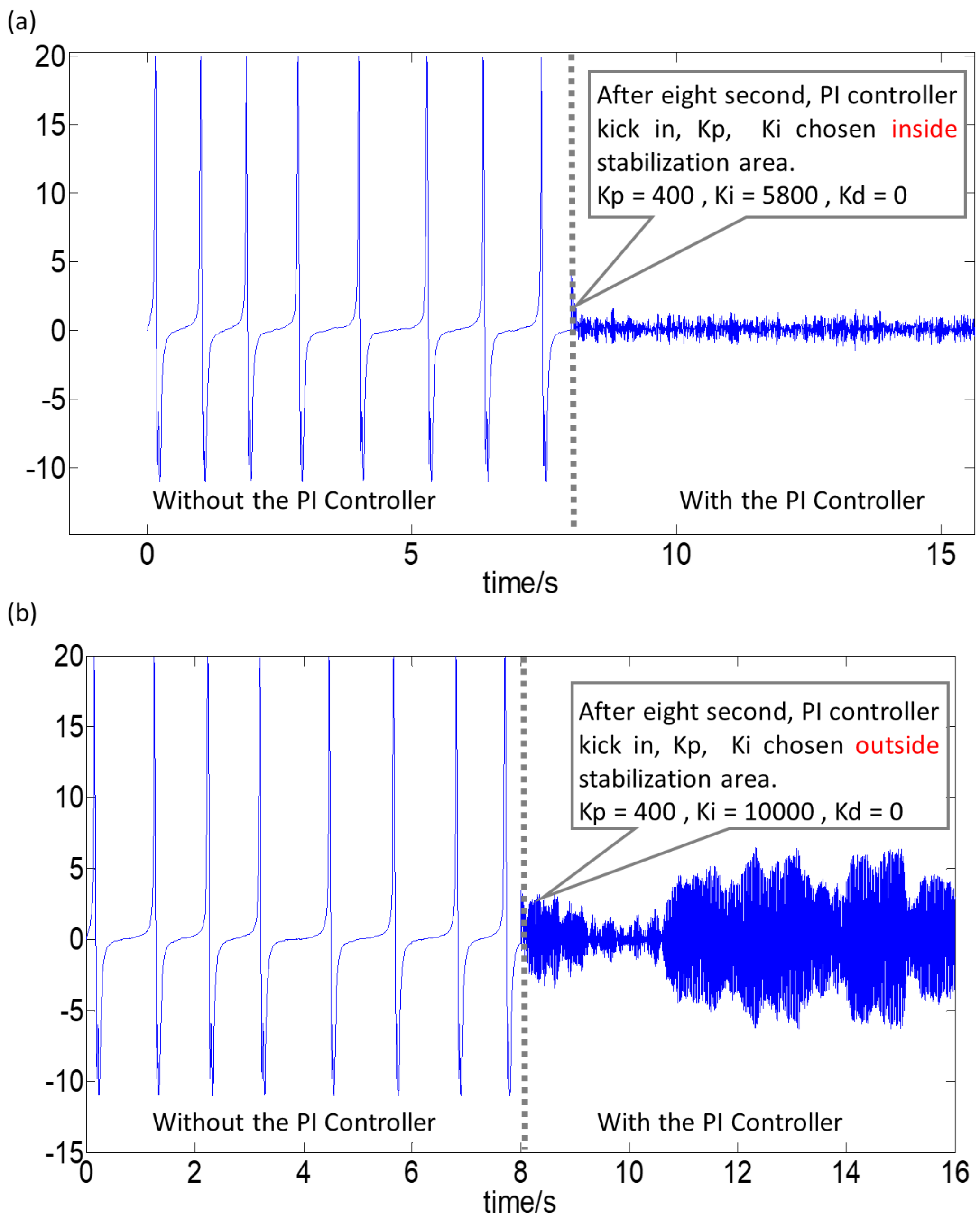
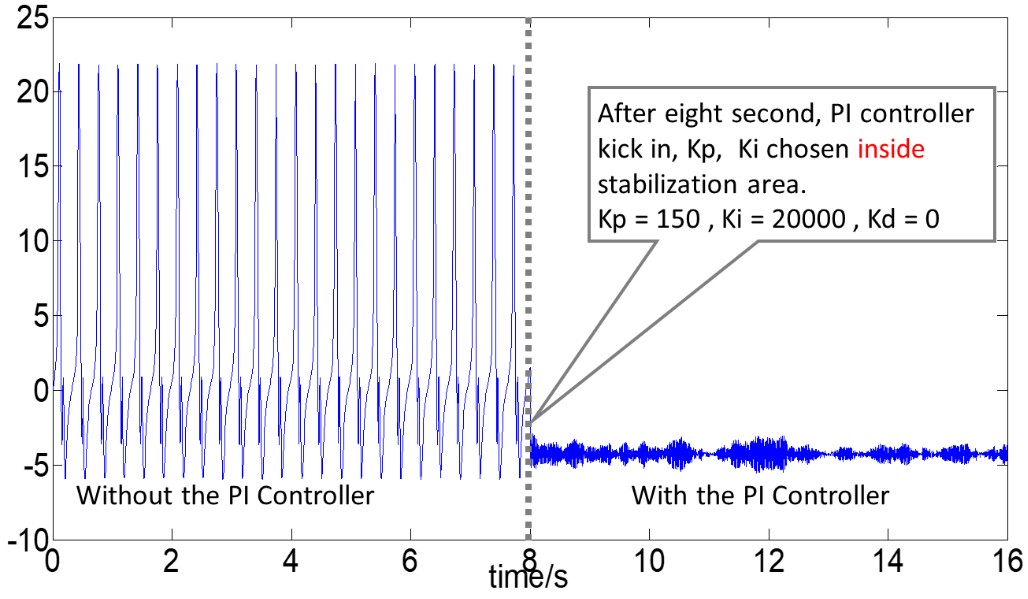


Figure 3.10: In the hyper-excitation scenario ($\boldsymbol{H_e = 7}$, $\boldsymbol{H_i = 22}$), the comparison of output of Jansen's NMM for the first eight-second simulation without the PI controller, and the second eight-second simulation with stimulation feedback from the PI controller. (a) A PI control gain set up is chosen inside stabilisation area ($\boldsymbol{K_p = 400, K_i = 5800, K_d = 0}$). (b) A PI control gain set up is chosen outside stabilisation area ($\boldsymbol{K_p = 400, K_i = 10000, K_d = 0}$).

- ***Low inhibition scenario***

(a)



(b)

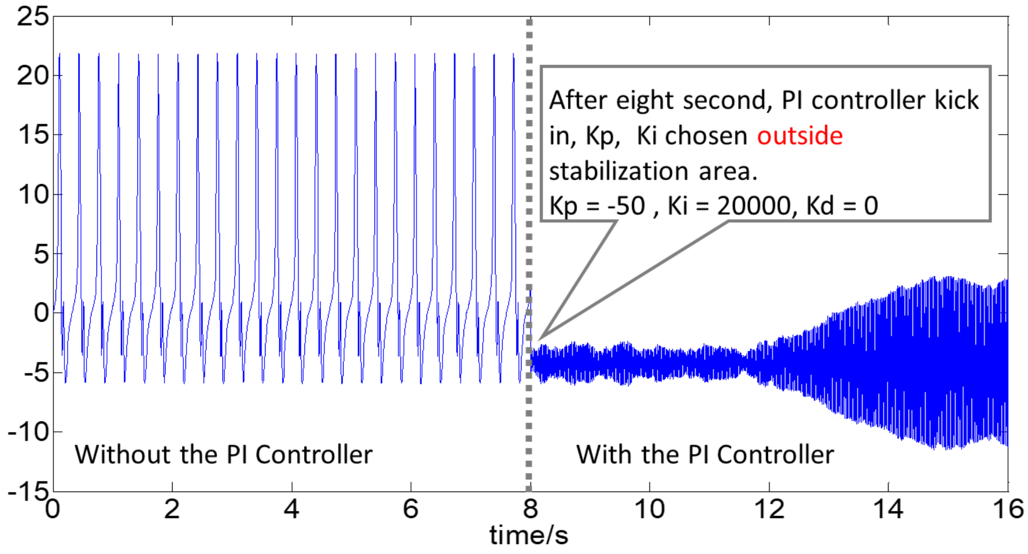


Figure 3.11: In the low inhabitation scenario ($H_e = 3.25$, $H_i = 17$), the comparison of output of Jansen's NMM for the first eight-second simulation without the PI controller, and the second eight-second simulation with stimulation feedback from the PD controller. (a) A PI control gain set up is chosen inside stabilisation area ($K_p = 150, K_i = 20000, K_d = 0$). (b) A PI control gain set up is chosen outside stabilisation area ($K_p = -50, K_i = 20000, K_d = 0$).

Simulation results show that the output signal (local field potential) of Jansen's NMM without the PD control and PI control feedback was high amplitude epileptic seizure activity, which then became low amplitude activity with feedback stimulation from the PD controller and PI controller. A graphical stability analysis method was employed to determine the stability region of the PD controller and PI controller for plotting the stabilised parameter space. As a result, stabilised regions of the PD controller and PI controller parameters have been simulated, which can provide proportion and derivative gain choice to be used for stabilising epileptic seizure activity in Jansen's NMM.

3.5 Controller design for proposed algorithms

This chapter has verified that PI controller design and PD controller design can help suppress high amplitude activity generated by neural mass models which represents high amplitude seizure activity. This also provides us with a solid computation proof that closed-loop control is a promising strategy to help control neurodisorder diseases. The next stage of this work is to implement plausible control algorithms in implantable hardware devices with minimal power consumption. The goal is to establish a quantitative relationship between the chosen controller parameters and the neural mass model's excitatory and inhibitory parameters. This provides a guideline for the choice of controller parameters to help suppress high amplitude seizure signals in the math model.

This modelling work suggests that the PI control algorithm and PD control algorithm could be potential candidates for pacemakers. The tradeoff of mapping the PI control algorithm and PD control algorithm in hardware can be listed as follows:

- A PI controller is more stable than a PD controller
- PI controller hardware implementation memory costs more than a PD controller, as PI controllers require continuous history and memory updates
- A PI controller costs more time to finish per computation than a PD controller

	Stability	Hardware Resources cost	Hardware computing time

			cost
PI control	Yes	Expensive	Long
PD control	No	Cheap	Short

In closed-loop biomedical control systems, we aim to create a closed-loop control system which aims for low computation time cost, meaning minimal computation cost. PI hardware design will take longer to compute and occupy larger memory to implement compared to PD hardware design. Even PIs can provide larger stability regions of gain parameters than PDs. By selecting proper gain parameters inside the PD stabilising area, we can still stabilise the neural mass model by selecting the proper parameters.

3.6 Conclusion

In this chapter, we introduced a mathematical computational study of the closed-loop control of the brain. Jansen's Neural Mass Model has been chosen as a test bench to mimic a human brain in section 3.2. Section 3.3 demonstrated how to apply PID control algorithms for generating feedback as the stimulus feeding into Jansen's neural mass model. Based on the stabilisation area analysis, we have found that with proper PI and PD gain parameters set up, we can stabilise the amplitude activity generated by Jansen's Neuron Mass Model in section 3.4.

This chapter also provides an analytical approach for closed-loop control of brain modelling by providing the flexibility to substitute math models or control algorithms for more exploratory efforts. Furthermore, we also presented an analytical approach to closed-loop PD controllers and PI controllers to determine the stimulation parameters for suppressing high-amplitude epileptic activity in the neural mass model.

The proposed graphical stability analysis approach method certifies that the design of this feedback controller was analytical, revealing a cause and effect relationship in a theoretical manner. This allows us to explore the relationship between the model parameters of inducing epileptic activity and feedback controller parameters, to form a better understanding of the mechanism of suppressing epileptic seizure activity by applying closed-loop feedback stimulation methodology (pharmacology stimulation, electrical stimulation and optogenetic stimulation etc.). Different parameter sets of PD and PI gains have been listed in the following table to provide a better understanding of the clinical seizure onset parameter choice.

PI and PD controller suggestions for NMM model

Closed-loop PD-NMM system	
Epilepsy scenario	**PD parameter suggestion**
Hyper-excitation scenario $H_e = 7$, $H_i = 22$	$\mathrm{K_p} = 100, \mathrm{K_i} = 0, \mathrm{K_d} = -2$
Low inhabitation scenario $H_e = 3.25$, $H_i = 17$	$\mathrm{K_p} = 30, \mathrm{K_i} = 0, \mathrm{K_d} = -1.3$
Closed-loop PI-NMM system	
Epilepsy scenario	**PI parameter suggestion**
Hyper-excitation scenario $H_e = 7$, $H_i = 22$	$\mathrm{K_p} = 400, \mathrm{K_i} = 5800, \mathrm{K_d} = 0$
Low inhabitation scenario $H_e = 3.25$, $H_i = 17$	$\mathrm{K_p} = 150, \mathrm{K_i} = 20000, \mathrm{K_d} = 0$

Relative contribution

Correspondent	Contribution
Miss Lijuan Xia Dr. Patrick Degenaar	1. A closed-loop PD-NMM model and a closed-loop PI-NMM model are investigated in this chapter, different sets of Kp, Kd gain and of Kp, Kd gain parameters inside stabilisation areas are chosen for simulations, and will be employed in next chapter for hardware implementation. 2. This modelling study suggests that PD controllers and PI controllers can help to suppress high amplitude seizure signals in a computational neuron mass model successfully.
Correspondent	**Future work**
Miss Lijuan Xia Dr.Yujiang Wang Dr. Patrick Degenaar	1. A spatial-temporal mathematical brain model needs to be investigated to mimic brain activities for leading a better understanding of brain function in diseased states. 2. Various seizure patterns apart from high amplitude epilepsy seizure-like signals need to be studied to imitate seizure signals. 3. Different control algorithms (PI, PID, machine learning algorithms) need to be applied to intervene with the neural mass models as a closed-loop control system. 4. The optogenetic stimulation math model needs to be conducted to interface with PID algorithms for supplying into the neuron mass model.

Chapter 4. Algorithm Hardware Implementation

4.1 Chapter Overview

The previous chapter has conducted a closed-loop brain control modelling study to suggest a plausible PD control algorithm for intervening with the neuron mass model to suppress epileptic seizures. The main objective of this chapter is to investigate the feasibility of a low power digital implementation of PD algorithm which will be optimized for minimal energy consumption in sparse sampling rate processing application. Figure 4-1 details a general overview of the framework of this chapter. Section 4.1 gives a general overview of this chapter. Section 4.2 compares different digital hardware platforms between the commercial off-the-shelf microcontrollers and FPGAs. Then the comparison results and analysis will be discussed in this section and the selection of the specific digital platform for our biomedical application will be shown as a conclusion. Section 4.3 depicts the design and development of PD firmware implementation for the entire system. Conclusions will be given in the final section 4.4.

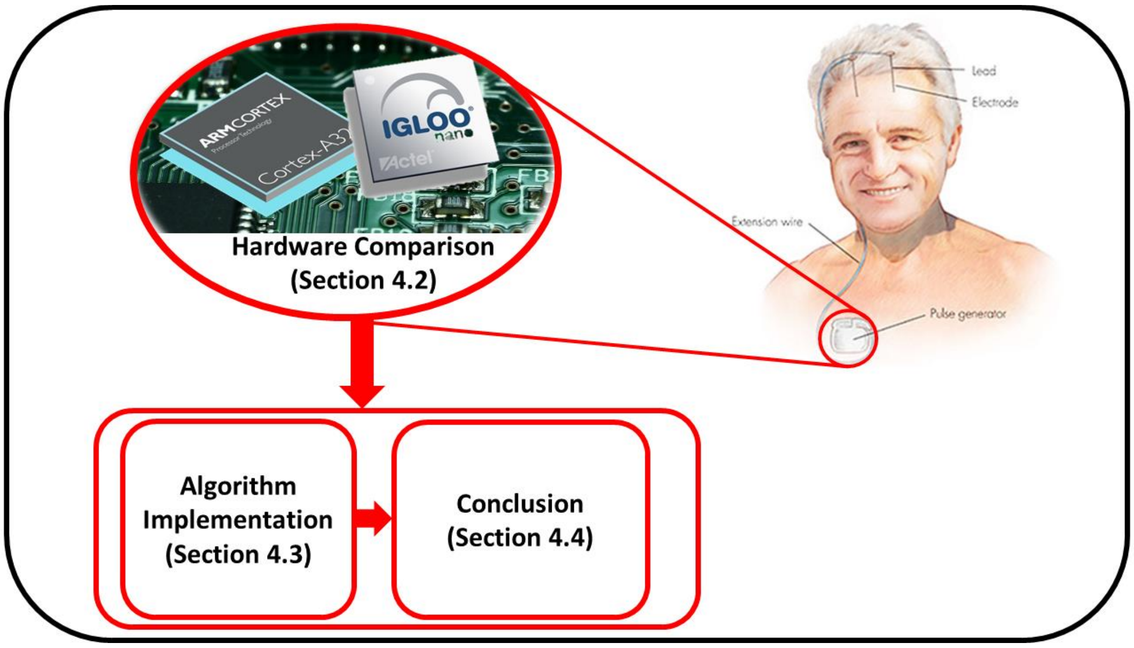


Figure 4-1: Chapter four overview. This chapter starts by describing different hardware platforms, then compares them from different perspectives, then selects a hardware platform for our application. Finally, we describe the firmware implementation of the chosen algorithm in our control unit.

4.2 Hardware Comparison

With the final aim of implanting neural interface for patients with neurological disorders, emerging evidence indicates there is an increasing need for developing wearable, low-power consumption, miniaturized embedded devices. For testing the performance and capabilities of these closed-loop neural interfaces, non-human primates and rodents are preferred by neuroscientists to use for neuroscience experimental models. Thus, there is of overwhelming interest in developing tools for closed-loop control experiments, for freely moving rodents. One of the design debates is a trade-off of the use of different hardware platforms between application specific integrated circuits (ASICs) and digital platforms. Three potential digital implementation platforms are discussed and compared in the following Table 4-1 for a further selection of one of the platforms:

Table 4-1: Comparison between different hardware platforms for embedded system applications. We mainly compare ASICs, FPGAs and MCUs as the implementation platforms regarding re-programmability, power consumption and speed. We also list the requirement for our biomedical application.

	Reprogrammability	**Power Consumption**	**Computing Time Cost**
ASIC	No	Low	Fast
FPGA	Yes	moderate	Fast
MCU	Yes	High	Moderate
Requirement	Yes	Low	Fast

In order to develop an optimal processing unit with a power management system which can be tested in freely moving rodents, we desire to design a platform capable of implementing the closed-loop optogenetic stimulation with tight real-time constraints, and low power consumption to enable a battery life of over 24 hours. Therefore, for this biomedical application, our key requirements are listed as follows:

1) ***<u>Reprogrammability:</u>*** A platform can communicate with the neural interface ASIC probe which constructs an ASIC finite state machine (FSM) of implementing a command interpreter that can send out LFP recordings and receive instructions to control LED emission. The commands are communicated by a digital interface using a serial

peripheral interface (SPI) protocol [63].

2) ***Low latency:*** A platform can require low processing power to perform low latency closed-loop control algorithms with tight time constraints for achieving minimal processing delay.

3) ***Small size:*** A platform can be sufficiently compact to be installed on the freely moving rat system for neuroscience experiments.

4) ***Lightweight:*** A platform need to be lightweight (<20g for a large rodent) to be mounted on the back and head of a freely moving rodent for long time neuroscience experiments.

Hence, we decide to compare two digital platforms between FPGAs and MCUs as digital processors can provide flexible programmability. A general overview of FPGAs and MCUs is given in the following sections.

4.2.1 FPGA Overview

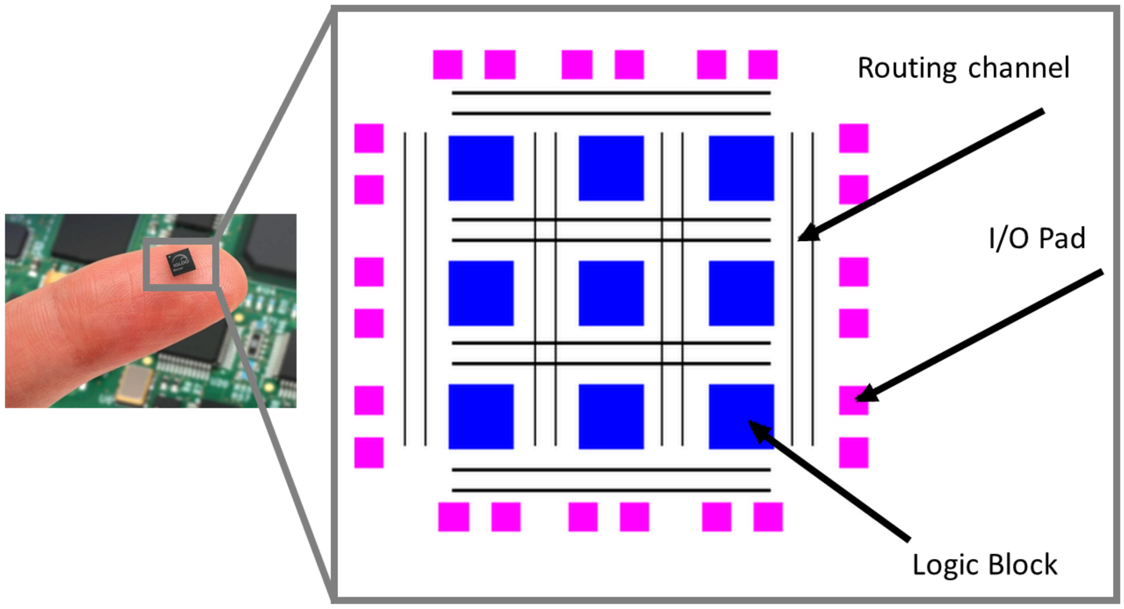


Figure 4-2: The conceptual scheme of FPGA architectures. A basic FPGA platform is composed of an array of logic block, a hierarchy of reconfigurable interconnects which allow the block to be wired together. For most FPGA platforms, logic blocks include memory elements that may be simple flip-flops ore more complete blocks of memory.

An FPGA is defined as a prefabricated silicon device that can be electrically programmed to become any arbitrary design of digital circuits and systems. The conceptual scheme of an FPGA is shown in Figure 4-2, it contains:

- Logic blocks
- Routing channels
- I/O interfaces

A FPGA includes an array of programmable logic blocks and a hierarchy of reconfigurable interconnects which makes the blocks to be wired together. Many logic gates can also be inter-wired in different configurations. The logic blocks can be programmed to perform complex combinational functions or only simple logic gates like AND and XOR. Logic blocks also contain memory elements that are simple flip-flops or more complete blocks of memory. The FPGA configuration is generally specified by the hardware description language (HDL).

There are two main commercial off-the-shelf FPGAs available on the market today based on the basic process technology: SRAM based FPGAs with static RAM memory cells holding their configuration patterns and flash-based logic arrays with nonvolatile memory cells. On one hand, SRAM memory is a volatile memory meaning that the configuration is lost when power is removed. On the other hand, the main other form of memory is flash memory. Flash memory evolves from EEPROM (Electrically erasable programmable read only memory). There are two main types of flash memory: NOR or NAND. Flash memory cell is effectively a transistor in nature. Flash is a similar in composition to a MOSFET with an added floating gate which acts as an electron trap.

The expectation for this project is that the SRAM based FPGAs will present higher static power than the flash-based FPGAs yet yield a lower dynamic power. IGLOO nano is a product nano of non-volatile FPGA released by Microsemi Company. IGLOO nano flash FPGA provides ultra-low static and low dynamic power consumption. The logic size is ranging from 10000 gates to 250000 gates. The unique capabilities of Flash*Freeze mode in the non-volatile FPGA fabrics can help reduce power dramatically.

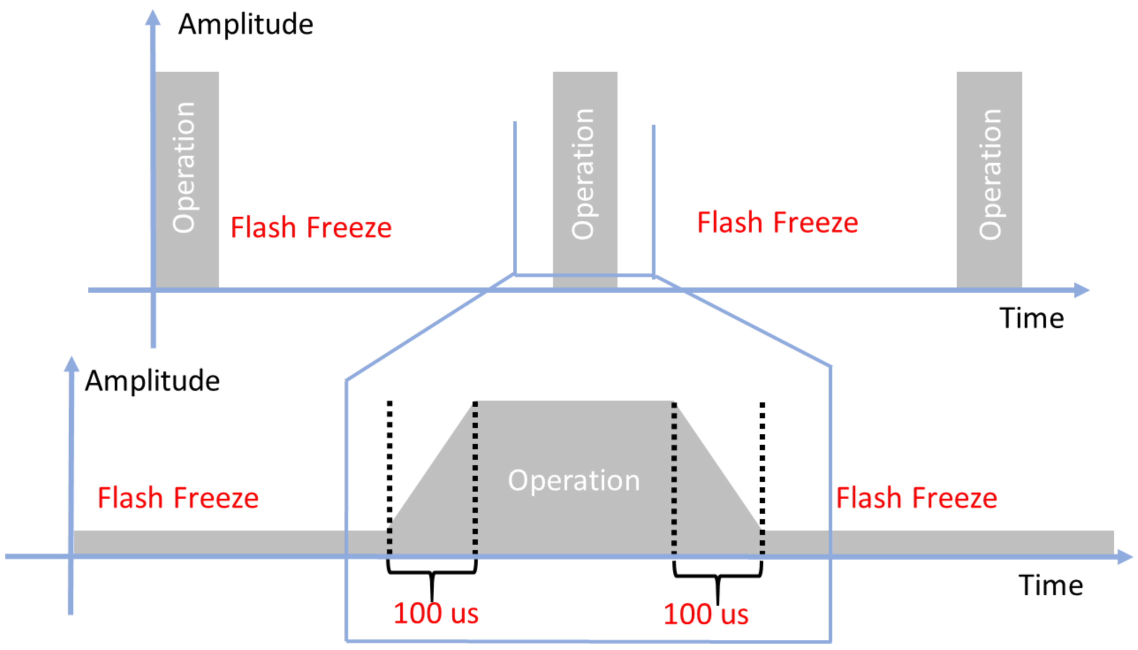


*Figure 4-3: Example system of enter and exit from Flash*Freeze mode to regular normal operation within 100 usec. Flash*Freeze mode allows the non-volatile FPGAs to enter a low power (~24µW) mode and retain all internal memory and flip-flop states.*

Flash*Freeze technology can enable the rapid stopping and starting of the FPGA fabrics and related I/O while preserving the state of FPGA fabrics shown in Figure 4-3. This mode will also allow the device to go into a low power mode that also holds all internal memory and flip-flop states as well as output values. There is a great potential medical application as a prime area for using flash freeze mode taking advantage of the relatively low sampling rate. This would allow for prolonged periods of down time yet remove the requirement for re-configuration and would also respond rapidly to wake-up requirements.

To conclude, the major advances in adopting non-volatile FPGAs than SRAM-based FPGAs for biomedical applications can be concluded into three reasons:

1. ***Lower System Cost:***

High-performance non-volatile FPGAs can deliver an analogous features and functions identical with SRAM-based FPGAs on the grounds that marvelous progresses have been made in shrinking flash memory cells and the capability to integrate the flash into unconventional logic processes.

2. ***Reduced System Footprint and Power Consumption***

An external configuration memory is not required for flash-based logic arrays. As a result, non-volatile FPGAs generally has reduced system footprint and lower power consumption.

3. ***Less Startup Time:***

For non-volatile FPGAs, the configuration memory is with the logic arrays on the same chip while SRAM-based FPGAs still require more startup time to load the configuration time.

4.2.2 MCU Overview

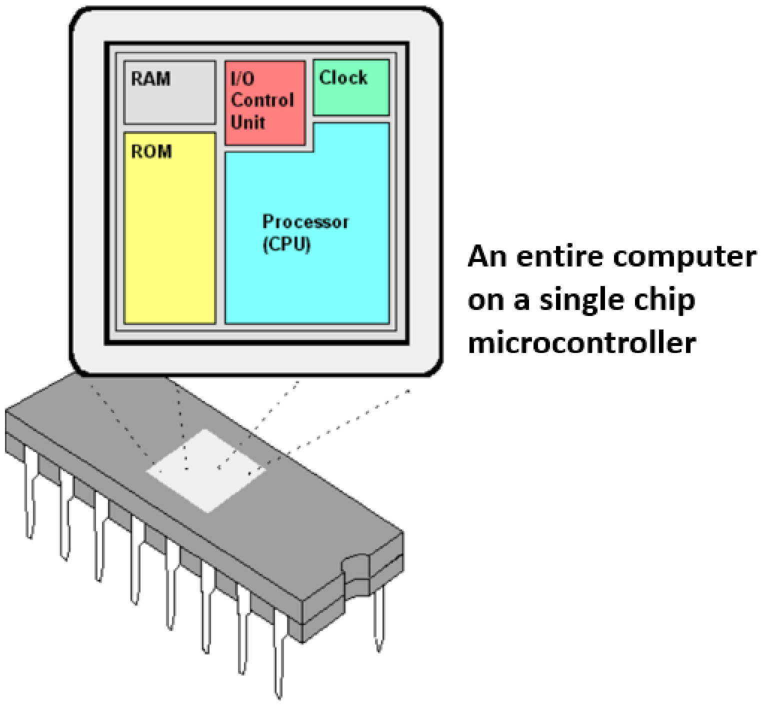


Figure 4-4: A simplified microcontroller scheme. A single chip microcontroller contains the processor includes the CPU (Processor), non-volatile memory for the program (ROM or Flash), volatile memory for input and output (RAM), clock module and I.O control unit.

A microcontroller (MCU) is a small computer on a single integrated circuit. Computer architecture can be thought as a set of rules and methods that describe the functionality, organization and implementation of a given computer system. In Figure 4-4, it can be seen that computer architecture involves the instruction set architecture design, microarchitecture design, logic design and implementation.

One of the key differences between powerful CPU and MCU is the way how the instruction sets are implemented to control the functionality of the processor. There are two main computer instruction sets shown in Figure 4-4.

- CISC (Complex Instruction Set Computer)
- RISC (Reduced Instruction Set Computer)

In this efforts, ARM Cortex M4 based architecture is chosen for its wide capability of core of DSP (Digital Signal Processor) with an FPU (Floating Point Unit), along with further instructions for handling single precision floating. These instructions operate on an extended register bank of 3 single precise registers and provide single precision floating point arithmetic, comparison, data transfer between the extension registers, core register and memory .

4.2.3 Comparison between MCUs and non-volatile FPGAs

In this section, we will compare a flash-based FPGA with a MCU in terms of power consumption and time cost. The experimental methodology is to apply two platforms to organize an optimized Linear Time-Invariant (LTI) filter implementation for providing a fair comparison between the MCU (ARM Cortex M4) and the non-volatile FPGA (IGLOO nano). Then we will choose a digital platform for our application based on the measured performance. The main reason we use FIR filter as an exemplar example to implement is that it is the fundamental unit of our closed-loop PID control algorithms which will be also demonstrated in the next section 4.3.

In digital hardware implementation, the PID digital implementation falls within the scheme of linear, time invariant (LTI) filters. In this case, the proportion, integration and derivative can be treated as an LTI filter which can be represented by convolution with a finite impulse response of truncated length impulse h(t). The LTI filter can be written as:

$$y = x(n) * h(n) \qquad (4\text{-}19)$$

The formula is read as y is the convolution of x and h where the operation of sum of products is called convolution. This formula can also be detailed as

$$y = \sum_{n=0}^{N-1} x(t-N)h(n) \tag{4-2}$$

Truncating the length of the corresponding FIR filter will impose a low frequency limit of the filter. It means the computed integral signal will be correct only above a frequency given by the reciprocal of the length of FIR filter. Convolution, a basic element of FIR filter, has been described in Figure 4-5. We will detail the FPGA implementation and MCU implementation of the convolution in the following sections.

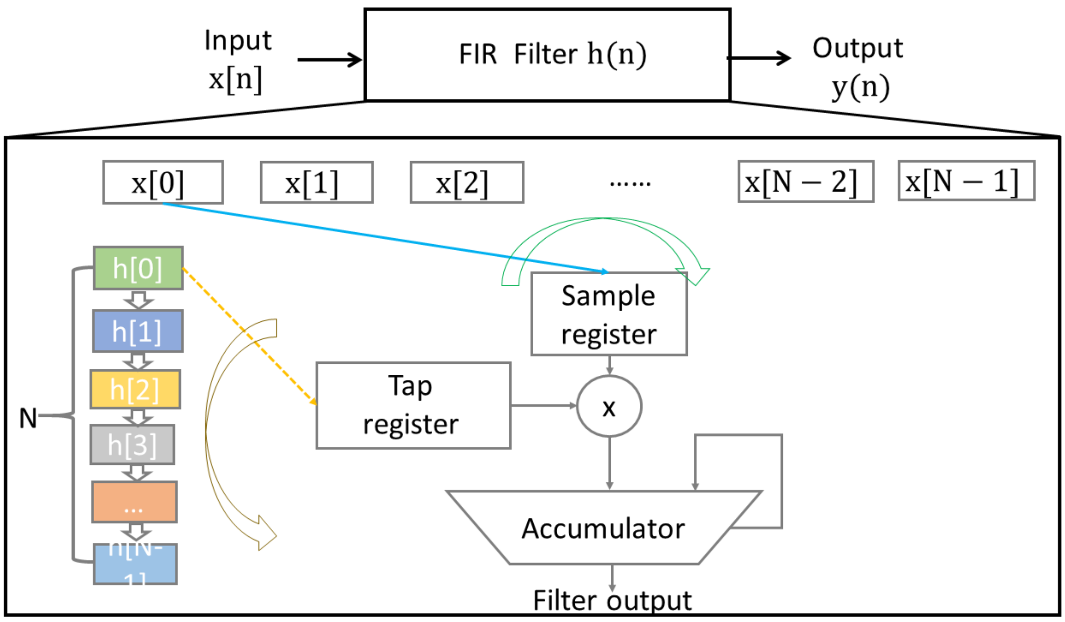


Figure 4-5: A simplified hardware design of implementing the linear time invariant digital design of FIR filter. The row array x_0, x_1, x_2, x_3 … … is used for saving incoming signal while one column array h_0, h_1, h_2, h_3 … … is applied for holding the pre-set FIR kernel of impulse response.

- *Optimized FPGA implementation of FIR filter*

In Figure 4-6, it shows how to construct a finite impulse response implementation (supposing the filter taps is 16 taps) in VHDL for FPGAs. A basic logic cell architecture of FIR filter is shown as follows.

i. A 16*1 row buffer bank of 8-bit row register is receiving local field potential from optrode for further PID control

ii. A 1x1 MUX is used for row buffer bank to select corresponding row register for multiplication

iii. Three 16*1 column buffer banks of 8 bit column register is used to hold proportional Dirac Delta kernel, integral unit step kernel , derivative Gaussian kernel as convolution kernel to do further multiplication.

iv. A 1x1 MUX is used for column buffer bank to select corresponding column buffer bank for multiplication

v. An accumulator adds all results of multiplication which has been send to output DACDAT buffer bank for generating further close loop neural stimulation.

vi. A 16*1 buffer bank of 8-bit register is PID control output for generating further pulse width modulation feedback to optrode for close loop optogenetic stimulation.

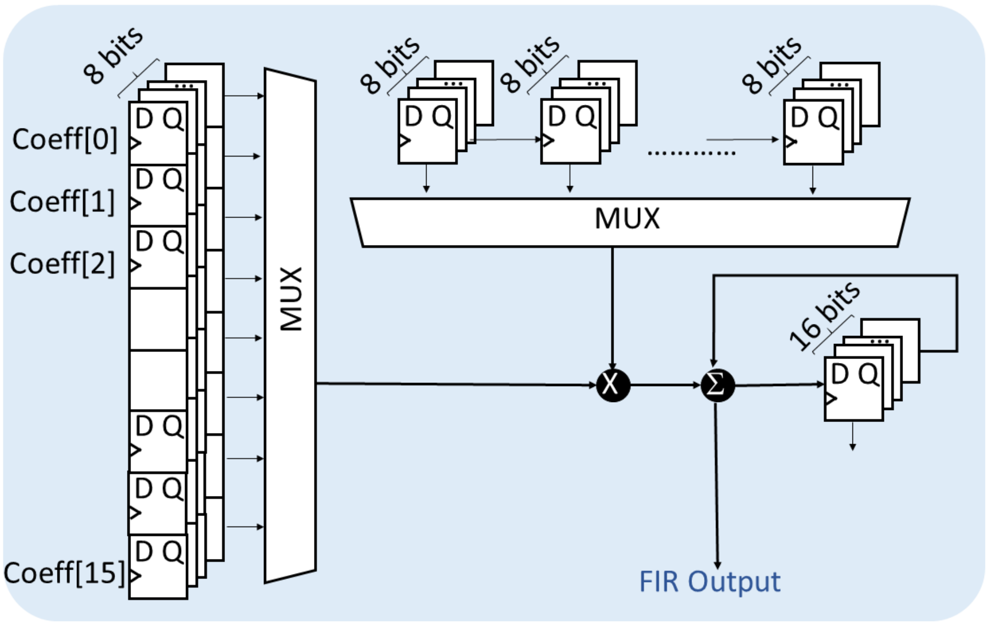


*Figure 4-6: Direct FPGA Implementation of FIR Filter Architecture. It describes an optimized FIR filter implementation in the non-volatile FPGA. A 16*1 row 8-bit buffer bank and a 1*16 column 8-bit buffer bank are employed to store incoming recordings*

and filter taps respectively. Two multiplexers are used to select incoming data and filter coefficients for multiplication to do further accumulation.

Listing 4-1: FPGA Convolution Algorithm in VHDL

```
shift_register_pro : process(fpga_clock)
begin
   If(rising_edge(fpga_clock)) then
    shift_buffer(0)                  <= fpga_in_8_bits;
    shift_buffer (taps downto 1)     <= shift_buffer (taps-1 DOWNTO 0);
end if;
end process shift_register_pro ;
accumulation_pro : process (SHIFT_BUFFER)
begin
   for i in 0 to 15 loop
     accumulator(15 downto 0)     <= accumulator (15 downto 0) + shift_buffer
(i)*coeff(i);
   end loop;
     filter_out(7 downto 0)       <= accumulator (7 DOWNTO 0);
end process accumulation_pro;
```

- *Optimized FIR Filter of MCU implementation*

In comparison, we construct an optimized FIR filter implementation in C++ language for microcontrollers shown in Figure 4-7

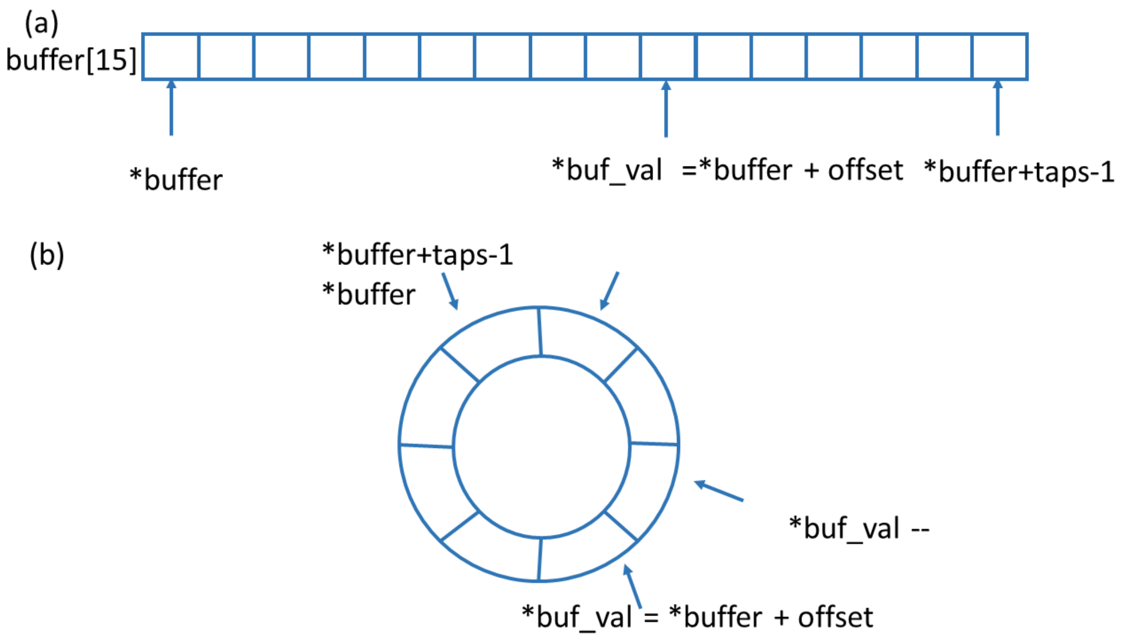


Figure 4-7: Ring buffer implementation of FIR filter in Microcontroller. (a) The row buffer is used to save input into the buffer array. Pointer is used to address the corresponding

buffer. (b) A ring representation of data buffer array.

Listing 4-2: MCU Convolution Algorithm in C++

```
int main(void)
{
PE_low_level_init();
  for(;;) {
  uint8_t *h_start  = h;
  uint8_t *h_end    = h + taps;
  uint8_t *buf_val  = buffer + offset
  AD1_Measure(TRUE);
  AD1_GetValue8(&input);
  *buf_val = input;
  while( buf_val >= buffer)
  output += *buf_val-- * *h_start++
;
  buf_val    = buffer + taps-1;
  while( h_start < h_end)
  output +=*buf_val-- * *h_start++;
  if(++offset >= taps)
  offset    = 0;
}
```

In Listing 4-2, It demonstrates an optimized ring buffer FIR Filter execution in microcontroller which is thus the process of an infinite for(;;) loop that triggers a new iteration every time that new sample input is available in AD1GetValue(); Two pointers are applied to save addresses of incoming data address and filter taps address separately. One multiplication is for multiplying incoming data and filter taps for further accumulation. Two pointers will shift with each iteration.

It mainly consists of the following instructions:

- We first digitize the incoming analogy signal for an ADC module for sampling.
- We then assign a buffer pointer for saving the address of the incoming buffer.
- After saving the measurement into the incoming buffer, we do the multiplication with kernels for accumulation as the filter output.
- The filter output can be pushed into a DAC module as a continuous output.

- *Measured FPGA Power Comparison*

After the FIR filter implementation of the MCU and the non-volatile FPGA, current consumption of the microcontroller and the FPGA have been tested a multimeter (Digital *multimeter, Truevolt Series 34465A) demonstrate in Figure 4-12(a). The measurement of the non-volatile FPGA implementation is comparatively obvious. Once the FPGA board is powered up, we can evaluate the power usage by current consumption, using the current measurement pins on the boards.*

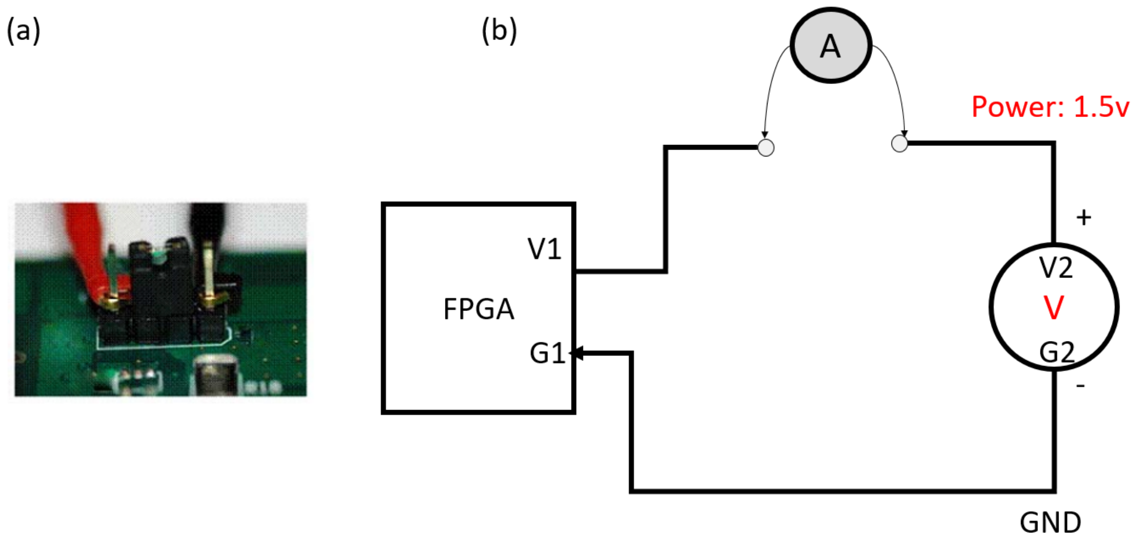


Figure 4-8: Set the multimeter to measure current and attach the probe of the multimeter to pin1 and pin4 when the board is in normal operation. (a) the photograph taken from the hardware testbench setup. (b) the simplified schematic demonstration of the closed-loop hardware set up.

Steps to measure the current consumption of the non-volatile FPGA:

- Take out the power (V1) and ground pin (G1) of the power analyzer (Agilent Technologies N6705B) DC Power Analyzer.
- Take out the power (V2) and ground pin (G2) of the non-volatile FPGA chip (IGLOO nano chip).
- Connect the power (V1) and ground pin (G1) of the power analyzer and the power (V2) and ground pin (G2) of the non-volatile FPGA.

- *Measured MCU vs FPGA Power Consumption*

To provide a fair comparison between the MCU and the non-volatile FPGA, we have implemented the optimized FIR filter in the flash FPGA and MCU. We first set the filter taps and architecture as 16 taps.

For further analysis, we decided to draw the energy consumption per convolution to provide comparison. We have set the microcontroller and the non-volatile FPGA frequency as 1MHz. Thus, comparison of energy per convolution has been compared in Figure 4-9. Red line plots how the energy cost per convolution of the non-volatile FPGA changes with filter tap increases. Blue line draws how the energy cost per convolution of microcontroller changes with filter tap increases. When filter taps are below 250, the energy cost per convolution of the non-volatile FPGA implementation is lower than the energy cost per convolution of microcontroller. If the filter taps are below 50, the non-volatile FPGA has an obvious edge over than microcontroller than microcontroller.

Then comparison of computing time cost per convolution has been compared in Figure 4-10. If we set the non-volatile FPGA and microcontroller as 1MHz, the computing time of filter implementation between the non-volatile FPGA and the MCU has been shown in Figure 4-10. It can be seen that the FIR filter taps are below 50 taps, the non-volatile FPGA has an edge over microcontroller. As FIR filter is implementation in a ring buffer way, so the time cost and current consumption of filter implementation of microcontroller will not change dramatically with taps. Then MCU shows its strength over the non-volatile FPGA.

To conclude, it can be seen that with the same frequency setting (1MHz), the non-volatile FPGA takes less time than microcontroller in implementing filter when filter tap numbers are less than 128. When filter taps is 16, the IGLOO nano non-volatile FPGA is 8 times less than microcontroller implementing 16 taps filter. Under this circumstance, the IGLOO nano FPGA takes 2.392 nJ energy to implement a 16 taps FIR filter while microcontroller Cortex M4 spends 34.479 nJ energy to execute 16 taps FIR filter.

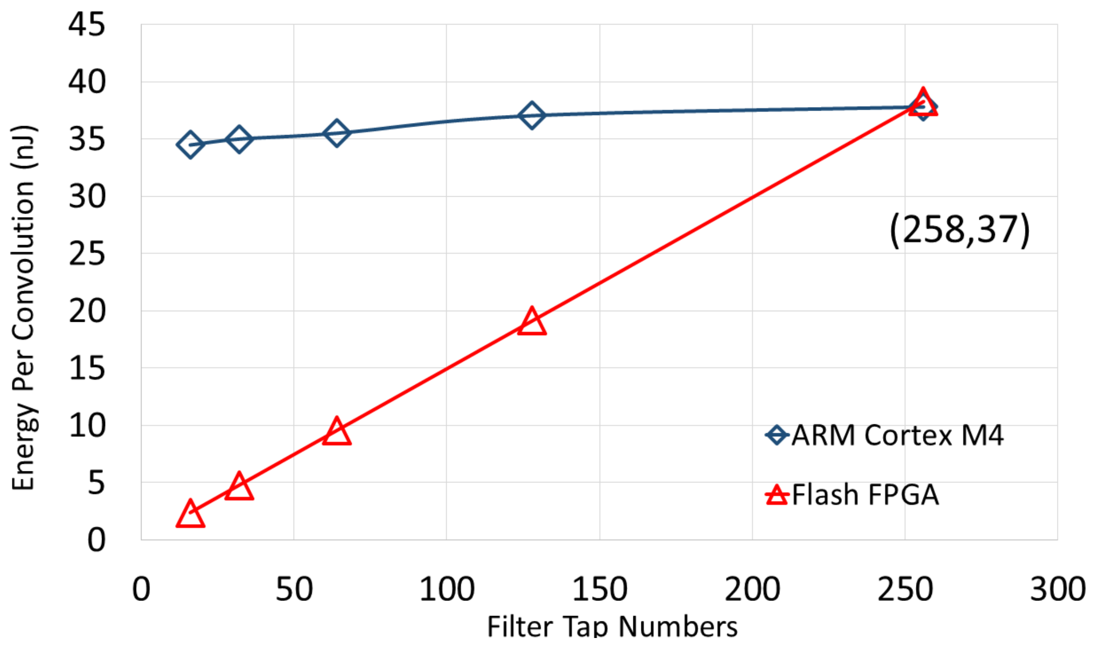


Figure 4-9: Comparison of measured time cost per optimized convolution (us) between the FRDM K22F MCU ARM CORTEX M4 and the IGLOO Nano FPGA. The blue line shows how the measured energy consumption per convolution in ARM Cortex M4 increasing with filter taps. The red line shows how the measured energy consumption per convolution in IGLOO nano increasing with filter taps.

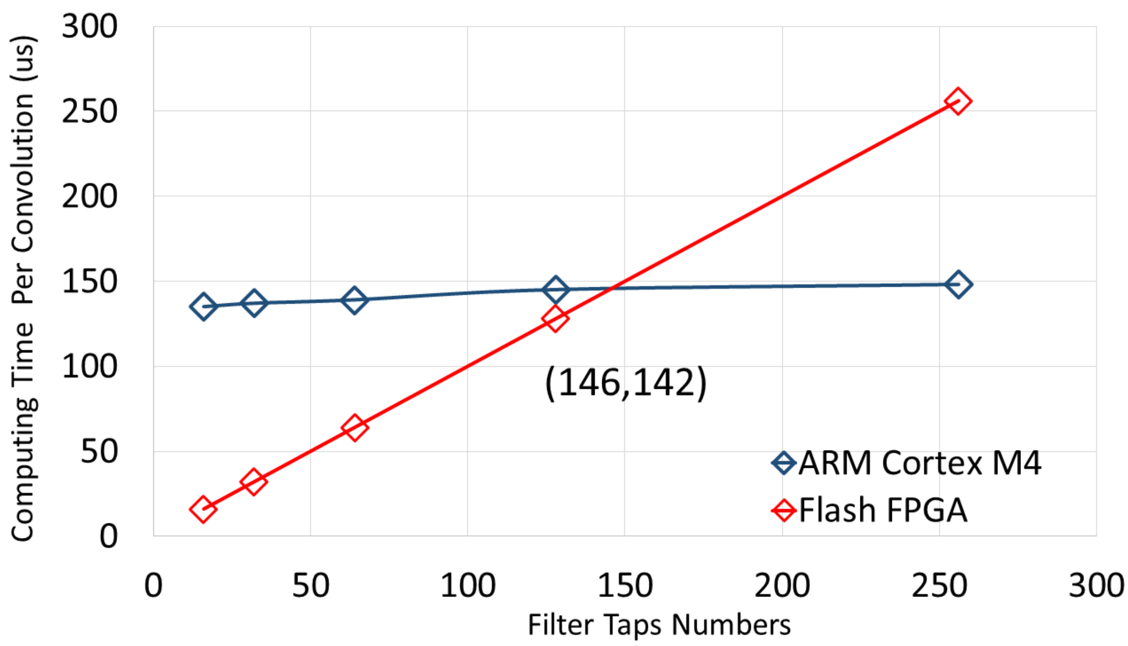


Figure 4-10: Comparison of measured energy consumption per optimized convolution (nJ) between the FRDM K22F MCU ARM CORTEX M4 and the IGLOO Nano FPGA. The blue line shows how the measured computing time cost per convolution in ARM Cortex M4 increasing with filter taps. The red line shows how the computing time cost per convolution in IGLOO nano increasing with filter taps.

Conclusion:

a. *Energy cost per convolution: The non-volatile FPGA has an edge over than microcontroller in energy cost per convolution algorithm implementation when filter taps is below 250 when the MCU and the non-volatile FPGA are set as the same frequency.*

b. *Time cost per convolution: The non-volatile FPGA has an edge over than microcontroller in time cost per convolution algorithm implementation when filter taps is below 150 when the MCU and the non-volatile FPGA are set as the same frequency.*

c. *Filter tap: With filter taps increase, the MCU will show strength over the non-volatile FPGA in time cost and current consumption.*

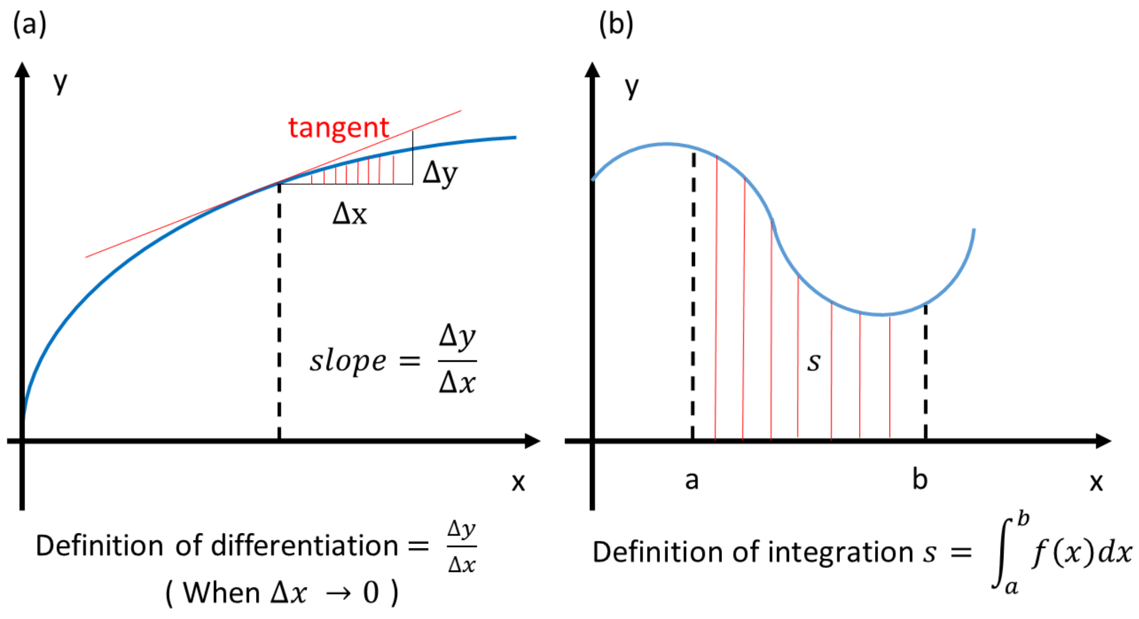
4.3 PD Algorithm Implementation

This closed loop microsystem is aimed at intervening neural network via generating closed-loop optogenetic feedback to control epilepsy. Based on the previous modelling effort in Chapter 3.2, the PID algorithm show it can be applied to intervene with the activity of neural mass model to suppress epilepsy activity.

In industry, proportional-integral-differential (PID) systems have become the most commonly used closed-loop controllers used within industry. This is due to their simplicity and effectiveness in processing an error signal (actual measured signal compared to input reference signal) and producing a response to reduce/remove this error. In the above system example, the desired signal, which could be a set or varying value, is supplied to the PID controller. This input will produce a response by the process that is being controlled. This response is fed back to the PID which subtracts it from the input reference to produce an error (E(t)). It is this error that is used within the PID process itself. Each element carries out its individual mathematical computation and the results are summed together. The resulting signal is then used to drive the process. This operation will continue in pursuit of an error signal equating to zero.

- *From Math to Hardware*

Figure 4-11 describes the math definition of derivative and integral. Mathematically, the derivative of a given function is the slope of the curve at any point. The integral of a given function is the area subtended under the curve between two points.



(a)
y
tangent
Δy
Δx
$slope = \frac{\Delta y}{\Delta x}$
x
Definition of differentiation $= \frac{\Delta y}{\Delta x}$
(When $\Delta x \rightarrow 0$)
(b)
y
s
a
b
x
Definition of integration $s = \int_a^b f(x)dx$

Figure 4-11: Mathematical definition of differentiation and integration. Differentiation can be defined as differential calculus which concerns with the study of the rates at which quantities change. Integration can be defined as integral calculus. A definite integral of a function can be represented as the signed area of the region bounded by its graph.

Here we need to present the effect of the differentiation and integration operator applied to recorded brain signals. Brain signal is obtained by first converting the brain signal in an electrical voltage signal, by means of a sensor. The electrical waveform is sampled by Analog-to-Digital converter, at a sample rate of 100 Hz (that is, 100 samples per second are collected). Each sample is a floating-point number. After the signal is sampled, we need to send it to the hardware for processing. After the hardware processing, it will send the digitalized signal to DAC for generating an analogy signal. Let's take an example. If we choose a filter length N=16 samples, and the sampling frequency is 100 Hz which means per second 100 samples are taken and the sampling rate is 0.01 second. It will give us a correct integral filtering above 100 Hz. The optrode sampling rate is 100 samples/second which represents every sample data costs 0.01

second, and the local field potential frequency band of interest is 1Hz to 100Hz. accordingly, we set the filter window taps is 100.

So, what is the time domain representation of FIR filter to achieve proportion, integration and differentiation?

Table 4-4: FIR Filter Parameters Taps Number Calculation Steps

Filter Parameter	**Filter Parameter Description**
Sampling Frequency	$f_s = 100Hz$ (4-3)
Sampling Ratio	$d_t = 0.01s$ (4-4)
Cut off frequency	$f_{cutoff} = 1Hz$ (4-5)
Filter Taps	$Ntap = \frac{1}{f_{cutoff} * d_t} = \frac{1}{1 * 0.01} = 100$ (4-6)

1) Proportion

If a signal is convolved with Dirac delta function $\delta(t)$, the result is identical to the original signal, except for a delay due to the position of the non-zero inside the Dirac Delta FIR Filter. In time domain, the $\delta(t)$ function is a null filter which only contains a zero amplitude for all samples except for one sample which contains a value of 1. Theoretically, the Dirac delta function $\delta(t)$ cannot be realised in hardware. A Gaussian curve can be used to realise the Dirac delta function $\delta(t)$ in hardware. It will generate the same theoretical convolution results but will produce incremental weighting to each input value as the filter output.

Figure 4-12 describes the theoretical delta function. The impulse response of proportional filter is shown Figure 4-13 (a) and the fast Fourier transform (FFT) of proportion kernel is displayed in Figure 4-13 (b). It can be seen frequency spectrum of proportional filter is ranging from 0 to 100Hz which means that the proportion filter is a low pass filter in frequency domain.

$$y(t) = x(t) * \delta(t) = x(t) \tag{4-7}$$

The proportional filter kernel can be written as:

$$h(t) = \delta(t) \tag{4-8}$$

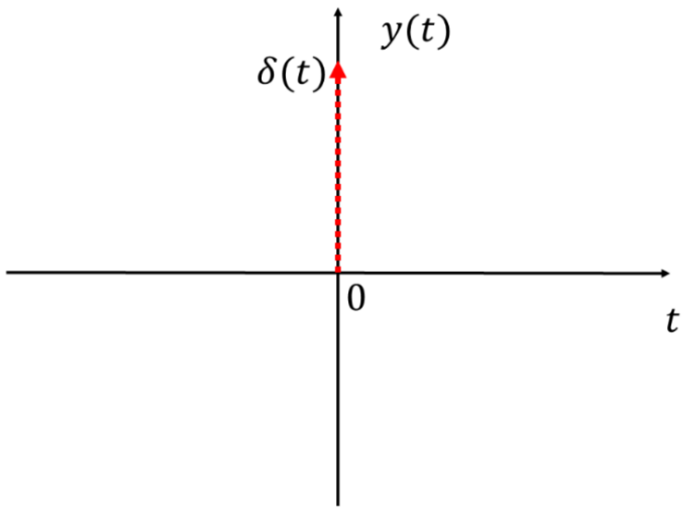


Figure 4-12: Schematic representation of the theoretical Dirac delta function by a line with an arrow. The height of the arrow is to specify the value of any multiplicative constant.

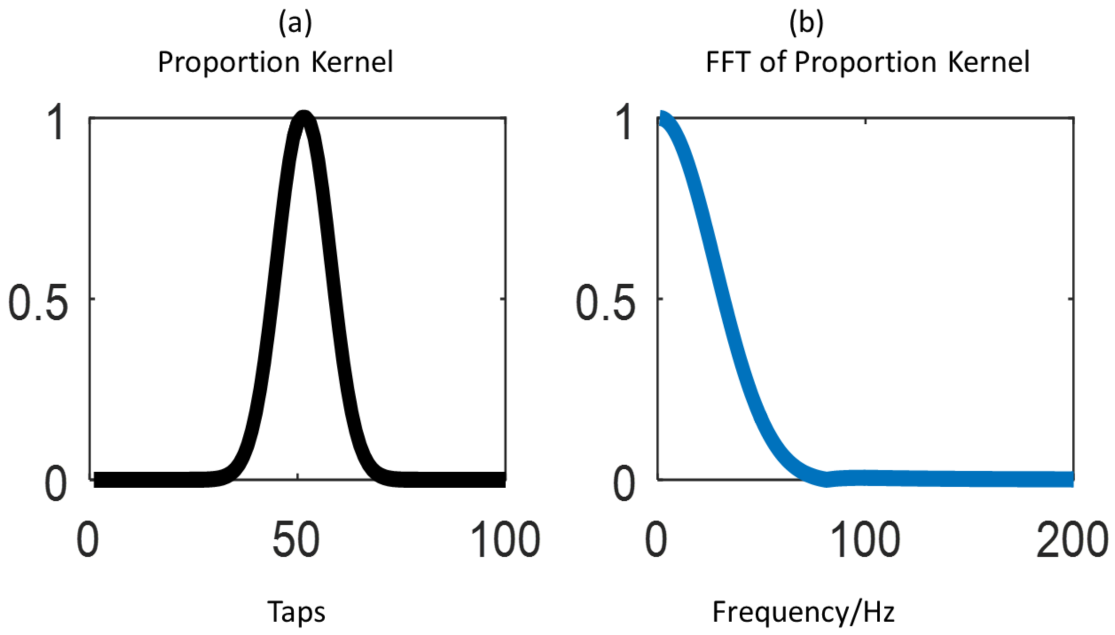


Figure 4-13: Proportional Kernel of Proportion Controller. (a) Impulse Response of Proportional Controller in Time Domain. (b) Frequency Response of Proportional Controller Analysed by Fourier Transform in frequency domain.

2) Integration

The integral operation will correspond to the past errors. The integration of error will accumulate over time which will allow the integral control to overcome the small current error. If the $x(t)$ signal is convolved with the integral $\delta(t)$, the output signal will be a time integral of the original signal. If we apply the integration of the $\delta(t)$, it will generate a unit step function also called Heaviside step function. Under this circumstance, we should continue the integration for a long time, ensuring to get a FIR filter for enough length, for obtaining proper integration over all the frequency spectrum. The theoretical Heaviside function is shown in Figure 4-14.

The integration filter kernel is shown in Figure 4-14. The impulse response of integration filter is shown in Figure 4-15(a), the FFT of integration kernel is displayed in Figure 4-15(b). From frequency domain, the integration filter is a low pass filter which the cut off frequency of integration filter:

$$y(t) = x(t) * \int \delta(t)dt = \int x(t)dt * \delta(t) = \int x(t)dt \tag{4-9}$$

The integration filter kernel can be written as:

$$h(t) = \int \delta(t)dt \tag{4-10}$$

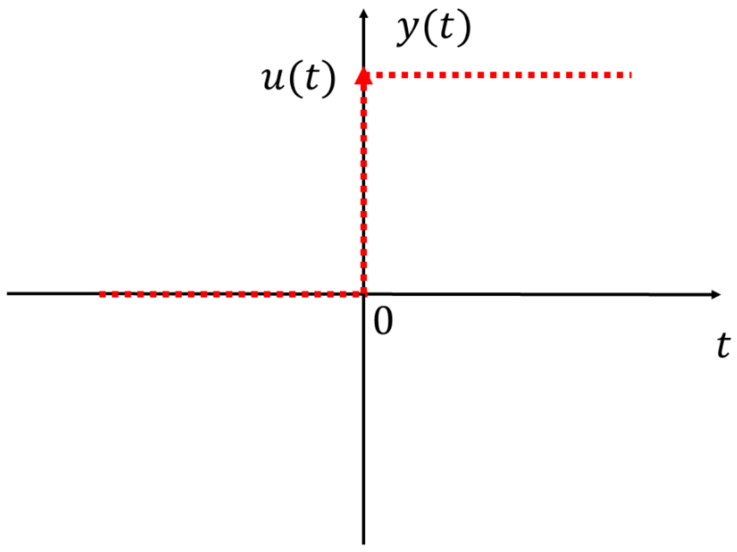


Figure 4-14: Schematic representation of the theoretical Heaviside function by a line with an arrow. The height of the arrow is to specify the value of any multiplicative constant.

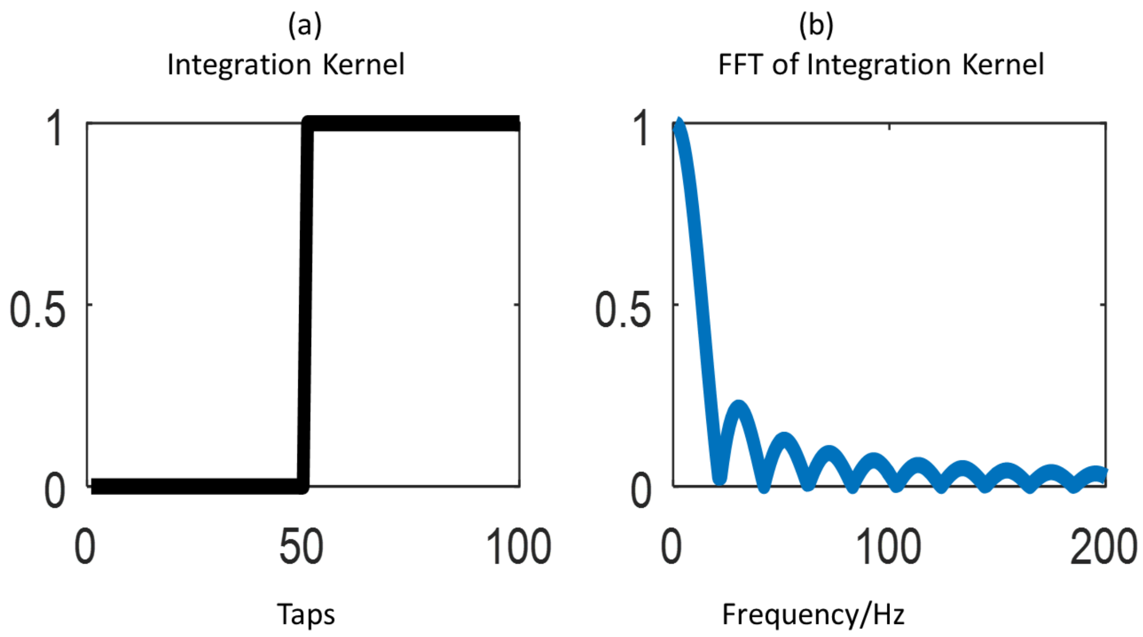


Figure 4-15: Integration Kernel of Integration Controller. (a) Impulse Response of Integration Controller in Time Domain. (b) Frequency Response of Integration Controller Analysed by Fourier Transform in frequency domain.

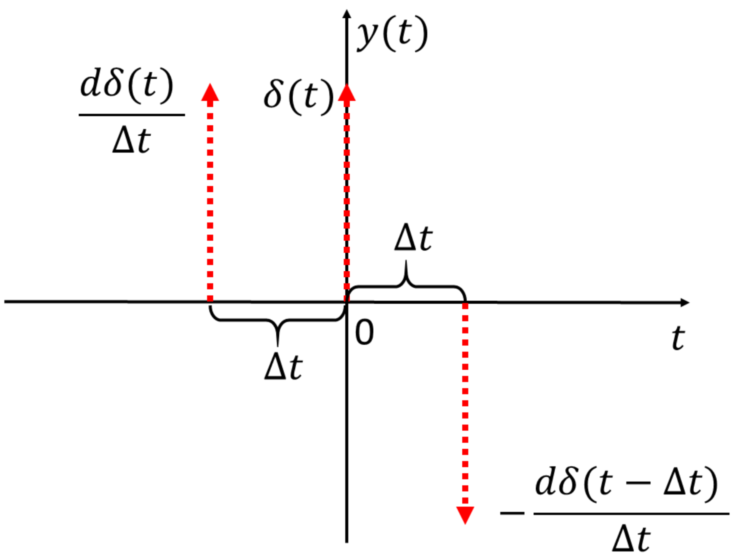
3) Differentiation

If the $x(t)$ signal is convolved with the differentiation of $\delta(t)$, the output signal will be a time derivative of original signal. If we differentiate the $\delta(t)$, we will get an impulse response of Unit Doublet function in mathematical definition. The differentiation filter kernel is shown in Figure 4-16. The impulse response of differentiation filter has displayed in Figure 4-17(a) and the FFT of differentiation kernel is displayed in Figure 4-17(b). From the frequency domain information of differentiation filter, differentiation filter is a band pass filter.

$$y(t) = x(t) * \frac{d\delta(t)}{dt} = \frac{dx(t)}{dt} * \delta(t) = \frac{dx(t)}{dt} \tag{4-11}$$

The differentiation filter kernel can be written as:

$$h(t) = \frac{d\delta(t)}{dt} = \frac{d\delta(t) - d\delta(t-\Delta t)}{\Delta t} = \frac{d\delta(t)}{\Delta t} - \frac{d\delta(t-\Delta t)}{\Delta t} \tag{4-12}$$



$y(t)$
$\frac{d\delta(t)}{\Delta t}$
$\delta(t)$
Δt
Δt
0
t
$-\frac{d\delta(t-\Delta t)}{\Delta t}$

Figure 4-16: Schematic representation of the derivation of the theoretical Dirac delta function by lines with arrows. The height and symbol of the arrow are to specify the value of any multiplicative constant.

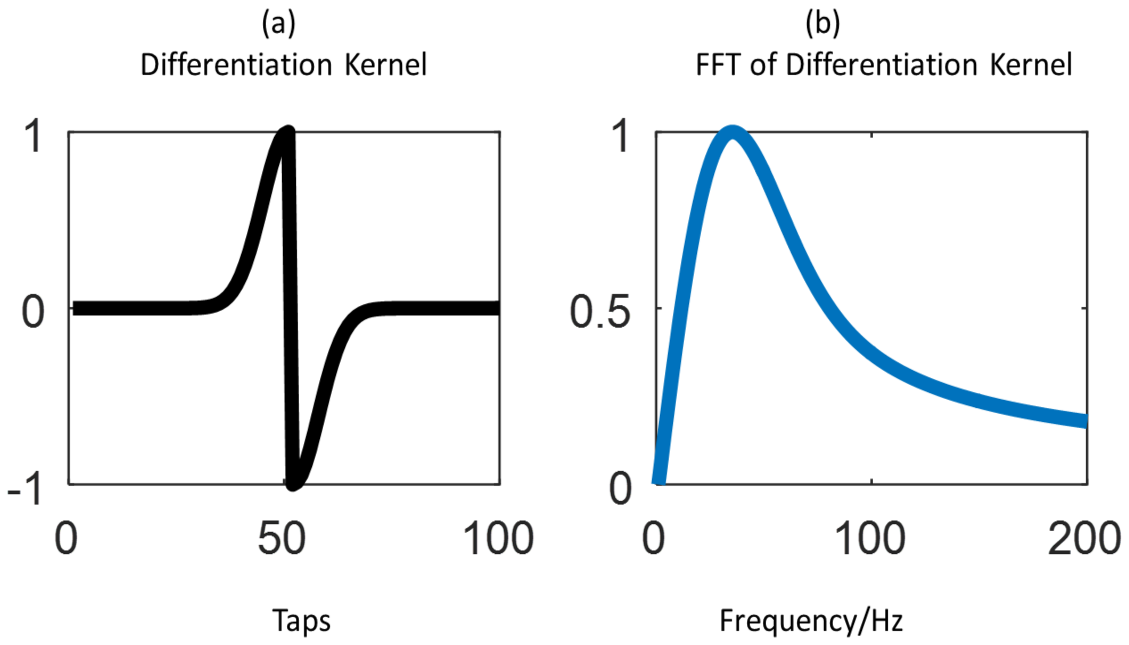


Figure 4-17: Differentiation Kernel of Differentiation Controller. (a) Impulse Response of Differentiation Controller in Time Domain. (b) Frequency Response of Differentiation Controller Analysed by Fourier Transform in frequency domain.

The proportional, integration, and differentiation kernels will be stored as lookup tables in the non-volatile FPGA implementation. Physically, these are stored as three column buffer banks consisting of an 8-bit column registers implementing on novel flash-based logic arrays. These three kernels will be employed to do multiplication with incoming digitalized local field potential recordings for achieving convolution.

4.4 Conclusion

This chapter reviews different reprogrammable digital hardware candidates from computing time and power consumption. Section 4.2 conducted a case study of comparing FIR filter implementation on a microcontroller and a non-volatile FPGA. The non-volatile FPGA outweigh microcontroller in terms of computing time and power consumption for our application. Section 4.3 introduces a hardware implementation of PID controller from theoretical analysis and digital hardware implementation. This chapter has laid the foundation for next chapter for hardware candidate selection and PID algorithm integration with a bi-directional neural interface.

Relative Contribution

Correspondent	Contribution
Miss Lijuan Xia Dr. Patrick Degenaar	1. The digital hardware evaluation between FPGAs and MCUs has been reviewed in this chapter. 2. The feasibility study of non-volatile FPGAs has been presented by comparing the convolution implementation on an off-the-shelf microcontroller and a non-volatile FPGA. 3. This chapter also reports the feasibility study of PID algorithm implementation onto a wearable digital processor in the next chapter.
Correspondent	**Future Work**
Miss Lijuan Dr. Patrick Degenaar	1. A more comprehensive comparison between non-volatile FPGAs and MCUs need to be studied to prove the feasibility of the non-volatile FPGA. 2. More digital processors (GPUs, DSPs) can be applied to conduct a more fair and comprehensive comparison.

Chapter 5. Closed-loop Energy-Efficient Digital Processor

5.1 Chapter Overview

Recent neuroscience studies have demonstrated that considerable information about brain states can be contained in low-frequency Local Field Potentials (lf–LFPs; below 5 Hz) with applications in real-time closed-loop neurostimulation for neurological disorders [64], [65], [66]. Given these signals can be sampled at low sampling rate (below 100 Hz) and thus provide a sparse data stream, there is an opportunity to design implantable neuroprosthesis with long battery lifetime and sufficient processing power to implement the long-term and real-time closed-loop control algorithms.

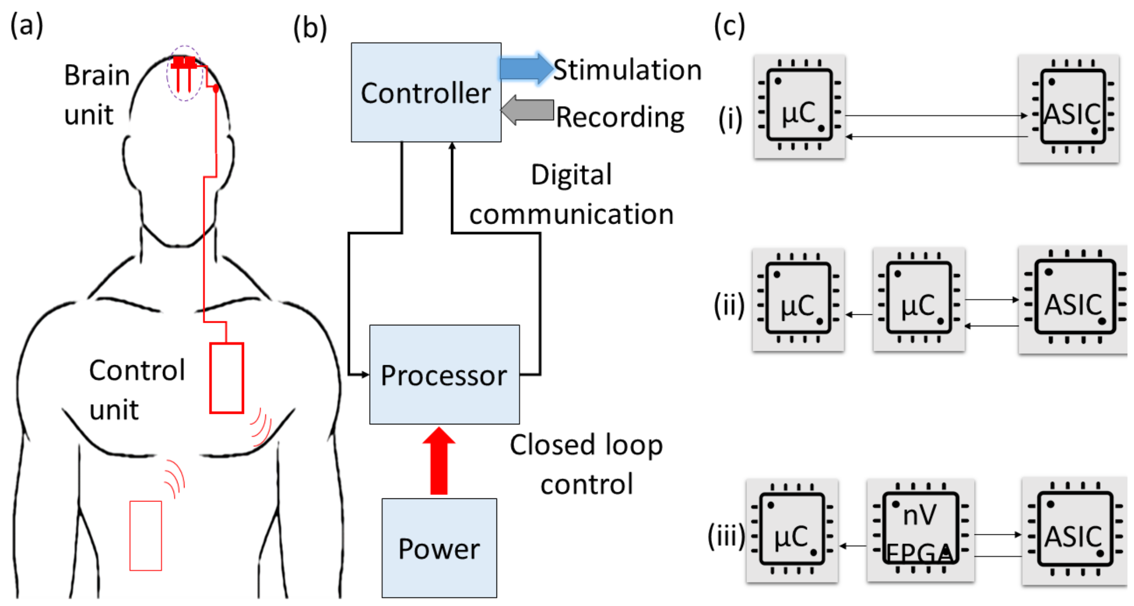


Figure 5-1: High Level Schematic of Closed-loop Brain Neuromodulation Control System. Shown are: (a) Scale diagram schematic prototype of brain neuromodulation system: Brain unit is for electrical recording and optogenetic stimulation; Controller unit is for data transmission. (b) Shows the schematics of closed-loop algorithm processing; (c) Compares different communication architectures between ASIC brain implant and control unit.

In this chapter, we explore the two candidate architectures shown in Figure 5-1(c, i) and Figure 5-1(c, iii). Our objective of this effort is to explore which of these is optimum digital architecture in realistic processing conditions. Hence an energy-efferent digital processor interfacing with an ASIC brain implant can be proposed for closed-loop brain neuromodulation. We therefore implement an exemplar PID control algorithm proposed in chapter 3 and chapter 4 on this digital processor for intervening with epileptic neuron networks to suppress seizures in neuroscience rodent experiments. The algorithm was optimized for each architecture and the total power consumption compared over respective wake-up and sleep processing cycles.

We used one of the most highly efficient microcontrollers currently available for this task which uses the 28nm technology node. In contrast, the only non-volatile FPGA (nv FPGA) uses the 90nm Programmable digital platforms (DSPs, MCUs, CPU, etc.) are highly flexible and have been typically used for developing neuroprosthesis systems. Examples of microcontroller digital implementations for closed-loop neuroprosthesis processors include [67] and [68]. Such systems could be implemented as shown in Figure 5-1(c, ii), assuming it to be desirable to have a separate microcontroller to ensure timing accuracy. However, general purpose systems lack the architectural efficiency of dedicated hardware.

5.2 System Architecture

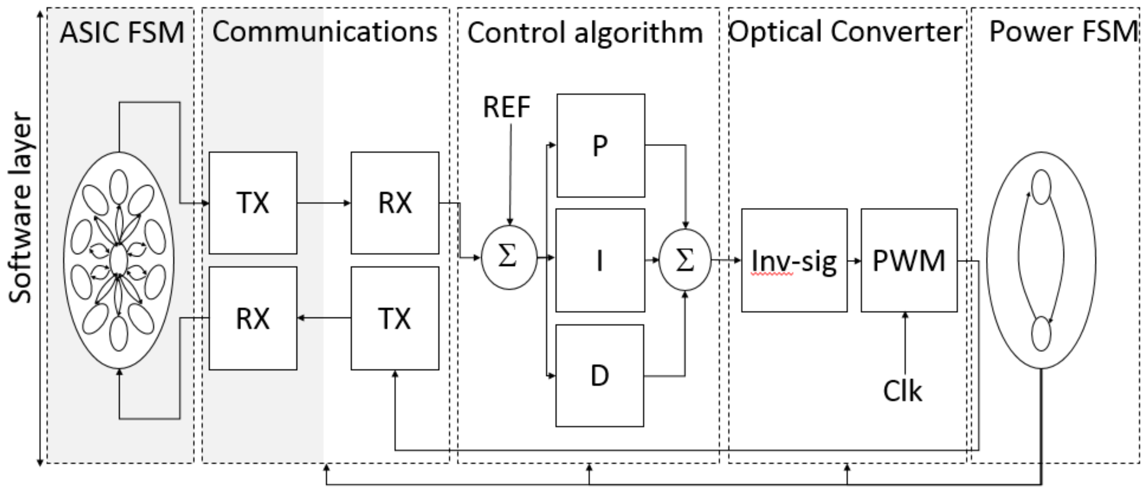


Figure 5-2: High block diagram of proposed closed-loop system design of software layer with implantable ASIC optrode [69] , bi-directional control system of the exemplar PID control algorithm.

This subchapter mainly describes the system architecture of the bio-directional ASIC optrode and the digital processor from the hardware and software level. Figure 5-2 shows a high-level block diagram of proposed closed-loop optogenetic stimulation system integration of the software layer. In the proposed software layer, it mainly contains three parts:

1. ASIC finite state machine information (ASIC FSM, communications)
2. Controller systems (control algorithms, optical converter)
3. Power unit (Power FSM).

Corresponding to the software layer, Figure 5-3 shows the hardware layer containing two parts: Brain implant (ASIC neural interface) and, Control unit (nvFPGA, microcontroller, power battery). The ASIC-based brain implant, which provides amplification, filtering and digitization of LFP signals as well as current sources for driving LEDs for optogenetics, has been described in the previous publication [18], [63] .

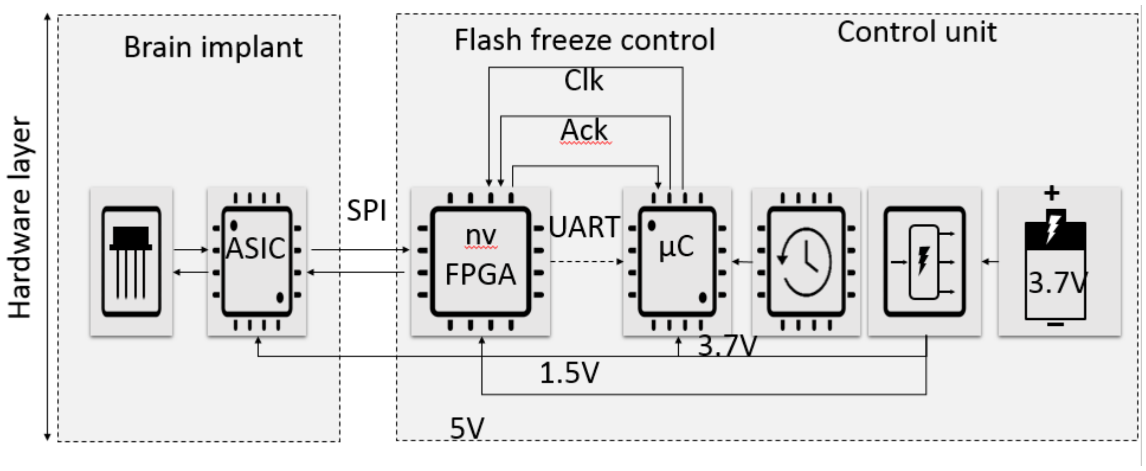


Figure 5-3: High block diagram of the proposed closed-loop system design of hardware layer and software layer with an implantable ASIC optrode, a bi-directional control system of the exemplar control algorithm.

For this specific application, a non-volatile FPGA (IGLOO nano FPGA: AGLN250V2-VQG100I) chip stands out with its flash freeze technology for significantly reducing standby current consumption. This non-volatile has been designed with a peripheral voltage circuit onto a 25mm*25mm flexible printed circuit board (PCB) shown in Figure 5-4 and Figure 5-5. A co-processor microcontroller is also employed to coordinate with this non-volatile FPGA to activate flash freeze mode to save on power consumption. Figure 5-4 shows the front end and back end design of the proposed PCB board which contains six-layer wire layout. Figure 5-5 is the photograph of the designed

25mm*25mm PCB board.

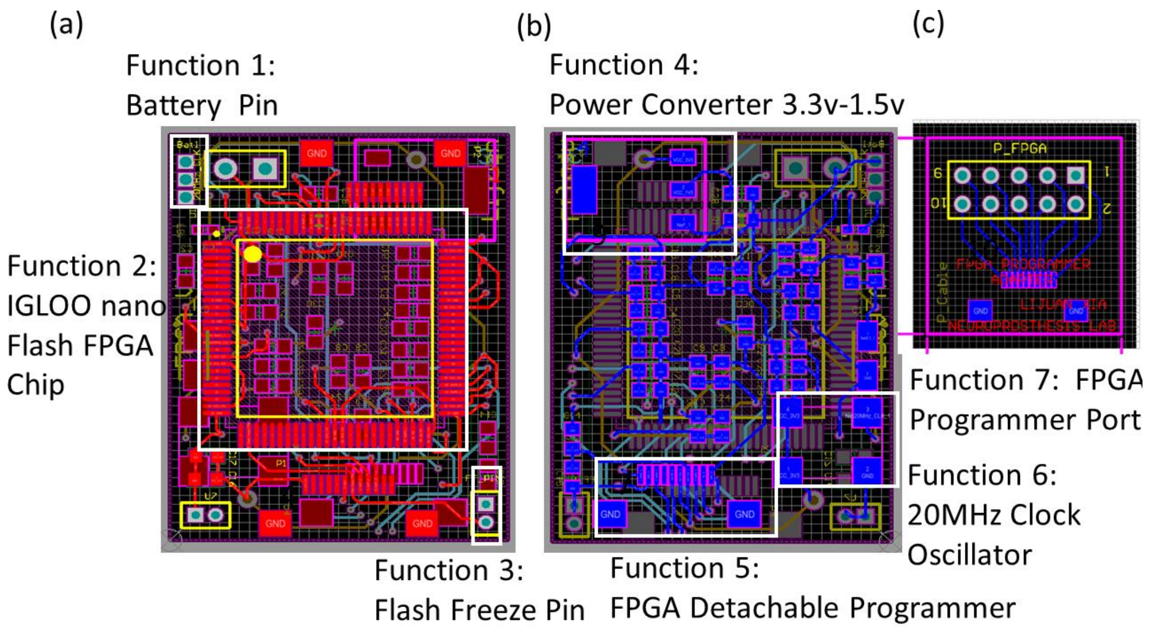


Figure 5-4: A detailed description of the proposed six-layer non-volatile FPGA PCB board in (a) front of the non-volatile FPGA board, (b) back view of the non-volatile FPGA board, (c) the FPGA programmer port for reprogramming. This PCB board is designed by Altium Designer Software.

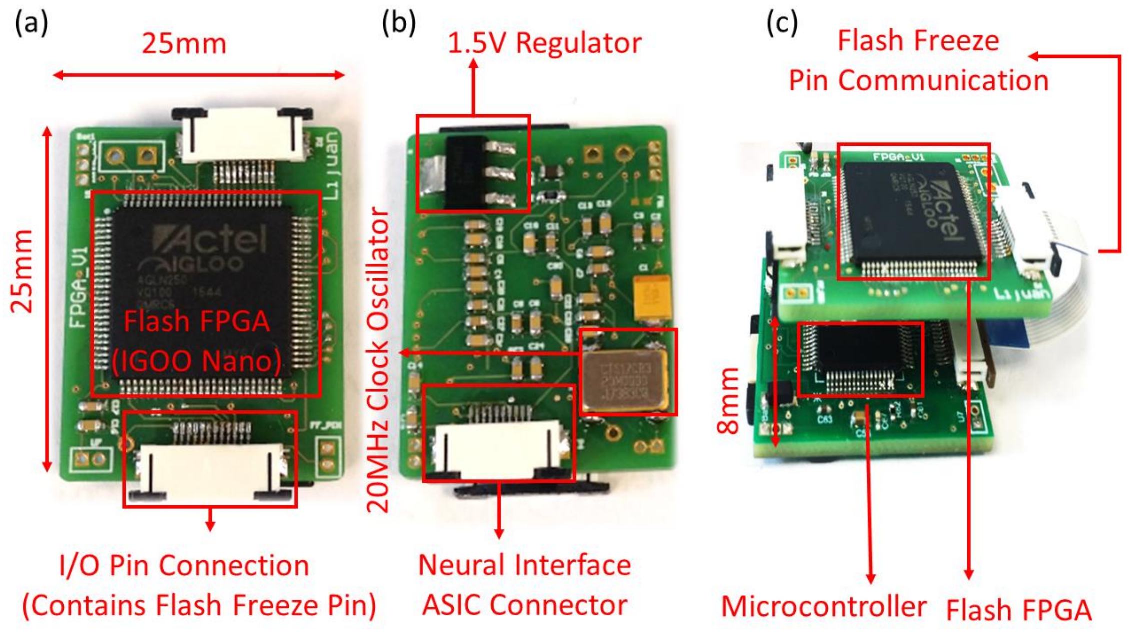


Figure 5-5: Photograph of the proposed PCB board in (a) front of the non-volatile FPGA board, (b) back view of the non-volatile FPGA board, (c) is the assembled version between the non-volatile FPGA and MCU board.

5.3 Processing Flow

This subchapter will introduce the processing flow in the FPGA based digital processor. First, we will present the PID algorithm and optical converter algorithm as an exemplar algorithm implementation. Following is the digital serial protocol implementation between digital processor of closed-loop algorithm and bi-directional neural interface. In other words, the FPGA based digital processor will receive some recordings from the implantable neural interface ASIC probe and performs the exemplar closed-loop control algorithms to convert the signal output into a stimulus pattern for further optogenetic means on the ASIC probe. The last section of this subchapter is to brief how to program a co-processor MCU to send a pulse width modulation signal to the FPGA directly to enable entering and exiting an ultra-low power Flash Freeze mode (8.032uA) to save energy consumption.

One of the most common used control algorithms in the engineering field is the PID (Proportional, Integral and Differential) algorithm, or PI, PD variants. It basically compares the signal with reference and determine the deviation (error) with respect to that reference. It is also applicable to closed-loop control of biological activity such as suppression of epilepsy seizures. A target of zero activity within a frequency range can be given. Then if the activity deviates too much, feedback can be provided to suppress activity. An exemplar of prior literature in this field has been proven in chapter 3 of this thesis. Furthermore, to achieve intervention we propose that optogenetics allows for closed-loop electrical recording and optical stimulation without interference [69]. As such, we also include an optical conversion algorithm based on the properties of channelrhodopsin-2 – the primary photosensitization agent used in optogenetics [70], [71].

5.3.1 PID Control Algorithm

In our application, the closed-loop algorithm is designed to intervene targeted neuron network through delivering continuous closed-loop optogenetic stimulation to suppress epilepsy seizures. The intervention philosophy of employing the linear PID algorithm to stabilize the neuron network has been proven in chapter 3.

Therefore, in the microsystem, we have defined the PID control algorithm as the summation of proportional operator, integral operator and derivative operator to filter the incoming real time LFP signal recorded by an exemplar ASIC optrode. The implementation of PID controller has been detailed in chapter 4.

LFP is sampled by means of an ASIC optrode at a sampling rate of 100Hz (that is, 100 8-bit samples per second are collected). As PID algorithms can be redesigned into the scheme of linear, time-invariant (LTI) filters, we need to design our LTI filter kernel in the time domain of impulse response for performing proportion, integration, differentiation operation.

5.3.2 Optical Converter

An optical optogenetic stimulus converter has been created for converting output of PID module to optogenetic Pulse Width Modulation (PWM) stimulus on the probe for modulating optogenetic infected neurons. Figure 5-6 describes two stages in the optical converter process, stage 1 is to adapt output of PID controller to total optical stimulus following on the non-linear inverse sigmoid transfer function shown in Figure 5-6, stage 2 employs a reciprocal counter for modulating duty ratio of fixed period pulses (10ms) for optogenetic neural stimulation [72].

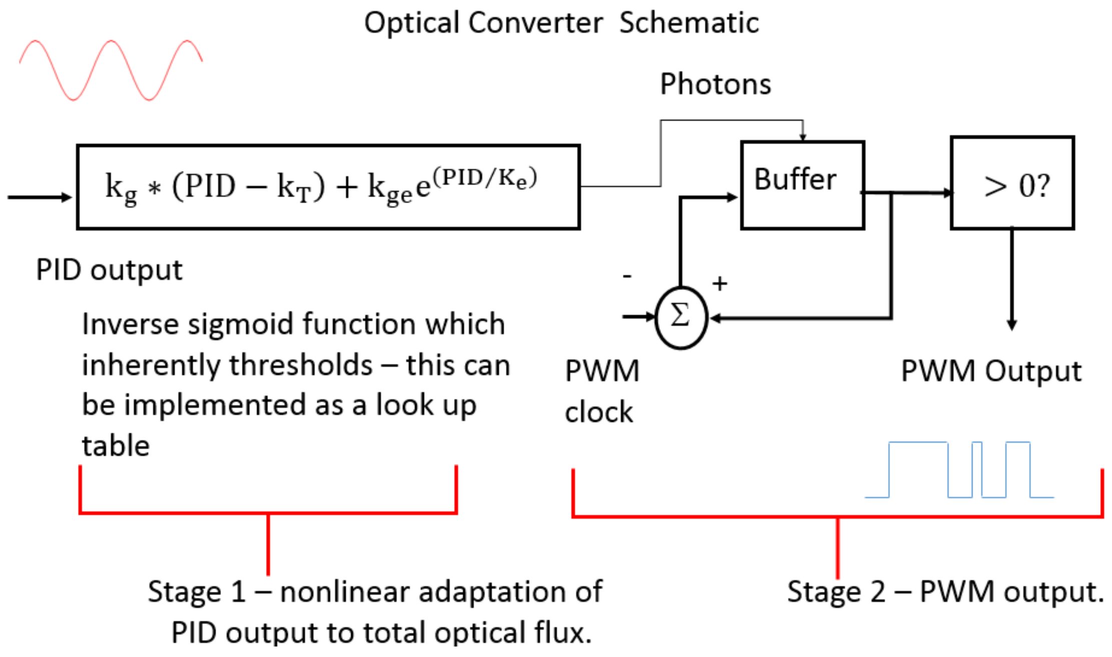

Optical Converter Schematic
Photons
$k_g * (PID - k_T) + k_{ge}e^{(PID/K_e)}$
Buffer
$> 0?$
PID output
-
+
Σ
Inverse sigmoid function which inherently thresholds – this can be implemented as a look up table
PWM clock
PWM Output
Stage 1 – nonlinear adaptation of PID output to total optical flux.
Stage 2 – PWM output.

Figure 5-6: Intensity dependent optical stimulus mechanism: (a) schematic for converting output of PID controller to width modulated pulse for delivering optogenetic stimulus (Kg =0.4, Kr = 128, Kge=0.1, Ke=34).

Figure 5-7 shows the relationship between the input and output of the optical converter design of inverse sigmoid function. The input of the optical converter is the output of PID control algorithm which can be defined as photon flux in terms of mW/mm2. The output of optogenetic stimulation from neural interface ASIC probe is defined as light intensity and PWM stimulation time. Figure 5-7 (a) represents the light intensity from the output of the inverse sigmoid function corresponding to light flux in terms of mW/mm2 for 10ms which limits the maximum PWM time (in ms) for an optogenetic stimulation on intervening epileptic seizure onset neurons. The neural response above is calibrated as the average plateau response resulting from continuous illumination. However, the main interested frequency range is in pulsed illumination with a defined PWM between 0.1 - 10ms (assuming 100Hz sampling - or at least 100Hz intervention). If the required light intensity is too high, the PWM time will exceed the maximum time allowable within a frame. Thus, this needs to saturate to the maximum time. Figure 5-7 (b) shows the PWM stimulation time corresponding to normalize neural response with a defined PWM time between 0.1 - 10ms.

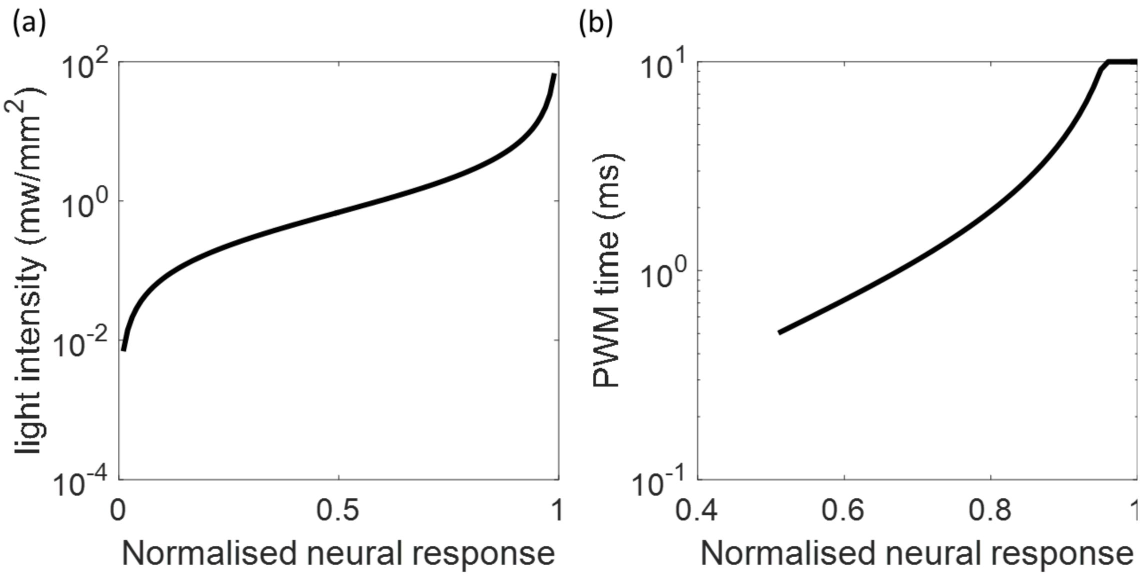


Figure 5-7: Relationship between the input and output of optical converter of inverse sigmoid function. (a) is the light flux response with the neural response, (b) is the light PWM stimulation time with the neural response.

5.3.3 Digital Neural Interface

Reza et al reported an application-specific integrated circuit (ASIC) brain implant of intelligent electrode recording and optical neural stimulation including a fully digital interface with a serial peripheral interface (SPI) to allow for use with embedded controllers [69]. The embedded SPI interface of their brain implant relates to a Finite State Machine (FSM) which implements a command interpreter which is capable of sending out LFP data whilst receiving instructions to control LED emission. Therefore, we incorporate a SPI master and corresponding state machine in our digital processor to interface with the operands in the ASIC. The digital ASIC command set with corresponding operands have been listed in Table5-1. Figure 5-8 describes the timing diagram of collaborating a microcontroller and a non-volatile FPGA to enter and exit flash freeze design of a non-volatile FPGA by driving flash freeze pin.

Table 5-1: Comparison results for different communication protocols.

ASIC Command Set with Corresponding Operands

Command	Purpose
0001xxxx	LED off
0010xxxx	LED On
0101xxxx	SET LED
1000xxxx	READ LFP

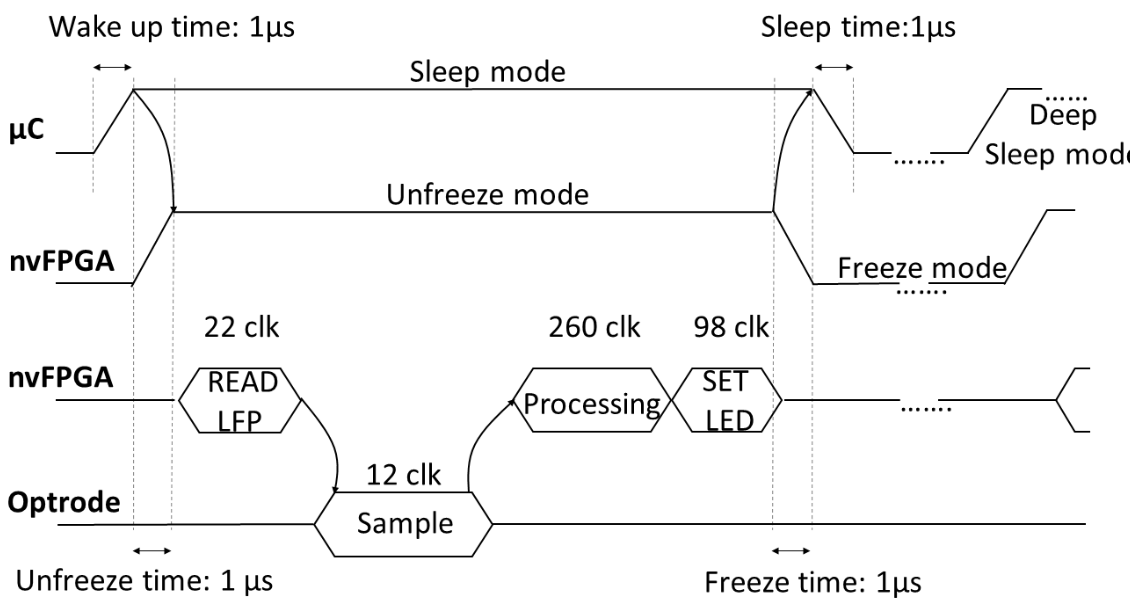
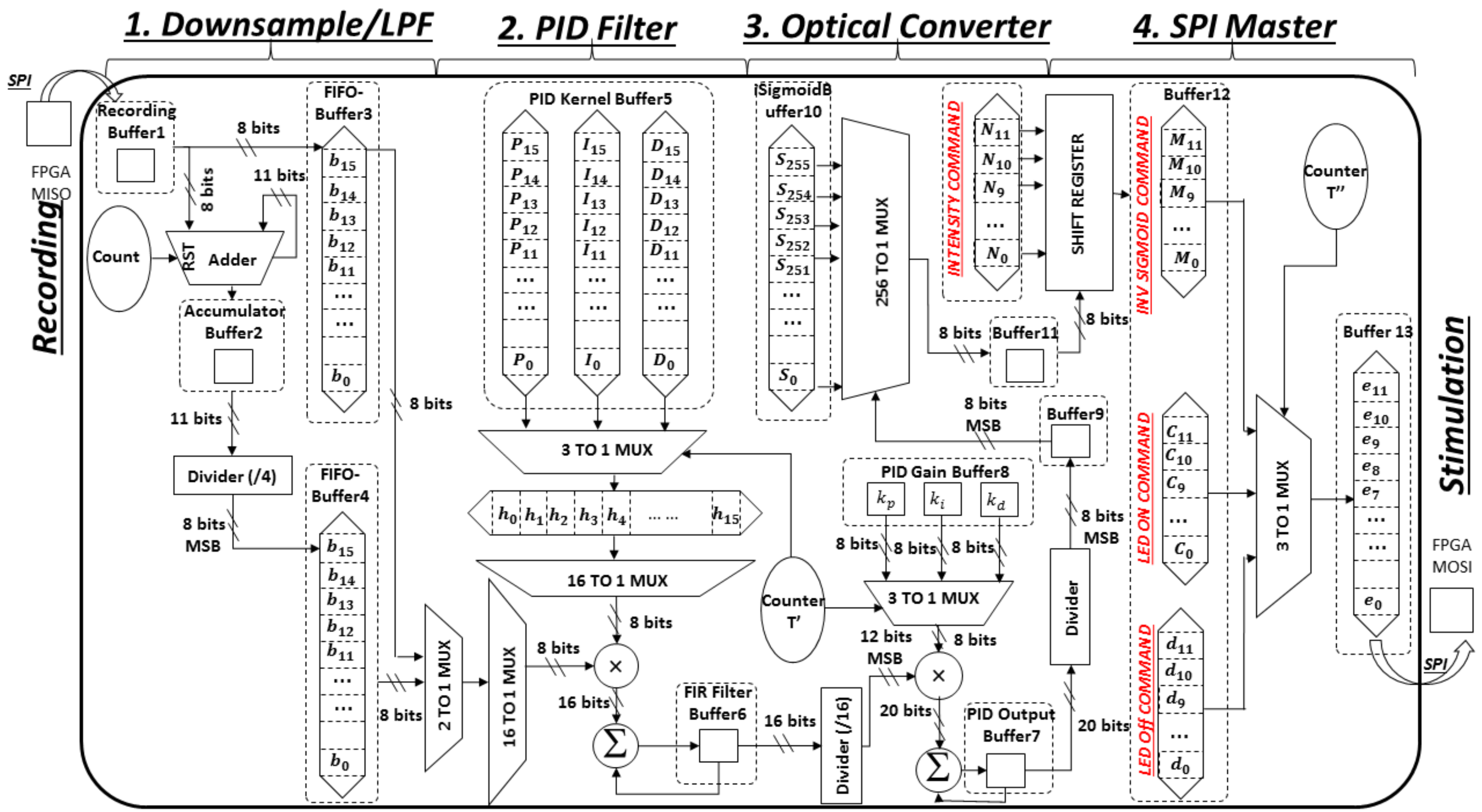


Figure 5-8: Timing diagram of collaborating the microcontroller and the non-volatile FPGA to enter and exit flash freeze design of the non-volatile FPGA by driving flash freeze pin.

Figure 5-9: the FPGA implementation of closed-loop control scheme: 1. Down-sampling, 2. PID filter, 3. Optical converter, 4.SPI master.



5.3.4 nvFPGA Version Implementation

The implemented processing architecture on the nvFPGA is shown in Figure 5-9, the basic state transition diagram for the chip-level FSM is shown in Table 5-1. The recorded data is stored in an 8-bit wide on-chip SRAM. The following closed-loop schematic consists of three main modules: recoding interface with the head-stage board of the probe, a closed loop algorithm, and stimulation interface interfacing with the head-stage board of the probe.

Recording Interface

In recording interface, a SPI slave has been designed for interfacing with the data transceiver and command interpreter to communicate with the master controller of the ASIC optrode. A SPI recording circuitry has been designed for interfacing with the probe to receive the command of "READ LFP ".

- *Closed-loop Algorithm Implementation*

In Figure 5-9, the digital FPGA architecture of an exemplar closed loop algorithm has been shown to have consisted of the following components:

1. A counter (T) and an 8-bit adder calculate for down sampling the input LFP recordings from Recording Buffer 1.

2. Two 8-bit 16*1 column buffer banks of 8-bits row registers FIFO Buffer3 and FIFO Buffer4 are used to save incoming recording of two separate frequencies.

3. Three 16*1 column registers PID Kernel Buffer5 of 8-bits are applied to hold proportional Dirac Delta kernel, integral unit step kernel and derivative Gaussian kernels (displayed in Figure 5-6) as FIR filter kernels to do further multiplication.

4. A 2 to 1 Mux is used for two column buffer banks FIFO Buffer 3, FIFO Buffer 4 to select either of the corresponding column register banks for the next 16 to 1 mux.

5. A16 to 1 Mux is used for one column buffer bank FIFO Buffer 3, FIFO Buffer 4 to select a corresponding column register for multiplication.

6. A 3 to 1 Mux is applied to select the corresponding PID kernel PID Kernel Buffer5 for further multiplication.

7. A16 to 1 Mux is used for one row buffer bank FIFO Buffer 3, FIFO Buffer 4 to select corresponding column register for multiplication.

8. An accumulator adds all results of multiplication which will be sent to recording buffer FIR Kernel Buffer 7 for generating further closed-loop neural stimulation.

9. Three registers PID gain Buffer 8 hold PID gain for a further 3 to 1 multiplier to do further multiplication.

10. Buffer PID output Buffer9 is employed to save the total PID output.

11. A 255*1 buffer bank Sigmoid Buffer 12 of an 8-bit register holds a sigmoid function look up table for modulating further pulse with feedback to optrode for closed-loop optogenetic stimulation.

12. A 256*1 multiplier selects sigmoid function output based on the incoming PID output Buffer10 to the register bank Intensity Command.

- *Stimulation Interface*

In stimulation interface, a SPI master has been designed for interfacing with FSM the master controller of the implantable probe for data transceiver and command interpreter. These commands will be sent off by the non-volatile FPGA based control unit after closed-loop algorithm processing. The digital ASIC command set with corresponding operands have been in Table 5-1.

5.3.5 Microcontroller implementation

The implementation of microcontroller (Kinetis K22 MCU: MK22FN512VLH12 MCU) of entire closed-loop algorithm has been simplified as follows. The recorded data is stored in an 8-bit wide buffer pointer. Then the input pointer will be passed to a PID filter which is organized into the ring buffer filter architecture of proportional filter, integral filter and derivative filter. The output of the PID controller is directly sent into optical stimulation commands to update LED status. The simplified description of microcontroller implementation is provided here as we will supply an energy consumption comparison of a microcontroller and a non-volatile FPGA in the results section to compare their power consumption performance.

5.3.6 System Integration FF Design

In Figure 5-10, a schematic circuitry description has shown for detailing the hardware of explaining flash freeze architecture inside the non-volatile FPGA. A pseudo code of driving flash and freeze mode has been listed.

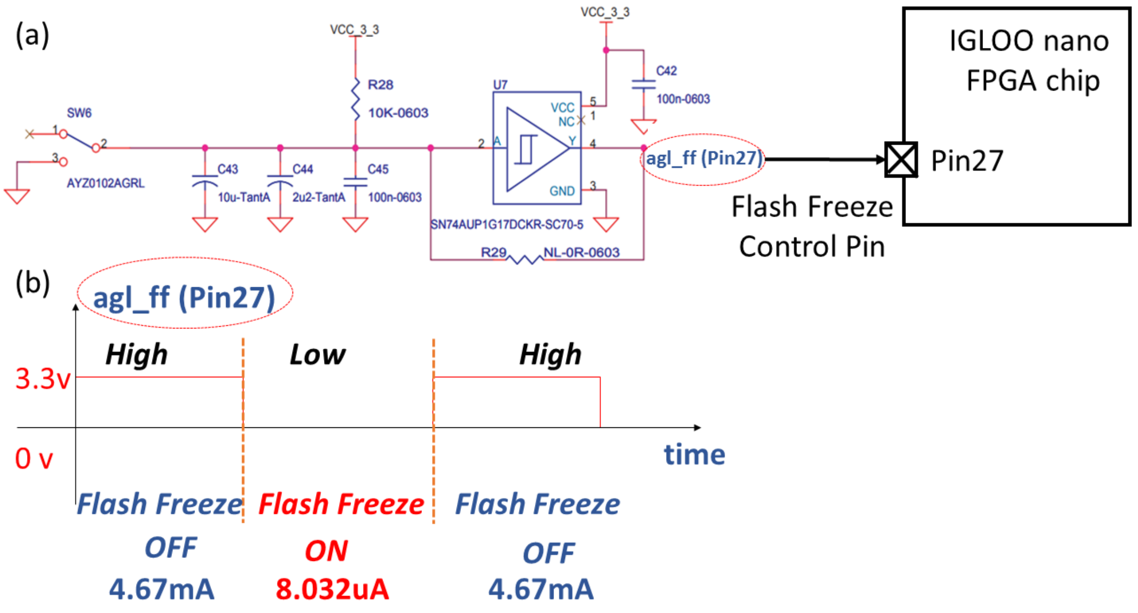
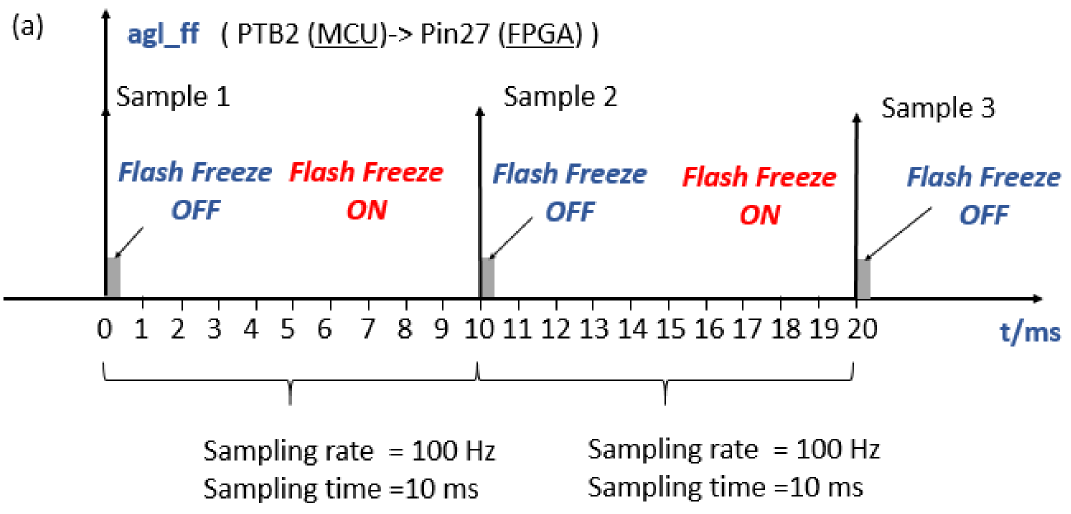


Figure 5-10: A simplified schematic diagram of flash freeze mechanism of the non-volatile FPGA. (a) The peripheral circuit design for getting the non-volatile FPGA to enter and exit the flash freeze mode by communicating with the agl_ff pin27 of the IGLOO nano FPGA chip. (b) The timing diagram of Flash Freeze agl_ff pin27 with closed-loop control design.

Figure 5-10 shows the proposed digital architecture for activating Flash Freeze mode by driving a flash freeze pin from the microcontroller to the non-volatile FPGA. A detailed breakdown of timing diagram of integrating a MCU and a non-volatile FPGA has been illustrated in Figure 5-11. To be specific, the entire closed-loop algorithm consumes 0.38ms, where data is sampled 100Hz (per sample every 10 ms), the design of the non-volatile FPGA (working frequency: 20MHz) is set to freeze mode after the processing is completed after 0.38ms. When we set the non-volatile FPGA working at 20MHz, the whole algorithm will cost 0.38ms. Then the neural interface ASIC sampled the incoming LFP at 100 Hz, which means per sample takes 10 ms. A pseudo code of emulating the flash*freeze mechanism of IGLOO nano chip has been listed. This digital architecture is designed for reducing static power consumption by setting the non-volatile FPGA into flash-freeze mode.

pseudo code of IGLOO nano Flash Freeze emulator

```
SIGNAL  FF_FLAG          : std_logic := '0'; -- example
   IF(INCOMING > 0 )
       FF_FLAG <= '1';
            IF( FF_FLAG <= '1' ) THEN
                   Recording  Circuit  ;
                   Algorithm  Circuit   ;
                   Stimulation Circuit ;
            END IF;
       FF_FLAG <= '0';
   ELSE
       FF_FLAG <= '0';
   END IF;
   AGL_FF_PIN <= FF_FLAG;
```



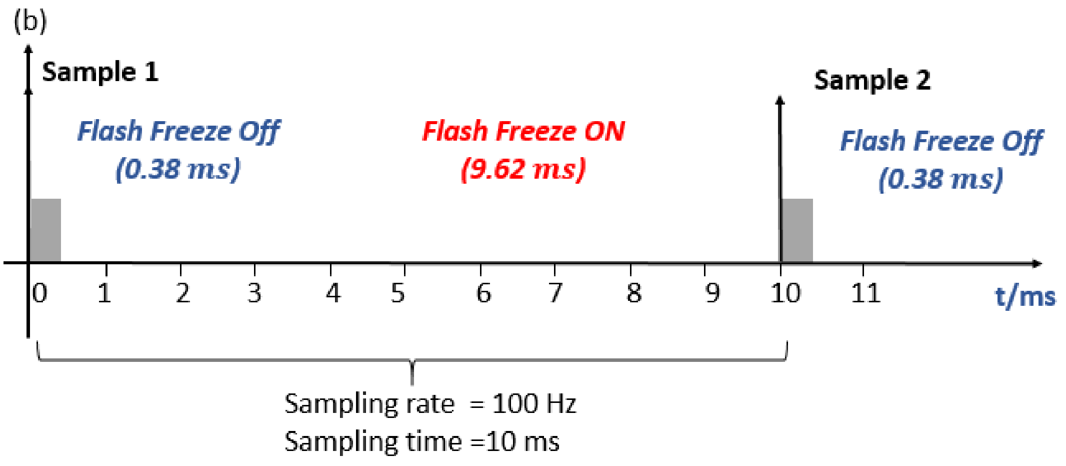


Figure 5-11: (a) two sampled Flash freeze mechanism of the non-volatile FPGA. (b) is the zoom up of one sampled Flash freeze mechanism of the non-volatile FPGA.

5.4 Results and Analysis

In order to demonstrate the functionality of the proposed closed-loop framework design, a dataset of 55 minutes (split in traces of 10 minutes) of prerecorded neocortical epileptic seizure local field potential (LFP) recordings were used to verify the performance of the exemplar closed-loop processing algorithm. The data was collected by neuroscientists working at the Institute of Neuroscience in Newcastle University from an epileptic adult rodent. The epileptic seizure recordings were emulated in a waveform signal generator (Keysight 33500B Series Waveform Generator) and then connected to the proposed digital processor for hardware verification.

5.4.1 Algorithm Verification

In order to verify the digital implementation of our FPGA implementation of closed-loop algorithms, we have used a microcontroller-based ADC and DAC module.

- ADC module of microcontroller FRDM K22F Cortex M4 is responsible for digitalizing the incoming local field potential (LFP).
- DAC module of microcontroller FRDM K22F Cortex M4 is to convert the digital output of the closed-loop algorithm into an analog output for a verification into oscilloscope.

For testing the frequency response of PID controller, we have utilized a sweep sin wave as the input.

a. **Input**: sin wave starting from 1Hz, end to 1KHz. Sweep time is 1 second
b. **Output**: the output of separate proportional, integral and derivative filter.

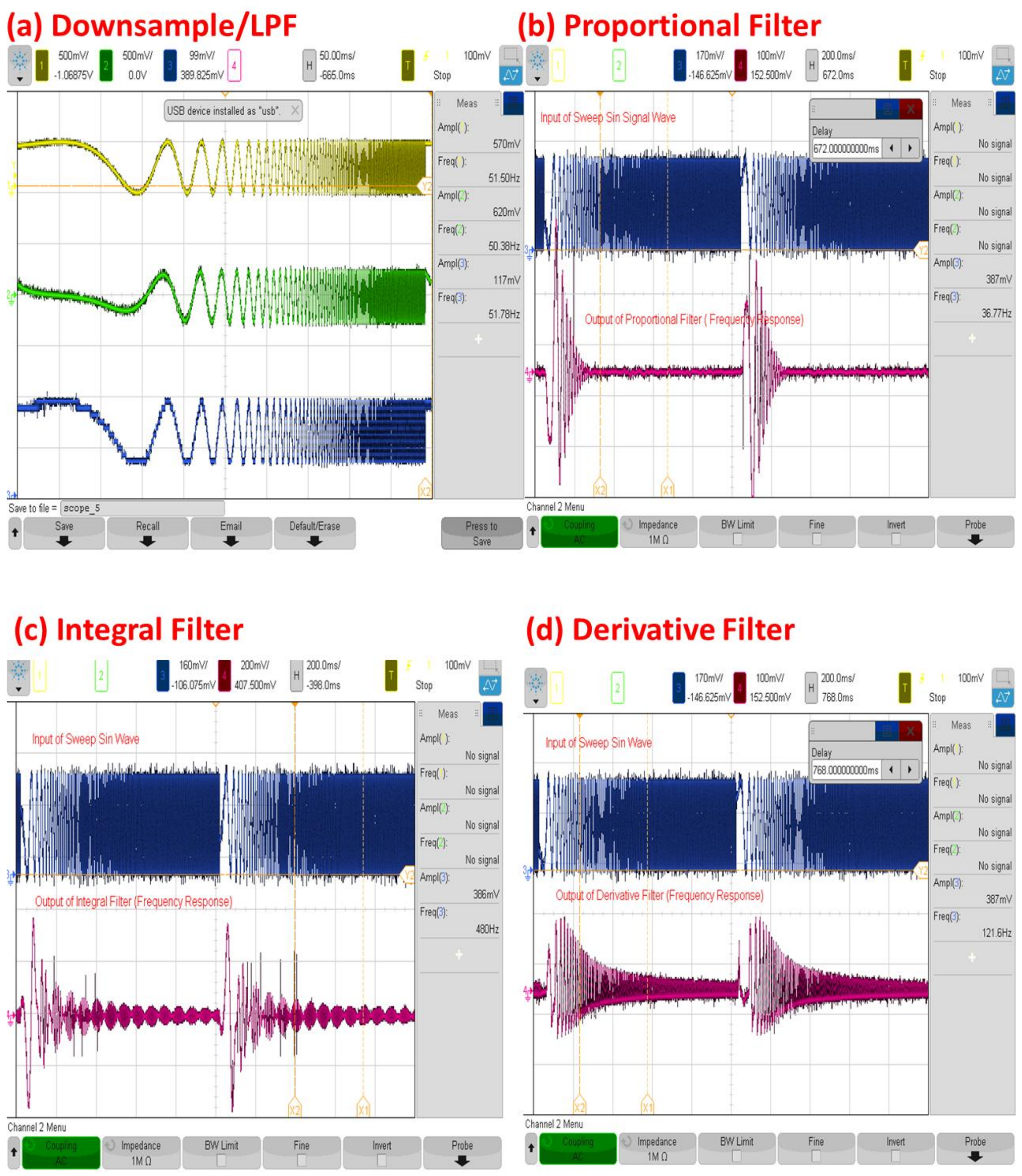

(a) Downsample/LPF
(b) Proportional Filter
Input of Sweep Sin Signal Wave
Output of Proportional Filter (Frequency Response)
(c) Integral Filter
Input of Sweep Sin Wave
Output of Integral Filter (Frequency Response)
(d) Derivative Filter
Input of Sweep Sin Wave
Output of Derivative Filter (Frequency Response)

Figure 5-12: (a) Comparison of input and output of down sample module with sweep sin wave signal as input signal for verification. (b) Comparison of input and output of proportional filter with sweep sin wave signal as input signal for verification. The measured frequency response of the proportional filter matches with Figure 4-23(b). (c) Comparison of input and output of integral filter with sweep sin wave signal as input signal for verification. The measured frequency response of the integral filter matches with Figure 4-25(b). (d) Comparison of input and output of derivative filter with sweep sin wave signal as input signal for verification. The measured frequency response of the derivative filter matches with Figure 4-27(b).

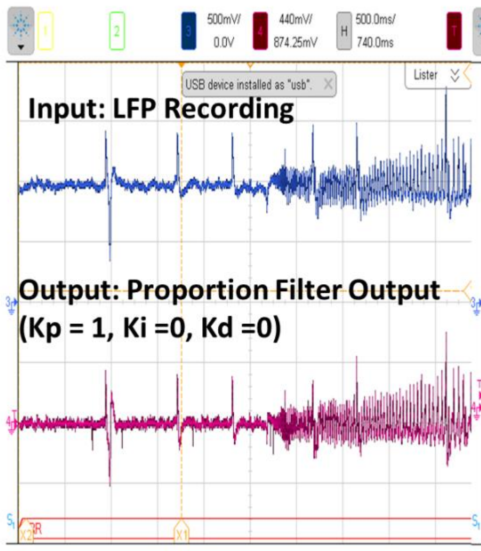
Figure 5-12 shows the hardware verification of the corresponding hardware design by capturing the output signal of each module verified by sending corresponding input data. Figure 5-12(a) describes the comparison of input and output of downsample module where blue is sweep sin wave and red is the output of proportion filter as a comparison. Figure 5-12(b) shows the comparison of input and output of proportional filter where blue is sweep sin wave and red is the output of proportion filter as a comparison. In addition, Figure 5-12(c) demonstrates the comparison of input and output of integral filter where blue is sweep sin wave and red is the output of integral filter as a comparison. Finally, Figure 5-12(d) illustrates the comparison of input and output of derivative filter where blue is sweep sin wave and red is the output of derivative filter as a comparison.

Furthermore, the pre-recorded seizure local field potential recordings are utilized to verify our digital processor for biomedical application.

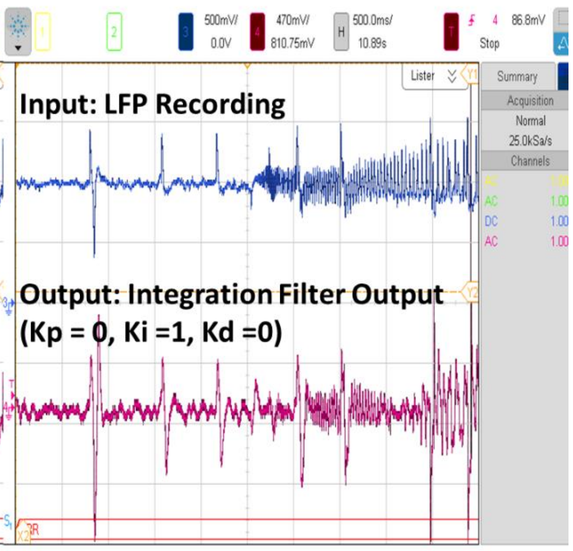
a. **Input**: epileptic local field potential recordings from rat cortex
b. **Output**: the output of separate proportional, integral and derivative filter.

Figure 5-13 shows the hardware verification of the corresponding hardware design by capturing the output signal of each module verified by sending corresponding input data. Figure 5-13(a) describes the comparison of input and output of downsample module where blue is epileptic LFP recordings and red is the output of proportion filter as a comparison. Figure 5-13(b) shows the comparison of input and output of proportional filter where blue is epileptic LFP recordings and red is the output of proportion filter as a comparison. In addition, Figure 5-13(c) demonstrates the comparison of input and output of integral filter where blue is epileptic LFP recordings and red is the output of integral filter as a comparison. Finally, Figure 5-13(d) illustrates the comparison of input and output of derivative filter where blue is epileptic LFP recordings and red is the output of derivative filter as a comparison.

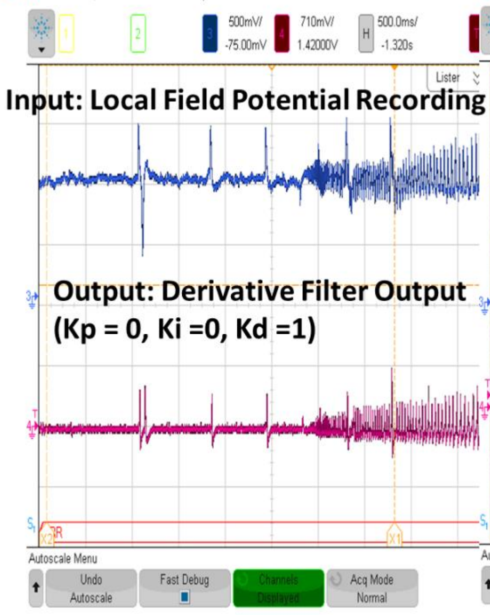
(a)



(b)



(c)



(d)

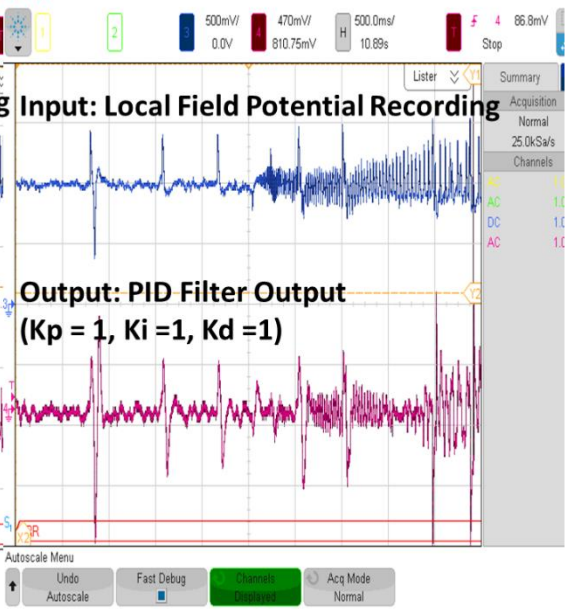


Figure 5-13: (a) Comparison of input and output of downsample module with epileptic seizure local field potential recordings of 10 seconds as input signal for verification. (b) Comparison of input and output of proportional filter with epileptic seizure local field potential recordings of 10 seconds as input signal for verification. (c) Comparison of input and output of integral filter with epileptic seizure local field potential recordings of 10 seconds as input signal for verification. (d) Comparison of input and output of derivative filter with epileptic seizure local field potential recordings of 10 seconds as input signal for verification.

In previous chapter 3, different PID gain parameterizations are suggested by closed-loop modelling work. An exemplar implementation is shown in Figure 5-14 with a proper PID K_p, K_i, K_d gain setup. We picked up PID gain parameters [$K_p \quad K_i \quad K_d$] as [100, 0,-2] from the stabilization area plotted in Figure 3-6(a). This PID set up is found to help control high amplitude epilepsy seizures successfully that is also shown in Figure 3-7(a). Under this circumstance, [$K_p \quad K_i \quad K_d$] as [100, 0, -2] have been chosen to tune the PID hardware implementation shown in Figure 5-14 as the first parameter setup for neuroscience rodent experiments to intervene the neuron network for controlling epilepsy seizures.

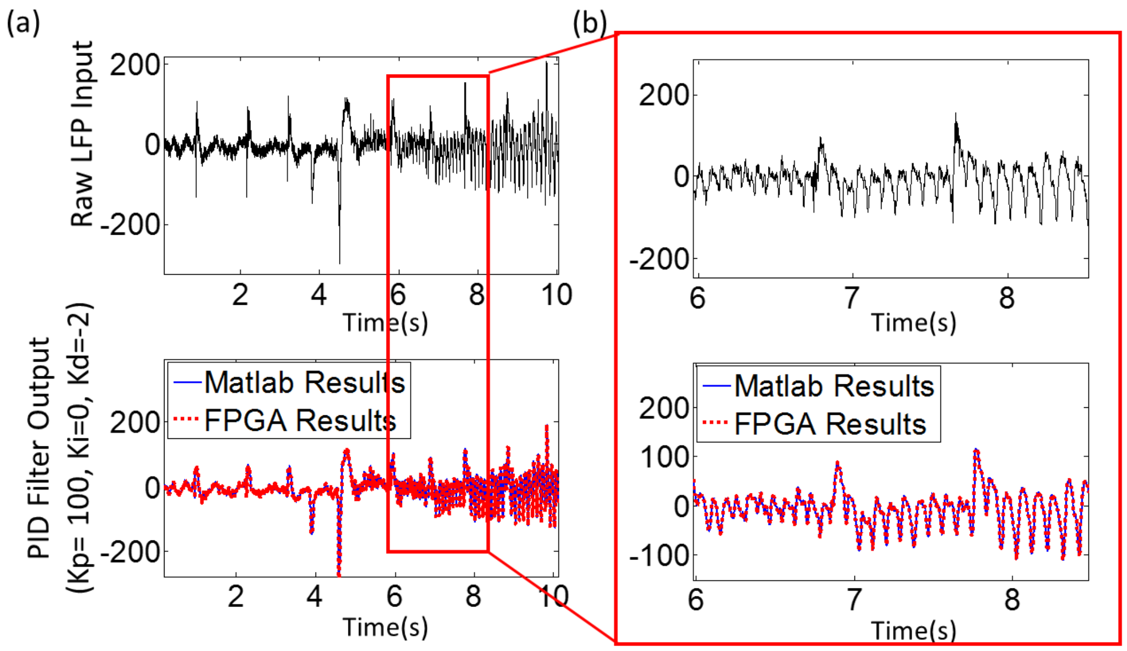


Figure 5-14: Measured non-volatile FPGA hardware results of input and output of PID controller ($\boldsymbol{K_p = 100, K_i = 0, K_d = -2}$). (a) Comparison of FPGA results and matlab results for 10 second local field potential recordings. (b) Zoomed up verification of comparison of FPGA results and matlab results from 7th second to 9th second local field recordings. The FPGA signals are converted into analogue signals by using an external DAC module based on FRDM K22F microcontroller.

Figure 5-14 has demonstrated the measured non-volatile FPGA hardware results of input and output of PID controller with comparison to the Matlab reference. The microcontroller implementation was identical, but we do not show here for brevity. Figure 5-14 (a) describes the comparison of input and output of downsample module where blue is epileptic seizure local field potential recordings of 10 seconds and red is the output of proportion filter as a comparison. Figure 5-14 (b) shows the comparison of input and output of proportional filter where blue is epileptic seizure local field potential recordings of 10 seconds and red is the output of proportion filter as a comparison.

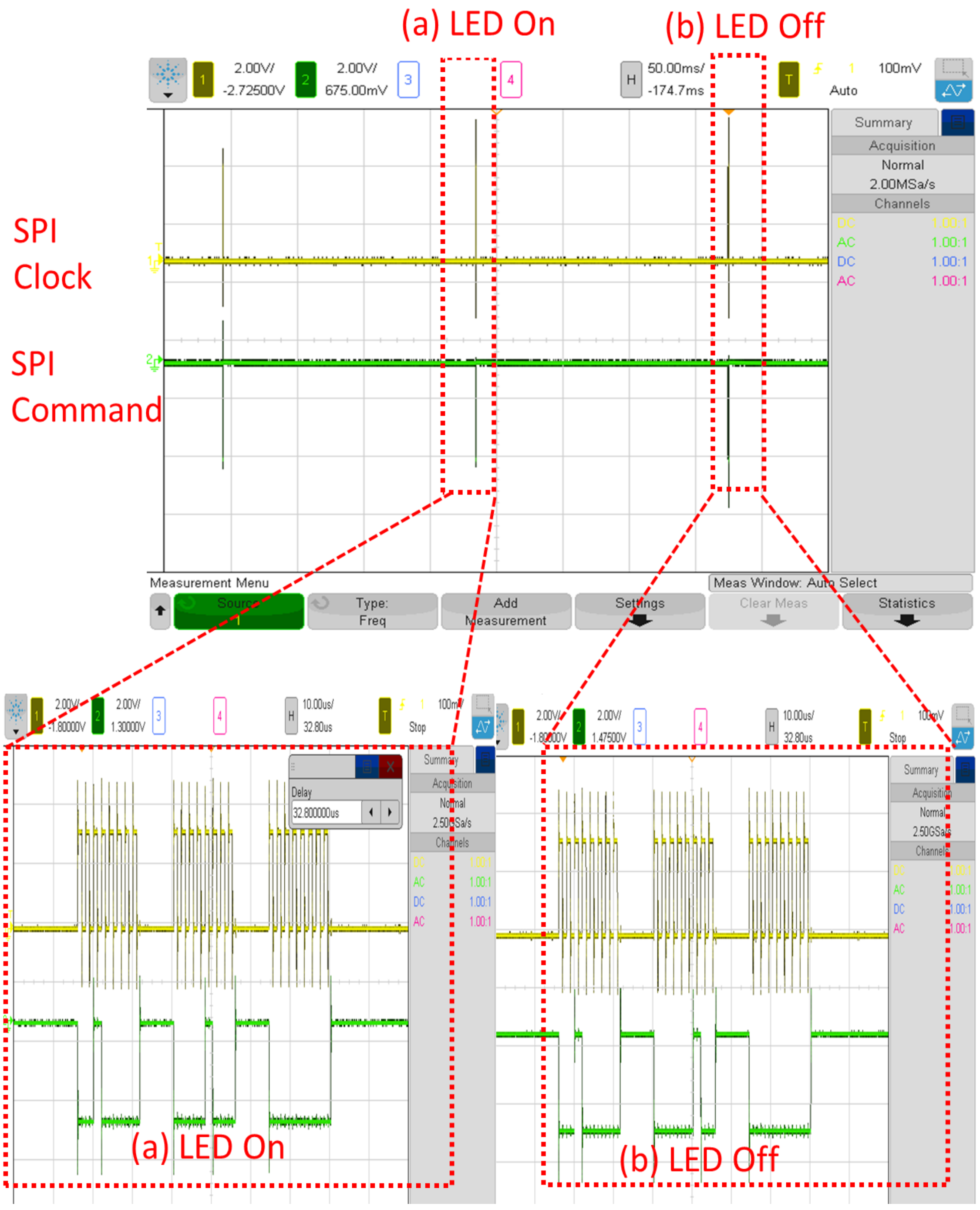


Figure 5-15: Oscilloscope verification of LED on and LED off command sent from the FPGA to the ASIC for turning on and turning off LED bonded in ASIC optrode. (a) The zoomed oscilloscope screenshot showing the data interface neural stimulation LED On command; (b) The zoomed oscilloscope screenshot showing the data interface neural stimulation LED Off command.

After the PID controller, the output of PID controller is sent into the inverse sigmoid optical converter for delivering the turn on and turn off the LED bonded on the optrode for delivering the optical stimulation. In order to verify the LED on and LED off command between the front-end FPGA and neural interface ASIC, a probe has been hooked up in the connection between the front-end FPGA and neural interface ASIC to display the signal in oscilloscope (Keysight MSOX4154A Mixed Signal Oscilloscope). Figure 5-15 demonstrates how the LED on and LED off be sent to drive the neural interface ASIC to the target LED. Figure 5-15 shows the zoomed up of the LED on and LED off command.

5.4.2 Flash Freeze Verification

A digital communication protocol has been designed for activating Flash Freeze mode by driving a flash freeze pin from the microcontroller to the non-volatile FPGA which is also demonstrated in Figure 5-16. A hardware oscilloscope screenshots verification of the timing-diagram of integrating the MCU and the non-volatile FPGA has been illustrated in Figure 5-16. As the entire closed-loop algorithm consumes 0.38ms, where data is sampled 100Hz (per sample every 10 ms), the design of the non-volatile FPGA (working frequency: 20MHz) is set to freeze mode after the processing is completed after 0.38ms. The flow mechanism of collaborating the front-end non-volatile FPGA with a microcontroller to enter and exit the Flash*Freeze is designed into three stages as follows:

1. Stage1: MCU->FPGA for unfreezing:

The MCU sends the non-volatile FPGA high voltage to enable exiting from flash*freeze mode (unfreeze mode).

2. Stage2: FPGA->MCU for freezing:

When the non-volatile FPGA finishes computing, it sends an ACK to the MCU, then the MCU send a low voltage to make the non-volatile FPGA enter flash freeze mode (freeze mode).

3. Stage3: MCU->FPGA for unfreezing:

Once the FPGA receives the next recordings, it will send an ACK to the microcontroller for sending high voltage to the FPGA to enter flash*freeze mode (unfreeze mode).

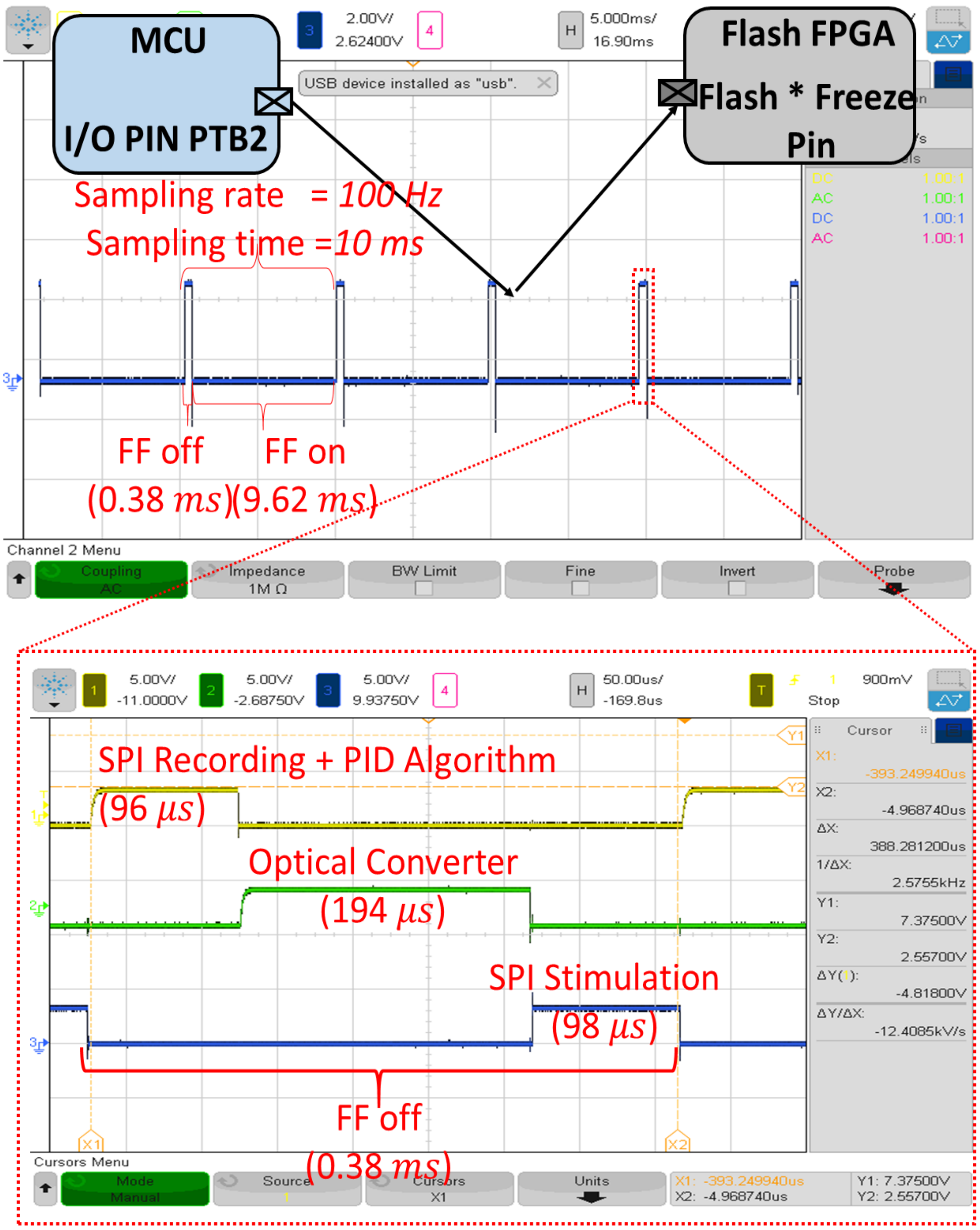


Figure 5-16: Oscilloscope capture of flash freeze active signal sent from microcontroller to the non-volatile FPGA flash freeze pin to help enter and exist flash freeze mode. The signal also represents the time cost for the flash freeze pin in which FF off lasts for 0.38ms and FF on lasts for 9.62ms in time domain. In FF off mode, the FPGA is in unfreeze mode where SPI recording and PID algorithm lasts for 96us, optical converter costs 194us and SPI stimulation takes 98us.

During the unfreeze mode, the computing time consisted of SPI recording, PID algorithm, optical converter and SPI stimulation. The breakdown of the computing time has been shown in Figure 5-16. To be specific, the flash freeze exploration can be split into the following steps:

- SPI recording with PID algorithm :96us
- Optical Converter:194us
- SPI stimulation: 98us

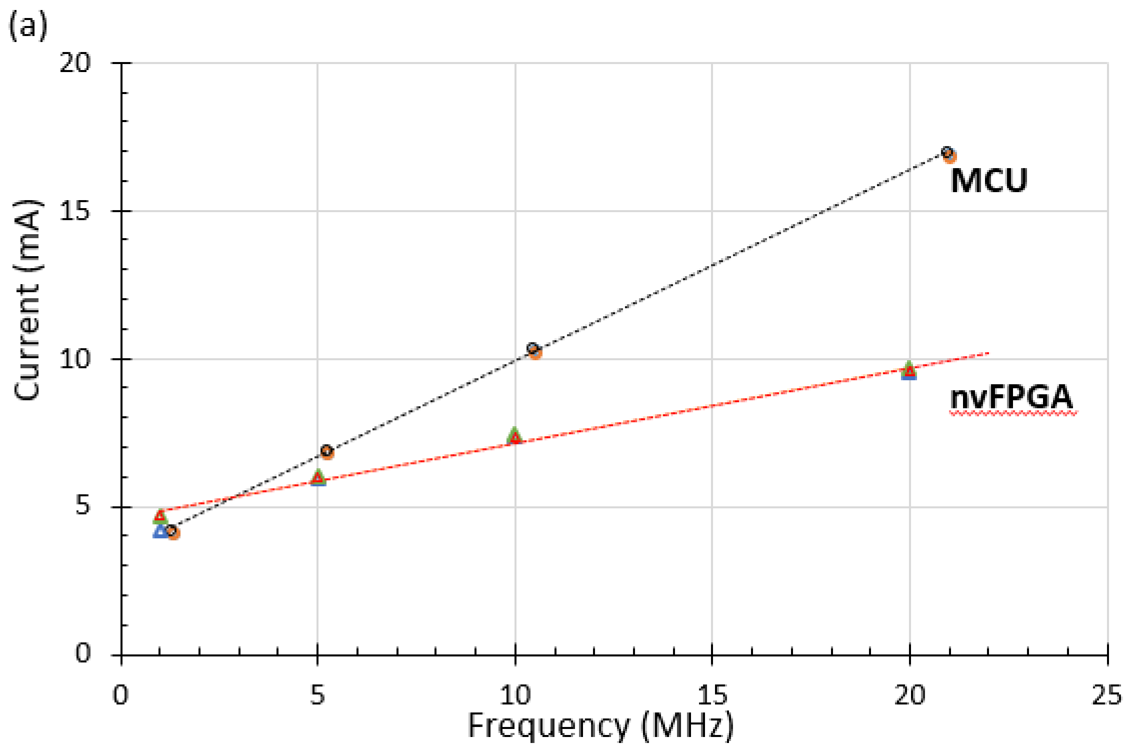
It means the whole processing unit can take 0.38ms, while the non-volatile FPGA is on flash freeze off mode.

5.4.3 Power Measurement Results

In this chapter, we mainly analyse how power consumption be distributed by exploring this nvFPGA architecture. We have listed the measured power consumption of the non-volatile FPGA and the microcontroller during one computing cycle in Table 5-4. It can be observed that the algorithm computing time for each sample takes 0.38ms. During this period, with a 20 MHz working frequency set up, the non-volatile FPGA consumes 4.78mA (flash freeze mode off) and while in sleeping mode with the non-volatile FPGA having flash freeze mode on, it costs 8.15uA.

Table 5-4: Measured power consumption of the non-volatile FPGA (IGLOO Nano) and the microcontroller (ARM Cortex M4) during one computing cycle

	Non-volatile FPGA		**MCU**
Work Mode	FF on	FF off	Interrupt Mode
Voltage Rail	1.2v	1.2v	3.3v
Current	4.78mA	8.15uA	0.25mA
Time	0.38ms	9.62ms	10ms



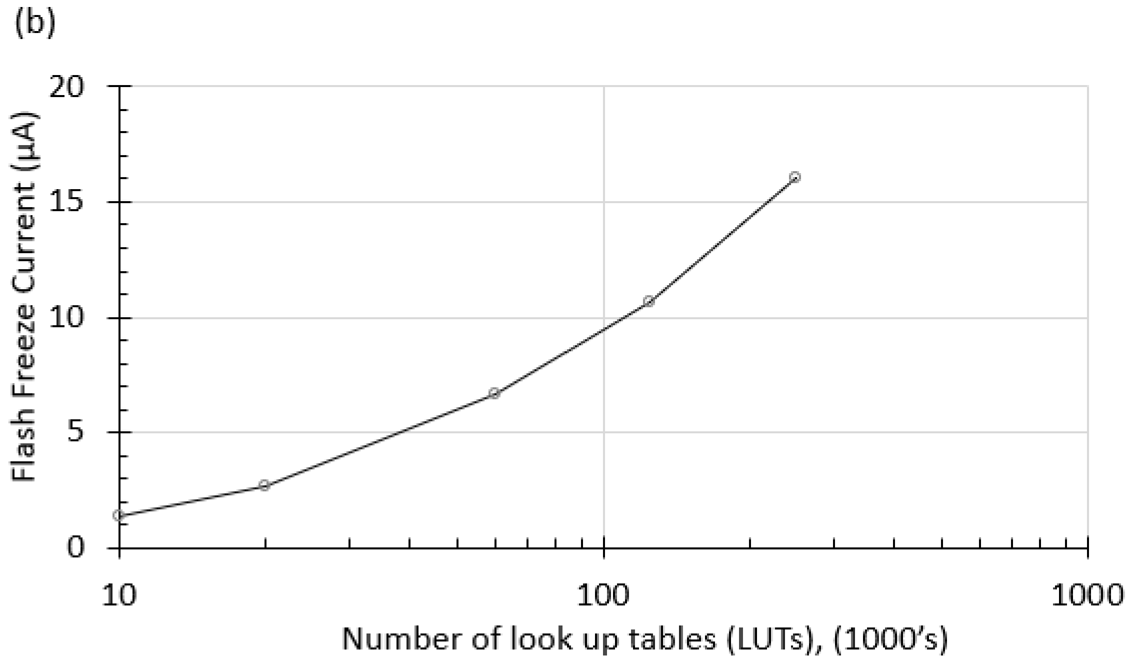


Figure 5-17: (a) The measured current consumption of the front-end non-volatile FPGA implementation and MCU implementation. (b) Scalability analysis of flash freeze current leakage with respect to look up table numbers of non-volatile FPGA.

In comparison, the microcontroller is programmed into sleep mode with a timer set to generate a pulse signal to the non-volatile FPGA for entering and exiting flash freeze mode. It costs 0.25mA for microcontroller to be a co-processor during the one computing cycle. On the flash freeze snippet mode (with aforementioned recording settings), the complete system measured in at 5.12mW, including all the I/O power (i.e. driving the PCB traces).

In order to conduct a proper power consumption comparison, we have implemented the same closed-loop algorithm in microcontroller with a sleep and wake up mode setup. Figure 5-17 (a) shows the measured power consumption comparison between the non-volatile FPGA (flash freeze mode on and off) and microcontroller (sleep and wake up on and off). It shows the strength of dynamic energy consumption of flash compared to commercial digital microcontrollers. Figure 5-17 (b) describes the scalability analysis of flash freeze current leakage with respect to look up table numbers of non-volatile FPGA. For non-volatile FPGA (AGLN010) which contains the least logic cell numbers of 10,000, the leakage power is 2 μw while for non-volatile FPGA (AGLN250) which contains the most logic cell numbers of 250,000, the leakage power is 24μw.

In our scenario, AGLN250 has been picked as it contains the maximum LUTs and 25.6% utilization of LUTs is used for our implementation, and 74.4% of the remaining LUTs can be applied to explore other algorithms implementation to conduct additional exploratory stimulation methodologies for controlling epileptic seizure neuron network for neuroscientists to work with. If we choose an exact AGLN010 FPGA based on our LUT estimation, it reveals that the power consumption can be further reduced.

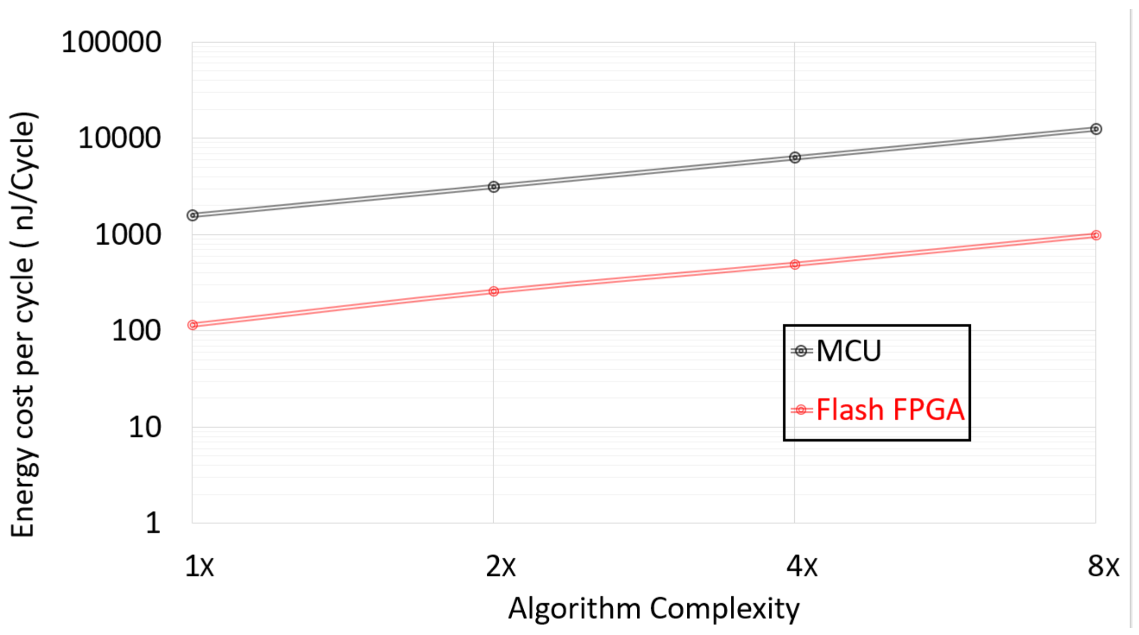


Figure 5-18: Comparison of energy cost per computing cycle of the non-volatile FPGA and the microprocessor. This figure also shows the energy cost per cycle with respect to algorithm complexity of the MCU and the non-volatile FPGA.

To provide a fair comparison, Figure 5-18 compares the measured energy cost per computing cycle of the non-volatile FPGA. When algorithm complexity increases, Figure 5-18 also highlight the non-volatile FPGA has a lower energy consumption strength over the commercial digital microcontrollers. The breakdown of the look up table has been investigated in the Table 5-5. Table 5-5 also shows the utilization of resources in the design of the closed-loop algorithm. It shows the whole design consumes 25.6% of the available resources on the non-volatile FPGA, which has been optimized for closed-loop algorithms.

Table 5-5: Measured Look up Table (LUT) Distribution of Closed-loop Algorithm Design implemented on the non-volatile FPGA IGLOO nano Chip

	LUT	**LUT Utilization**	**Utilization**
LUT Utilization of Entire Architecture			
SPI Slave for Recording	32	32/6144	0.5%
Closed-loop Algorithm	827	827/6144	13.5%
SPI Master for stimulation	712	712/6144	11.6%
Total	1571	1571/6144	25.6%
Break Down of the Closed-loop Architecture			
Downsample	128	128/6144	2.1%
PID Algorithm	444	444/6144	7.2%
Optical Converter	255	255/6144	2.3%

Figure 5-19 also uses the pie chart to demonstrate the breakdown of logic cells utilization of closed-loop digital implementation. In this design, it is possible to add more filters working in parallel with each other, with not adding massive computing latency by taking advantage of the FPGA parallel computing architecture.

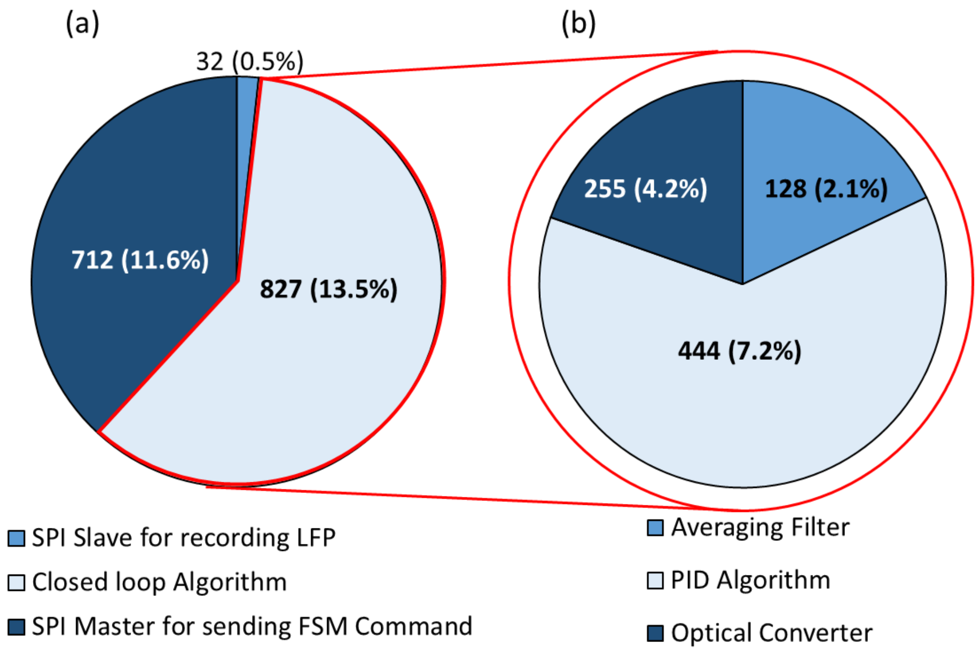


Figure 5-19: Breakdown of power consumption of look up table distribution of closed-loop architecture. Shown are: (a) Look Up Table Utilization of Overall Architecture. (b) Break Down of the Exemplar Closed-loop algorithm.

5.5 Conclusion

In this chapter, we have presented an exploratory energy-efficient digital processor architecture built with the commercial off-the-shelf non-volatile FPGA and microcontroller for sparse data processing of brain neuromodulation. Taking a commonly-used algorithm with reference target application, the front-end non-volatile FPGA is used to implement the exemplar algorithm implementation and a MCU co-processor is applied to coordinate to enable entering and exiting an ultra-low power Flash*Freeze mode of the front-end non-volatile FPGA. The main features of this effort are as follows.

i. The first key advancement is that we develop and implement a new power computing diagram based on the FPGA+MCU architecture. This time-driven approach significantly reduced power consumption which suggests that a digital combined control system of the non-volatile FPGA and micro controller outweighs a digital control system of microcontroller with microcontroller regarding computing time cost and energy consumption.

ii. The second key improvement of this work is that its potential flexibility to be employed in neuroscience research experiments. This work presents a digital implementation of an exemplar Proportional-Integral-Derivative (PID) control algorithm which can be applied theoretically to suppress epileptic seizure neuron networks by setting up proper gain parameters in neuroscience research experiments.

iii. Furthermore, a 55-minutes dataset of offline seizure LFP recordings from rat cortex has been applied to verify this digital processor with closed-loop algorithm implementation. It also shows the efficient-energy consumption of 116 nJ/computing cycle which means the wearable digital processor can runs for more than 14 days on a wearable 3.7V LiPo 180mAh Battery.

iv. This non-volatile FPGA digital architecture can be further translated to a System on Chip (SoC) design for integrating with an implantable neural interface (ASIC) chip to do electrical recording and optogenetic stimulation to form a closed-loop SoC.

This is the first cohort exploratory study to apply such an energy-efficient digital architecture to interface with brain implants for controlling neural networks with optogenetic stimulation to treat epilepsy. The small size and low power consumption can enable new neuroscience experiments in the study of neural control behaviour. Although this digital architecture was conducted in the field of brain implants, this digital architecture might also have great potential to impact clinical applications. This digital processor can also be further adapted to other embedded electronic devices for sparse signal processing to achieve lower energy consumption (IoT, cellphones, cardiac pacemaker, etc.).

Relative Contribution

Correspondent	Contribution
Miss Lijuan Xia Dr. Patrick Degenaar Mr. Dimitrios Firfilionis	1. A new power computing diagram based on the FPGA+MCU architecture has been proposed in this chapter. 2. The proposed Proportional-Integral-Derivative (PID) control algorithm which can be applied theoretically to suppress epileptic seizure neuron networks by setting up proper gain parameters in neuroscience research experiments. 3. This non-volatile FPGA digital architecture can be further translated to a System on Chip (SoC) design for integrating with an implantable neural interface (ASIC) chip to do electrical recording and optogenetic stimulation to form a closed-loop SoC.
Correspondent	**Future Work**
Miss Lijuan Dr. Patrick Degenaar	1. More algorithms apart from PID controller can be explored for the rest of look up tables on the non-volatile FPGA to integrate with the FSM implementation for benefitting neuroscience research experiments. 2. For the wireless power transfer, the required components are the battery, embedded system power management, and the power of the transmitter and receiver.

Chapter 6. Conclusion

This thesis concerns a closed-loop control system of brain which demonstrates that closed-loop stimulation methodology can alter ongoing epileptiform activity in vitro. A closed-loop computational modelling work has been proposed to show that with proper PI controller gain set up and PD controller gain set up, closed-loop PI controller and PD controller can help suppress high amplitude epilepsy seizure-like activities. Furthermore, different digital hardware platforms have been examined for an energy-efficient hardware implementation of PI and PD controller implementation for closed-loop optogenetics experiments in rodent brain slices. Last but not least, the proposed embedded PI and PD controller have been designed to connect with a bidirectional intelligent optoelectronic probe for closed-loop electronic recording and optogenetic stimulation.

This chapter will summary the main contribution of each chapter in this thesis and identify the future work. And the final part of this chapter will demonstrate concluding remarks.

6.1 Original Contributions

The major contribution of this work can be presented by the two following points:

Chapter 2 has given a basic overview of the human brain function and some neurological diseases. Epilepsy and epilepsy treatments are also examined in this chapter. Anti-epileptic drugs (AEDs) are mainstays in stopping epileptic seizures from happening. However, there are also other options for those patients whose seizures are not stopped by taking medication. An operation on the brain can help control seizures and improve their life quality. Firstly, surgical resective surgery can be used to remove the focal onset part of patients' brain that causes seizure. Secondly, the Vagus nerve stimulation can also be employed to disrupt the nerve pathways that seizure impulses take through your brain. Finally, the deep brain stimulation and closed-loop responsive neurostimulator systems can be adopted to implant a brain probe device for delivering stimulation to the target nervous system. Closed-loop neuroprosthesis systems are reviewed from the hardware and control algorithm perspectives.

Chapter 3 has depicted a closed-loop computational modelling work to deliver closed-loop stimulation for intervening the neuron mass model. In this study, we took the Jansen's neuron mass model as a test bed to develop a closed-loop control system for controlling high amplitude epileptic-like signal. PI controller and PD controller are used to deliver optogenetic stimulation for intervening the neuron mass network. A graphical stability method was used to determine the stability area of PI type controller and PD type controller in the control parameter space for the proposed neuron mass model, which shows a theoretical guideline for the parameter choosing of PI control and PD controller. The real time simulation results show that with appropriate PI and PD gain choosing from the derived stabilization area, the PI controller and PD controller can help to suppress high amplitude epileptic like seizure signals in the proposed neural mass model.

Chapter 4 has reviewed different hardware platforms to suggest a proper hardware platform satisfying the proposed specifications in terms of reprogrammability, power consumption and computation time cost for our biomedical application. After a comparison case study, the non-volatile FPGA was chosen for our biomedical application as it has reasonable power consumption and computing time cost with a great reprogrammability. Additionally, a low-power optimized digital implementation of PID control algorithm suggested from the computational modelling efforts in Chapter3 was described in this chapter.

Chapter 5 highlights how to integrate the PID controller implementation with an FSM command interpreter of an ASIC-based neural interface to drive the bi-directional neural interface optrode to receive electrical recording and deliver optogenetic stimulation based on the proposed closed-loop controller. By exploiting the flash freeze function of the non-volatile FPGA, a co-processor microcontroller is programmed to send a pulse width modulation (PWM signal) to the non-volatile FPGA directly for enable entering and exiting an ultra-low power flash freeze mode to save power consumption. A portable 2.5cm*2.5cm PCB board has been designed for the proposed non-volatile FPGA chip with peripheral voltage converter. The proposed PCB design can offer the feasibility for neuroscientists to work with for rodent epilepsy control experiments, with the long-term goal of employing them into real human surgery trial in the following years.

6.2 Future Work

The suggested advancement of each chapter has been listed in the end of each chapter. To summarize, it can be highlighted from two viewpoints:

- ***<u>Algorithm research</u>***

1. Neuron mass models:

Since the Jansen's neural mass model disregards spatial variations in activity, more detailed and comprehensive neural mass models need to be advanced in spatial domain to represent more numbers of neuron populations with higher precision and higher accuracy.

2. Different seizure patterns:

Seizures can be categorised into different patterns: high amplitude, high frequency and high amplitude oscillations etc. In chapter 3, we justify that the appearance of epilepsy seizure onset might be caused by the imbalance of excitatory and inhibitory parameters. However, seizures can also be caused by the imbalance of excitatory and inhibitory neuron connectivity, or the stimulus strength. More seizure patterns need to be discovered to help build a better understanding of seizure onset and then form a better understanding of how to build the next generation of closed-loop control systems to treatment epilepsy.

3. Closed-loop algorithm:

In this study, we used the Jansen's neural mass model to develop a systematic design approach to determine the control parameters of the proposed closed-loop controllers. It should be highlighted that the proposed design methodology is independent of the specific neuron mass model. More reasonable algorithms (machine learning, effective seizure detection, adaptive learning etc.) need to be explored to integrate with the neural mass model for providing the closed-loop stimulation to control epileptic signals.

- ***Hardware research:***

1. Different hardware platforms

In this thesis, two digital platforms (MCU, FPGAs) have been reviewed in terms of reprogrammability, power consumption and computational time cost. More digital hardware platforms (CPU, GPU, MCU, DSP, FPGA, CPLD etc.) need to be reviewed and compared to provide a fair and comprehensive comparison to benefit brain machine interfaces research field.

2. Optimized digital implementation of algorithm

When the closed-loop algorithms are suggested by computational modelling work, the optimized digital implementation of closed-loop algorithms need to be examined and demonstrated. Smaller size PCB boards of digital implementation are also required to benefit the long-term animal experiences.

6.3 Concluding Remarks

It is my firm belief that brain machine interface (BMI) will be a great asset for disabled individuals with neurological disorder diseases and motor or sensory impairments as an assisted living device. Recent progresses in brain machine interface research has led neuro-engineers and neuroscientists to record the electroencephalogram (EEG) or local field potential (LFP), analyse recordings and deliver subsequent treatments in real time by means of a closed-loop control system. Neural Interface, a subspecialty of BMI, aims to use tiny implantable/wearable devices to change precise electrical signals in nerves for the treatment of a range of debilitating chronic diseases.

This thesis has presented a system-level design of miniaturized, low-power neural interface implementation of novel closed-loop control algorithms to generate real time stimulation for seizure suppression. As the project (CANDO) we involved in is in the fourth year of its seven-year journey, more and more algorithms are expected to be on trial for rodent control experiments to test their algorithm performance on seizure suppression. I believe my thesis effort will be a solid proof of PID algorithm to be tested in neuroscience experiments and will provide a solid foundation and contribution for future neuroscience research field for seizure suppression.

References

[1] B. K. MacDonald, “The incidence and lifetime prevalence of neurological disorders in a prospective community-based study in the UK,” *Brain*, vol. 123, no. 4, pp. 665–676, 2000.

[2] M. Vila and S. Przedborski, “Targeting programmed cell death in neurodegenerative diseases,” *Nat. Rev. Neurosci.*, vol. 4, no. 5, pp. 1–11, 2003.

[3] X. Liu, H. Zhu, M. Zhang, A. G. Richardson, T. H. Lucas, and J. Van Der Spiegel, “Design of a low-noise, high power efficiency neural recording front-end with an integrated real-time compressed sensing unit,” *Proc. - IEEE Int. Symp. Circuits Syst.*, vol. 2015-July, pp. 2996–2999, 2015.

[4] M. Yin, D. A. Borton, J. Aceros, W. R. Patterson, and A. V. Nurmikko, “A 100-channel hermetically sealed implantable device for chronic wireless neurosensing applications,” *IEEE Trans. Biomed. Circuits Syst.*, vol. 7, no. 2, pp. 115–128, 2013.

[5] E. Krook-Magnuson, C. Armstrong, M. Oijala, and I. Soltesz, “On-demand optogenetic control of spontaneous seizures in temporal lobe epilepsy,” *Nat. Commun.*, vol. 4, pp. 1376–1378, 2013.

[6] M. T. Salam, M. Mirzaei, M. S. Ly, D. K. Nguyen, and M. Sawan, “An implantable closedloop asynchronous drug delivery system for the treatment of refractory epilepsy.,” *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 20, no. 4, pp. 432–42, 2012.

[7] F. T. Sun, M. J. Morrell, R. E. Wharen, S. F.T., M. M.J., and W. J. R.E., “Responsive Cortical Stimulation for the Treatment of Epilepsy,” *Neurotherapeutics*, vol. 5, no. 1, pp. 68–74, 2008.

[8] S. Ramgopal *et al.*, “Seizure detection, seizure prediction, and closed-loop warning systems in epilepsy,” *Epilepsy Behav.*, vol. 37, pp. 291–307, 2014.

[9] S. Santaniello, G. Fiengo, L. Glielmo, and W. M. Grill, “Closed-loop control of deep brain stimulation: a simulation study.,” *IEEE Trans Neural Syst Rehabil Eng*, vol. 19, no. 1, pp. 15–24, 2011.

[10] B. Shan, J. Wang, B. Deng, X. Wei, H. Yu, and H. Li, “UKF-based closed loop iterative learning control of epileptiform wave in a neural mass model,” *Cogn. Neurodyn.*, vol. 9, no. 1, pp. 31–40, 2014.

[11] B. J. Gluckman, H. Nguyen, S. L. Weinstein, and S. J. Schiff, “Adaptive electric field control of epileptic seizures.,” *J. Neurosci.*, vol. 21, no. 2, pp. 590–600, 2001.

[12] T. K. T. Nguyen *et al.*, “Closed-loop optical neural stimulation based on a 32-channel low-noise recording system with online spike sorting,” *J. Neural Eng.*, vol. 11, no. 4, 2014.

[13] N. Verma *et al.*, “A Micro-Power EEG Acquisition SoC With Integrated Feature Extraction Processor for a Chronic Seizure Detection System,” *IEEE J. Solid-State Circuits*, vol. 11, no. 4, pp. 804–817, 2014.

[14] L. V Borovikova *et al.*, “Vagus nerve stimulation attenuates the systemic inflammatory response to endotoxin.,” *Nature*, vol. 405, no. 6785, pp. 458–462, 2000.

[15] L. Jehi, “Responsive neurostimulation: The hope and the challenges,” *Epilepsy Curr.*, vol. 14, no. 5, pp. 270–271, 2014.

[16] V. Gilja, C. A. Chestek, I. Diester, J. M. Henderson, K. Deisseroth, and S. V. Krishna, “Challenges and Opportunities for Next-Generation Intacortically Based Neural Prostheses,” *IEEE Trans. Biomed. Eng.*, vol. 58, no. 7, pp. 1891–1899, 2011.

[17] Y. Liu *et al.*, “A 64-Channel Versatile Neural Recording SoC With Activity-Dependent Data Throughput,” vol. 11, no. 6, pp. 1344–1355, 2017.

[18] R. Ramezani *et al.*, “On-Probe Neural Interface ASIC for Combined Electrical Recording and Optogenetic Stimulation,” *IEEE Trans. Biomed. Circuits Syst.*, vol. 12, no. 3, pp. 576–588, 2018.

[19] S. Farshchi *et al.*, “A 128-channel 6 mW wireless neural recording IC with spike feature extraction and UWB transmitter,” *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 17, no. 4, pp. 312–321, 2014.

[20] R. R. Harrison *et al.*, “A low-power integrated circuit for a wireless 100-electrode neural recording system,” *IEEE J. Solid-State Circuits*, vol. 42, no. 1, pp. 123–133, 2007.

[21] M. S. Chae, Z. Yang, M. R. Yuce, L. Hoang, and W. Liu, “A 128-channel 6 mW wireless neural recording IC with spike feature extraction and UWB transmitter,” *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 17, no. 4, pp. 312–321, 2009.

[22] W. Biederman *et al.*, “A 4.78 mm2 Fully-Integrated Neuromodulation SoC Combining 64 Acquisition Channels with Digital Compression and Simultaneous Dual Stimulation,” *IEEE J. Solid-State Circuits*, vol. 50, no. 4, pp. 1038–1047, 2015.

[23] V. K. Jirsa, W. C. Stacey, P. P. Quilichini, A. I. Ivanov, and C. Bernard, “On the nature of seizure dynamics,” *Brain*, vol. 137, no. 8, pp. 2210–2230, 2014.

[24] P. Anantachaisilp and Z. Lin, “Fractional Order PID Control of Rotor Suspension by Active Magnetic Bearings,” *Actuators*, vol. 6, no. 1, p. 4, 2017.

[25] A. Jackson, “Neuroscience: Brain-controlled robot grabs attention,” *Nature*, vol. 485, no. 7398, pp. 317–318, 2012.

[26] I. Osorio, M. G. Frei, and S. B. Wilkinson, “Real-time automated detection and quantitative analysis of seizures and short-term prediction of clinical onset.,” *Epilepsia*, vol. 39, no. 6, pp. 615–627, 1998.

[27] A. Berenyi, M. Belluscio, D. Mao, and G. Buzsáki, “Closed-Loop Control of Epilepsy by Transcranial Electrical Stimulation,” *Science (80-.).*, vol. 337, no. 6095, pp. 735–737, 2012.

[28] X. Liu, M. Zhang, B. Subei, A. G. Richardson, T. H. Lucas, and J. Van Der Spiegel, “The PennBMBI: Design of a general purpose wireless brain-machine-brain interface system,” *IEEE Trans. Biomed. Circuits Syst.*, vol. 9, no. 2, pp. 248–258, 2015.

[29] R. S. Fisher *et al.*, “Epileptic seizures and epilepsy: Definitions proposed by the International League Against Epilepsy (ILAE) and the International Bureau for Epilepsy

(IBE)," *Epilepsia*, vol. 46, no. 4, pp. 470–472, 2005.

[30] C. K. Mbuba, A. K. Ngugi, C. R. Newton, and J. A. Carter, "The epilepsy treatment gap in developing countries: A systematic review of the magnitude, causes, and intervention strategies," *Epilepsia*, vol. 49, no. 9, pp. 1491–1503, 2008.

[31] S. Li, A. L. Zaninotto, I. S. Neville, W. S. Paiva, D. Nunn, and F. Fregni, "Clinical utility of brain stimulation modalities following traumatic brain injury: Current evidence," *Neuropsychiatr. Dis. Treat.*, vol. 11, pp. 1573–1586, 2015.

[32] A. J. Rush *et al.*, "Vagus nerve stimulation (VNS) for treatment-resistant depressions: A multicenter study," *Biol. Psychiatry*, vol. 47, no. 4, pp. 276–286, 2000.

[33] M. J. Morrell and R. N. S. S. in E. S. Group, "Responsive cortical stimulation for the treatment of medically intractable partial epilepsy.," *Neurology*, vol. 77, no. 13, pp. 1295–1304, 2011.

[34] M. A.H., H. C.H., and B. G.H., "Vagus Nerve Stimulation in the Treatment of Refractory Epilepsy," *Neurotherapeutics*. 2009.

[35] A. Beuter, J. P. Lefaucheur, and J. Modolo, "Closed-loop cortical neuromodulation in Parkinson's disease: An alternative to deep brain stimulation?," *Clin. Neurophysiol.*, vol. 125, no. 5, pp. 874–885, 2014.

[36] T. Wichmann and M. R. DeLong, "Deep Brain Stimulation for Movement Disorders of Basal Ganglia Origin: Restoring Function or Functionality?," *Neurotherapeutics*, vol. 13, no. 2, pp. 264–283, 2016.

[37] P. J. Grahn *et al.*, "A neurochemical closed-loop controller for deep brain stimulation: Toward individualized smart neuromodulation therapies," *Front. Neurosci.*, vol. 8, no. 8 JUN, pp. 1–11, 2014.

[38] B. C. Lega, C. H. Halpern, J. L. Jaggi, and G. H. Baltuch, "Deep brain stimulation in the treatment of refractory epilepsy: Update on current data and future directions," *Neurobiol. Dis.*, vol. 38, no. 3, pp. 354–360, 2010.

[39] A. P. Amar, M. L. Levy, C. Y. Liu, and M. L. J. Apuzzo, "Vagus Nerve Stimulation," *Proc. IEEE*, vol. 96, no. 7, pp. 1142–1151, 2008.

[40] C. M. DeGiorgio *et al.*, "Prospective long-term study of vagus nerve stimulation for the treatment of refractory seizures," *Epilepsia*, vol. 41, no. 9, pp. 1195–1200, 2000.

[41] A. Hartshorn and B. Jobst, "Responsive brain stimulation in epilepsy," *Ther. Adv. Chronic Dis.*, vol. 9, no. 7, pp. 135–142, 2018.

[42] "RNS System User Manual," 2010.

[43] E. Ben-Menachem, "Neurostimulation-past, present, and beyond," *Epilepsy Curr.*, vol. 12, no. 5, pp. 188–191, 2012.

[44] G. Santhanam, S. I. Ryu, B. M. Yu, A. Afshar, and K. V. Shenoy, "A high-performance brain-computer interface," *Nature*, vol. 442, no. 7099, pp. 195–198, 2006.

[45] S. Zanos, A. G. Richardson, L. Shupe, F. P. Miles, and E. E. Fetz, "The neurochip-2: An autonomous head-fixed computer for recording and stimulating in freely behaving

monkeys," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 19, no. 4, pp. 427–435, 2011.

[46] G. Deco, V. K. Jirsa, P. A. Robinson, M. Breakspear, and K. Friston, "The dynamic brain: From spiking neurons to neural masses and cortical fields," *PLoS Comput. Biol.*, vol. 4, no. 8, 2008.

[47] J. Jacobs, P. LeVan, R. Chander, J. Hall, F. Dubeau, and J. Gotman, "Interictal high-frequency oscillations (80-500 Hz) are an indicator of seizure onset areas independent of spikes in the human epileptic brain," *Epilepsia*, vol. 49, no. 11, pp. 1893–1907, 2008.

[48] Y. Wang, M. Goodfellow, P. N. Taylor, and G. Baier, "Dynamic Mechanisms of Neocortical Focal Seizure Onset," *PLoS Comput. Biol.*, vol. 10, no. 8, 2014.

[49] O. Faugeras, "Bifurcation Analysis of Jansen ' s Neural Mass Model," *Neural Comput.*, vol. 18, no. 12, pp. 3052–3068, 2006.

[50] B. A. Lopour and A. J. Szeri, "A model of feedback control for the charge-balanced suppression of epileptic seizures," *J. Comput. Neurosci.*, vol. 28, no. 3, pp. 375–387, 2010.

[51] J. Wang, E. Niebur, J. Hu, and X. Li, "Suppressing epileptic activity in a neural mass model using a closed-loop proportional-integral controller," *Sci. Rep.*, vol. 6, no. May, pp. 1–12, 2016.

[52] L. Xia, A. Soltan, Y. Wang, A. Jackson, G. Chester, and P. Degenaar, "Closed-loop Proportion-Derivative Control Stabilization Analysis for Suppressing Epileptic Seizures in a Neural Mass Model," vol. 1, pp. 1–4.

[53] O. David, D. Cosmelli, and K. J. Friston, "Evaluation of different measures of functional connectivity using a neural mass model," *Neuroimage*, vol. 21, no. 2, pp. 659–673, 2004.

[54] B. H. Jansen, G. Zouridakis, and M. E. Brandt, "A neurophysiologically-based mathematical model of flash visual evoked potentials," *Biol. Cybern.*, vol. 68, no. 3, pp. 275–283, 1993.

[55] J. Touboul, F. Wendling, P. Chauvel, and O. Faugeras, "Neural mass activity, bifurcations, and epilepsy," *Neural Comput.*, vol. 23, no. 12, pp. 3232–3286, 2011.

[56] A. Spiegler, S. J. Kiebel, F. M. Atay, and T. R. Knösche, "Bifurcation analysis of neural mass models: Impact of extrinsic inputs and dendritic time constants," *Neuroimage*, vol. 52, no. 3, pp. 1041–1058, 2010.

[57] E. M. IZHIKEVICH, "Neural Excitability, Spiking and Bursting," *Int. J. Bifurc. Chaos*, vol. 10, no. 06, pp. 1171–1266, 2000.

[58] S. ichi Amari, "Dynamics of pattern formation in lateral-inhibition type neural fields," *Biol. Cybern.*, vol. 27, no. 2, pp. 77–87, 1977.

[59] P. Kaps and P. Rentrop, "Generalized Runge-Kutta methods of order four with stepsize control for stiff ordinary differential equations," *Numer. Math.*, vol. 33, no. 1, pp. 55–68, 1979.

[60] V. N. Murthy and E. E. Fetz, "Oscillatory activity in sensorimotor cortex of awake monkeys: synchronization of local field potentials and relation to behavior," *J. Neurophysiol.*, vol. 76, no. 6, pp. 3949–3967, 1996.

[61] E. L. White and D. Czeiger, “Synapses made by axons of callosal projection neurons in mouse somatosensory cortex: Emphasis on intrinsic connections,” *J. Comp. Neurol.*, 1991.

[62] X. B. Liu, Z. H. Zheng, M. C. Xi, and C. P. Wu, “Distribution of synapses on an intracellularly labeled small pyramidal neuron in the cat motor cortex,” *Anat. Embryol. (Berl).*, 1991.

[63] S. S. Ghoreishizadeh *et al.*, “Four-Wire Interface ASIC for a Multi-Implant Link,” pp. 1–12, 2017.

[64] R. A. Andersen, S. Musallam, and B. Pesaran, “Selecting the signals for a brain-machine interface,” *Curr. Opin. Neurobiol.*, vol. 14, no. 6, pp. 720–726, 2004.

[65] B. Engelhard, N. Ozeri, Z. Israel, H. Bergman, and E. Vaadia, “Inducing Gamma Oscillations and Precise Spike Synchrony by Operant Conditioning via Brain-Machine Interface,” *Neuron*, vol. 77, no. 2, pp. 361–375, 2013.

[66] A. Jackson and T. M. Hall, “Decoding Local Field Potentials for Neural Interfaces,” *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 25, no. 10, pp. 1705–1714, 2017.

[67] G. Gagnon-Turcotte *et al.*, “A wireless optogenetic headstage with multichannel electrophysiological recording capability,” *Sensors (Switzerland)*, vol. 15, no. 9, pp. 22776–22797, 2015.

[68] C. P. Young, S. F. Liang, D. W. Chang, Y. C. Liao, F. Z. Shaw, and C. H. Hsieh, “A portable wireless online closed-loop seizure controller in freely moving rats,” *IEEE Trans. Instrum. Meas.*, vol. 60, no. 2, pp. 513–521, 2011.

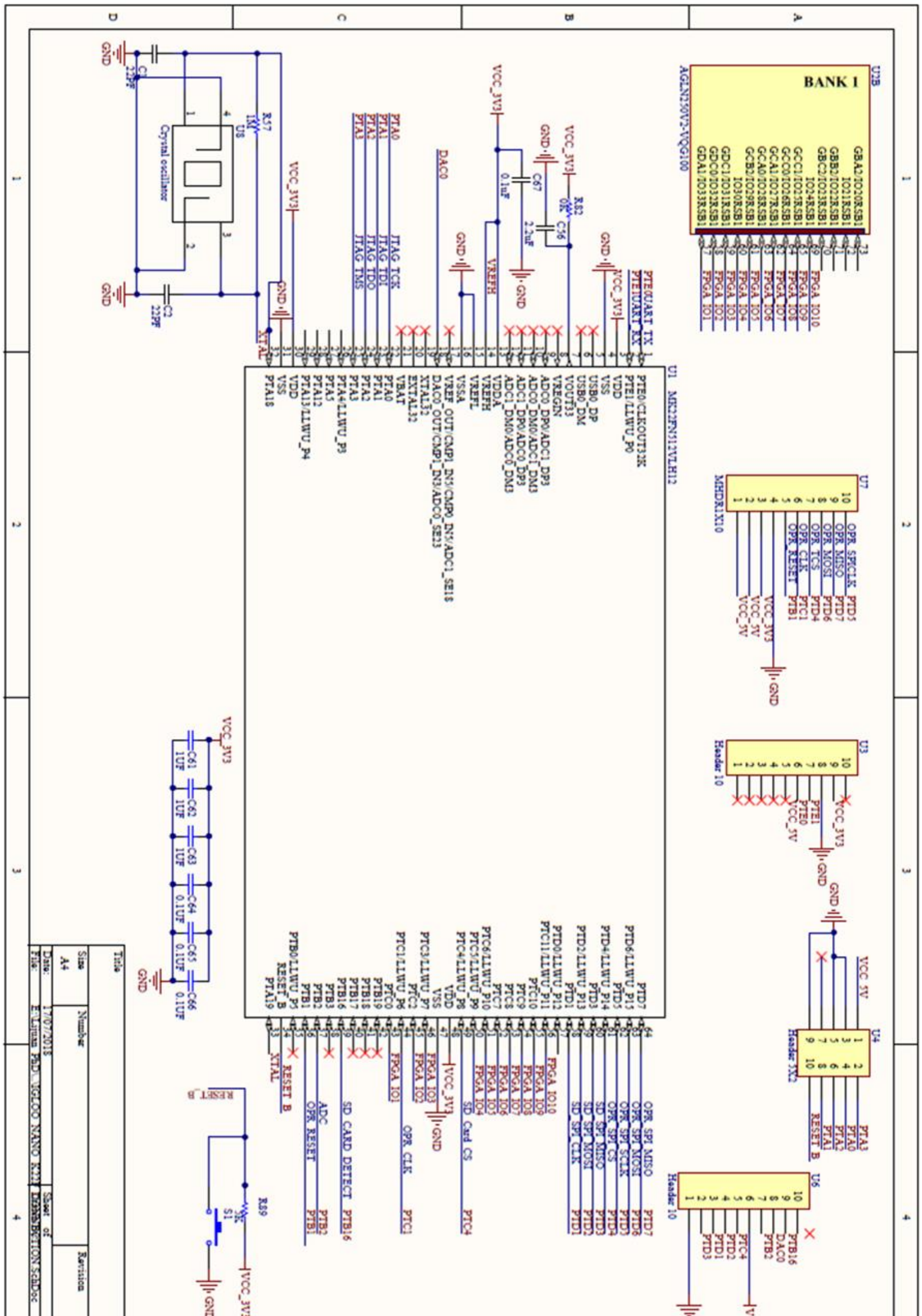
[69] R. Ramezani *et al.*, “On-Probe Neural Interface ASIC for Combined Electrical Recording and Optogenetic Stimulation,” vol. XX, pp. 1–9, 2017.

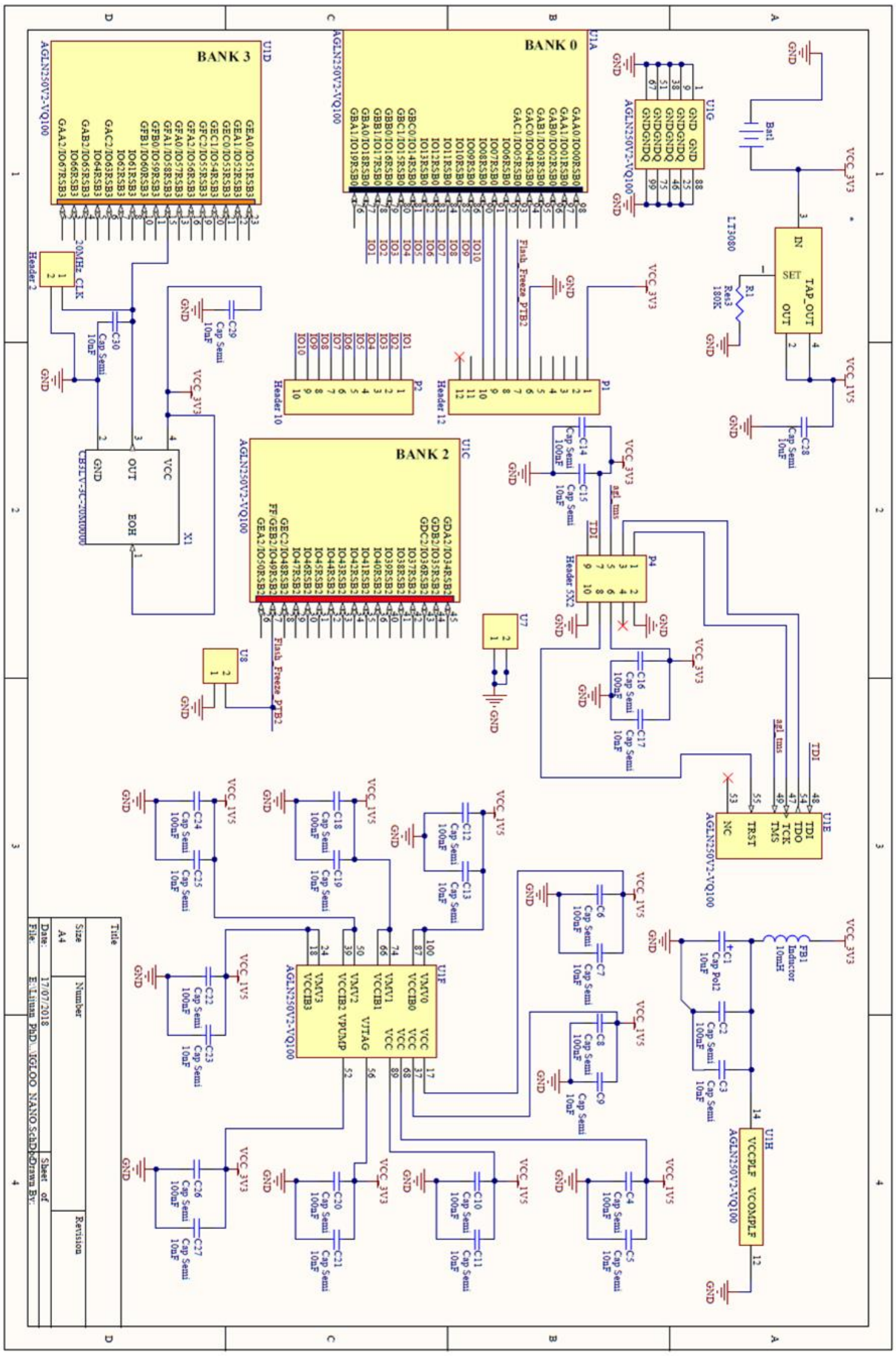
[70] K. Nikolic, N. Grossman, M. S. Grubb, J. Burrone, C. Toumazou, and P. Degenaar, “Photocycles of channelrhodopsin-2,” *Photochem. Photobiol.*, vol. 85, no. 1, pp. 400–411, 2009.

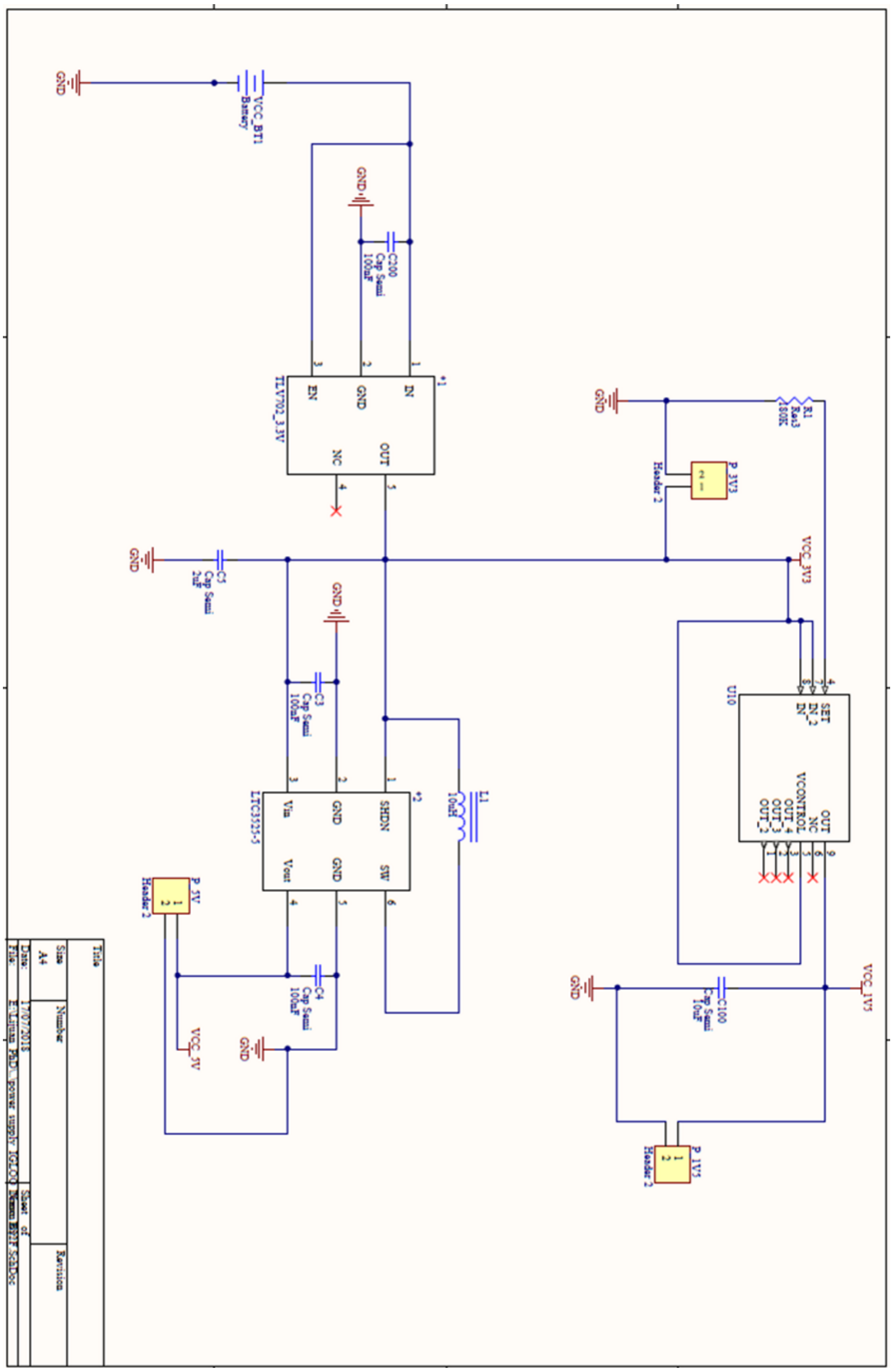
[71] N. Grossman, K. Nikolic, M. S. Grubb, J. Burrone, C. Toumazou, and P. Degenaar, “High-frequency limit of neural stimulation with ChR2,” *Proc. Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. EMBS*, pp. 4167–4170, 2011.

[72] J. J. Nassi, M. C. Avery, A. H. Cetin, A. W. Roe, and J. H. Reynolds, “Optogenetic Activation of Normalization in Alert Macaque Visual Cortex,” *Neuron*, vol. 86, no. 6, pp. 1504–1517, 2015.

Appendix A. Schematic Circuits of MCU Circuits

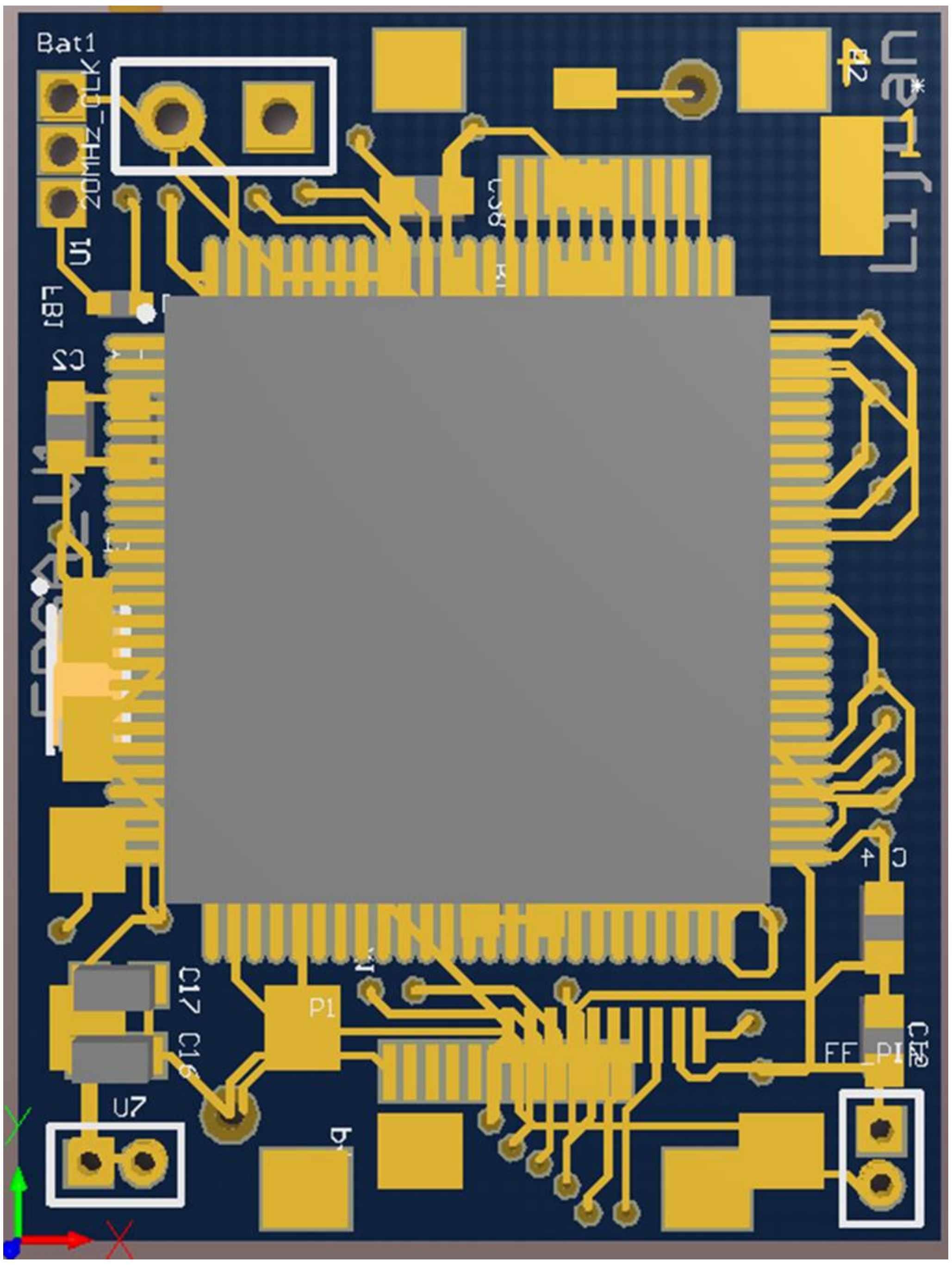



BANK 0
BANK 2
BANK 3
U1A
U1C
U1D
U1E
U1F
U1G
U1H
AGLN250V2-VQ100
LT3080
Bat1
Header 12
Header 10
Header 5X2
Header 2
20MHz_CLK
Flash_Freeze_PTB2
VCC_3V3
VCC_1V5
GND
TDI
TDO
TCK
TMS
TRST
FB1
Inductor
Title
Size
A4
Number
Revision
Date:
17/07/2018
Sheet of
Drawn By:

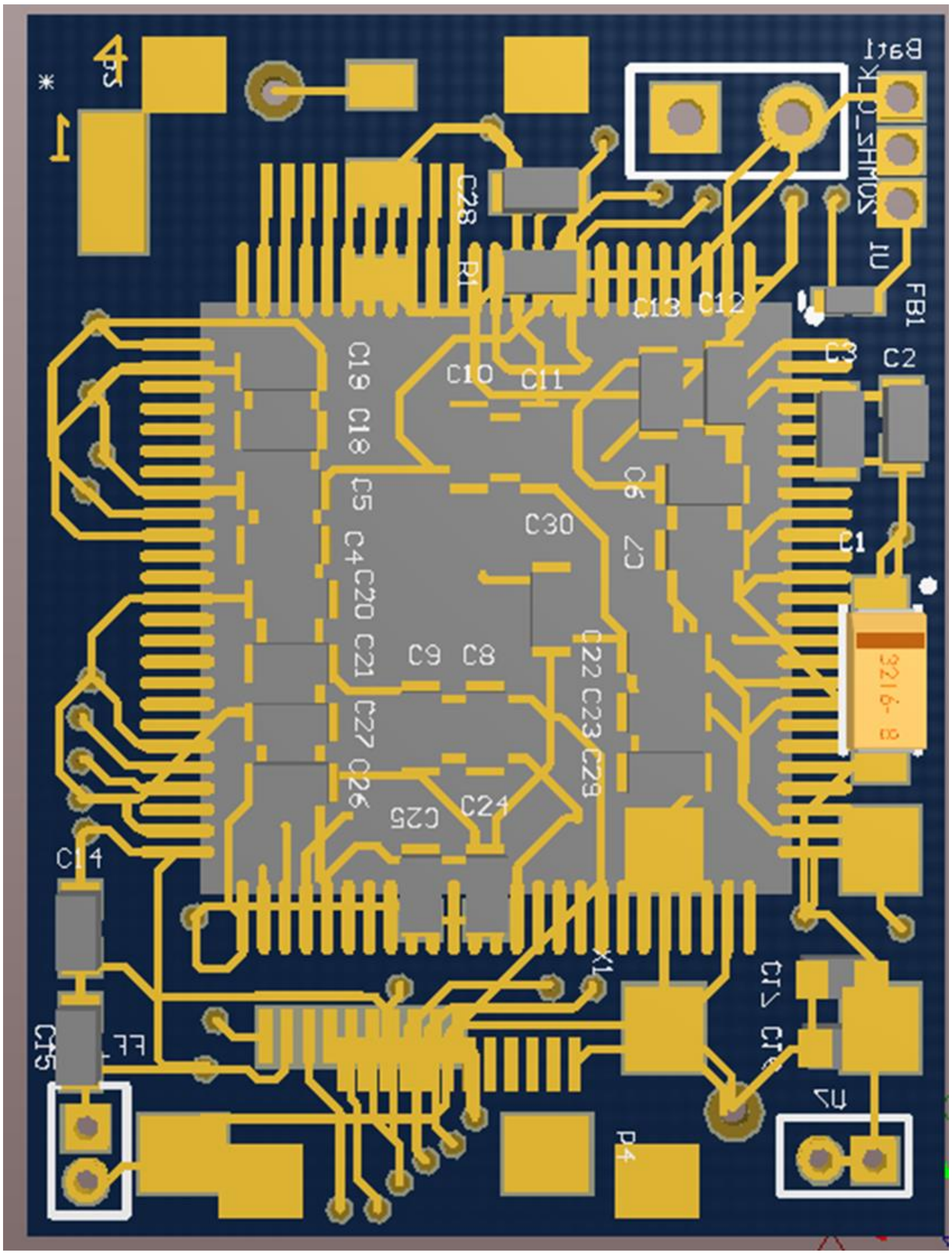


GND
VCC_BT1
Battery
C100
Cap Semi
100nF
TLV702_3.3V
EN
GND
IN
NC
OUT
GND
R1
Res3
180K
P_3V3
Header 2
VCC_3V3
C5
Cap Semi
2uF
C3
Cap Semi
100nF
LTC3525-5
Vin
GND
SHDN
Vout
SW
L1
10uH
P_5V
Header 2
C4
Cap Semi
100nF
VCC_5V
U10
SET
IN_2
IN
VCONTROL
OUT
NC
OUT_4
OUT_3
OUT_2
C100
Cap Semi
10uF
VCC_1V5
P_1V5
Header 2
Title
Size
A4
Number
Revision
Date:
17/07/2018
Sheet of
File:

Front

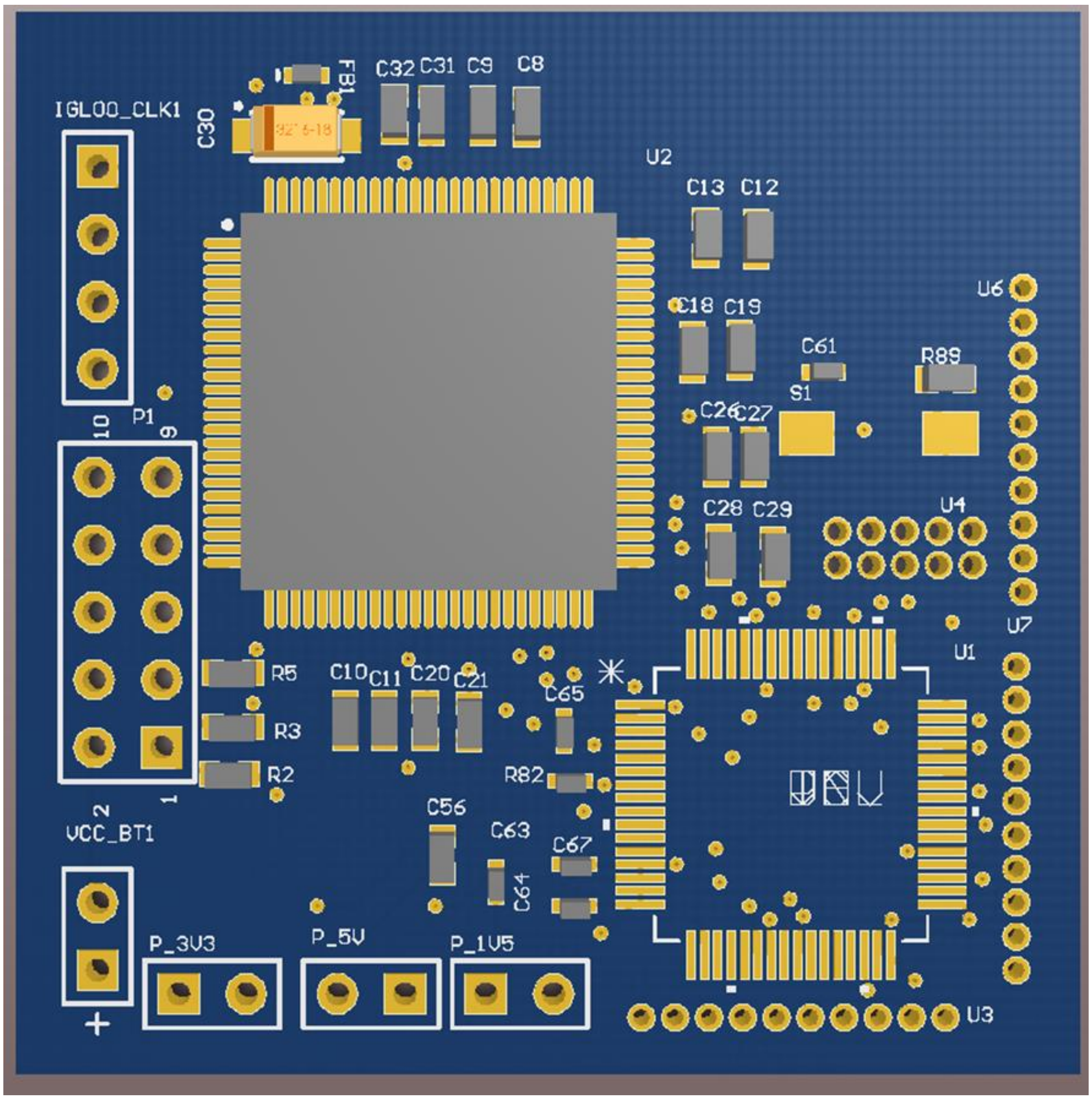


End

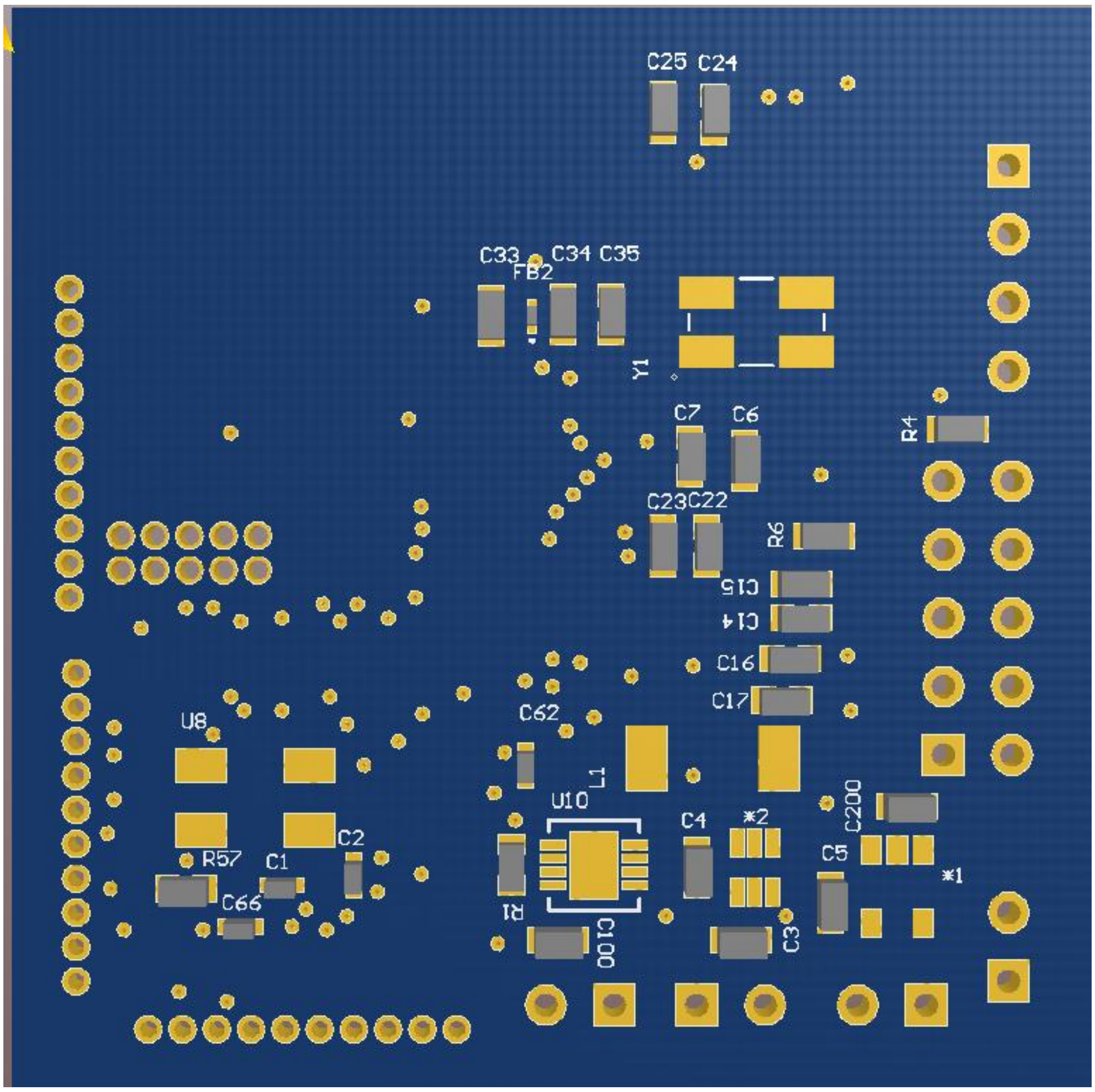


C28
R1
C19
C18
C5
C4
C20
C21
C27
C26
C25
C24
C9
C8
C10
C11
C13
C12
C30
C6
C22
C23
C29
C3
C2
C1
FB1
Batt
3216-B
C14
C15
X1
P4

Front


IGLOO_CLK1
C30
FB1
C32
C31
C9
C8
U2
C13
C12
U6
C18
C19
C61
R89
S1
C26
C27
10
P1
9
C28
C29
U4
U7
U1
R5
C10
C11
C20
C21
C65
R3
R82
R2
1
2
C56
VCC_BT1
C63
C67
C64
P_3V3
P_5V
P_1V5
U3
+

End



C25
C24
C33
FB2
C34
C35
Y1
C7
C6
R4
C23
C22
R6
C15
C14
C16
C17
C62
U8
L1
U10
C200
C4
*2
C5
*1
R57
C1
C2
C66
R1
C100
C3

```
--------------------------------------------------------------
--------
-- Company: MICROSEMI
-- File: SPI_MASTER.vhd
-- <Revision number>: <Date>: <Comments>
--    Targeted    device:    <Family::IGLOO>    <Die::AGLN250V2>
<Package::100 VQFP>
-- Author: LIJUAN XIA
--------------------------------------------------------------
--------

LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.numeric_std.all;
USE ieee.std_logic_signed.all;

entity SPI_MASTER_V16 is

GENERIC (
    CMD_BITS                                        :   POSITIVE: = 7;
-- 8-1 BITS OF EACH CMD, 8 BITS ARE INFORMATION
    MISO_TAPS                                      :   POSITIVE: = 15;
-- THIS IS FOR COUNTING RECORDING DATA FOR FURTHER FILTERING
    FPGA_SCLK_PRSCL                                :   POSITIVE: = 9;
-- SPI CLOCK: 1000KHZ, 1000 NS (50*20) 20/2 = 10
    TAPS                            :  POSITIVE: = 10
);

Port (
   FPGA_CLOCK                        : IN  std_logic;
   -- FPGA CLOCK : 20MHZ , 50 NS ,       PIN15
   FPGA_MISO                         : IN  std_logic;
   -- MCU  MOSI  :                       PTD2
   FPGA_SCLK                         : IN  std_logic;
   -- SPI  CLOCK : 1MHZ , 1000 NS, 1MS, PTD1
   FPGA_CS                           : IN  std_logic;
   -- SPI  CS    :                       PTC4
    FPGA_OUT_16_BITS                   : OUT std_logic_vector (15
DOWNTO 0)
);
End SPI_MASTER_V16;

Architecture architecture_SPI_MASTER_V16 of SPI_MASTER_V16 is
   -- signal, component etc. declarations
    SIGNAL   FIR_FILTER_FLAG          : std_logic := '0';
    SIGNAL   FPGA_MISO_REG            : std_logic := '0';
    SIGNAL  FPGA_SCLK_REG           : std_logic := '0';
    SIGNAL  FPGA_CS_REG             : std_logic := '1' ;
    SIGNAL  FPGA_FLAG               : std_logic := '0';

    SIGNAL  FPGA_MISO_PARALLEL      : std_logic_vector (CMD_BITS
DOWNTO 0)  := (OTHERS => '0');
    SIGNAL  INDEX_MISO             : INTEGER RANGE 0 TO 7 := 0 ;
    SIGNAL  INDEX_SHIFT_REGISTER   : INTEGER RANGE 0 TO 27:= 0 ;

    TYPE     SHIFT_REGISTER  IS  ARRAY  (MISO_TAPS  downto  0)  OF
std_logic_vector (7 DOWNTO 0) ;
    SIGNAL       SHIFT_BUFFER     :     SHIFT_REGISTER          :=
(                      b"00000000",b"00000000",b"00000000",b"00
000000",  b"00000000",b"00000000",b"00000000",b"00000000",

b"00000000",b"00000000",b"00000000",b"00000000",b"00000000",b"
00000000",b"00000000",b"00000000" );

    TYPE      KERNEL_VALUE  IS  ARRAY  (MISO_TAPS  downto  0)  OF
std_logic_vector (7 DOWNTO 0);
    SIGNAL  KERNEL   : KERNEL_VALUE   := (

b"00000001",b"00000001",b"00000001",b"00000001",b"00000001",b"
00000001",b"00000001",b"00000001" ,

b"00000001",b"00000001",b"00000001",b"00000001",b"00000001",b"
00000001",b"00000001",b"00000001" );

--  SIGNAL  KERNEL   : KERNEL_VALUE   := (
--
b"11111111",b"11111111",b"11111111",b"11111111",b"11111111",b"
11111111",b"11111111",b"11111111",
--
b"11111111",b"11111111",b"11111111",b"11111111",b"11111111",b"
11111111",b"11111111",b"11111111");

    SIGNAL  ACCUMULATOR            : STD_LOGIC_VECTOR  (15  DOWNTO
0)  :="0000000000000000";
    SIGNAL  SIG_OUT                 : STD_LOGIC_VECTOR  (7  DOWNTO
0)   :="00000000";
    SIGNAL   SHIFT_BUFFER_0        :  STD_LOGIC_VECTOR  (7  DOWNTO
0)   :="00000000";
    SIGNAL   I                 : INTEGER RANGE 0 TO MISO_TAPS  :=
0;

begin
INDEX_SHIFT_REGISTER_PRO : PROCESS (INDEX_MISO)
BEGIN

   IF (INDEX_MISO = 0) THEN
    -- FPGA_OUT_8_BITS (7 DOWNTO 0) <= FPGA_MISO_PARALLEL;
    --  OUTPUT 8 BIT DATA FOR NEXT MODULE (DAC/OPTICAL CONVERTER)
```

```
          INDEX_SHIFT_REGISTER  <= INDEX_SHIFT_REGISTER + 1;
    -- COUNTING 8 BIT PARALLEL DATA INTO SHIFT BUFFER
           SHIFT_BUFFER (0)      <= FPGA_MISO_PARALLEL;
           SHIFT_BUFFER (MISO_TAPS DOWNTO 1)     <=
           SHIFT_BUFFER (MISO_TAPS-1 DOWNTO 0);
                FIR_FILTER_FLAG <= '1';
                 ELSE
                FIR_FILTER_FLAG <= '0';
                 END IF;

END PROCESS INDEX_SHIFT_REGISTER_PRO;


--ACCUMULATOR_PRO : PROCESS(FPGA_CLOCK,INDEX_MISO)
ACCUMULATOR_PRO : PROCESS(INDEX_MISO,FPGA_CLOCK)
BEGIN

IF ( FPGA_CLOCK'EVENT AND FPGA_CLOCK = '1' ) THEN

    IF (INDEX_MISO = 0) THEN
        SHIFT_BUFFER_0                  <= FPGA_MISO_PARALLEL;
        ACCUMULATOR(15  DOWNTO   0)                        <=
SHIFT_BUFFER(0)*KERNEL(0)   +      SHIFT_BUFFER(1)*KERNEL(1)   +
SHIFT_BUFFER(2)*KERNEL(2)   +      SHIFT_BUFFER(3)*KERNEL(3)   +
SHIFT_BUFFER(4)*KERNEL(4)   +      SHIFT_BUFFER(5)*KERNEL(5)   +
SHIFT_BUFFER(6)*KERNEL(6)   +      SHIFT_BUFFER(7)*KERNEL(7)   +
SHIFT_BUFFER(8)*KERNEL(8)   +      SHIFT_BUFFER(9)*KERNEL(9)   +
SHIFT_BUFFER(10)*KERNEL(10)  +    SHIFT_BUFFER(11)*KERNEL(11)  +
SHIFT_BUFFER(12)*KERNEL(12)  +    SHIFT_BUFFER(13)*KERNEL(13)  +
SHIFT_BUFFER(14)*KERNEL(14) +  SHIFT_BUFFER(15)*KERNEL(15);
        FPGA_OUT_16_BITS                     <= ACCUMULATOR(15
DOWNTO 0);
    END IF;
END IF;
END PROCESS ACCUMULATOR_PRO;

MISO_SHIFT_REGISTER_PRO : PROCESS(FPGA_SCLK)
BEGIN

          FPGA_MISO_REG                     <=        FPGA_MISO;
-- PUT THE INPUT SERIAL MISO FROM MCU/ASIC TO "FPGA_MISO_REG"

          IF(FPGA_CS        =         '0')                   THEN
-- SPI CS ACTIVE LOW
                 IF (FPGA_SCLK'EVENT AND FPGA_SCLK = '1') THEN
-- RISING EDGE OF FPGA_SCLK
                         IF (INDEX_MISO = 0) THEN
                             FPGA_MISO_PARALLEL(0)           <=
FPGA_MISO_REG;
                             INDEX_MISO <= INDEX_MISO + 1;
                         ELSIF (INDEX_MISO = 1) THEN
                             FPGA_MISO_PARALLEL(1)           <=
FPGA_MISO_REG;
                                        INDEX_MISO <= INDEX_MISO + 1;
                                  ELSIF (INDEX_MISO = 2) THEN
                                        FPGA_MISO_PARALLEL(2)              <=
FPGA_MISO_REG;
                                        INDEX_MISO <= INDEX_MISO + 1;
                                  ELSIF (INDEX_MISO = 3) THEN
                                        FPGA_MISO_PARALLEL(3)              <=
FPGA_MISO_REG;
                                        INDEX_MISO <= INDEX_MISO + 1;
                                  ELSIF (INDEX_MISO = 4) THEN
                                        FPGA_MISO_PARALLEL      (4)        <=
FPGA_MISO_REG;
                                        INDEX_MISO <= INDEX_MISO + 1;
                                  ELSIF (INDEX_MISO = 5) THEN
                                        FPGA_MISO_PARALLEL      (5)        <=
FPGA_MISO_REG;
                                        INDEX_MISO <= INDEX_MISO + 1;
                                  ELSIF (INDEX_MISO = 6) THEN
                                        FPGA_MISO_PARALLEL(6)              <=
FPGA_MISO_REG;
                                        INDEX_MISO <= INDEX_MISO + 1;
                                  ELSIF(INDEX_MISO = 7) THEN
                                        FPGA_MISO_PARALLEL      (7)        <=
FPGA_MISO_REG;
                                        INDEX_MISO <= 0;
                                  END IF; --// (INDEX_MISO = 0)
                           END IF; --// (FPGA_SCLK'EVENT AND FPGA_SCLK =
'1')
----------------------------------------  THIS  IS  FOR  MISO  TO
RECEIVE 8 BIT DATA INTO PARALLEL ------------------------------
------------------


              END IF; --// (FPGA_CS = '0')


END PROCESS MISO_SHIFT_REGISTER_PRO;

end architecture_SPI_MASTER_V16;
```

Appendix G. VHDL for FDM Stimulation Implementation

```
--------------------------------------------------------------
------------------
-- Company: NEWCASTLE UNIVERSITY
```

```
--
-- File: SPI_STIMULATION.vhd
-- File history:
--      SPI_STIMULATION: 4/1/2017:
-- Description:
--
--    Targeted    device:    <Family::IGLOO>    <Die::AGLN250V2>
<Package::100 VQFP>
-- Author: LIJUAN XIA
--
---------------------------------------------------------------
------------------
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;
USE ieee.std_logic_signed.all;

entity SPI_STIMULATION is
GENERIC (
 --   BAUDRATE: Positive: = 4160
    TAPS                                  :   POSITIVE  := 10  ;
-- THIS IS FOR CMD NUMBER , THIS IS FOR SENDING 3*3 CMD TO TURN
LED ON OR LED OFF + 2 FOR TIME DELAY
    MISO_TAPS                             :   POSITIVE  := 15  ;
-- THIS IS FOR COUNTING RECORDING DATA FOR FURTHER FILTERING
    CMD_BITS                              :   POSITIVE  := 11  ;
-- 8+4-1 BITS OF EACH CMD, 8 BITS ARE INFORMATION, 4 BITS ARE
TIME DELAY
    CMD_NUMBERS                             :   POSITIVE  := 1;
-- 2 CMD SO FAR: LED ON, LED OFF

    FPGA_SCLK_PRSCL                       :   POSITIVE  := 19  ;
-- SPI CLOCK : 500KHZ, 2000 NS (50*40) 40/2 = 20
    OPTRODE_CLK_PRSCL                    :   POSITIVE  := 6    ;
-- OPTRODE CLOCK : 1.6MHZ , 625 NS (50 * 12.5) 12.5/2 = 6.25
-- PACKET_WAIT : , 200 MS / 50 NS = 200 000000/50 = 4000000 ,
FOR EACH PACKET, THE INTERVALS ARE 200MS
   PACKET_WAIT_PRSCL_NUM                  :   POSITIVE   :=  31;
-- THIS IS A BUFFER SIZE FOR DETERMINING HOW LONG WILL EACH
PACKET WILL WAIT
   BAUDRATE                     :  POSITIVE := 100
);

port (
    FPGA_CLOCK                            :   IN     STD_LOGIC;
-- PIN15 : FPGA CLOCK 20MHZ, 50NS
    FPGA_RESET_BUTTON                   :   IN     STD_LOGIC;
-- PIN10 : RESET_BUTTON:  RESETTING
    PACKET_WAIT_PRSCL_IN           :   IN    STD_LOGIC_VECTOR(15
DOWNTO 0);        --

    FPGA_SCLK_OUT                         :   OUT    STD_LOGIC;
-- SPI CLOCK : 500KHZ, 2000 NS (50*40)
    FPGA_MOSI_OUT                               :    OUT    STD_LOGIC;
-- SPI MOSI
    FPGA_CS_OUT                                 :    OUT    STD_LOGIC;
-- SPI CHIP SELECT
    PACKET_FINSHED_TAG_OUT                   :     OUT     STD_LOGIC;
-- PACKET FINSHED TAG OUT

    OPTRODE_RST                                 :    OUT    STD_LOGIC;
-- OPTRODE RESET
    OPTRODE_CLK_1600KHZ                        :     OUT     STD_LOGIC
-- OPTRODE CLOCK : 1.6MHZ , 625 NS (50 * 12.5) 12.5/2 = 6.25
);
End SPI_STIMULATION;

Architecture architecture_SPI_STIMULATION of SPI_STIMULATION is
   -- signal, component etc. declarations

  -- signal, component etc. declarations
-- THIS SIGNAL DEFINITION IS FOR DEFINING PACKET SIGNAL
SIGNAL   PACKET_WAIT_PRSCL             : INTEGER     := 800;

SIGNAL   PACKET_WAIT_PRSCL_REG_ON      : INTEGER     := 0;
SIGNAL   PACKET_WAIT_PRSCL_REG_OFF     : INTEGER     := 0;

SIGNAL    FPGA_SCLK_COUNTER                     :  INTEGER  RANGE  0  TO
FPGA_SCLK_PRSCL :=0;

SIGNAL   FPGA_SCLK_REG               : STD_LOGIC := '0';

SIGNAL    OPTRODE_CLK_COUNTER                   :  INTEGER  RANGE  0  TO
OPTRODE_CLK_PRSCL :=0;
SIGNAL   OPTRODE_CLK_REG             : STD_LOGIC := '0';

SIGNAL    OPTRODE_RESET_COUNTER                 :  INTEGER  RANGE  0  TO
OPTRODE_CLK_PRSCL :=0;
SIGNAL   OPTRODE_RESET_REG           : STD_LOGIC := '0';
SIGNAL   OPTRODE_RESET_I             : INTEGER RANGE 0 TO 1  :=0;

SIGNAL   PRSCL                     : INTEGER RANGE 0 TO 5208:=0;
SIGNAL   DATAFLL                    : STD_LOGIC_VECTOR(10 downto
0) := "00000000000";
SIGNAL    INDEX                                 :  INTEGER  RANGE  0  TO
CMD_BITS:=0;
SIGNAL    INDEX2                                :  INTEGER  RANGE  0  TO
(TAPS+1):= 0;
SIGNAL    INDEX_MISO                            :  INTEGER  RANGE  0  TO
(MISO_TAPS+1)  := 0  ;

SIGNAL   SPI_CS_REG                  : STD_LOGIC  :='1';
TYPE          SHIFT_REGISTER  IS  ARRAY  (TAPS  DOWNTO  0)  OF
STD_LOGIC_VECTOR (CMD_BITS DOWNTO 0);
```

```
SIGNAL    SHIFT_REGISTER_BUFFER_LED_ON                              :
SHIFT_REGISTER                                                     :=
(                                                                  --
LED ON
b"111100000000", b"111100010000" , b"111100000110",
b"111111111111",     b"111100000000"     ,     b"111100100000",
b"111100000101", b"111111111111" , b"111100000000",
b"111100100000", b"111100000100"
 );


SIGNAL    SHIFT_REGISTER_BUFFER_LED_OFF                             :
SHIFT_REGISTER                                                     :=
(                                                                  --
LED OFF
b"111100000000", b"111100010000" , b"111100000110",
b"111111111111", b"111100000000" , b"111100010000",
b"111100000101", b"111111111111" , b"111100000000",
b"111100010000", b"111100000100"
 );

SIGNAL  SHIFT_REGISTER_BUFFER            : SHIFT_REGISTER := (
b"000000000000", b"000000000000", b"000000000000",
b"000000000000", b"000000000000", b"000000000000",
b"000000000000", b"000000000000", b"000000000000",
b"000000000000", b"000000000000"
 );

TYPE     SHIFT_REGISTER_MISO IS ARRAY (MISO_TAPS DOWNTO 0) OF
STD_LOGIC_VECTOR (7 DOWNTO 0);
SIGNAL SHIFT_REGISTER_MISO_BUFFER: SHIFT_REGISTER_MISO:= (
b"000000000000", b"000000000000", b"000000000000",
b"000000000000", b"000000000000", b"000000000000",
b"000000000000", b"000000000000", b"000000000000",
b"000000000000", b"000000000000"
 );

SIGNAL   FPGA_MISO_PARALLEL         :  STD_LOGIC_VECTOR(7 downto
0)         := "00000000";
SIGNAL   FPGA_MOSI_PARALLEL          :  STD_LOGIC_VECTOR(CMD_BITS
DOWNTO 0)  := SHIFT_REGISTER_BUFFER(0);
SIGNAL   FPGA_MOSI_REG               :  STD_LOGIC_VECTOR(CMD_BITS
DOWNTO 0)  := "000000000000";

SIGNAL  CNT                     : INTEGER     : = 0;
SIGNAL  RESET_CNT               : INTEGER     : = 0;
SIGNAL  OPTRODE_RST_REG         : STD_LOGIC   : ='0';
SIGNAL  PACKET_WAIT_CNT         : INTEGER     : = 0;
SIGNAL  PACKET_WAIT_MAKER       : STD_LOGIC   : ='0';
SIGNAL  PACKET_FINSHED_TAG      : STD_LOGIC   : ='0';
SIGNAL  INDEX_SCHEDULE_CMD      : STD_LOGIC := '0';

-- PWM PARAMETERS
TYPE    PWM_REGISTER IS ARRAY (PACKET_WAIT_PRSCL_NUM DOWNTO 0)
OF INTEGER;

SIGNAL  PWM            : PWM_REGISTER := (                  -
- DEFINE DUTY CYCLES FOR LED ON AND LED OFF,  4000000 , FOR EACH
PACKET, THE INTERVALS ARE 200MS

--400,400,100,100,      200,200,800,800,      100,100,100,100,
800,200,800,800,
--100,100,100,100,      200,800,800,100,      200,200,100,800,
800,200,800,800

--0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
--0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0

1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1
 );


--SIGNAL  PACKET_WAIT_PRSCL                     : INTEGER    := 1;
SIGNAL  INDEX_PACKET                          : INTEGER    := 0;

begin


-- architecture body
SHIFT_REGISTER_PRO: PROCESS( FPGA_CLOCK )
BEGIN

PACKET_FINSHED_TAG_OUT <= PACKET_FINSHED_TAG;

IF (FALLING_EDGE ( FPGA_CLOCK )) THEN

-- RESET ALL THE COUNTERS TO AN APPRORAITE STATE
      IF ( FPGA_RESET_BUTTON = '0') THEN
             CNT                        <= 0   ;
             OPTRODE_RST_REG            <= '0';
             PACKET_FINSHED_TAG         <= '1';
      ELSIF ( CNT = 1000 ) THEN
             CNT                        <= 0   ;
             RESET_CNT                  <= RESET_CNT + 1;
             IF(RESET_CNT = 0) THEN
                   OPTRODE_RST_REG      <= '0';
             ELSIF(RESET_CNT = 1 ) THEN
                   OPTRODE_RST_REG        <= '1'  ;          --
RESETTING HIGH
                   PACKET_FINSHED_TAG   <= '0';
                   SHIFT_REGISTER_BUFFER                     <=
SHIFT_REGISTER_BUFFER_LED_ON ;

            ELSIF(RESET_CNT = 2 ) THEN
                  OPTRODE_RST_REG        <= '0';
                  RESET_CNT              <= 2;
            END IF;--// IF(RESET_CNT = 0) THEN
      ELSE
            CNT                          <= CNT + 1;
      END IF;  --//IF ( FPGA_RESET_BUTTON = '0') THEN


-- WHEN PACKET_FINSHED_TAG IS '0', MEANS PACKET IS TRANSMITTING
      IF ( PACKET_FINSHED_TAG = '0') THEN
            IF     (     OPTRODE_RST_REG     =     '1')     THEN
-- RESET   FROM FPGA BUTTON OF SW1, PINOUT 20
                  INDEX                <= 0;                    -
- INDEX:  THIS IS FOR TRANSFERRING BIT  BY BIT COUNTING
                  INDEX2               <= 1;                    -
- INDEX2: THIS IS FOR TRANSFERRING BTYE BY BYTE COUNTING
                  INDEX_MISO               <= 0;

                  FPGA_SCLK_REG                           <='1';
-- INTERNAL FPGA SCLK COUNTER (500KHZ)
                  OPTRODE_CLK_REG                         <='0';
-- INTERNAL OPTRODE SCLK COUNTER (1.6 MHZ)

                  PACKET_FINSHED_TAG                <=      '0';
-- COUNTING DOES 3*3 COMMANDS SEND
                  FPGA_MOSI_PARALLEL                             <=
SHIFT_REGISTER_BUFFER(0);     -- MOSI  COUNTER  :  LST  OF  SHIFT
REGISTER BUFFER[0]
             ELSE
                      IF      (       (FPGA_SCLK_COUNTER       =
FPGA_SCLK_PRSCL) and ( FPGA_SCLK_REG = '0') ) THEN   -- IN EACH
FPGA SCLK RISING EDGE , COUNTING INDEX FOR MOSI

                          IF(INDEX     <     CMD_BITS)     THEN
-- INDEX: USED TO COUNT CMD BITS OUT FOR FPGA_MOSI
                             INDEX <= INDEX + 1;
                          ELSE
                             INDEX  <= 0;
                             INDEX2     <=     INDEX2     +     1;
-- INDEX2: USED TO COUND CMD NUMBERS FOR EACH PACKET: 3*3 CMD +
2*'11111111111' FOR TIME DELAY
                             INDEX_MISO <= INDEX_MISO +1 ;

                               IF    (INDEX2    =    TAPS)    THEN
-- TAPS : 11 (3*3 + 2 )
                                    INDEX2 <= 0;
                               ELSIF(INDEX2 = 0 ) THEN
                                    PACKET_FINSHED_TAG    <=
'1';                -- IF INDEX2 IS 0 , IT MEANS EACH PACKET HAS
```

```
FINISHED TRANSMITTING
                                                   INDEX_SCHEDULE_CMD    <=
NOT INDEX_SCHEDULE_CMD ;

------------------------------------------------------ THIS IS
FOR DELIVERING VARIABLE DUTY PWM TO STIMULATION ---------------


                                                        --
PACKET_WAIT_PRSCL_IN

IF(INDEX_PACKET < PACKET_WAIT_PRSCL_NUM) THEN
            PWM(0)                                                   <=
TO_INTEGER(UNSIGNED(PACKET_WAIT_PRSCL_IN));
PWM(PACKET_WAIT_PRSCL_NUM DOWNTO 1)  <=
PWM (PACKET_WAIT_PRSCL_NUM-1 DOWNTO 0);
                   PACKET_WAIT_PRSCL <=  PWM(0);
-- PUT DUTY ON AND DUTY OFF IN A SHIFT BUFFER
-- PACKET_WAIT_PRSCL <=  PWM(INDEX_PACKET);  -- PUT DUTY ON AND
DUTY OFF IN A SHIFT BUFFER
-- INDEX_PACKET <= INDEX_PACKET + 1 ;
ELSE
                        INDEX_PACKET <= 0;
END IF;
-- THIS IS FOR DELIVERING VARIABLE DUTY PWM TO STIMULATION ----
------

END IF;  --// IF (INDEX2 = TAPS) THEN

  IF (INDEX_MISO = MISO_TAPS) THEN
          INDEX_MISO <= 0;
  END IF;  --//  IF (INDEX_MISO = MISO_TAPS) THEN
 END IF;  --//  IF(INDEX < CMD_BITS) THEN
END IF;  --//  IF ((FPGA_SCLK_COUNTER = FPGA_SCLK_PRSCL) and
( FPGA_SCLK_REG = '0')) THEN


IF (FPGA_SCLK_COUNTER < FPGA_SCLK_PRSCL) THEN
-- SPI CLOCK COUNTER : 500KHZ = 2000 NS,  (50NS*40)/50NS,  40/2
= 20
      FPGA_SCLK_COUNTER <= FPGA_SCLK_COUNTER +1;
   ELSE
      FPGA_SCLK_COUNTER <= 0;
      FPGA_SCLK_REG     <= NOT FPGA_SCLK_REG;
  END IF; --//IF (FPGA_SCLK_COUNTER < FPGA_SCLK_PRSCL) THEN
END IF;  --// IF ( OPTRODE_RST_REG = '1') THEN

IF ( OPTRODE_RST_REG = '1') THEN
   FPGA_SCLK_OUT       <= FPGA_SCLK_REG AND '0';
ELSE

   IF( INDEX = 8 OR INDEX = 9 OR INDEX = 10 OR INDEX = 11 OR
INDEX2 =4 OR INDEX2 = 8 ) THEN
 -- DEALY1 : THIS IS FOR GENERATING LOW VOLTAGE FOR SCLK FOR 4
CLOCK CYCLES BETWEEN EACH 2 CMD
                FPGA_SCLK_OUT              <= FPGA_SCLK_REG AND
'0';
                   ELSE
                FPGA_SCLK_OUT            <= FPGA_SCLK_REG;
            END IF;
         END IF;

 IF ( (FPGA_SCLK_COUNTER = FPGA_SCLK_PRSCL) and ( FPGA_SCLK_REG
= '0') and ( INDEX = CMD_BITS)) THEN   -- SENDING OFF MOSI
  FPGA_MOSI_PARALLEL       <= SHIFT_REGISTER_BUFFER(INDEX2);
             END IF;
       FPGA_MOSI_OUT       <= FPGA_MOSI_PARALLEL(INDEX);


      ELSIF(PACKET_FINSHED_TAG = '1') THEN

     PACKET_WAIT_PRSCL_REG_ON       <=    PACKET_WAIT_PRSCL*100;
-- 8000000=800*10000 means 400ms
     PACKET_WAIT_PRSCL_REG_OFF     <=     PACKET_WAIT_PRSCL*100;
-- 4000000=400*10000 means 200ms
-- 6542600=65426*100
--------------------------------------------------------------
--------

             IF (INDEX_SCHEDULE_CMD = '1') THEN    -- LED OFF
                   IF         (        PACKET_WAIT_CNT          =
PACKET_WAIT_PRSCL_REG_OFF ) THEN
                     PACKET_WAIT_CNT        <= 0 ;
                     PACKET_WAIT_MAKER                 <=   NOT
PACKET_WAIT_MAKER ;
                     PACKET_FINSHED_TAG     <= '0';
                   ELSE
                     PACKET_WAIT_CNT         <= PACKET_WAIT_CNT
+ 1;

                     FPGA_SCLK_OUT          <= '0';
                   END IF;
             ELSE
                    IF         (        PACKET_WAIT_CNT         =
PACKET_WAIT_PRSCL_REG_ON ) THEN
                     PACKET_WAIT_CNT        <= 0 ;
                     PACKET_WAIT_MAKER                 <=   NOT
PACKET_WAIT_MAKER ;
                     PACKET_FINSHED_TAG     <= '0';
                   ELSE
                     PACKET_WAIT_CNT         <= PACKET_WAIT_CNT
+ 1;

                     FPGA_SCLK_OUT          <= '0';
                   END IF;
             END IF;
             IF (INDEX_SCHEDULE_CMD = '1') THEN     -- LED OFF
              SHIFT_REGISTER_BUFFER                               <=
SHIFT_REGISTER_BUFFER_LED_OFF ;
             ELSE
              SHIFT_REGISTER_BUFFER                               <=
SHIFT_REGISTER_BUFFER_LED_ON ;
             END IF;
--------------------------------------------------------------
-------------------------------

      END IF; --//IF ( PACKET_FINSHED_TAG = '0') THEN

IF (OPTRODE_CLK_COUNTER < OPTRODE_CLK_PRSCL) THEN
       OPTRODE_CLK_COUNTER <= OPTRODE_CLK_COUNTER +1;
ELSE
       OPTRODE_CLK_COUNTER <= 0;
       OPTRODE_CLK_REG <= NOT OPTRODE_CLK_REG;
END IF;

OPTRODE_CLK_1600KHZ   <=  OPTRODE_CLK_REG;                    --
OPTRODE_CLOCK : 1.6 MHz
END IF; --//IF (FALLING_EDGE ( FPGA_CLOCK )) THEN
END PROCESS SHIFT_REGISTER_PRO;

FPGA_CS_OUT             <= SPI_CS_REG;
OPTRODE_RST             <= OPTRODE_RST_REG;

end architecture_SPI_STIMULATION;
```

Appendix H. PID Phase Shift Analysis

```
%% This is matlab code for PID Kernel magnitude response and
phase response analysis

close all;
clc;
clear all;

**************************************************************
********
% Author:                 Lijuan & Patrick %
% First Version:          18/1/2018 Created by Lijuan %
% Second Version:         PID kernel analysis
% Description:
**************************************************************
********
```

```
close all;
clc;
clear all;

%% impulse response
% Andy Phase Shift Kernel Frequency Response Analysis
%           ____________
%   x(t) |             |  y(t)
%-------->|    h(t)     |------>
%         |____________|
%
%      h(t) = e^(-kt)cos( wt + phase)
%
%
%              cos(phase)s + k*cos(phase) -(2*pi*f)*sin(phase)
%   => H(s) =  ------------------------------------------------
-------
%                          s^2 + k^2 + 2ks + w^2
%
%                                 s^2 + k^2 + 2ks + w^2
% transfer function = -----------------------------------------
------
%                cos (phase)s + k*cos(phase) -(2*pi*f)*sin(phase)
w     = linspace (0,100,512);
k     = 1.25;
f     = 2;
figure;
for phase   = 0:45:360
    G_AD_phase          =     tf([1*cos(phase)     k*cos(phase)-
(2*pi*f)*sin(phase)], [1 2*k k*k+(2*pi*f)^2]);
    [G_AD_phase, P_AD_phase] = bode(G_AD_phase, w);
    subplot(211); plot(w,G_AD_phase(:));
    hold on;
    subplot(212); plot(w,P_AD_phase(:));
    hold on;
end
```

```
% impulse response
% PID Phase Shift Kernel Frequency Response Analysis
%           ____________
%   x(t) |             |  y(t)
%-------->|    h(t)     |------>
```

```
%         |____________|
%                                                               d(delta(t))
%       h(t) = Kp*delta(t)+ Ki*integral(delta(t))+Kd*----------
%                                                                  dt
%                kd*s^2 + Kp*s + Ki
%       H(s) = -------------------------
%                        s
%
%                                   s
%      transfer function = ------------------------
%                              kd*s^2 + Kp*s + Ki

wpid    = linspace (0,100,512);


% kp      = 10.2;
% ki      = 58.8;
% kd      = 123.4;

 kp      = 0.01;
 ki      = 1.6;
 kd      = 0.01;

figure
for ki     = 1.4:0.2:2
% for kp     = 0.01:0.04:0.13
%  for kd      = 123:10:153
G_PID_phase  = tf([1 1], [kd kp ki]);
[G_PID_phase, P_PID_phase] = bode(G_PID_phase, wpid);


%% option1

% subplot(121);plot(wpid,G_PID_phase(:)/100);
% legend('Kp = 0.01','Kp = 0.05','Kp = 0.09','Kp = 0.13')
% hold on;
% subplot(122);plot(wpid,P_PID_phase(:));
% legend('Kp = 0.01','Kp = 0.05','Kp = 0.09','Kp = 0.13')
% hold on;


subplot(121);plot(wpid,G_PID_phase(:)/100);
legend('Ki = 1.4','Ki = 1.6','Ki = 1.8','Ki = 2')
hold on;
subplot(122);plot(wpid,P_PID_phase(:));
legend('Ki = 1.4','Ki = 1.6','Ki = 1.8','Ki = 2')
hold on;

 end
```

```
close all;
clc;
clear all;


****************************************************************
********
% Author:              Lijuan & Patrick %
% First Version:       28/9/2017 Created by Patrick %
% Second Version:      2/10/2017 Created by Lijuan %
%  Description:            Inverse sigmoid function of optical
converter  analysis,  define  photon  flux  in  terms  of  mW/mm2.
Typical range would be 1e-3 to 1e1. The neural response should
be 50% at 0.7mW/mm2

****************************************************************
********


% ----------------------------------------------------------------
-------
%% Sigmoid function
% ----------------------------------------------------------------
-------
% define photon flux in terms of mW/mm2. Typical range would be
1e-3 to 1e1.

% the neural response should be 50% at 0.7mW/mm2
f = 1e-6;
for n =1:70
    f = f * 1.4;
    flux(n)  = f;
end

Vt = 1.45;

% sigmoid function in the logarithmic domain
Response = (Vt*flux)./(1+(Vt*flux));

% Return the neuron response at 0.7mW/mm2.
% The result should be 0.5
Neural_Response_at_threshold = Response(40)

figure;
```

```
semilogx(flux,Response,'k');
title('ChR2 neural response vs light intensity');
set(gcf, 'color', 'w')
xlim([1e-4 1e2])
xlabel('light intensity (mw/mm^{2})');
ylabel('Normalised neural response');

% -------------------------------------------------------------
-------------
%% Inverse Sigmoid function
% -------------------------------------------------------------
-------------

%settings
R_threshold = 0.7;          % Determines the normalised neural
threshold
                            % for which to intervene. This needs
to take
                            %  into  account  many  variables
including genetic
                            % expression, LED light intensity
and optical
                            % traversal through the tissue. It
will
                            %   ultimately   have   to   be
experimentally
                            % calibrated for each LED.
minPWM = 0.5;                 % minimal PWM time (in ms) for a
stimulation
                            % frame. This will be related to the
                            % intervention frequency e.g. 100Hz,
which may
                            % be separate to recording frequency
maxPWM = 10;                  % maximum PWM time (in ms) for a
stimulation
                            % frame This will be related to the
                            % intervention frequency e.g. 100Hz,
which may
                            % be separate to recording frequency

% Define response in terms of maximum possible response. i.e.
between 0 -
% 1;
Response = 0:0.01:1;

% inverse sigmoid function with light flux in terms of mW/mm2
for 10ms
lightFlux = (Response ./(Vt * (1-Response)));

% The neural response above is calibrated as the average plateau
response
% resulting from continuous illumination. HOWEVER, we are
```

```
interested in
% pulsed illumination with a defined PWM between 0.1 - 10ms
(assuming 100Hz
% sampling - or at least 100Hz intervention).
PWM_time = lightFlux * R_threshold;

for n = 1: length(PWM_time)

    % for neural responses resulting in ultra-short PWM times,
simply set
    % the output to zero. This is effectively a lower end
threshold
    if PWM_time(n) < minPWM; PWM_time(n) = 0; end

    % If the required light intensity is too high, the PWM time
will exceed
    % the maximum time allowable within a frame. Thus this needs
to
    % saturated to that maximum time.
    if PWM_time(n) > maxPWM; PWM_time(n) = maxPWM; end

end

% write LUT to file
PWMLUT(:,1) = Response';
PWMLUT(:,2) = round(PWM_time,1)';
csvwrite('pwm_LUT.csv',PWMLUT);

% Plot figures
figure;
semilogy(Response, lightFlux);
title('Light flux vs neural response');
%set(gcf, 'color', 'w')
xlabel('Normalised neural response');
ylabel('light intensity (mw/mm^{2})');

figure;
semilogy(Response, PWM_time);
title('Light PWM vs neural response');
%set(gcf, 'color', 'w')
xlabel('Normalised neural response');
ylabel('PWM time (ms)');

figure;
subplot(121);
semilogy(Response, lightFlux);
title('Light flux vs neural response');
%set(gcf, 'color', 'w')
xlabel('Normalised neural response');
ylabel('light intensity (mw/mm^{2})');
subplot(122);
semilogy(Response, PWM_time);
```

```
title('Light PWM vs neural response');
%set(gcf, 'color', 'w')
xlabel('Normalised neural response');
ylabel('PWM time (ms)');
```

Appendix J. Neuron Mass Modelling

```
*************************************************************
********
% Author:                 Lijuan & Patrick %
% First Version:          2/6/2016 Created by Lijuan %
% Second Version:         14/10/2017 Created by Lijuan %
% Description:            Neuron Mass Modelling with PID feedback

*************************************************************
********
```

```
%%  Neural Mass Model Transfer Function
% Neunal Mass Model Transfer Function in jw domain
% Figure 3-6 Plotting and Figure 3-9

clear all
close all
clc

%%  Neunal Mass Model Transfer Function
% He     = 3.25;
% Te     = 0.0108;
% Ge     = tf(  [0 He*Te], [Te*Te 2*Te 1] );    % this is laplace
transform of He
% figure;
% bode(Ge);
%
% Hi     = 22;
% Ti     = 0.02;
% Gi     = tf(  [0 Hi*Ti], [Ti*Ti 2*Ti 1] );    % this is laplace
transform of Hi
% figure;
% bode(Gi);
%

% C1     = 135
% C2     = 0.8*C1;
% C3     = 0.25*C1;
% C4     = 0.25*C1;
% s      = tf('s');
% v0     = 6;
% e0     = 2.5
% r      = 0.56
% Ks     = e0*r/2;
%  Gnmm    =  Ge/(1+Ks^2*Ge*(C3*C4*Gi  -  C1*C2*Ge));  %  laplace
transform of NMM

%  Gnmm    =  Ge/(1+Ks^2*Ge*(C3*C4*Gi  -  C1*C2*Ge));  %  laplace
transform of NMM
%
% figure;
% impulse(Gnmm);
%
% figure;
% step(Gnmm);

% Neunal Mass Model Transfer Function in jw domain
for He_w    = 5:2:9

w             = 0:0.01:500;
%He_w          = 4.5
% Ks_w          = 2.5*0.56*2./(j*w);
Ks_w          = 2.5*0.56/2;
```

```
Te_w          = 0.0108;
Ge_w            = He_w*Te_w./[(j*w*Te_w + 1).^2];      % this is
laplace transform of He

Hi_w          = 17;
Ti_w          = 0.02;
Gi_w            = Hi_w*Ti_w./[(j*w*Ti_w + 1).^2];      % this is
laplace transform of Hi

C1            = 135
C2            = 0.8*C1;
C3            = 0.25*C1;
C4            = 0.25*C1;
v0            = 6;
e0            = 2.5
r             = 0.56
Ks_w          = e0*r/2;

Gnmm_w                 =   Ge_w./(1+Ks_w.*Ks_w.*Ge_w.*(C3*C4*Gi_w  -
C1*C2*Ge_w)); % laplace transform of NMM

Real_NMM     = real(Gnmm_w);
Imag_NMM     = imag(Gnmm_w);

%
figure;subplot(211);plot(log10(w),abs(Gnmm_w));subplot(212);pl
ot(log10(w),phase(Gnmm_w));set(gca, 'xscale', 'log');
%      figure;subplot(211);plot(log10(w),sqrt(Real_NMM.^2      +
Imag_NMM.^2  ));subplot(212);plot(log10(w),atan(Imag_NMM.*(Real
_NMM).^(-1)));set(gca, 'xscale', 'log');

%% Ki, Kp
% den_Gnmm_w   = Real_NMM.^2 + Imag_NMM.^2;
den_Gnmm_w   = Real_NMM.^2 + Imag_NMM.^2;
Ki           = -w.*Imag_NMM./den_Gnmm_w ;
Kp           = -Real_NMM ./den_Gnmm_w;
Kd           = Imag_NMM./(w.*den_Gnmm_w) ;
%       figure(2);       subplot(121);plot(Kp,Ki,'linewidth',2);
xlabel('Kp');ylabel('Ki');title('PI Controller');
%                      subplot(122);plot(Kp,Kd,'k','linewidth',2);
xlabel('Kp');ylabel('Ki');title('PD Controller');

%subplot(221);plot(Kp,Ki);         legend('He=5','He=7','He=9');
xlabel('Kp');ylabel('Ki');title('PI Controller');hold on;
subplot(121);plot(Kp,Kd);          legend('He=5','He=7','He=9');
xlabel('Kp');ylabel('Kd');title('PD Controller');hold on;
end
%
for Hi_w     = 15:2:19
w             = 0:0.01:500;
He_w          = 3;
% Ks_w            = 2.5*0.56*2./(j*w);
```

```
Ks_w          = e0*r/2;
Te_w          = 0.0108;
Ge_w            = He_w*Te_w./[(j*w*Te_w + 1).^2];      % this is
laplace transform of He

% Hi_w        = 17;
Ti_w          = 0.02;
Gi_w            = Hi_w*Ti_w./[(j*w*Ti_w + 1).^2];      % this is
laplace transform of Hi

C1           = 135
C2           = 0.8*C1;
C3           = 0.25*C1;
C4           = 0.25*C1;
Gnmm_w                = Ge_w./(1+Ks_w.*Ks_w.*Ge_w.*(C3*C4*Gi_w  -
C1*C2*Ge_w)); % laplace transform of NMM

Real_NMM     = real(Gnmm_w);
Imag_NMM     = imag(Gnmm_w);

%
figure;subplot(211);plot(log10(w),abs(Gnmm_w));subplot(212);pl
ot(log10(w),phase(Gnmm_w));set(gca, 'xscale', 'log');
%      figure;subplot(211);plot(log10(w),sqrt(Real_NMM.^2      +
Imag_NMM.^2  ));subplot(212);plot(log10(w),atan(Imag_NMM.*(Real
_NMM).^(-1)));set(gca, 'xscale', 'log');

%% Ki, Kp
% den_Gnmm_w  = Real_NMM.^2 + Imag_NMM.^2;
den_Gnmm_w   = Real_NMM.^2 + Imag_NMM.^2;

Ki           = -w.*Imag_NMM./den_Gnmm_w ;
Kp           = -Real_NMM ./den_Gnmm_w;
Kd           = Imag_NMM./(w.*den_Gnmm_w)  ;

%
% Ki           = -w.*Imag_NMM ;
% Kp           = Real_NMM ;
% Kd           = -1000*Imag_NMM./w ;

%subplot(223);plot(Kp,Ki);      legend('Hi=15','Hi=17','Hi=19');
xlabel('Kp');ylabel('Ki');title('PI Controller');hold on;
subplot(122);plot(Kp,Kd);       legend('Hi=15','Hi=17','Hi=19');
xlabel('Kp');ylabel('Kd');title('PD Controller');hold on;
end
```

```
%%  Figure3-7 Plotting
clear all
close all
clc
%%  PARAMETER SET UP
tstart              = 0;                        % START TIME
tend                = 8 ;                       % END TIME
tinterp             = 1;                        % NORMALIZED STEO
SIZE
h                   = 0.001;                    % STEP SIZE
T                  = tstart:h*tinterp:tend;     % STIMULATION TIME
Nl                  = length(T);                % TIME SIMULATION
NUMBERS
nsq                 = length(T);
y0                   = zeros(6, Nl);              % SIX VARAIBLES
FOR DIFFERENTIAL EQUATIONS PAIRS

He                 = 7;                   % He : average excitory
synaptic gain
Hi                 = 22;                  % Hi : average inhitory
synaptic gain

%%  OPEN LOOP
tic

[ Y ]               = runSheetPRamp_LJ(y0,T,He,Hi);
y_1                 = Y(1,:);
y_2                 = Y(2,:);
y_3                 = Y(3,:);
y_4                 = Y(4,:);
y_5                 = Y(5,:);
y_6                 = Y(6,:);
LFP                 = y_3 - y_5 ;


%%  CLOSED LOOP
Kp                   = 90;                    % proportional term
Kp
```

```
Ki                   = 2;                           % Integral term
Ki
Kd                 = 0;                    % derivative term    Kd
y_initial          = Y;

[Y1]               = runSheetPRamp_PID_LJ(y0,T,Kp,Ki,Kd,He,Hi);
y1_1               = Y1(1,:);
y1_2               = Y1(2,:);
y1_3               = Y1(3,:);
y1_4               = Y1(4,:);
y1_5               = Y1(5,:);
y1_6               = Y1(6,:);
LFP1               = y1_3 - y1_5 ;
toc

%% plotting
T2                 = tend:h*tinterp:tend*2;
T_CLOSED            = tstart:h*tinterp:tend*2;      % STIMULATION
TIME
LFP_CLOSED_LOOP   = [LFP,LFP1*30];
figure;     subplot(121);     plot(T,LFP);         legend('without
PI');xlabel('time/s');ylabel('y(t)(mv)')
        subplot(122);      plot(T2,LFP1,'k');      legend('with
PI');xlabel('time/s');ylabel('y(t)(mv)')
figure;
plot(T_CLOSED,LFP_CLOSED_LOOP(1:length(T_CLOSED)));xlabel('tim
e/s');ylabel('y(t)(mv)')


%%  Figure3-8 Plotting
clear all
close all
clc

%%  PARAMETER SET UP
tstart             = 0;                          % START TIME
tend               = 8 ;                         % END TIME
tinterp            = 1;                          % NORMALIZED STEO
SIZE
h                  = 0.001;                      % STEP SIZE
T                 = tstart:h*tinterp:tend;      % STIMULATION TIME
Nl                 = length(T);                  % TIME SIMULATION
NUMBERS
nsq                = length(T);
y0                  = zeros(6, Nl);                % SIX VARAIBLES
FOR DIFFERENTIAL EQUATIONS PAIRS

He                   = 7;                           % He: average
excitatory synaptic gain
Hi                   = 17;                          % Hi: average
inhitatory synaptic gain
```

```
%%  OPEN LOOP
tic
[ Y ]              = runSheetPRamp_LJ(y0,T,He,Hi);

y_1                = Y(1,:);
y_2                = Y(2,:);
y_3                = Y(3,:);
y_4                = Y(4,:);
y_5                = Y(5,:);
y_6                = Y(6,:);
LFP                = y_3 - y_5 ;


%%  CLOSED LOOP
Kp                  = 25;                          % proportional term
Kp
Ki                    = 0;                            % Integral term
Ki
Kd                   = 2;                          % derivative term
Kd
y_initial          = Y;

[Y1]               = runSheetPRamp_PID_LJ(y0,T,Kp,Ki,Kd,He,Hi);
y1_1               = Y1(1,:);
y1_2               = Y1(2,:);
y1_3               = Y1(3,:);
y1_4               = Y1(4,:);
y1_5               = Y1(5,:);
y1_6               = Y1(6,:);
LFP1               = y1_3 - y1_5 ;
toc

%% plotting
T2                 = tend:h*tinterp:tend*2;
T_CLOSED            = tstart:h*tinterp:tend*2;      % STIMULATION
TIME
LFP_CLOSED_LOOP   = [LFP,LFP1*10-5];
figure;     subplot(121);     plot(T,LFP);        legend('without
PD');xlabel('time/s');ylabel('y(t)(mv)')
        subplot(122);      plot(T2,LFP1,'k');      legend('with
PD');xlabel('time/s');ylabel('y(t)(mv)')
figure;
plot(T_CLOSED,LFP_CLOSED_LOOP(1:length(T_CLOSED)));xlabel('tim
e/s');ylabel('y(t)(mv)')
%% Library 1: Open loop Stimulation

function  [ Y ]=runSheetPRamp_LJ(y0,T,A,B)
                                                         % y0: 12006*1
                                                         % T : 1*2001

%% Parameter set up
e0                          =  2.5 ;                  % maximum firing
rate of the neural population
```

```
r                    =  0.56 ;                 % steepmess
v0                   =  6   ;                  % firing rate

a                     =  100;                   % a parameter of
PSP is inversely proportional to the duration of PSP
b                     =  50;                    % b parameter of
PSP is inversely proportional to the duration of PSP

C                     =  135;                    % C is to vary
under different physiolodical constrants
C1                   =  C;                     % C1 accounts for
synaptic depletion
C2                   =  0.8*C;
C3                   =  0.25*C;
C4                   =  0.25*C;

%%  white noise
% the random white noise input p(t) will have an amplitude
varying
% between 120 and 320 pulses per second

sigma                = 2.4;                   % mean value : 2.4
standard_deviation    = 2  ;                          % standard
deviation : 2
p                                                      =  sigma  +
standard_deviation.*randn(length(T),1);
h                    = 0.001;                  % STEP SIZE
Nl                   = length(T);              % 2001
nsq                  = length(T);

Y                   = y0;                     % y0=zeros(6*Nl,1);
                                               %  SIX  VARAIBLES
FOR DIFFERENTIAL EQUATIONS PAIRS
%for i = 1 : Nl
for i = 1 : Nl-1

% y_1                  = Y(1:nsq,i);           % 1:nsq = 1:2001,
i
% y_2                   = Y(nsq+1:2*nsq,i);
% y_3                   = Y(2*nsq+1:3*nsq,i);
% y_4                   = Y(3*nsq+1:4*nsq,i);
% y_5                   = Y(4*nsq+1:5*nsq,i);
% y_6                   = Y(5*nsq+1:6*nsq,i);

y_1                  = Y(1,i);             % 1:nsq = 1:2001, i
y_2                  = Y(2,i);
y_3                  = Y(3,i);
y_4                  = Y(4,i);
y_5                  = Y(5,i);
y_6                  = Y(6,i);

dy_1dt                                                         =
y_2;                                                                 %
y(1)<-y0, y(2)<-y3
dy_2dt                  = A*a*(2*e0./( 1 + exp( r.*(v0- (y_3 -
y_5)))) - 2*a*y_2-a^2*y_1;
dy_3dt                                                               =
y_4;                                                                 %
y(3)<-y1, y(4)<-y4
dy_4dt                = A*a*(p(i,:) + C2*(2*e0./( 1 + exp( r.*(v0-
( C1*y_1)))))) - 2*a*y_4-a^2*y_3;
dy_5dt                                                               =
y_6;                                                                 %
y(5)<-y2, y(6)<-y5
dy_6dt                   = B*b*(C4*  (2*e0./( 1  + exp( r.*(v0-
( C3*y_1 )))))) - 2*b*y_6 -b^2*y_5;

%Y(:,i+1)                                             =   Y(:,i)    +
h*[dy_1dt;dy_2dt;dy_3dt;dy_4dt;dy_5dt;dy_6dt];
increase                                                             =
h*[dy_1dt;dy_2dt;dy_3dt;dy_4dt;dy_5dt;dy_6dt]; % 6*1 matrix
Y(:,i+1)              = Y(:,i) + increase ;

% LFP                 = y1 - y2 = y(3)-y(5);
end

%Y                     = single(Y)';
end
```

```
%% Library 2: Closed loop Stimulation

function    [Y] = runSheetPRamp_PID_LJ(y0,T,Kp,Ki,Kd,A,B)
%% parameter set up
e0                      =  2.5 ;                       % maximum firing
rate of the neural population
r                      =  0.56 ;                      % steepmess
v0                     =  6  ;                        % firing rate

a                      =  100;
b                      =  50;

C                      =  135;
C1                     =  C;
C2                     =  0.8*C;
C3                     =  0.25*C;
C4                     =  0.25*C;

%% white noise
% the random white noise input p(t) will have an amplitude
varying
% between 120 and 320 pulses per second
sigma                  = 2.4;                        % mean value : 2.4
standard_deviation   = 2 ;                                 % standard
deviation : 2
h                      = 0.001;                       % STEP SIZE
Nl                     = length(T);
nsq                    = length(T);
Y                      = y0;
p                 = sigma + standard_deviation.*randn(length(T),1);

desired                = 0;                           % desired output,
or reference point

%%
for i = 1 : Nl-1
% y_1                    = Y(1:nsq,i);
% y_2                    = Y(nsq+1:2*nsq,i);
% y_3                    = Y(2*nsq+1:3*nsq,i);
% y_4                    = Y(3*nsq+1:4*nsq,i);
% y_5                    = Y(4*nsq+1:5*nsq,i);
% y_6                 = Y(5*nsq+1:6*nsq,i);

y_1                  = Y(1,i);           % 1:nsq = 1:2001, i
y_2                  = Y(2,i);
y_3                  = Y(3,i);
y_4                  = Y(4,i);
y_5                  = Y(5,i);
y_6                  = Y(6,i);
%% PID Controller
y_t(:,i)            =  y_3 - y_5;                               %
local field potential
Error(:,i+1)         =  desired - y_t(:,i);                     %
error with reference equal to zero
Prop(:,i+1)          =  Error(:,i+1);                            %
error of proportional term
Der(:,i+1)           = (Error(:,i+1) - Error(:,i));             %
derivative of the error
Int(:,i+1)           = (Error(:,i+1) + Error(:,i));             %
integration of the error
I(:,i+1)             = Int(:,i+1);                               %
the sum of the integration of the error
PID(:,i+1)          = Kp*Prop(:,i) + Ki*I(:,i+1)+ Kd*Der(:,i);  %
the three PID terms

%%  NMM Model
dy_1dt              = y_2;                   % y(1)<-y0 , y(2)<-
y3
dy_2dt                  = A*a*(2*e0./( 1 + exp( r.*(v0- (y_3 -
y_5))))) - 2*a*y_2-a^2*y_1;
dy_3dt              = y_4;                   % y(3)<-y1,  y(4)<-
y4
dy_4dt                 = A*a*(p(i,:) +  PID(:,i+1) + C2*(2*e0./( 1
+ exp( r.*(v0- ( C1*y_1)))))) - 2*a*y_4-a^2*y_3;
dy_5dt                 = y_6;                                    %
y(5)<-y2,  y(6)<-y5
dy_6dt                    = B*b*(C4*  (2*e0./( 1  +  exp(  r.*(v0-
( C3*y_1 )))))) - 2*b*y_6 -b^2*y_5;


%Y(:,i+1)                                          =   Y(:,i)   +
h*[dy_1dt;dy_2dt;dy_3dt;dy_4dt;dy_5dt;dy_6dt];
increase                                                         =
h*[dy_1dt;dy_2dt;dy_3dt;dy_4dt;dy_5dt;dy_6dt]; % 6*1 matrix
Y(:,i+1)             = Y(:,i) + increase ;

% LFP                = y1 - y2 = y(3)-y(5);
end

%Y                     = single(Y)';
end
```