7-7-2020

Spiking Neural Networks: Modification and Digital Implementation

Moslem Heidarpur
University of Windsor

SPIKING NEURAL NETWORKS: MODIFICATION AND DIGITAL IMPLEMENTATION

by

Moslem Heidarpur

A Dissertation
Submitted to the Faculty of Graduate Studies through the
Department of Electrical and Computer Engineering in Partial Fulfillment
of the Requirements for the Degree of Doctor of Philosophy at the
University of Windsor

Windsor, Ontario, Canada

2020

Spiking Neural Networks: Modification and Digital Implementation

by

Moslem Heidarpur

APPROVED BY:

N. Dimopoulos, External Examiner
University of Victoria

R. Riahi
Department of Mechanical, Automotive and Materials Engineering

---

R. Rashidzadeh
Department of Electrical and Computer Engineering

M. Khalid
Department of Electrical and Computer Engineering

A. Ahmadi, Co-Advisor
Department of Electrical and Computer Engineering

M. Ahmadi, Co-Advisor
Department of Electrical and Computer Engineering

May 1, 2020

Declaration of Co-Authorship / Previous Publication

I hereby declare that this thesis incorporates material that is result of joint research, as follows:

Chapter 2 of this thesis was co-authored with Dr. Mostafa Rahimi Azghadi, Dr. Arash Ahmadi, and Prof. Majid Ahmadi. The design, simulation, implementation, analysis of results and writing the manuscript were performed by the author whereas Dr. Mostafa Rahimi edited the manuscript, Dr. Arash Ahmadi and Prof. Majid Ahmadi supervised the research.

Chapter 3 of this thesis was co-authored with Dr. Parvin Khosravifar, Dr. Arash Ahmadi, and Prof. Majid Ahmadi. In all cases, the design, simulation, implementation, analysis of results and writing the manuscript were performed by the author while Dr. Parvin Khosravifar provided medical consultant, Dr. Arash Ahmadi and Prof. Majid Ahmadi supervised the research.

Chapter 4 of this thesis was co-authored with Dr. Arash Ahmadi, and Prof. Majid Ahmadi. The circuit designed, simulated and analyzed in term of performance and resources by the author while Dr. Arash Ahmadi, and Prof. Majid Ahmadi provided supervision and edited the manuscript.

Chapter 5 of this thesis was co-authored with Dr. Arash Ahmadi, and Prof. Majid Ahmadi. The design, simulation, implementation, analysis of results and writing were

performed by the author. The contribution of Dr. Arash Ahmadi and Prof. Majid Ahmadi was to oversee the research, provide feedback and give comments to improve the manuscript.

I am aware of the University of Windsor Senate Policy on Authorship and I certify that I have properly acknowledged the contribution of other researchers to my thesis, and have obtained written permission from each of the co-authors to include the above materials in my thesis.

I certify that, with the above qualification, this thesis, and the research to which it refers, is the product of my own work.

This thesis includes 4 original papers that have been previously published/accepted in peer reviewed journals and conferences, as follows:

Thesis Chapter	Publication Title	Publication status
Chapter 2	M. Heidarpur, A. Ahmadi, M. Azghadi and M. Ahmadi, "CORDIC-SNN: On-FPGA STDP Learning and Izhikevich Neuron Model," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 66, no. 7, pp. 2651-2661, Jul. 2019	Published
Chapter 3	M. Heidarpur, P. Khosravifar, A. Ahmadi and M. Ahmadi, "CORDIC-Astrocyte: A Tripartite Glutamate-IP3-Ca2+ Interaction Dynamics on FPGA," in IEEE Transactions on Biomedical Circuits and Systems, vol. 14, no. 1, pp. 36-47, Feb. 2020	Published
Chapter 4	M. Heidarpur, A. Ahmadi and M. Ahmadi, "An efficient digital implementation of a detailed biological model of astrocyte , in Advances in Computational Intelligence, Springer International Publishing, Switzerland, Cham, 2019, pp. 857-868.	Published
Chapter 5	M. Heidarpur, A. Ahmadi, M. Ahmadi, "Time step impact on performance and accuracy of izhikevich neuron: Software simulation and hardware implementation,"2020 IEEE International Symposium on Circuits & Systems (ISCAS)	Accepted

I certify that I have obtained a written permission from the copyright owners to include the above published materials in my thesis. I certify that the above material describes work completed during my registration as graduate student at the University of Windsor.

I declare that, to the best of my knowledge, my thesis does not infringe upon anyone's copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my

thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that I have included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Canada Copyright Act, I certify that I have obtained a written permission from the copyright owners to include such materials in my thesis.

I declare that this is a true copy of my thesis, including any final revisions, as approved by my thesis committee and the Graduate Studies office, and that this thesis has not been submitted for a higher degree to any other University or Institution.

Abstract

Real-time large-scale simulation of biological systems is a challenging task due to nonlinear functions describing biochemical reactions in the cells. Being fast, cost and power efficient alongside of capability to work in parallel have made hardware an attractive choice for simulation platform.

This thesis proposes a neuromorphic platform for online Spike Timing Dependant Plasticity (STDP) learning, based on the COordinate Rotation DIgital Computer (CORDIC) algorithms. The implemented platform comprises two main components. First, the Izhikevich neuron model is modified for implementation using the CORDIC algorithm and simulated to ensure the model accuracy. Afterwards, the model was described as hardware and implemented on Field Programmable Gate Array (FPGA). Second, the STDP learning algorithm is adapted and optimized using the CORDIC method, synthesized for hardware, and implemented to perform on-FPGA online learning on a network of CORDIC Izhikevich neurons to demonstrate competitive Hebbian learning.

The implementation results are compared with the original model and state-of-the-art to verify accuracy, effectiveness, and higher speed of the system. These comparisons confirm that the proposed neuromorphic system offers better performance and higher accuracy while being straightforward to implement and suitable to scale.

New findings show that astrocytes are important parts of the information processing in brain and believed to be responsible for some brain diseases such as Alzheimer

and Epilepsy. Astrocytes generate Ca^{2+} waves and release neuro-transmitters over a large area. To study astrcoytes, one need to simulate large number of biologically realistic models of these cells alongside neuron models. Software simulation is flexible but slow.

This thesis proposes a high-speed and low-cost digital hardware to replicate biological-plausible astrocyte and glutamate release mechanism. The nonlinear terms of these models were calculated using high-precision and cost-efficient algorithms. Subsequently, the modified models were simulated to study and validate their functions. Several hardware were developed by setting different constraints to investigate trade-offs and achieve best possible design. As proof of concept, the design was implemented on a FPGA device. Hardware implementation results confirmed the ability of the design to replicate biological cells in detail with high accuracy. As for performance, the proposed design turned out to be far more faster and area efficient than previously published works that targeted digital hardware for biological-plausible astrocytes.

Spiking neurons, the models that mimic the biological cells in the brain, are described using ordinary differential equations. A common method to numerically solve these equations is Euler's method. An important factor that has a significant impact on the performance and cost of the hardware implementation or software simulation of spiking neural networks and yet its importance has been neglected in the published literature, is the time step in Euler's method. In this thesis, first the Izhikevich neuron's accuracy as a function of the time step was measured. It was uncovered that the threshold time step that Izhikevich neuron becomes unstable is an exponential function of the input current.

Software simulation performance, including total computational time and memory usage were compared for different time steps. Afterwards, the model was synthesized and implemented on the FPGA. Hardware performance metrics such as speed, area and power consumption were measured for each time step. Results indicated that time step has a negative linear effect on the performance. It was concluded that by determining maximum input current to the neuron, larger time steps comparable to those used in the previous works could be employed.

Lovingly dedicated to

my wife Parvin,

for her love, support and faith

Acknowledgment

Foremost, I would like to express my sincere gratitude to my advisor Professor Majid Ahmadi for the continuous support of my Ph.D study and research, for his patience, motivation, enthusiasm, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better advisor and mentor for my Ph.D study.

My deep gratitude also goes to my co-advisor, Doctor Arash Ahamdi for providing me with invaluable guidance throughout this research. His dynamism, vision, sincerity and motivation have deeply inspired me.

Moreover, I would like to thank the rest of my thesis committee: Doctor Rashid Rashidzadeh, Doctor Mohammed Khalid and Doctor Reza Riahi for their encouragement and insightful comments. I am extending my thanks to students of Research Centre for Integrated Microsystems (RCIM) for their support during my research work.

Last but not the least, I would like to thank my parents for their unconditional love, support and spiritual guidance.

Contents

List of Figures

List of Tables

List of Abbreviations

AER	Address Event Representation
AFM	Amplitude Frequency Modulation
AM	Amplitude Modulation
ANN	Artifical Neural Network
ASIC	Application Specific Integrated Circuit
CCF	Computaional Cost Factor
CORDIC	COordinate Rotation DIgital Computer
EIF	Exponential Integrate and Fire
ER	Endoplasmic Reticulum
FM	Frequency Modulation
FPGA	Field Programmable Gate Array
FSM	Finite State Machine
GPF	Global Performance Factor
GPU	Graphics Processing Unit

HDL	Hardware Description Language
IF	Integrate and Fire
IP3	Inositol trisPhosphate
LUT	Look Up Table
MD	Maximum Deviation
NRMSD	Root Mean Square Deviation
NRMSE	Root Mean Square Error
ODE	Ordinary Differential Equation
PWL	Piece-Wise Linear
SNN	Spiking Neural Network
STDP	Spike Time Dependent Plasticity
UART	Universal Asynchronous Receiver Transmitter
USB	Universal Serial Bus
VHDL	Very high speed integrated circuits Hardware Description Language
VLSI	Very Large Scale Integration
mGluRs	metabotropic Glutamate Receptors

Introduction

First Section of this Chapter presents a brief summary of Neuromorphic systems and their properties and applications, introduces Spiking Neural Network (SNN) as the third generation of neural networks and its comparison with previous generations. The second Section discuses the current challenges and problems in the field of neuromorphic engineering and SNNs and a literature review for state-of-the-art solutions. Finally, Section 1.3 presents the proposed solutions.

1.1 Background

1.1.1 Neuromorphic Systems

During the last decades, researchers have been trying to replicate and study brain for both medical and information processing applications. Some medical objectives include understanding neurological psychiatric diseases [1, 2], simulating drug treatment [3] alongside designing brain computer interfaces for the people with sensory, motor and cognitive disabilities [4, 5]. However, this study is most important from information processing point of view, where it can eventually lead to a new generation of computational devices [6]. Resembling to brain, such processors expected to be intelligent, low power and fast [7, 8, 9, 10]. Nevertheless, contrary to nowadays computers, brain inspired systems are tolerant to both hardware and data failures [11].

These neuromorphic systems include a large number of neurons, synapses and their interconnecting structure on hardware [12]. Highly parallel, energy efficient, fault tolerant, and compact, these systems promise alternative devices for solving engineering problems [13] and powerful tools to understand properties of biological neural networks [14]. Several such systems have already been introduced and used [15, 16, 17, 18, 19] for various applications such as pattern recognition, signal processing, and autonomous robots [20, 21, 22, 23, 24, 25].

1.1.2 Spiking Neural Network

Artificial Neural Networks (ANNs) are simplified imitations of biological neurons replicating basic elements of the nervous system. Based on the in/out signal types, there are three classes of neural networks. The first one represents neurons as perceptrons, which are composed of two parts: sum and threshold. If the sum of weighted inputs reaches a threshold, the output will be one, otherwise it will be zero. This is useful for Boolean function implementation. In the second class, neurons activation functions are Sigmoids. This type also consists of two parts: sum and sigmoid evaluator. In addition to digital functions, this vlass is also capable of approximating analog functions as well.

Spiking neural networks are the third generation of neural networks where neuron models communicate by sequences of spikes. SNNs use more biologically meaningful neuron models and incorporates spikes and spatiotemporal data processing schemes [26, 27, 28]. The neurons of this class have three computational stages: sum of inputs, integrate the sum over time, and a threshold. In this model, when the membrane potential reaches the threshold, the neuron fires a spike and resets its potential. In general, these types of spiking neuron models are mathematically described by Ordinary Differential Equations (ODEs). The normalized firing rate in a period of time, also called the rate coding, determines the output of the network [29].

Although it is considered a strong computational model, there are other processes in the brain, such as classifying visual objects, that are too fast to be explained using only rate coding algorithms [30]. Both experimental and theoretical research suggests

that sequence timing of the signals must be considered as another essential parameter of neural coding [31, 32].

SNNs have become an interesting subject for a variety of research fields including computational neuroscience, cognitive computing, artificial intelligence, and neuromorphic. There is a wide range of applications related to this topic such as pattern recognition, data processing, medical diagnoses, autonomous robotics, game playing, etc [33].

Current neuromorphic research has led to the development of a plethora of models to mimic real neurons with different levels of abstraction in biological details. Biologically-plausible models, such as Hodgkin Huxley [34] describe cellular phenomena and properties of the individual biological components. Such low-level models, impose more computation cost, making it difficult to simulate large-scale networks. On the other hand, biologically-inspired models such as Izhikevich [35] and models in [36, 37, 38, 39, 40], aim to mimic the biological neurons to the best degree of accuracy. Such models can reproduce most of the firing patterns of real neurons and are easier to couple to other spike-oriented units.

Moreover, high-level Integrate and Fire (IF) [41] is another computationally efficient neural model, but cannot exhibit many essential features of the biological neurons as observed in experiments [42]. As far as neuromorphic computing is concerned, simpler models are cheaper, faster, and more energy efficient. Nevertheless, the choice of models depends on the application of the device to be designed. To perform computations with SNNs only a simple IF or Exponential IF (EIF) may be enough to act as a thresholding box. However, for research in neuroscience, biologically plausible models have higher flexibility in mimicking biology.

After selecting the neuron model, a proper spiking neural network topology should be chosen. This depends on a number of factors such as the level of abstraction, the targeted application, available hardware, and the learning algorithm. A variety of spiking network topologies have been used in neuromorphic systems such as recurrent [43], feed-forward [44], winner-take-all [45], and probabilistic [46]. Subsequently, the SNN learning method should be selected based on factors such as network topol-

ogy, whether the learning should be on-chip or off-chip, be supervised or unsupervised, etc.

1.2 Problem Statement

1.2.1 The Challenge

One approach to brain study is dividing it into two hierarchical sub levels of components and architectural. Components level, which is the lowest, concerns studying and mathematically modelling properties of cells and the way they are interconnected and interact with each other.

In architectural level, brain cells are connected together through their different interaction mechanisms and their systematic behaviour is analyzed. This is useful both to study biological functions and disorders as well as discovering computational algorithms underlying information processing, learning, memory, etc.

However, high complexity of such systems, due to large number of cells [47] and numerous communication pathways, make such simulation a difficult task. The performance of such systems at a higher level depends on the neuron, synapse and learning models and at a lower level on the circuits realizing such units. Therefore, optimization of SNN is important to increase performance and reduce cost of implementation.

1.2.2 Implementation Platforms

Neuron models could be either implemented in Very Large Scale Integration (VLSI) systems or simulated by computer (software) codes. Computer simulations have the advantage of flexibility, full control over model parameters, could be clock driven (synchronous) or event driven (asynchronous), and is capable of simulating and connecting different neuron models, but even for medium scale networks it falls behind the real time simulation [48]. Since the real power of the brain is believed to stem from massive parallelism, the classic fetch and decode computers do not seem like a very

good choice. Many projects have been simulating neural networks on supercomputers [49, 50, 51] or in some cases software has been developed for PCs [52, 53, 54], although supercomputers are expensive and are not available for the majority of researchers.

Unlike general purpose architectures, hardware simulators include well developed and dedicated functional units for simulation capable of working in parallel, which is very desirable in the simulations of SNNs. Such hardware could be categorized into analog and digital neuromorphic systems. Analog systems, in general, are: more energy efficient, use smaller die area, continuous in time and therefore have a higher speed, could easily communicate with real world signals taking advantage of the fact that actual neurons are analog. Having said that, analog implementations generally suffer from some limitations: analog design is not very straightforward and any change in a parameter of the model may leads to redesigning the whole hardware. They are also very sensitive to process variability and suffer from noise. Furthermore, analog spiking neurons require rather large capacitance, could have a limited number of post-synaptic connections due to circuit fan-out constraints, are not sufficiently reliable [55], and mostly rely on digital parts for full functionality [56]. Some of these analog implementations could be find in the references: [57, 58, 59, 60, 61, 62, 63].

Digital systems, on the other hand, are more power consuming, require larger silicon area, discrete in the time, and rather slower; but, they are highly flexible, straightforward to design, can simulate multiple neuron models, not sensitive to process variability and noise, stable, and can benefit from micro electronic technology scalability. The combination of these two techniques, have also been used in some projects that uses digital system for the connections and analog neuron components.

Among the learning rules for SNNs, Spike-Timing Dependent Plasticity (STDP) is the most favored for unsupervised online training of feed-forward networks which is believed to be closer to biology [64]. As a result, many neuromorphic architectures have used various techniques to implement STDP-based spiking networks [65, 66, 67, 68, 69, 70].

1.2.3 ASICs Versus FPGAs

Two classes of digital systems are Field Programmable Gate Arrays (FPGAs) and Application Specified Integrated Circuits (ASICs). Comparing these two classes, logic components in FPGA devices could easily change with a configuration bitstream result from HDL synthesizers providing a cheap and flexible platform. In ASIC devices, on the other hand, a simple change in design could result in a new development cycle, which is expensive and prolonged. However, when using FPGAs as the implementation platform, one should take into consideration the limited FPGAs resources, which makes it crucial to employ them effectively for the best performance and the lowest cost.

1.2.4 Objectives

Neuromorphic systems are comprises of large number of neurons, synapses and challenges. Therefore, it is essential to design an effective hardware for these building block to make the system more efficient. The obkective of this dissertation is use the state-of-art techniques to design effective hardware in the terms of area and power consumtion and yet fast to for digital implementation of spiking neural networks.

1.3 Proposed Solutions

One of the factors that makes implementation of biological systems a challenging task, is nonlinear functions describing biochemical reactions in the cells. Such nonlinear functions includes division, non-integer roots, multiplication, quadratic, cubic, quartic etc. Implementing and calculating these terms requires extensive use of resources which are limited in the target implementation platform: FPGA. Besides, calculating such terms will greatly affect the speed and throughput of the system. In this disseration, to overcome this problem, the nonlinear terms were calculated in a way that both reduce the implementation resources and increase the speed of the hardware. These methods are as follows: Linearization and COordinate Rotation DIgital Computer (CORDIC)

1.3.1 Linearization

In this technique nonlinear terms were replaced with a sequence of linear terms. This would results in a small error and deviation from the original term but considerably improves the efficiency of the hardware. The number of the linear segments is a trade-off between complexity and accuracy. More number of nonlinear terms results in higher accuracy and lower error but impose more hardware resources. On the other hand, designs with fewer segments are faster and require less area but have higher error. To compare the results several hardware were developed by setting different constraints to investigate trade-offs and achieve best possible design.

1.3.2 CORDIC

CORDIC is an iterative algorithm originally developed in [71] and thereafter generalized for calculation of hyperbolic and exponential functions, multiplications, divisions and square roots. CORDIC only requires simple shift and addition operations, which can be cheaply implemented on hardware hence making it an appropriate choice for fast and low-cost hardware implementations.

Comparing with the previous method of linearization which induces errors and deviations, this algorithm has a very high precision while it is very well suited for hardware implementation. This algorithm was used to calculate nonlinear terms in the differential equations describing spiking neural networks.

1.4 Outline of disseration and list of the contributes

In this dissertation, several hardware were designed to efficient implement spiking neural networks:

- Chapter 2 proposes a neuromorphic platform for on-FPGA online STDP learning, based on the CORDIC algorithm. The implemented platform comprises of two main components. First, the Izhikevich neuron model is modified for implementation using the CORDIC algorithm, simulated to ensure the model ac-

curacy, described as hardware, and implemented on FPGA. Second, the STDP learning algorithm is adapted and optimized using the CORDIC method, synthesized for hardware, and implemented to perform on-FPGA online learning on a network of CORDIC Izhikevich neurons to demonstrate competitive Hebbian learning. The implementation results are compared with the original model and state-of-the-art to verify accuracy, effectiveness, and higher speed of the system. These comparisons confirm that the proposed neuromorphic system offers better performance and higher accuracy while being straightforward to implement and suitable to scale.

- Chapter 3 proposes a high-speed and low-cost digital hardware to replicate biological-plausible astrocyte and glutamate release mechanism. The nonlinear terms of these models were calculated using a high-precision and cost-efficient algorithm. Subsequently, the modified models were simulated to study and validate their functions. Several hardware were developed by setting different constraints to investigate trade-offs and achieve best possible design. Hardware implementation results confirmed the ability of the design to replicate biological cells in detail with high accuracy. As for performance, the proposed design turned out to be far more faster and area efficient than previously published works that targeted digital hardware for biological-plausible astrocytes.

- Chapter 4 presents a digital hardware which can effectively implement nonlinear differential equations of astrocyte by modifying the model for implementation purpose. Further, the modified models were simulated to ensure its accuracy and proper functioning . The main difficulty for the circuit implementation of the astrocyte model lies in nonlinearity of the expressions which describe the biochemical reactions. As proof of concept, the design implemented on a FPGA device. As the results indicated, proposed hardware was capable of replicating the astrocyte in cellular level.

- Chapter 5 Izhikevich neuron's accuracy as a function of the time step was measured. It was uncovered that the threshold time step that Izhikevich neuron

becomes unstable is an exponential function of the input current. Software simulation performance, including total computational time and memory usage were compared for different time steps. Afterwards, the model was synthesized and implemented on the FPGA. Hardware performance metrics such as speed, area and power consumption were measured for each time step. Results indicated that time step has a negative linear effect on the performance. It was concluded that by determining maximum input current to the neuron, larger time steps comparable to those used in the previous works could be employed.

References

[1] E. Capecci, F. C. Morabito, M. Campolo, N. Mammone, D. Labate, and N. Kasabov, *A Feasibility Study of Using the NeuCube Spiking Neural Network Architecture for Modelling Alzheimer's Disease EEG Data.* Cham: Springer International Publishing, 2015, pp. 159–172.

[2] S. Ghosh-Dastidar and H. Adeli, "A new supervised learning algorithm for multiple spiking neural networks with application in epilepsy and seizure detection," *Neural Networks*, vol. 22, no. 10, pp. 1419–1431, Dec 2009.

[3] I. Wallach, M. Dzamba, and A. Heifets, "Atomnet: A deep convolutional neural network for bioactivity prediction in structure-based drug discovery," *CoRR*, vol. abs/1510.02855, Oct 2015.

[4] Y. Li and C. S. Nam, "Collaborative brain-computer interface for people with motor disabilities," *IEEE Computational Intelligence Magazine*, vol. 11, no. 3, pp. 56–66, Aug 2016.

[5] F. O. Carreon, J. G. G. Serna, A. M. Rendon, N. G. Franco, A. M. P. Jimenez, and J. H. Gomez, "Induction of emotional states in people with disabilities through film clips using brain computer interfaces," *IEEE Latin America Transactions*, vol. 14, no. 2, pp. 563–568, Feb 2016.

[6] X. Zeng, T. Song, X. Zhang, and L. Pan, "Performing four basic arithmetic operations with spiking neural p systems," *IEEE Transactions on NanoBioscience*, vol. 11, no. 4, pp. 366–374, Dec 2012.

[7] K. E. Friedl, A. R. Voelker, A. Peer, and C. Eliasmith, "Human-inspired neurorobotic system for classifying surface textures by touch," *IEEE Robotics and Automation Letters*, vol. 1, no. 1, pp. 516–523, Jan 2016.

[8] J. A. Wall, L. J. McDaid, L. P. Maguire, and T. M. McGinnity, "Spiking neural network model of sound localization using the interaural intensity difference," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 4, pp. 574–586, Apr 2012.

[9] J. H. Lee, T. Delbruck, M. Pfeiffer, P. K. J. Park, C. W. Shin, H. Ryu, and B. C. Kang, "Real-time gesture interface based on event-driven processing from stereo silicon retinas," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 12, pp. 2250–2263, Dec 2014.

[10] N. Srinivasa and Y. Cho, "Self-organizing spiking neural model for learning fault-tolerant spatio-motor transformations," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 10, pp. 1526–1538, Oct 2012.

[11] M. Samie, G. Dragffy, A. M. Tyrrell, T. Pipe, and P. Bremner, "Novel bio-inspired approach for fault-tolerant vlsi systems," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 21, no. 10, pp. 1878–1891, Oct 2013.

[12] M. R. Azghadi, N. Iannella, S. F. Al-Sarawi, G. Indiveri, and D. Abbott, "Spike-based synaptic plasticity in silicon: design, implementation, application, and challenges," *Proceedings of the IEEE*, vol. 102, no. 5, pp. 717–737, 2014.

[13] K. Boahen, "A neuromorph's prospectus," *Computing in Science and Engg.*, vol. 19, no. 2, pp. 14–28, Mar. 2017.

[14] L. A. Pastur-Romay, F. Cedron, A. Pazos, and A. B. Porto-Pazos, "Deep artificial neural networks and neuromorphic chips for big data analysis: Pharmaceutical and bioinformatics applications," *International Journal of Molecular Sciences*, vol. 17, no. 8, 2016.

[15] B. V. Benjamin, P. Gao, E. McQuinn, S. Choudhary, A. R. Chandrasekaran, J. M. Bussat, R. Alvarez-Icaza, J. V. Arthur, P. A. Merolla, and K. Boahen, "Neurogrid: A mixed-analog-digital multichip system for large-scale neural simulations," *Proceedings of the IEEE*, vol. 102, no. 5, pp. 699–716, May 2014.

[16] P. Merolla, J. Arthur, F. Akopyan, N. Imam, R. Manohar, and D. S. Modha, "A digital neurosynaptic core using embedded crossbar memory with 45pj per spike in 45nm," in *2011 IEEE Custom Integrated Circuits Conference (CICC)*, Sept 2011, pp. 1–4.

[17] S. B. Furber, F. Galluppi, S. Temple, and L. A. Plana, "The spinnaker project," *Proceedings of the IEEE*, vol. 102, no. 5, pp. 652–665, May 2014.

[18] J. Schemmel, D. Briiderle, A. Griibl, M. Hock, K. Meier, and S. Millner, "A wafer-scale neuromorphic hardware system for large-scale neural modeling," in *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*, May 2010, pp. 1947–1950.

[19] R. Wang, C. S. Thakur, G. Cohen, T. J. Hamilton, J. Tapson, and A. van Schaik, "Neuromorphic hardware architecture using the neural engineering framework for pattern recognition," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 11, no. 3, pp. 574–584, June 2017.

[20] C. S. Thakur, J. Molin, G. Cauwenberghs, G. Indiveri, K. Kumar, N. Qiao, J. Schemmel, R. Wang, E. Chicca, J. O. Hasler *et al.*, "Large-scale neuromorphic spiking array processors: A quest to mimic the brain," *arXiv preprint arXiv:1805.08932*, 2018.

[21] B. Sen-Bhattacharya, S. James, O. Rhodes, I. Sugiarto, A. Rowley, A. B. Stokes, K. Gurney, and S. B. Furber, "Building a spiking neural network model of the basal ganglia on spinnaker," *IEEE Transactions on Cognitive and Developmental Systems*, pp. 1–1, 2018.

[22] J. P. Dominguez-Morales, A. Rios-Navarro, D. Gutierrez-Galan, R. Tapiador-Morales, A. Jimenez-Fernandez, E. Cerezuela-Escudero, M. Dominguez-Morales, and A. Linares-Barranco, "Multilayer spiking neural network for audio samples classification using spinnaker," in *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2017, pp. 1–1.

[23] D. Khodagholy, J. N. Gelinas, T. Thesen, W. Doyle, O. Devinsky, G. G. Malliaras, and G. Buzsáki, "Neurogrid: recording action potentials from the surface of the brain," *Nature neuroscience*, vol. 18, no. 2, p. 310, 2015.

[24] S. Scholze, H. Eisenreich, S. Höppner, G. Ellguth, S. Henker, M. Ander, S. Hänzsche, J. Partzsch, C. Mayr, and R. Schüffny, "A 32 gbit/s communication soc for a waferscale neuromorphic system," *INTEGRATION, the VLSI journal*, vol. 45, no. 1, pp. 61–75, 2012.

[25] M. Chu, B. Kim, S. Park, H. Hwang, M. Jeon, B. H. Lee, and B. G. Lee, "Neuromorphic hardware system for visual pattern recognition with memristor array and cmos neuron," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 4, pp. 2410–2419, April 2015.

[26] F. Ponulak and A. Kasinski, "Introduction to spiking neural networks: Information processing, learning and applications," *Acta neurobiologiae experimentalis*, vol. 71, no. 4, pp. 409–433, 2011.

[27] A. Grüning and S. M. Bohte, "Spiking neural networks: Principles and challenges," in *In: ESANN 2014. 22nd European symposium on artificial neural networks, computational intelligence and machine learning*, Oct 2014, pp. 1–10.

[28] J. Vreeken, *Spiking Neural Networks: An Introduction*, Artificial Intelligence laboratory, Intelligent Systems Group, Univ. Utrecht, 2003.

[29] W. Maass, G. Schnitger, and E. Sontag, "On the computational power of sigmoid versus boolean threshold circuits," in *Foundations of Computer Science, 1991. Proceedings., 32nd Annual Symposium on*, Oct 1991, pp. 767–776.

[30] W. Gerstner, R. Kempter, J. L. van Hemmen, and H. Wagner, "A neuronal learning rule for sub-millisecond temporal coding," *Nature*, vol. 383, no. 6595, pp. 76–78, 1996.

[31] R. Eckhorn, R. Bauer, W. Jordan, M. Brosch, W. Kruse, M. Munk, and H. Reitboeck, "Coherent oscillations: A mechanism of feature linking in the visual cortex?" *Biological Cybernetics*, vol. 60, no. 2, pp. 121–130, 1988.

[32] C. von der Malsburg and W. Schneider, "A neural cocktail-party processor," *Biological Cybernetics*, vol. 54, no. 1, pp. 29–40, 1986.

[33] J. Misra and I. Saha, "Artificial neural networks in hardware: A survey of two decades of progress," *Neurocomputing*, vol. 74, no. 1-3, pp. 239 – 255, 2010.

[34] A. L. Hodgkin and A. F. Huxley, "A quantitative description of membrane current and its application to conduction and excitation in nerve," *The Journal of physiology*, vol. 117, no. 4, p. 500, 1952.

[35] E. M. Izhikevich, "Simple model of spiking neurons," *IEEE Transactions on neural networks*, vol. 14, no. 6, pp. 1569–1572, 2003.

[36] R. Brette and W. Gerstner, "Adaptive exponential integrate-and-fire model as an effective description of neuronal activity," *Journal of neurophysiology*, vol. 94, no. 5, pp. 3637–3642, 2005.

[37] R. FitzHugh, "Impulses and Physiological States in Theoretical Models of Nerve Membrane," *Biophysical Journal*, vol. 1, pp. 445–466, jul 1961.

[38] C. Morris and H. Lecar, "Voltage oscillations in the barnacle giant muscle fiber," *Biophysical Journal*, vol. 35, no. 1, pp. 193–213, 1981.

[39] H. R. WILSON, "Simplified dynamics of human and mammalian neocortical neurons," *Journal of Theoretical Biology*, vol. 200, no. 4, pp. 375–388, 1999.

[40] R. M. Rose and J. L. Hindmarsh, "The assembly of ionic currents in a thalamic neuron i. the three-dimensional model," *Proceedings of the Royal Society B: Biological Sciences*, vol. 237, no. 1288, pp. 267–288, 1989.

[41] W. Gerstner and W. M. Kistler, *Spiking neuron models: Single neurons, populations, plasticity*. Cambridge university press, 2002.

[42] W. Gerstner, W. M. Kistler, R. Naud, and L. Paninski, *Neuronal dynamics: From single neurons to networks and models of cognition*. Cambridge University Press, 2014.

[43] P. U. Diehl, G. Zarrella, A. Cassidy, B. U. Pedroni, and E. Neftci, "Conversion of artificial recurrent neural networks to spiking neural networks for low-power neuromorphic hardware," in *2016 IEEE International Conference on Rebooting Computing (ICRC)*, Oct 2016, pp. 1–8.

[44] D. Martí, M. Rigotti, M. Seok, and S. Fusi, "Energy-efficient neuromorphic classifiers," *Neural computation*, vol. 28, no. 10, pp. 2011–2044, 2016.

[45] R. Kreiser, T. Moraitis, Y. Sandamirskaya, and G. Indiveri, "On-chip unsupervised learning in winner-take-all networks of spiking neurons," in *2017 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, Oct 2017, pp. 1–4.

[46] H. Y. Hsieh, P. Y. Li, C. H. Yang, and K. T. Tang, "A high learning capability probabilistic spiking neural network chip," in *2018 International Symposium on VLSI Design, Automation and Test (VLSI-DAT)*, April 2018, pp. 1–4.

[47] F. A. Azevedo, L. R. Carvalho, L. T. Grinberg, J. M. Farfel, R. E. Ferretti, R. E. Leite, W. J. Filho, R. Lent, and S. Herculano-Houzel, "Equal numbers of neuronal and nonneuronal cells make the human brain an isometrically scaled-up primate brain," *The Journal of Comparative Neurology*, vol. 513, no. 5, pp. 532–541, Apr 2009.

[48] S. Davies, "Learning in the spiking neural networks," Ph.D. dissertation, Univ. of Manchester, Manchester, 2012.

[49] "The human brain project," 2016. [Online]. Available: https://www.humanbrainproject.eu

[50] "Bluebrain — epfl," 2016, last visited 2012-03-18. [Online]. Available: http://bluebrain.epfl.ch

[51] "Cognitive computation project," 2016. [Online]. Available: http://ibm.com

[52] "Nest simulator — the neural simulation tool," 2016. [Online]. Available: http://www.nest-simulator.org/

[53] "The brain spiking neural network simulator," 2016. [Online]. Available: http://briansimulator.org/

[54] "Nemo," 2016. [Online]. Available: http://nemosim.sourceforge.net/

[55] A. Joubert, B. Belhadj, O. Temam, and R. Heliot, "Hardware spiking neurons design: Analog or digital?" in *Neural Networks (IJCNN), The 2012 International Joint Conference on*, June 2012, pp. 1–5.

[56] A. S. Cassidy, J. Georgiou, and A. G. Andreou, "Design of silicon brains in the nano-cmos era: Spiking neurons, learning synapses and neural architecture optimization," *Neural Networks*, vol. 45, pp. 4 – 26, 2013, neuromorphic Engineering: From Neural Systems to Brain-Like Engineered Systems.

[57] O. Sharifipoor and A. Ahmadi, "An analog implementation of biologically plausible neurons using {CCII} building blocks," *Neural Networks*, vol. 36, pp. 129 – 135, 2012.

[58] S. Millner, A. Grübl, K. Meier, J. Schemmel, and M. olivier Schwartz, "A vlsi implementation of the adaptive exponential integrate-and-fire neuron model," in *Advances in Neural Information Processing Systems 23*, J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta, Eds. Curran Associates, Inc., 2010, pp. 1642–1650.

[59] S. Brink, S. Nease, P. Hasler, S. Ramakrishnan, R. Wunderlich, A. Basu, and B. Degnan, "A learning-enabled neuron array ic based upon transistor channel models of biological phenomena," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 7, no. 1, pp. 71–81, Feb 2013.

[60] Y. Yamashita and H. Torikai, "A novel pwc spiking neuron model: Neuron-like bifurcation scenarios and responses," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 59, no. 11, pp. 2678–2691, Nov 2012.

[61] J. A. Starzyk and Basawaraj, "Memristor crossbar architecture for synchronous neural networks," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 61, no. 8, pp. 2390–2401, Aug 2014.

[62] Y. Wang and S. C. Liu, "A two-dimensional configurable active silicon dendritic neuron array," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 58, no. 9, pp. 2159–2171, Sept 2011.

[63] G. Indiveri, B. Linares-Barranco, T. J. Hamilton, A. van Schaik, R. Etienne-Cummings, T. Delbruck, S.-C. Liu, P. Dudek, P. Häfliger, S. Renaud, J. Schemmel, G. Cauwenberghs, J. Arthur, K. Hynna, F. Folowosele, S. SAÏGHI, T. Serrano-Gotarredona, J. Wijekoon, Y. Wang, and K. Boahen, "Neuromorphic silicon neuron circuits," *Frontiers in Neuroscience*, vol. 5, no. 73, 2011.

[64] A. Morrison, M. Diesmann, and W. Gerstner, "Phenomenological models of synaptic plasticity based on spike timing," *Biological cybernetics*, vol. 98, no. 6, pp. 459–478, 2008.

[65] E. Covi, S. Brivio, M. Fanciulli, and S. Spiga, "Synaptic potentiation and depression in al: Hfo2-based memristor," *Microelectronic Engineering*, vol. 147, pp. 41–44, 2015.

[66] M. R. Azghadi, S. Moradi, D. B. Fasnacht, M. S. Ozdas, and G. Indiveri, "Programmable spike-timing-dependent plasticity learning circuits in neuromorphic vlsi architectures," *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, vol. 12, no. 2, p. 17, 2015.

[67] C. Mayr, J. Partzsch, M. Noack, S. Hanzsche, S. Scholze, S. Hoppner, G. Ellguth, and R. Schuffny, "A biological-realtime neuromorphic system in 28 nm cmos using low-leakage switched capacitor circuits," *IEEE transactions on biomedical circuits and systems*, vol. 10, no. 1, pp. 243–254, 2016.

[68] R. M. Wang, T. J. Hamilton, J. Tapson, and A. van Schaik, "A mixed-signal implementation of a polychronous spiking neural network with delay adaptation," *Frontiers in neuroscience*, vol. 8, p. 51, 2014.

[69] S. Yang, J. Wang, B. Deng, C. Liu, H. Li, C. Fietkiewicz, and K. A. Loparo, "Real-time neuromorphic system for large-scale conductance-based spiking neural networks," *IEEE Transactions on Cybernetics*, pp. 1–14, 2018.

[70] K. Isobe and H. Torikai, "A novel hardware-efficient asynchronous cellular automaton model of spike-timing-dependent synaptic plasticity," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 63, no. 6, pp. 603–607, June 2016.

[71] J. E. Volder, "The cordic trigonometric computing technique," *Electronic Computers, IRE Transactions on*, vol. EC-8, no. 3, pp. 330–334, Sept 1959.

CORDIC-SNN: On-FPGA STDP Learning with Izhikevich Neurons

2.1 Introduction

Highly parallel, energy efficient, fault tolerant, and compact neuromorphic learning systems promise alternative devices for solving engineering problems [1] and powerful tools to understand properties of biological neural networks [2]. Several such systems have already been introduced and used [3, 4, 5, 6, 7] for various applications such as pattern cognition, signal processing, and autonomous robots [8, 9, 10, 11, 12, 13].

These neuromorphic systems typically include a large number of neurons, synapses and their interconnecting structure on hardware. They provide real-time simulation, regardless of the size of the network, are parallel, and energy efficient [14]. The performance of such systems at a higher level depends on the neuron, synapse and learning models and at a lower level on the circuits realizing such units [15].

Current neuromorphic research has led to the development of a plethora of models to mimic real neurons with different levels of abstraction in biological details. Biologically-plausible models, such as Hodgkin Huxley [16] describe cellular phenomena and properties of the individual biological components. Such low-level models, impose more computation cost, making it difficult to simulate large-scale networks. On the other hand, biologically-inspired models such as Izhikevich [17] and models

in [18, 19, 20, 21, 22], aim to mimic the biological neurons to the best degree of accuracy. Such models can reproduce most of the firing patterns of real neurons and are easier to couple to other spike-oriented units. Moreover, high-level Integrate and Fire (IF) [23] is another computationally efficient neural model, but cannot exhibit many essential features of the biological neurons as observed in experiments [24]. As far as neuromorphic computing is concerned, simpler models are cheaper, faster, and more energy efficient. Nevertheless, the choice of models depends on the application of the device to be designed. To perform computations with SNNs only a simple IF or Exponential IF (EIF) may be enough to act as a thresholding box. However, for research in neuroscience, biologically plausible models have higher flexibility in mimicking biology. Here, we have chosen the Izhikevich neuron for simulation and Field Programmable Gate Array (FPGA) implementation, because while being computationally efficient, it produces biologically plausible firing patterns.

After selecting the neuron model, a proper Spiking Neural Network (SNN) topology should be chosen. This depends on a number of factors such as the level of abstraction, the targeted application, available hardware, and the learning algorithm. A variety of spiking network topologies have been used in neuromorphic systems such as recurrent [25], feed-forward [26], winner-take-all [27], and probabilistic [28]. Subsequently, the SNN learning method should be selected based on factors such as network topology, whether the learning should be on-chip or off-chip, be supervised or unsupervised, etc. Previous hardware implementations of SNN adopt many of these approaches [29, 30, 31, 32]. Among them, Spike-Timing Dependent Plasticity (STDP) is the most favored for unsupervised online training of feed-forward networks which is believed to be closer to biology [33]. As a result, many neuromorphic architectures have used various techniques to implement STDP-based spiking networks [34, 35, 36, 37, 38, 39]. Similarly, this paper uses a novel technique based on CORDIC algorithm, described in the following sections, to realize an online STDP-learning architecture in hardware.

Considering hardware implementation platforms, they could be divided into three major categories as analog [6, 34, 36, 40, 41], digital [4, 7, 42] or mixed analog-

digital [3, 37, 35] systems, each with its advantages and disadvantages. Two classes of digital systems are FPGAs and Application Specified Integrated Circuits (ASICs). Comparing these two classes, logic components in FPGA devices could easily change with a configuration bitstream result from Hardware Description Language (HDL) synthesizers providing a cheap and flexible platform. In ASIC devices, on the other hand, a simple change in design could result in a new development cycle, which is expensive and prolonged. However, when using FPGAs as the implementation platform, one should take into consideration the limited FPGAs resources, which makes it crucial to employ them effectively for the best performance and the lowest cost.

To that end, the first challenge is to implement the neuron model as efficient and fast as possible. This paper utilizes CORDIC to calculate Izhikevich neuron differential equations. CORDIC is used to exclude the use of multipliers which are area-intensive and slow arithmetic operators in FPGAs. In order to increase the performance and size of the network, several techniques have been previously utilized to decrease the multiplication cost. These include bit serial and reduced range precision multipliers, stochastic-based neurons, replacing multiplication with add & shift operations, and Look Up Tables (LUTs) [43].

A number of FPGA implementations of Izhikevich neuron are available in literature. In [44], a rotate-and-fire digital spiking neuron model has been implemented that can reproduce five type of inhibitory responses as an asynchronous sequential logic circuit. In [45], an asynchronous cellular automata-based neuron model is presented. In [46], the continuous nullclines are approximated to cellular space for a low-cost neuron implementation. Reference [47] presents a piece-wise linear approximation [48] of the Izhikevich model to achieve multiplier-less hardware for lower cost and higher speed. Further, reference [49] utilizes CORDIC algorithm to design a low power digital circuit for this neuron. Compared with previous works, the CORDIC-based method presented here results in neurons requiring fewer resources and operating at a higher frequency.

In addition, to implement the STDP algorithm, the CORDIC exponential core

in [50] was adopted to compute STDP function with high precision while requiring low resources.

Different method have been used by researchers to implement STDP algorithm. One of the common methods is to use Address-Event Representation (AER) data protocol [35]. Reference [51] utilizes piece-wise linear approximation (PWL) technique to implement the exponential term in STDP and a counter to store spike events. Moreover, a dedicated plasticity processor was used in [52]. In another paper, authors used a simplified multiplier to reduce the STDP implementation cost[53]. In this work, to implement the STDP algorithm, the CORDIC exponential core in [50] was adopted to compute STDP function with high precision while requiring low resources. To account for the spike timings required for STDP, a shift register was utilized to store the firing times of pre and post-synaptic neurons in order to determine the time differences and calculate synaptic weight updates. This is a novel technique that exploits a distributed memory to realize biological networks.

The rest of this Chapter is organized as follows. Section 2.2 and 2.3 review the Izhikevich neuron and STDP learning algorithm and further presents CORDIC modified models, computer simulations, and investigation of accuracy through errors analysis and studying the network behaviors. Section 2.4 and 2.5 discuss FPGA implementation procedure and compare achieved result with previous works. Finally, Section 2.6 concludes the paper.

2.2 CORDIC neuron

2.2.1 Izhikevich neuron

Izhikevich neuron is a two-dimensional model, which consists of two coupled Ordinary Differential Equations (ODEs) as:

$$\frac{dv}{dt} = 0.04v^2 + 5v + 140 - u + I \tag{2.1}$$

$$\frac{du}{dt} = a(bv - u) \tag{2.2}$$

and a reset condition as:

$$if \quad v > 30mv \quad then \quad \begin{cases} v \rightarrow v_r \\ u \rightarrow u_r = u + d. \end{cases} \tag{2.3}$$

Here, Eqs. 2.1 and 2.2 describe membrane potential v, recovery variable u and applied current I. Other dimensionless parameters are:

- a: Time scale of the recovery variable;
- b: Sensitivity of the u to v;
- c: After-spike reset value of v;
- d: After-spike reset value of u;

With adjustment of these variables, Izhikevich model is capable of replicating several firing patterns exhibited by biological neurons such as tonic spiking, adaptation, initial or regular bursting, transient spiking, and irregular spiking [54].

2.2.2 CORDIC Izhikevich

CORDIC is an iterative algorithm originally developed in [55] and thereafter generalized for calculation of hyperbolic and exponential functions, multiplications, divisions and square roots. CORDIC only requires simple shift and addition operations, which can be cheaply implemented on hardware hence making it an appropriate choice for fast and low-cost hardware implementations.

The algorithm for CORDIC calculation of square term in Eq. 2.1 is shown in Fig. 2.1. The FOR loop in line 4 calculates the square(x) to the n bit precision. The x register keeps track of rotation direction in each iteration where z accumulates the result. In this approach, calculating to $k + n$ bit precision is equal to rounding of multiplication to $k + n$ bit without calculating unnecessary bits. Choosing n is a trade-off between computation complexity and precision where k depends on the domain of the square function. Since the membrane potential of the neuron ranges between -100 and 30, k is set to 6 so that its two's power ($2^6 = 64$) is greater than

```
square(x)
y=x;
z=0;
for i=-k:n
{
  if (x > 0)
  {
   x = x - 2^(-i);
   z = z + y*2^(-i);
  }
  else
  {
  x = x + 2^(-i);
  z = z - y*2^(-i);
  }
}
return z;
```

Figure 2.1: The CORDIC code for calculation of square function.

$100/2 = 50$ and therefore the algorithm can keep up splitting the v to reach the value of v^2. To further evaluate the effect of n on the neuron behavior, we define four models with n=6, n=8, n=10 and n=12, naming them $IzhCOR6$, $IzhCOR8$, $IzhCOR10$, and $IzhCOR12$, respectively. This will help to compare simulation and implementation results in terms of deviation from the original model and hardware cost. The indicated code is most useful for a fixed point hardware but it can be modified to make it applicable to floating point hardware as well.

2.2.3 Simulation Results

Fig. 2.2 compares computer simulation of multiplication and CORDIC-based square functions. As this figure shows, two graphs are very close and only by zooming in small range the difference is visible (Error analysis is further presented in the next Section). Fig. 2.3 shows the computer simulation of Izhikevich and modified CORDIC model for different neuronal behaviors. For an identical applied current, responses are very similar and there is no distinctive difference. However, these results only indicate resemblance of models for one specified value of applied current. Therefore, the resemblance of models for a wide range and different random values of stimulation

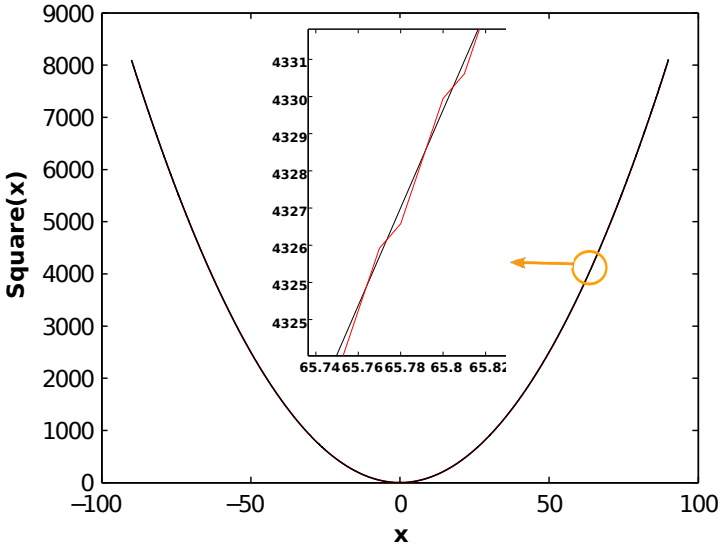
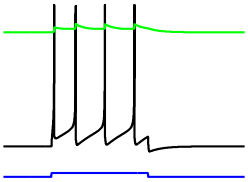
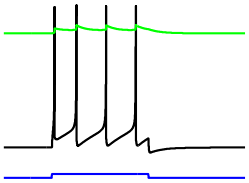


Figure 2.2: Computer simulation of multiplication and CORDIC-based square function. Here, black and red lines show multiplication, and CORDIC square functions, respectively. The difference between two lines is only visible by zooming in a small range.

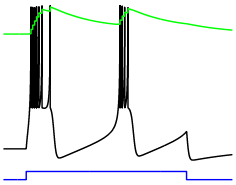
currents are investigated as follows. First, ODEs in both modified and original models



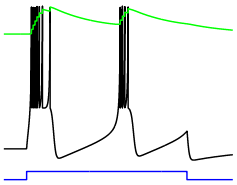
Tonic Spiking-Original Model



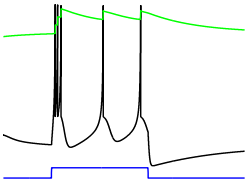
Tonic Spiking-Proposed Model



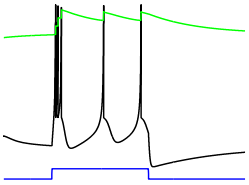
Tonic Bursting-Original Model



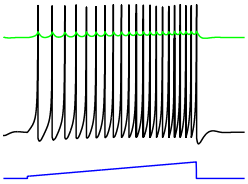
Tonic Bursting-Proposed Model



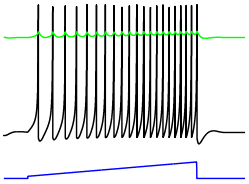
Mixed Mode-Original Model



Mixed-Proposed Model



Class II Excitability-Original Model



Class II Excitability-Proposed Model

Figure 2.3: Computer simulation of the original and the proposed modified CORDIC models for different neuronal behaviors. The black and green lines show membrane potential and recovery variable respectively. The applied current is illustrated by the blue line.

was set equal to zero as

$$\frac{dv}{dt} = 0 \quad and \quad \frac{du}{dt} = 0, \tag{2.4}$$

to depict nullclines in the phase planes of the systems. The result is displayed in Fig. 2.4, where the first row (a and c) shows nullclines for low values of stimulation where there are two fixed points. The second row (b and d), on the other hand, shows the responses of the models for higher stimulation current where those fixed points merge and annihilate simultaneously in both CORDIC and Izhikevich model phase plane.

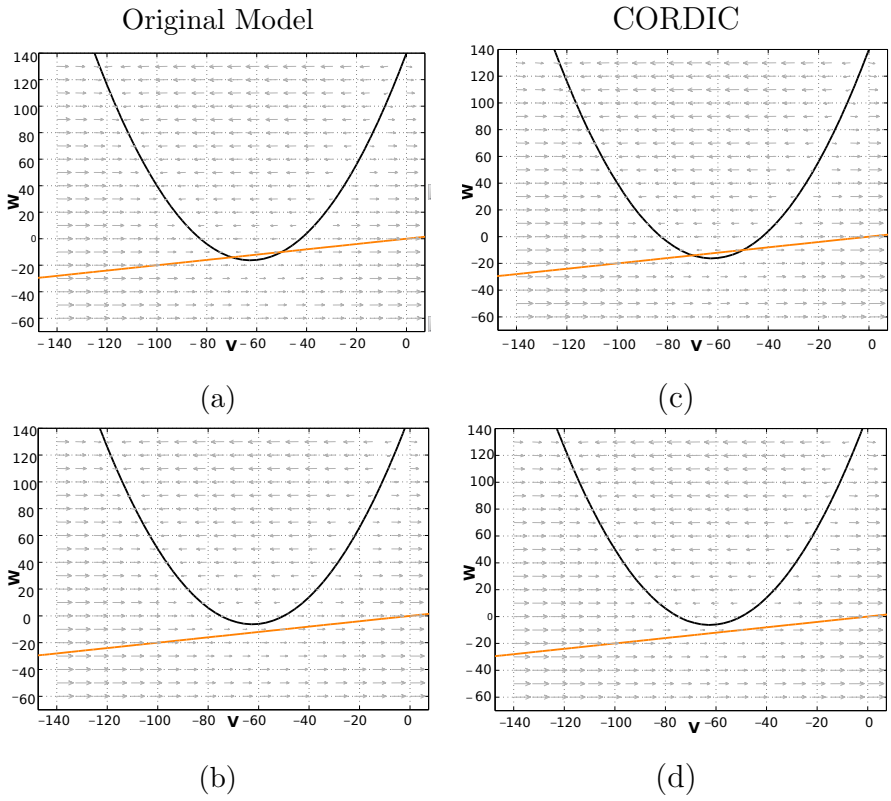
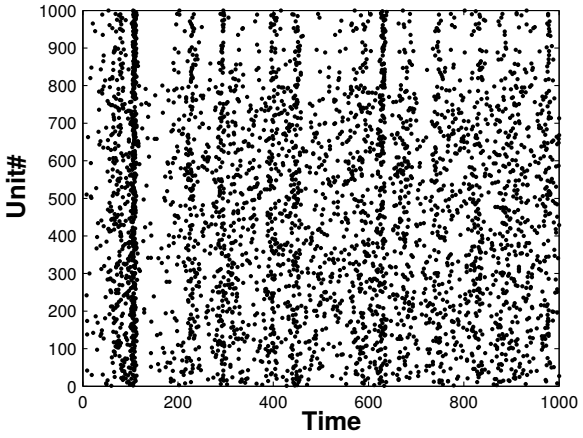
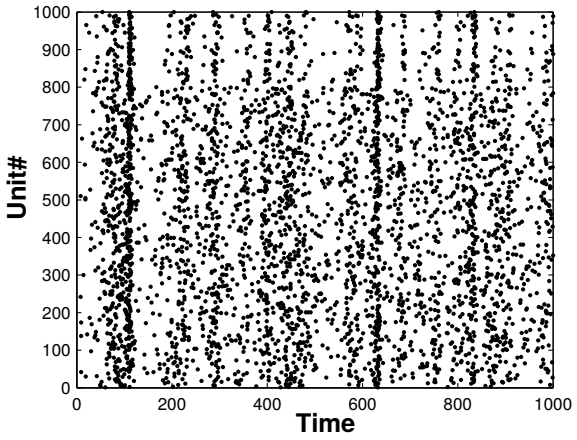


Figure 2.4: Nullclines of original and CORDIC model. In the first row, similar to the original model, CORDIC model has two interaction points for low injected current; the second row shows the state of models for higher injected current where those intersection points merged and annihilated.

Second, Fig. 2.5 compares the raster diagram of 1000 randomly coupled instances



(a)



(b)

Figure 2.5: The spike raster for a population of 1000 tonic bursting neurons which are coupled randomly. The utilized neuron models are (a) original Izhikevich and (b) proposed CORDIC.

of original Izhzikevich (a) and the proposed CORDIC neuron models. Here, each dot represents a specific neuron spiking at a specific time. Despite the differences in the details of the two models used, in general they are very much alike. Both Figures 2.4 and 2.5 demonstrate that the proposed CORDIC implementation of Izhikevich neuron can show qualitatively similar behavior to the original model. Further quantitative error analysis is presented in the following subsection.

2.3 Network and STDP rule

2.3.1 Models Numerical Analysis

To investigate the accuracy of the proposed model in generating Izhikevich behavior, two types of time domain errors were examined as follows.

ERRT: Modification in the neuron model may cause difference in spike timing and lag in the spike train of the modified model compared to the original one. For quantitative measuring of this error, first, two spike trains were synced and then time to next spike for original and CORDIC models was considered for the calculation of a timing error (named ERRT) as shown in Fig. 2.6. Here,

$$\begin{aligned} ERRT &= \left| \frac{\Delta t_c - \Delta t_o}{\Delta t_o} \right| \times 100, \\ \Delta t &= t_{s2} - t_{s1}, \end{aligned} \tag{2.5}$$

where Δt_c, and Δt_o are time intervals between the second (t_{s2}) and first spike (t_{s1}), for CORDIC and original model, respectively.

NRMSD: The Normalized Root Mean Square Deviation (NRMSD) [56] error is also used to measure the similarity of spike shapes in CORDIC and the original model. Low values for this error indicate more resemblance of output spikes. This error is defined as

$$RMSD = \sqrt{\frac{\sum_{i=1}^{n} (v_c(n) - v_o(n))^2}{n}}, \tag{2.6}$$

and normalized as

$$NRMSD = \frac{RMSD}{v_{max} - v_{min}}, \tag{2.7}$$

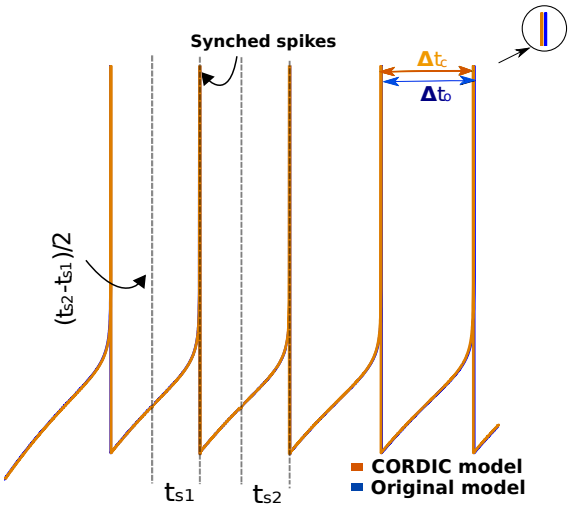


Figure 2.6: ERRT: The difference of time interval between two spikes in the original and CORDIC model obtained from computer simulations [50]

where V_c and V_o are the wave forms of the CORDIC and Izhikevich model, respectively. Here, V_{max} and V_{min} are the maximum and minimum values of V_o in its domain. For instance, for the curve in Fig. 2.2 at the range of [-100 100], NRMSD was calculated as 5.2177×10^{-5}, confirming a very small error between CORDIC squaring and squaring using normal multiplication operation. To measure the similarity of output spikes, first two spikes were synced as shown in Fig. 2.6 and thereafter NRMSD was evaluated for the half of time interval between these two spikes. Table 2.1 presents values of ERRT and NRMSD for computer simulation of modified CORDIC models. As expected, a higher value of n will result in smaller error values, where the IZH-COR12 has a negligible deviation from the Izhikevich neuron. It is worth noting that these errors were calculated for a value of input current and only between two spikes of the original and proposed neuron. To compare results, the value of NRMSD and ERRT error were calculated for different values of the input current and presented in Table 2.2 As data in this table indicates, the NRMSD error is higher for larger input cuurent. Hwoever, ERRT is decling since higher input currenr results in higher frequency and therefore shorter spike period.

Furthermore, the value of error was calculated for several consecutive spikes.The

Table 2.1: ERRT and NRMSD for tonic spiking and regular bursting.

	Model	Error Type	Ton. Spiking	Reg. Bursting
Computer simulation	IZHCOR6	Errt	%0.2549	%0.0000
		NRMSD	%0.0034	%0.0705
	IZHCOR8	Errt	%0.2049	%0.0000
		NRMSD	%0.0006	%0.0136
	IZHCOR10	Errt	%0.1025	%0.0000
		NRMSD	%0.0001	%0.0082
	IZHCOR12	Errt	%0.0000	%0.0000
		NRMSD	%0.0000	%0.0063
FPGA implementation	IZHCOR6	Errt	%0.0191	%0.0000
		NRMSD	%0.3951	%2.0631

Table 2.2: NRMSD and ERRT for different values of input current

		I=4mA	I=16mA	I==16mA	I=32mA
IZHCOR6	NRMSD%	13.34 10^{-4}	49.4 10^{-4}	64.25 10^{-4}	85.33 10^{-4}
	ERRT%	0.2159	0.0000	0.0000	0.0000
IZHCOR8	NRMSD%	2.68 10^{-4}	3.26 10^{-4}	18.80 10^{-4}	36.40 10^{-4}
	ERRT%	0.0000	0.0000	0.0000	0.0000

Table 2.3: NRMSD for different number of consecutive spikes

Number of consecutove spikes	1	5	10	15
NRMSD%	0.0041	0.0050	0.0066	0.054

mean value of NRMSD for different number of consecutive spikes are presented in Table 2.3. As data in this table indictes, the value of error slightly changes for different number of sequences but it remain almost steady.

2.3.2 Network Topology

In this study, a two-layer spiking neural network as shown in Fig. 2.7 was formed. The first layer consisting of 20 neurons acts as an input layer while the second one with a single neuron is the output. A uniform random spike train input, with the mean firing rate of 7 Hz, was applied to each input neuron, which made them fire (defined as the state where membrane potential become greater than 30mv). For the output neuron, the input current is considered to be the sum of the currents received from the input layer spiking neurons as

$$I_o = \sum_{i=1}^{20} w(i,1) f(i) \tag{2.8}$$

where $w(i,1)$, which was initially set to 96 (by trial and error), is the synaptic weight connecting the input layer i to the output neuron. The value of $f(i)$ is 1 if the corresponding neuron fires and is 0 otherwise.

2.3.3 STDP Learning

In STDP, analogous to biology, the synaptic weight changes when a pre-synaptic neuron fires in a short time before or after the post-synaptic neuron, strengthening or weakening the neuron connection accordingly. Such a change is determined as an exponential of the time difference between two events and is formulated as

$$\begin{cases} w_i(\Delta t) = A_+ \, e^{-\Delta t/\tau_+} & if \quad \Delta t > 0 \\ w_i(\Delta t) = -A_- \, e^{\Delta t/\tau_-} & if \quad \Delta t \leq 0, \end{cases} \tag{2.9}$$

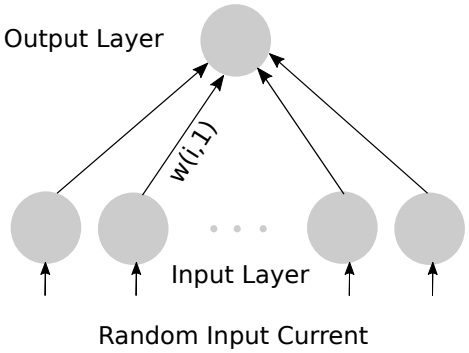


Figure 2.7: The topology of the utilized spiking neural network.

where $\Delta t = t_{post} - t_{pre}$ is the time span between pre- and post-synaptic spikes. Here, τ_+ and τ_- are STDP learning windows, which determine any time differences greater than them is considered to have a small effect on the synaptic weights and could be disregarded. These windows were set to $\tau_+ = \tau_- = 20ms$ in our experiments. In addition, A_+ and A_- are gain parameters set to 2 and 4 respectively considering the fact that in biology too, synapses tend to be more depressed than potentiated. Overall, these five parameters determine the magnitude of weight change.

Furthermore, as in biological synapses, the weight should be confined between $w_{min} < w < w_{max}$. The STDP mechanism of weakening and strengthening of synapses will eventually lead to a bi-modal distribution of weights, which is a result of competitive Hebbian learning [57]. This rule applies to the utilized network in Fig. 2.7 as well. STDP learning in this two-layer network leads to a bi-modal weight distribution as shown in Fig. 2.8. This figure depicts the evolving of network weights over the simulation time to distribute into two extreme weight values of 0 and 200.

2.3.4 CORDIC STDP

The main challenges in implementing STDP are its exponential terms and the memory required to store and retrieve spike timing. Here we implemented the exponential function required for STDP, using a modified version of the CORDIC algorithm presented in [50]. The algorithm for calculating the exponential of x (e^x) is shown in Fig. 2.9. Here, variables x and $expx$ are used to store input and output values, re-

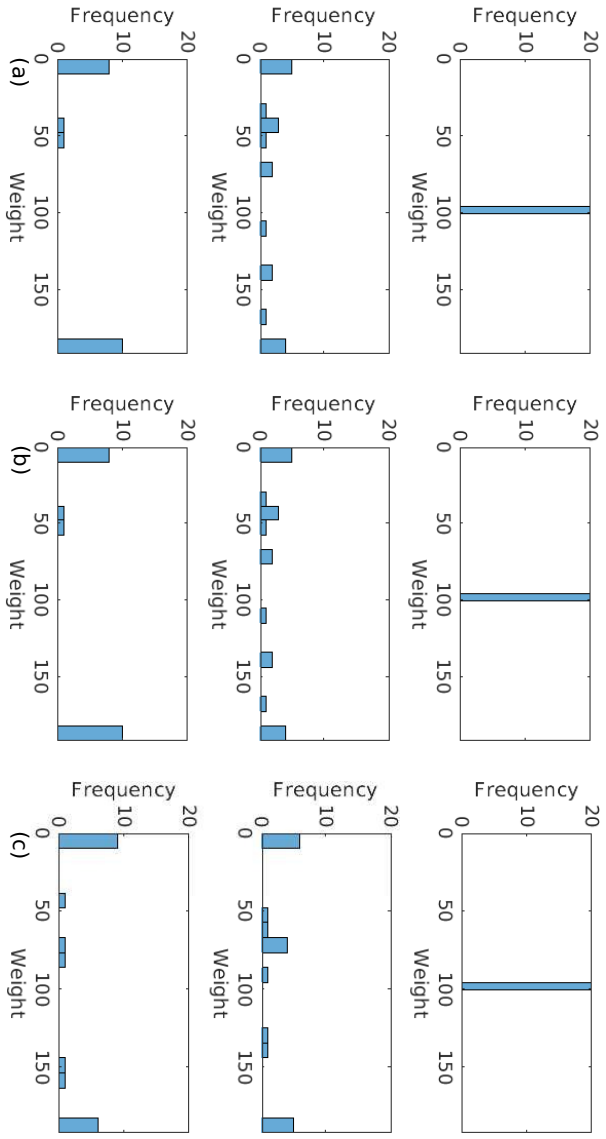


Figure 2.8: Weight distribution after STDP learning in the network of (a) original, (b) CORDIC, and (c) 2^x-based approximation model. All the 20 synaptic weights are initially set to 96 as shown in the first row. The second row displays weight distribution and their frequency after half of the simulation time. Finally, the third row depicts the weight distribution at the end of simulation, where the weights have mostly evolved to be either zero or the maximum possible value of $Wmax = 192$.

spectively. As part of the algorithm, the pre-calculated values of $e^{(\frac{1}{2})}, e^{(\frac{1}{4})}, \cdots, e^{(\frac{1}{n})}$

```
1  //assign initial values
2 z=fraction(x);poweroftwo=0.5;
3 expx=1;
4  //pre-calculated a elements
5 a=[exp((1/2)*(1:n))]
6  //Determine the weights
7  //and calculate products
8 for i from 0 to n do
9 {
10   if ( poweroftwo < z )
11   {
12    z=z-poweroftwo;
13    expx = expx * a(i);
14   }
15   poweroftwo=poweroftwo/2;
16 }
```

Figure 2.9: The pseudo code of CORDIC exponential.

are stored in an array, as shown in line 5. The *FOR* loop in line 8 calculates the exponential function for the fraction part of x with e^{-n} precision. In this work, n is set to 8, but higher values of n could be selected in the case of the need for higher precision exponential function. However, this will in turn slightly increases implementation cost. Our proposed algorithm is simpler than that of [50], because the range of x, for which we need to calculate the exponential function, is between -1 and 0. Fig. 2.10 demonstrates the very good approximation in implementing the exponential function achieved using our proposed CORDIC algorithm. In this figure, the blue curve shows the computer simulation of exponential function, while the red curve is the exponential approximation using CORDIC. The NRMSD error calculated for these curve was 2.38×10^{-3}, which further verify the high accuracy of the proposed CORDIC algorithm. To further verify the effectiveness of the proposed CORDIC algorithm in replicating the STDP model, the simple 2^x function was used as another method to approximate exponential function, because the value of x is always negative and in the range of -1 and 0. Such a term could be cheaply implemented on hardware using shift registers.

To test the accuracy of the approximated STDP models compared to the original

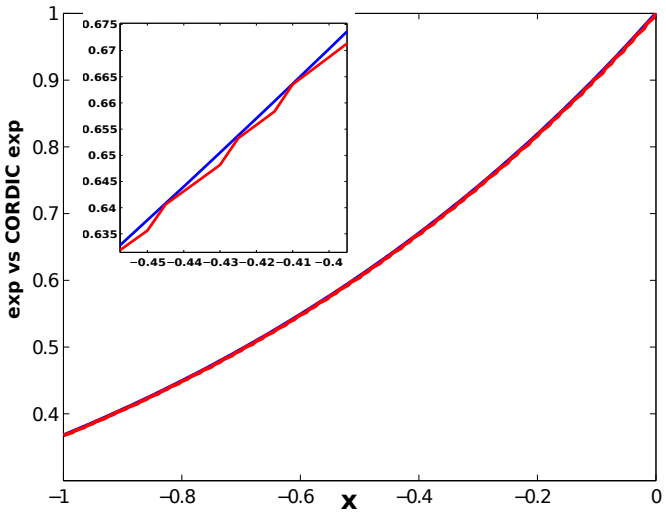


Figure 2.10: Software simulation of: exponential function (blue line) and the one calculated by CORDIC algorithm (red line) for n=8 indicating the resemblance of both functions.

model, two networks with the topology shown in Fig. 2.7 were formed. The first network consisted of original Izhikevich neurons and original STDP rule, while the second one used CORDIC STDP to connect CORDIC (IZHCOR8) neurons. Next, the same random current was applied to the input layer of both networks. Fig. 2.8(a) and (b) show the evolution of weight for the original and CORDIC networks, respectively. Due to the high precision of the CORDIC algorithm, the resulted weight distributions are very similar to the network implementing original STDP and Izhikevich models. Furthermore, Fig. 2.8(c) is the weight distribution achieved using the 2^x function instead of the exponential functions. As seen, the weight distribution is different from the original and CORDIC models but a similar bi-modal distribution could be observed.

2.4 FPGA implementation

2.4.1 Architecture of Izhikevich neuron

This section presents FPGA implementation of the proposed CORDIC Izhikevich neuron. Since the primary objective of this paper is to reduce the implementation cost and improve hardware speed, fixed-point arithmetic was used in our implementations. For solving the Izhikevich Ordinary Differential Equations (ODEs) shown in Eq. 2.1, and 2.2, they were discretized and simple Euler method was used that resulted in the following Equations.

$$\begin{aligned} v[n+1] = (0.04\ CORDIC\_Mul(v[n]) + 5v[n]... \\ + 140 - u[n] + I[n])dt + v[n], \end{aligned} \tag{2.10}$$

$$u[n+1] = a(bv[n] - u[n])dt + u[u]. \tag{2.11}$$

By choosing small step sizes and with the help of the reset equation, which keeps v bounded to help the stability of Euler method, this method produced stable outputs. In addition, multiplications by constant numbers were approximated to the closest possible values with the sum of a series of power of two numbers ($\sum_{-l}^{k} 2^n$), thereby reducing multiplications to simple shifts and adds. Obviously, a higher value of $k+l$ increases the multiplication precision but it also requires more shift registers and adders leading to a higher hardware resource requirement. The Control Data Flow Graph (CDFG) [58] of the Izhikevich CORDIC model is shown in Fig. 2.11. In this figure, operation blocks and registers are represented with circles and rectangles, respectively. In this graph, block A calculates the square function while block B solves the Euler method presented in Eq. 2.10 and 2.11. Arithmetic shift operations are shown by "¿¿" and "x¡¡" means shift by "x" position while adding the results, which implements $\sum_{-l}^{k} 2^n$.

For the model to work properly, optimum word length for the architecture should be determined. This can be specified when considering the minimum number of integer bits to correctly represent the range of variables, and the number of fraction bits for the minimum required precision. In addition, extra bits are required to

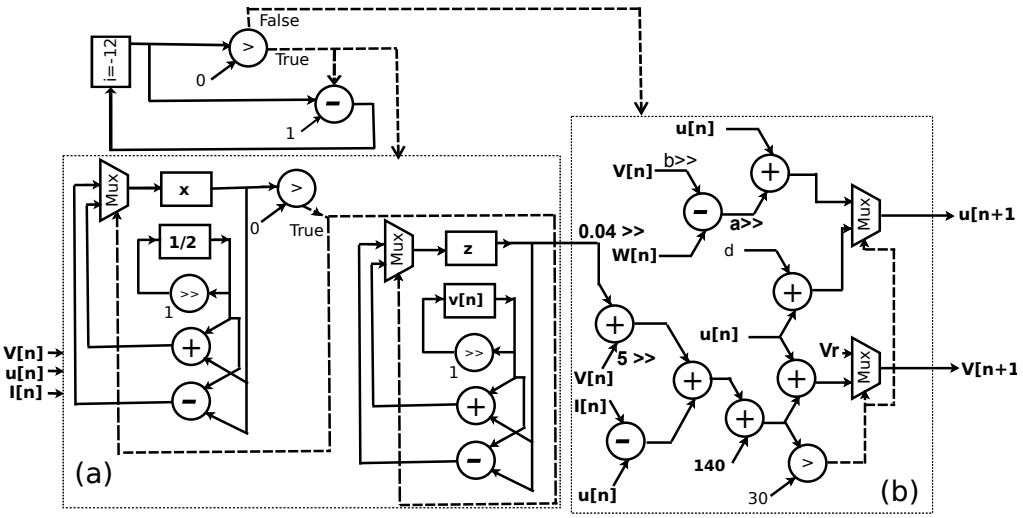


Figure 2.11: Control data flow graph for Spartan-6 XC6LX75 FPGA implementation of CORDIC Izhikevich neuron. First, block (a) calculates the square function as per the pseudo code in Fig. 2.1 . The counter which is showed at the top of this block, counts from -6 to 5, enabling this block for 12 iterations. In each iteration, $2^{(-i)}$ is added or subtracted from x register based on the sign of x. Eventually, this register's value tends to zero as iterations continues. The same scenario applies to the z register. The value of $2^{(-i)} * v[n]$ will be added or subtracted from the z register depending on the sign of z. After 12 iterations, the multiplication result is ready and the (b) block is enabled. This block solves the Euler method in the eq. 2.10 and 2.11. At the last stage of this block, v is compared with the threshold value of 30. If v is grater than this threshold, the multiplexers reset the v and u according to eq. 2.3. The plain lines show the flow of data and the dashed lines indicate the jumps and decision signals.

prevent from over and under flow in the shift&add operations. Considering all the requirements, and to avoid overflow and precision loss, 14 and 16 bits were dedicated for the fraction and integer parts, respectively.

2.4.2 Architecture of Network and STDP rule

Similar to the neuron, the Euler method was used to solve the discretized version of Eq. 2.9 to implement STDP. The CDFG for calculating the exponential term in this equation is presented in Fig. 2.12. Comparing to the flow graph used for calculating exponentials in [50], this one is simpler because here x is in the range of -1 and 0, and therefore no shift by e is needed. Nonetheless, this design also only uses shift & add operations, so it is hardware friendly.

The flow graph for implementation of the spiking network with STDP learning is shown in Fig. 2.13. The post-synaptic input current is the sum of the synaptic weights of all the pre-synaptic neurons that fire. As shown in the block (a) of the figure, a 41-bit shift register is used to record the spike timing of pre- and post-synaptic neurons. Every time a neuron fires or is silent, the register shifts to left and the least significant bit updates with 1 (spike) or 0 (silence), accordingly. Here, sampling time is controlled by a counter to act as enable signal for the shift register.

Online STDP learning rule is implemented in block (b) of Fig. 2.13. Here, the middle bit (Reg[20]) of the 41-bit shift register that records pre-synaptic spike times, enables STDP mechanism. This is to account for, and enable STDP, in response to future (Reg[19:0]) and past (Reg[40:21]) spike events. If a pre-synaptic neuron spikes, the time of that spike is compared to the time of post-synaptic spikes and the difference will be divided by τ (using shift operations) and passed to the exponential CORDIC calculator unit. Next, based on the sign of the time difference, the new weight will be determined and compared to the boundaries. The same approach could be used for calculating STDP but using the 2^x model. That way, the *exp_cordic* unit (in Fig. 2.13(b)), should be replaced with a 2^x calculator.

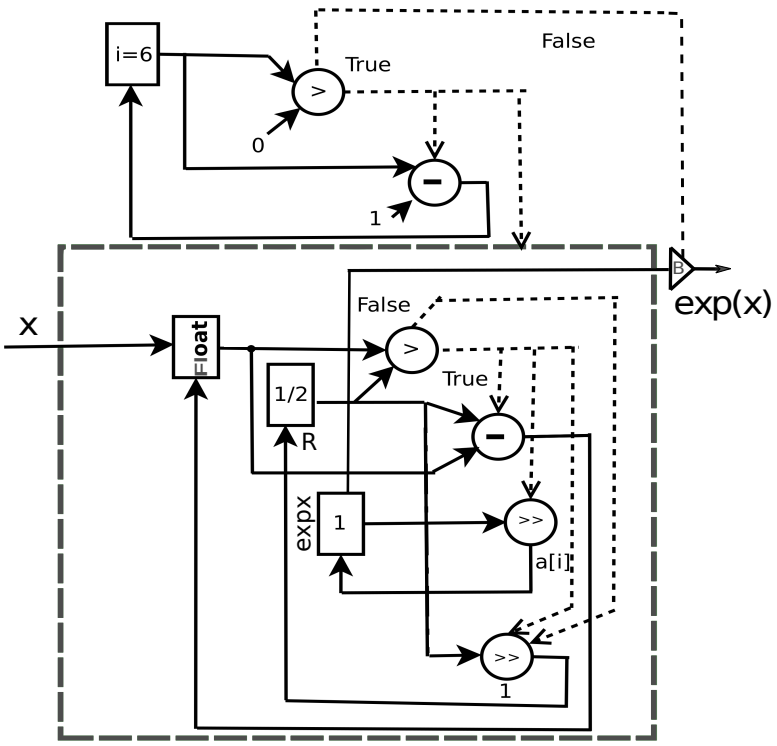


Figure 2.12: Control data flow graph for digital implementation of the exponential function for the range of -1¡x¡0 according the pseudo code shown in the Fig 2.9. Since the value of the input x is always smaller than one, the word length of this architecture was considered as total number of fraction bits. The calculations complete in 6 iterations as the counter at the top of the figure counts from 1 to 6. In each iteration, float register is compared with the 2^{-i}. If it is greater, the register is subtracted from 2^{-i}. Moreover, expx register, which it's initial value is 1, is multiplied by constant $a(i)$ with performing shift and add operations. Upon completion of iterations, the counter enables the out signal.

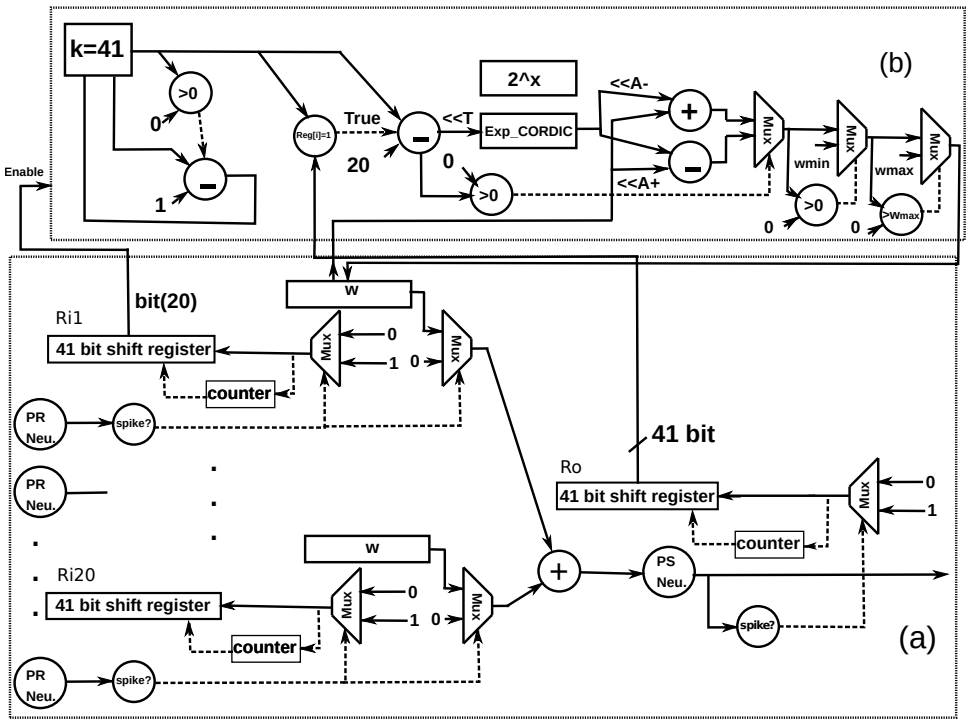


Figure 2.13: Control data flow graph for digital implementation of STDP algorithm. Block (a) is the hardware presented for the network in Fig. 2.7 and recording spike times. Block (b) implements STDP to calculate weight changes and update weights. In this block, either of Exp_CORDIC or 2x̂ blocks could be used for approximating the STDP exponential term.

Table 2.4: Resources used to implement different proposed CORDIC based Izhikevich models on Spartan-6 XC6LX75.

Model	Slice Registers	Slice LUT's	Max Speed (MHz)
IZHCOR6	229	410	183.4
IZHCOR8	232	413	182.7
IZHCOR10	234	418	181.4
IZHCOR12	236	421	180.1

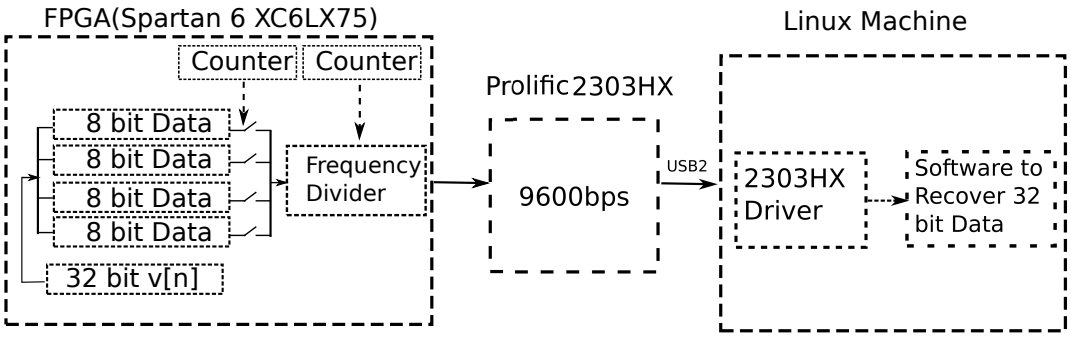


Figure 2.14: The method of transferring on-FPGA spiking neuron outputs to PC for analysis.

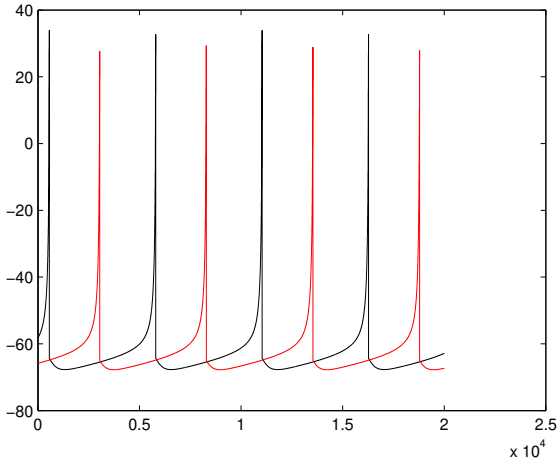
2.4.3 FPGA Implementation

Data flow graphs in Fig. 2.11 to 2.13 were described with VHDL hardware description language using Finite State Machines (FSMs). Further, the developed codes were first simulated using Modelsim for validation. Afterwards, the codes were synthesized by XILINX ISE XST synthesizer and implemented on the 45nm technology XILINX Spartan-6 XC6SLX75 FPGA.

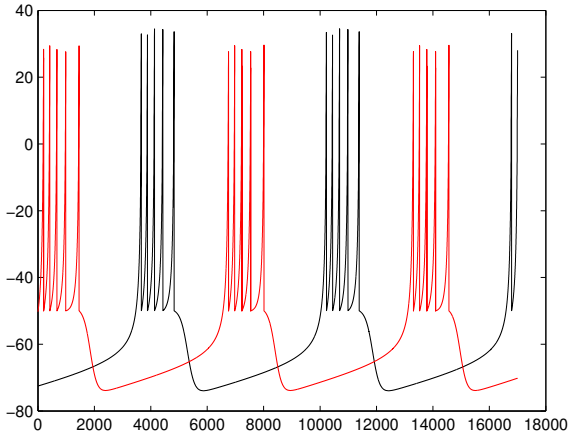
Since the utilized FPGA only supports Universal Asynchronous Receiver Transmitter (UART) port, a Prolific 2303HX chip and its driver were used to create virtual UART port in PC, through Universal Serial Bus (USB) port, to transfer the data from FPGA to PC for analysis. Furthermore, a UART transmitter and receiver module was added to the neuron VHDL code and implemented on FPGA as shown in Fig. 2.14. A counter was used to divide FPGA operation frequency to the chosen baud rate of the UART port (9600 bps). Data stream was structured as one start bit, eight data bits, one stop bit, and no parity. An additional counter was used to break the thirty-bit data in register V[n] into four bytes and send to UART port. Further, software was developed to receive and recover data from virtual UART port on Linux PC. Fig. 2.15 demonstrates the FPGA implementation and simulation results for two cases of tonic spiking and regular bursting of IZHCOR6. As it can be seen from the figure the FPGA implementation results well resemble the computer simulation results of the original Izhikevich model. To have a better comparison between the simulation and

Table 2.5: Comparison between proposed method and previously published works

Refrence	Slice Registers	Slice LUT's	Speed (MHz)	DSPs	NRMSD%	Errt%	Device
Soleimani et al [46].	493	617	241.9	0	-	1.54	Virtex-II Pro XC2VP30
Gomar et al. [59]	388	1279	190	0	4.02	-	Virtex-II Pro XC2VP30
Hayati et al. [60]	476	856	135	0	3.7	-	Virtex-II Pro XC2VP30
Grassia et al. [61]	646	1048	105	22	-	-	Virtex-5 XC5VLX50
Heidarpur et al. [50]	829	1221	134.3	0	0.04	0.39	Spartan-6 XC6SLX9
Shimada et al. [62]	357	1776	Asynchronous	-	-	-	Zync-7000 XC7Z020
IZHCOR6 (Area opt.)	229	410	183.4	0	0.003	0.26	Spartan-6 XC6SLX75
IZHCOR6 (Speed opt.)	280	469	212.8	0	0.003	0.26	Spartan-6 XC6SLX75



(A)



(B)

Figure 2.15: Spartan-6 XC6LX75 FPGA Implementation of CORDIC Izhikevich (red) and computer simulation of Izhikevich model (black). The FPGA Data was transferred to PC via UART-USB port. (A) Tonic Spiking and (B) Regular Bursting. Please note that the implementation data is scaled and an offset was added to it for closer behavior to the simulation.

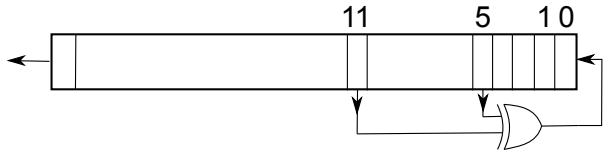


Figure 2.16: Linear Feedback Shift Register (LFSR) technique was used to generate semi-random input currents to feed the SNN input neurons.

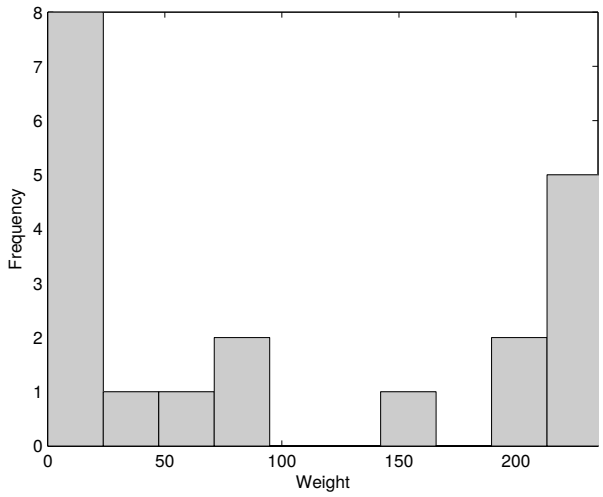


Figure 2.17: Bi-modal weight distribution reached after execution of the online on-FPGA STDP learning on a network of Izhikevich neurons.

Table 2.6: Total number and highest speed of CORDIC-based and original (implemented using DSP 36-bit multipliers) Izhikevich neurons that can be implemented on various FPGA devices.

Device	CORDIC		DSP Multiplier	
	Number	Speed	Number	Speed
Spartan-6 XC6LX75	110	183 MHz	22	44 MHz
Virtex-5 XC5VSX240T	365	220 MHz	176	102 MHz
Virtex-6 XC6VLX550T	835	332 MHz	144	111 MHz
Virtex-7 XC7VX980T	1490	370 MHz	600	130 MHz

Table 2.7: Utilized resources to implement the CORDIC (IZHCOR6) and original Izhikevich neuron on Spartan-6 XC6LX75

Resource	CORDIC	Original
Slice LUTs	410	370
Slice Registers	229	211
DSPs	0	6

Table 2.8: CORDIC and original neuron model on-FPGA power (reported by XILINX XPower Analyzer for the same frequency)

	CORDIC neuron	Original Model
On-FPGA power	71 mW	73 mW

the FPGA outputs, NRMSD and ERRT were calculated as shown in Table 2.1. These errors further confirm that the digital implemented CORDIC neuron has a similar behavior to the original model. The value of error is different from simulation results since the word length of fixed point hardware is 30 bit but computer simulations are performaed with single precision floating point numbers which has considerably higher precision.

To implement the spiking neural network with STDP learning and demonstrate its bi-modal behavior on FPGA, first, a Linear-Feedback Shift Register (LFSR) unit was designed to generate semi-random input spikes for the first layer neurons in the network shown in Fig. 2.7. The implemented LFSR is shown in Fig. 2.16. It is worth noting that, for other applications, the LFSR that generates random currents could be replaced with the spiking output of event-based sensors such as a silicon retina or cochlea. Fig. 2.17 demonstrates the bi-modal behavior reached after stimulating the implemented SNN on FPGA, with randomly generated input spikes generated using the on-FPGA LFSRs.

Table 2.9: Total Spartan-6 XC6LX75 FPGA utilization for implementation of CORDIC and 2^x online STDP on a network of CORDIC (IZHCOR8) Izhikevich neurons with topology of Fig. 2.7. These results includes semi-random input generator mechanism as well.

	Slice Registers	Utilization Perc.	Slice LUT's	Utilization Perc.	Max Speed (MHz)
CORDIC STDP	7,088	7%	10,376	22%	84.1
2^x STDP	7,047	7%	10,234	21%	84.5

2.5 Results and Discussion

Table 2.4 shows the amount of resources used to implement different CORDIC models of the Izhikevich neuron and the maximum speed reached using each of these models. As can be seen in this table, the resource usages of the four different implementations are close but for each higher precision model, extra time is needed to produce the new value of v. This delay can be calculated as:

$$T = \frac{1}{frequency} * n \tag{2.12}$$

Where T is the total time required to calculate the CORDIC square function and n is the number of the iterations. Considering the IZHCOR6 model (Area optimization goal) and frequency of device as 184 Mhz, total delay to calculate the result will be $6 * 5.5ns = 33ns$. DSP multiplier on the other side, operate at lower frequency of 44 Mhz but it need one clock to complete the results. DSP's total time can be calculated as $1 * 22.7ns = 22.7ns$. Still, the total delay is less than CORCID method. However, the architecture presented in this paper is not only consisted of the neurons, but also include the hardware to store the the spike times and the STDP algorithm to calculate and update the synaptic weights. The DSP multiplier reduces the frequency of the FPGA, resulting in other units to perform much slower. This in turn, increases total delay and reduces the throughput of the system.

In addition, in Table 2.5, the device utilization, speed, NRMSD, and ERRT are compared with some previously published works where a single neuron model is implemented on FPGA. Since the FPGA devices and synthesizer used are different in these works, this table results should be considered relatively. However, it can be seen that the proposed Izhikevich device consumes fewer resources while having higher speed compared to previous works.

Furthermore, Table 2.6 shows the number of CORDIC and original Izhikevich neurons that could be implemented on some FPGAs devices and compares the speed of both methods. The resources utilized for implementation of the CORDIC and DSP based neuron is presented in the Table 2.7. In implementation of CORDIC model, the number of neurons is limited by available LUTs in FPGA. Total number

of LUTs in Spartan-6 XC6LX75 is 46648. Therefore, the number of neurons was calculated as:

$$N = \frac{Available\ LUTs}{Utilized\ LUTs} = \frac{46648}{410} \approx 110 \tag{2.13}$$

In the case of DSP based implementation, the number of neurons is limited by available DSPs. In Spartan-6 XC6LX75, there are 132 DSP slices available. Thus, dividing 132 to the number of utilized DSP slices which is 6, gives maximum number of neurons.

$$N = \frac{Available\ DSPs}{Utilized\ DSPs} = \frac{132}{6} = 22 \tag{2.14}$$

To implement the original model on FPGAs, 36-bit DSP multipliers were used and the results were truncated to 36 bits. However, multiplication in constants were still performed with shift and add operations, the same way as performed in the proposed CORDIC device. Despite this simplification in the original model, the proposed CORDIC method allowed a higher number of faster neurons to be implemented on all FPGAs.

Power consumption and density is another important concern when designing hardware. It is also, one of major issues that need to be resolved for massive large scale implementation of neuromorphic systems considering that building such systems has been one of the main motivations of this work. To measure the on-FPGA power, first we generated a value change dump file and then the XILINX XPower Analyzer was used to determine the circuits power. For the fair comparison, it is presumed that both circuits work at the same frequency (40 MHz). As it is shown in this table 2.8, the CORDIC neuron consumes slightly less power than the original neuron model implementation.

To evaluate the cost of the total SNN with STDP learning and random input spike generation, the network with the topology of Fig. 2.7 consisting of CORDIC Izhikevich neurons (Fig. 2.11), semi-random input generators (Fig. 2.16), and STDP learning synapses (Fig. 2.13 and Fig. 2.12), was implemented on FPGA. This is the same network that was used to successfully generate the bi-modal weight distribution due to competitive Hebbian Learning of STDP synapses. Table 2.9 reports the total resources and speed of the implemented network. As the table indicates, this STDP

learning spiking network only consumes around 29% of available FPGA resources and could therefore be scaled almost 3.5 times on a fairly cheap device like Spartan XA6SLX75.

In addition, the second row in Table 2.9 presents implementation result for 2^x method, which uses lower resources and has higher speed in comparison with the previous methods. However, as discussed earlier, the accuracy of this model is lower than the proposed CORDIC model.

Overall, the above results confirm the reliable functionality of the proposed CORDIC-based SNN with STDP Learning. These results also show that the proposed design can lead to more efficient and faster FPGA-based SNNs compared to the literature. It can therefore contribute to the design and implementation of low-cost and high-speed large-scale digital neuromorphic systems exploring unsupervised STDP learning. It is important to note that FPGA devices utilize more resources for hardware implementation than that of ASICs. Implementing such hardware on silicon will have considerably less cost and have better performance.

2.6 Conclusion

In this paper, a novel hardware was presented based on the CORDIC method for on-FPGA online STDP learning. This hardware proved to be accurate while requiring less FPGA resources and having higher speed compared to the original models and state-of-the-art designs. The CORDIC method was utilized because of the simplicity of its structure, since it only uses add and shift operations which could be cheaply implemented on hardware. In order to implement the proposed learning system, first, the CORDIC method was used to implement Izhikevich neurons and its accuracy was analyzed. Second, the STDP algorithm was adopted for online learning and modified using the CORDIC algorithm to improve hardware efficiency. Furthermore, error analysis was performed on computer simulation data to ensure the accuracy of the implemented CORDIC models. Consequently, hardware was designed, described in VHDL, and simulated for both neuron and learning mechanism. Finally, the models

were implemented on FPGA to form a spiking neural network composed of Izhikevich neurons and STDP synapses to demonstrate competitive Hebbian learning. The proposed CORDIC-based FPGA spiking network with STDP learning is a step toward simpler and more efficient hardware design for SNN with unsupervized STDP learning implemented on FPGAs and digital platforms.

References

[1] K. Boahen, "A neuromorph's prospectus," *Computing in Science and Engg.*, vol. 19, no. 2, pp. 14–28, Mar. 2017.

[2] L. A. Pastur-Romay, F. Cedron, A. Pazos, and A. B. Porto-Pazos, "Deep artificial neural networks and neuromorphic chips for big data analysis: Pharmaceutical and bioinformatics applications," *International Journal of Molecular Sciences*, vol. 17, no. 8, 2016.

[3] B. V. Benjamin, P. Gao, E. McQuinn, S. Choudhary, A. R. Chandrasekaran, J. M. Bussat, R. Alvarez-Icaza, J. V. Arthur, P. A. Merolla, and K. Boahen, "Neurogrid: A mixed-analog-digital multichip system for large-scale neural simulations," *Proceedings of the IEEE*, vol. 102, no. 5, pp. 699–716, May 2014.

[4] P. Merolla, J. Arthur, F. Akopyan, N. Imam, R. Manohar, and D. S. Modha, "A digital neurosynaptic core using embedded crossbar memory with 45pj per spike in 45nm," in *2011 IEEE Custom Integrated Circuits Conference (CICC)*, Sept 2011, pp. 1–4.

[5] S. B. Furber, F. Galluppi, S. Temple, and L. A. Plana, "The spinnaker project," *Proceedings of the IEEE*, vol. 102, no. 5, pp. 652–665, May 2014.

[6] J. Schemmel, D. Briiderle, A. Griibl, M. Hock, K. Meier, and S. Millner, "A wafer-scale neuromorphic hardware system for large-scale neural modeling," in *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*, May 2010, pp. 1947–1950.

[7] R. Wang, C. S. Thakur, G. Cohen, T. J. Hamilton, J. Tapson, and A. van Schaik, "Neuromorphic hardware architecture using the neural engineering framework for pattern recognition," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 11, no. 3, pp. 574–584, June 2017.

[8] C. S. Thakur, J. Molin, G. Cauwenberghs, G. Indiveri, K. Kumar, N. Qiao, J. Schemmel, R. Wang, E. Chicca, J. O. Hasler *et al.*, "Large-scale neuromor-

phic spiking array processors: A quest to mimic the brain," *arXiv preprint arXiv:1805.08932*, 2018.

[9] B. Sen-Bhattacharya, S. James, O. Rhodes, I. Sugiarto, A. Rowley, A. B. Stokes, K. Gurney, and S. B. Furber, "Building a spiking neural network model of the basal ganglia on spinnaker," *IEEE Transactions on Cognitive and Developmental Systems*, pp. 1–1, 2018.

[10] J. P. Dominguez-Morales, A. Rios-Navarro, D. Gutierrez-Galan, R. Tapiador-Morales, A. Jimenez-Fernandez, E. Cerezuela-Escudero, M. Dominguez-Morales, and A. Linares-Barranco, "Multilayer spiking neural network for audio samples classification using spinnaker," in *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2017, pp. 1–1.

[11] D. Khodagholy, J. N. Gelinas, T. Thesen, W. Doyle, O. Devinsky, G. G. Malliaras, and G. Buzsáki, "Neurogrid: recording action potentials from the surface of the brain," *Nature neuroscience*, vol. 18, no. 2, p. 310, 2015.

[12] S. Scholze, H. Eisenreich, S. Höppner, G. Ellguth, S. Henker, M. Ander, S. Hänzsche, J. Partzsch, C. Mayr, and R. Schüffny, "A 32 gbit/s communication soc for a waferscale neuromorphic system," *INTEGRATION, the VLSI journal*, vol. 45, no. 1, pp. 61–75, 2012.

[13] M. Chu, B. Kim, S. Park, H. Hwang, M. Jeon, B. H. Lee, and B. G. Lee, "Neuromorphic hardware system for visual pattern recognition with memristor array and cmos neuron," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 4, pp. 2410–2419, April 2015.

[14] S. Davies, "Learning in the spiking neural networks," Ph.D. dissertation, Univ. of Manchester, Manchester, 2012.

[15] M. R. Azghadi, N. Iannella, S. F. Al-Sarawi, G. Indiveri, and D. Abbott, "Spike-based synaptic plasticity in silicon: design, implementation, application, and challenges," *Proceedings of the IEEE*, vol. 102, no. 5, pp. 717–737, 2014.

[16] A. L. Hodgkin and A. F. Huxley, "A quantitative description of membrane current and its application to conduction and excitation in nerve," *The Journal of physiology*, vol. 117, no. 4, p. 500, 1952.

[17] E. M. Izhikevich, "Simple model of spiking neurons," *IEEE Transactions on neural networks*, vol. 14, no. 6, pp. 1569–1572, 2003.

[18] R. Brette and W. Gerstner, "Adaptive exponential integrate-and-fire model as an effective description of neuronal activity," *Journal of neurophysiology*, vol. 94, no. 5, pp. 3637–3642, 2005.

[19] R. FitzHugh, "Impulses and Physiological States in Theoretical Models of Nerve Membrane," *Biophysical Journal*, vol. 1, pp. 445–466, jul 1961.

[20] C. Morris and H. Lecar, "Voltage oscillations in the barnacle giant muscle fiber," *Biophysical Journal*, vol. 35, no. 1, pp. 193–213, 1981.

[21] H. R. WILSON, "Simplified dynamics of human and mammalian neocortical neurons," *Journal of Theoretical Biology*, vol. 200, no. 4, pp. 375–388, 1999.

[22] R. M. Rose and J. L. Hindmarsh, "The assembly of ionic currents in a thalamic neuron i. the three-dimensional model," *Proceedings of the Royal Society B: Biological Sciences*, vol. 237, no. 1288, pp. 267–288, 1989.

[23] W. Gerstner and W. M. Kistler, *Spiking neuron models: Single neurons, populations, plasticity*. Cambridge university press, 2002.

[24] W. Gerstner, W. M. Kistler, R. Naud, and L. Paninski, *Neuronal dynamics: From single neurons to networks and models of cognition*. Cambridge University Press, 2014.

[25] P. U. Diehl, G. Zarrella, A. Cassidy, B. U. Pedroni, and E. Neftci, "Conversion of artificial recurrent neural networks to spiking neural networks for low-power neuromorphic hardware," in *2016 IEEE International Conference on Rebooting Computing (ICRC)*, Oct 2016, pp. 1–8.

[26] D. Martí, M. Rigotti, M. Seok, and S. Fusi, "Energy-efficient neuromorphic classifiers," *Neural computation*, vol. 28, no. 10, pp. 2011–2044, 2016.

[27] R. Kreiser, T. Moraitis, Y. Sandamirskaya, and G. Indiveri, "On-chip unsupervised learning in winner-take-all networks of spiking neurons," in *2017 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, Oct 2017, pp. 1–4.

[28] H. Y. Hsieh, P. Y. Li, C. H. Yang, and K. T. Tang, "A high learning capability probabilistic spiking neural network chip," in *2018 International Symposium on VLSI Design, Automation and Test (VLSI-DAT)*, April 2018, pp. 1–4.

[29] J. s. Kim and S. Jung, "Implementation of the rbf neural chip with the on-line learning back-propagation algorithm," in *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, June 2008, pp. 377–383.

[30] E. Stromatias and J. S. Marsland, "Supervised learning in spiking neural networks with limited precision: Snn/lp," in *2015 International Joint Conference on Neural Networks (IJCNN)*, July 2015, pp. 1–7.

[31] A. M. Sheri, A. Rafique, W. Pedrycz, and M. Jeon, "Contrastive divergence for memristor-based restricted boltzmann machine," *Engineering Applications of Artificial Intelligence*, vol. 37, pp. 336 – 342, 2015.

[32] F. Grassia, L. Buhry, T. Levi, J. Tomas, A. Destexhe, and S. Saighi, "Tunable neuromimetic integrated system for emulating cortical neuron models," *Frontiers in NEUROSCIENCE*, vol. 5, p. 134, 2011.

[33] A. Morrison, M. Diesmann, and W. Gerstner, "Phenomenological models of synaptic plasticity based on spike timing," *Biological cybernetics*, vol. 98, no. 6, pp. 459–478, 2008.

[34] E. Covi, S. Brivio, M. Fanciulli, and S. Spiga, "Synaptic potentiation and depression in al: Hfo2-based memristor," *Microelectronic Engineering*, vol. 147, pp. 41–44, 2015.

[35] M. R. Azghadi, S. Moradi, D. B. Fasnacht, M. S. Ozdas, and G. Indiveri, "Programmable spike-timing-dependent plasticity learning circuits in neuromorphic vlsi architectures," *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, vol. 12, no. 2, p. 17, 2015.

[36] C. Mayr, J. Partzsch, M. Noack, S. Hanzsche, S. Scholze, S. Hoppner, G. Ellguth, and R. Schuffny, "A biological-realtime neuromorphic system in 28 nm cmos using low-leakage switched capacitor circuits," *IEEE transactions on biomedical circuits and systems*, vol. 10, no. 1, pp. 243–254, 2016.

[37] R. M. Wang, T. J. Hamilton, J. Tapson, and A. van Schaik, "A mixed-signal implementation of a polychronous spiking neural network with delay adaptation," *Frontiers in neuroscience*, vol. 8, p. 51, 2014.

[38] S. Yang, J. Wang, B. Deng, C. Liu, H. Li, C. Fietkiewicz, and K. A. Loparo, "Real-time neuromorphic system for large-scale conductance-based spiking neural networks," *IEEE Transactions on Cybernetics*, pp. 1–14, 2018.

[39] K. Isobe and H. Torikai, "A novel hardware-efficient asynchronous cellular automaton model of spike-timing-dependent synaptic plasticity," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 63, no. 6, pp. 603–607, June 2016.

[40] M. R. Azghadi, S. Al-Sarawi, D. Abbott, and N. Iannella, "A neuromorphic VLSI design for spike timing and rate based synaptic plasticity," *Neural Networks*, vol. 45, pp. 70–82, 2013.

[41] M. R. Azghadi, S. Al-Sarawi, N. Iannella, and D. Abbott, "Tunable low energy, compact and high performance neuromorphic circuit for spike-based synaptic plasticity," *PLoS ONE*, vol. 9, no. 2, p. art. no. e88326, 2014.

[42] C. Lammie, T. Hamilton, and M. R. Azghadi, "Unsupervised character recognition with a simplified fpga neuromorphic system," in *2018 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2018.

[43] L. P. Maguire, T. M. McGinnity, B. Glackin, A. Ghani, A. Belatreche, and J. Harkin, "Challenges for large-scale implementations of spiking neural networks on fpgas," *Neurocomputing*, vol. 71, no. 1, pp. 13–29, 2007.

[44] T. Matsubara, H. Torikai, and T. Hishiki, "A generalized rotate-and-fire digital spiking neuron model and its on-fpga learning," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 58, no. 10, pp. 677–681, Oct 2011.

[45] N. Shimada and H. Torikai, "A novel asynchronous cellular automaton multicompartment neuron model," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 62, no. 8, pp. 776–780, 2015.

[46] H. Soleimani and E. M. Drakakis, "An efficient and reconfigurable synchronous neuron model," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. PP, no. 99, pp. 1–1, 2017.

[47] H. Soleimani, A. Ahmadi, and M. Bavandpour, "Biologically inspired spiking neurons: Piecewise linear models and digital implementation," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 59, no. 12, pp. 2991–3004, Dec 2012.

[48] M. Heidarpur, A. Ahmadi, and N. Kandalaft, "A digital implementation of 2d hindmarsh–rose neuron," *Nonlinear Dynamics*, vol. 89, no. 3, pp. 2259–2272, Aug 2017.

[49] A. Elnabawy, H. Abdelmohsen, M. Moustafa, M. Elbediwy, A. Helmy, and H. Mostafa, "A low power cordic-based hardware implementation of izhikevich neuron model," in *2018 16th IEEE International New Circuits and Systems Conference (NEWCAS)*. IEEE, 2018, pp. 130–133.

[50] M. Heidarpour, A. Ahmadi, and R. Rashidzadeh, "A cordic based digital hardware for adaptive exponential integrate and fire neuron," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 63, no. 11, pp. 1986–1996, Nov 2016.

[51] S. Gomar and M. Ahmadi, "Digital realization of pstdp and tstdp learning," in *2018 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2018, pp. 1–5.

[52] B. Belhadj, J. Tomas, O. Malot, G. N'Kaoua, Y. Bornat, and S. Renaud, "Fpga-based architecture for real-time synaptic plasticity computation," in *2008 15th IEEE International Conference on Electronics, Circuits and Systems*, Aug 2008, pp. 93–96.

[53] C. Lammie, T. J. Hamilton, A. van Schaik, and M. R. Azghadi, "Efficient fpga implementations of pair and triplet-based stdp for neuromorphic architectures," *IEEE Transactions on Circuits and Systems I: Regular Papers*, pp. 1–13, 2018.

[54] E. M. Izhikevich, "Which model to use for cortical spiking neurons?" *IEEE transactions on neural networks*, vol. 15, no. 5, pp. 1063–1070, 2004.

[55] J. E. Volder, "The cordic trigonometric computing technique," *Electronic Computers, IRE Transactions on*, vol. EC-8, no. 3, pp. 330–334, Sept 1959.

[56] R. J. Hyndman and A. B. Koehler, "Another look at measures of forecast accuracy," *International Journal of Forecasting*, vol. 22, no. 4, pp. 679–688, 2006.

[57] C. Borgers, *Spike Timing-Dependent Plasticity (STDP)*. Cham: Springer International Publishing, 2017, pp. 349–359.

[58] S. P. Mohanty, *Low-power high-level synthesis for nanoscale CMOS circuits*. Springer, 2008.

[59] S. Gomar and A. Ahmadi, "Digital multiplierless implementation of biological adaptive-exponential neuron model," *Circuits and Systems I: Regular Papers, IEEE Transactions on*, vol. 61, no. 4, pp. 1206–1219, April 2014.

[60] M. Hayati, M. Nouri, S. Haghiri, and D. Abbott, "Digital multiplierless realization of two coupled biological morris-lecar neuron model," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 62, no. 7, pp. 1805–1814, July 2015.

[61] F. Grassia, T. Levi, T. Kohno, and S. Saïghi, "Silicon neuron: digital hardware implementation of the quartic model," *Artif Life Robotics*, vol. 19, no. 3, pp. 215–219, 2014.

[62] N. Shimada and H. Torikai, "A novel asynchronous cellular automaton multi-compartment neuron model," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 62, no. 8, pp. 776–780, Aug 2015.

CORDIC-Astrocyte: A Tripartite Glutamate-IP3-Ca^{2+} Interaction Dynamics on FPGA

3.1 Introduction

During the last decades, researchers have been trying to replicate and study brain for both medical and information processing applications. Some medical objectives include understanding neurological psychiatric diseases [1, 2], simulating drug treatment [3] alongside designing brain computer interfaces for the people with sensory, motor and cognitive disabilities [4, 5]. However, this study is most important from information processing point of view, where it can eventually lead to a new generation of computational devices [6]. Resembling to brain, such processors expected to be intelligent, low power and fast [7, 8, 9, 10]. Nevertheless, contrary to nowadays computers, brain inspired systems are tolerant to both hardware and data failures [11].

One approach to brain study is dividing it into two hierarchical sub levels of components and architectural. Components level, which is the lowest, concerns studying and mathematically modelling properties of cells and the way they are interconnected and interact with each other. The architectural level, however, deals with complex behaviours that arise when large number of these components are connected which

include learning and information processing algorithms in the brain.

At the cell level, this work concentrates on the astrocyte cells. As the new researches revealed, beside physiological roles, astrocytes also are active partners of the neurons in the processing of the information in the brain. Astrocytes sense and respond to synaptic activity by releasing glio-transmitters and activation of different receptors and transporters. Thereby, they effect and regulate synaptic transmission and neuronal plasticity [12, 13, 14, 15]. Considering astrocyte as the third partner, the new synapse, called triparitie synapse, now consists of the post-synpatic neuron, pre-synpatic neuron and astrocyte [16]. Each astrocyte cell can potentially regulate function of not only one, but several neighbouring synapses [17]. Moreover, astrocytes are connected to each other through gap junctions, forming their own astrocytic network. Such connectivity is believed to be associated with processes leading to epilepsy disease [18]. Astrocytes are also involved in, and considered as a target for therapies of neurodegenerative disorders such as Alzheimer [19] and motor neuron diseases [20]. Additionally, experimental results suggest that astrocytes are associated with sleep development and functions, where impact of a astrocyte on a synapse could transform into large scale neural modulations [21]. Furthermore, neural networks that include astrocytes alongside neurons, are shown to exhibit self-detection of faults and self-repairing capability [22]. Given all the mentioned roles, astrocytes are important players and need to be incorporated for studies of biological neural networks.

Several models presented to describe astrocytes [23, 24, 25, 26, 27, 28]. In [23], a generalized and simplified model for astrocyte and neural-astrocyte interactions is presented. Postnov model is helpful to study the way astrocytes affect responses and the dynamical patterns in a network of simplified two-dimensional neurons. Examples of these neurons are models in [29, 30]. On the other hand, there are biologically-plausible astrocyte models with biophysical parameters that describe cellular phenomena and cell processes. Recently, these models are gaining more attention by researchers. For instance, they used such models to evaluate role of calcium dynamics in brain [31] [32].

In this work, the biological-plausible model in [25] was adopted for simulation

and hardware implementation for following reasons: First, astrocytes do not generate action potentials the way neurons do, but propagate Ca^{2+} waves and release transmitters in response to stimulation. Therefore, they require a mathematical description to account for ion concentrations variations, conductance of cell for ion inward and outward, steady state activation/inactivation and properties of the individual components and their effects of the cells. Second, it is among the few models that incorporates the glutamate regulation of Inositol triphosphate (IP3) and Ca^{2+} dynamics. Third, by changing biophysical parameters in this model, glutamate stimulation can act as Frequency Modulator (FM), Amplitude Modulator (AM) or both Frequency-Amplitude Modulator (AFM) of the Ca^{2+} oscillation. Overall, the simulation objectives and computational resources determine what models with what level of biological details are needed.

In architectural level, brain cells are connected together through their different interaction mechanisms and their systematic behaviour is analyzed. This is useful both to study biological functions and disorders as well as discovering computational algorithms underlying information processing, learning, memory, etc. However, high complexity of such systems, due to large number of cells [33] and numerous communication pathways, make such simulation a difficult task. For the simulation platform, computers are flexible and the best choice for small networks [34]. However, for large networks, supercomputers, which are not available to everyone, must be utilized [35] to run the simulation in a practical time span. Moreover, some computers are modified for efficient simulation of neural networks [36]. Researchers also used Graphics Processing Unit (GPU) to run and simulate spiking neural networks [37].

Another platform is dedicated Application Specified Integrated Circuit (ASIC) hardware which have several advantages over the computers. First, they are faster and, unlike fetch and decode computers, all processing units (cells) operate in parallel analogous to biological networks. ASICs are more efficient in the term of the power and cost comparing to super computers but still the design process requires a lot of engineering work and time. Besides, once the design fabricated, it would not be possible to update or change the design. As another alternative, re-configurable

digital platforms, i.e. Field Programmable Gate Arrays (FPGA), are widely used because they provide a re-configurable, easy to work with, cheap and reliable platform. However, resource limitation is one of main challenges of large scale neural network FPGA implementations [38]. The number of neural elements and size of the network could be limited by available resources like the slice registers, LUTs, block RAMs or DSPs depending on the design [39].

Several analog, digital and hybrid VLSI neuromorphic hardware [40, 41, 42, 43] are available with different type and number of neurons. However, the gap is, glia cells (almost half of brain) are missing in such hardware and no project includes them in a neuromorphic chip. To pave the way, researchers have proposed different analog [44, 45, 46] and digital [47, 48, 49, 50, 51, 52] circuits for astrocyte hardware implementation.

Liu et al. [47] propose hierarchical network-on-chip structure of neurons and astrocytes cells based on work in the [48] where FPGA implementation cost is reduced by sampling the constituent nonlinear curves. Karim et al. [50] implemented the biological-plausible model in FPGA using reduced bit precision. Reference [52] implemented the Postnov model [23] using Piece Wise Liner approximation (PWL) [53] technique to approximate the nonlinear terms in the astrocyte equations. Some of the problems associated with this method is outlined in the [54]. Beside those, the model proposed by Postnov is a high-level model, where biological reactions and functions are simplified. This work presents a COordinate Rotation Digital Computer (CORDIC) [55] based digital implementation of a biologically plausible astrocyte and glutamate release mechanism. This method can calculate the nonlinear functions with very high precision while it is very well suited for digital hardware implementation.

The rest of this Chapter is organized as follows. The biological-plausible model for astrocyte and glutamate release is introduced in Section 3.2. Section 3.3 presents CORDIC based models and evaluates the accuracy of the model through simulations. Hardware design and FPGA implementation are discussed in Section 3.4 and 3.5. The chapter concludes in Section 3.6.

3.2 Background

In the following sections, synaptic transmission mechanism and the way it is affected by astrocyte is briefly explained.

3.2.1 Pre and Post-Synaptic Neurons

When a pre-synaptic neuron fires an action potential, Ca^{2+} in the pre-synaptic bouton will increase through fast and slow pathways. This increase, activates glutamate release from the bouton to synaptic cleft. Subsequently, this glutamate binds to the post-synaptic terminal receptors which may result in depolarization of the post-synaptic terminal and eventually, in a excitatory post-synaptic action potential. These processes are illustrated in the Fig. 3.1.

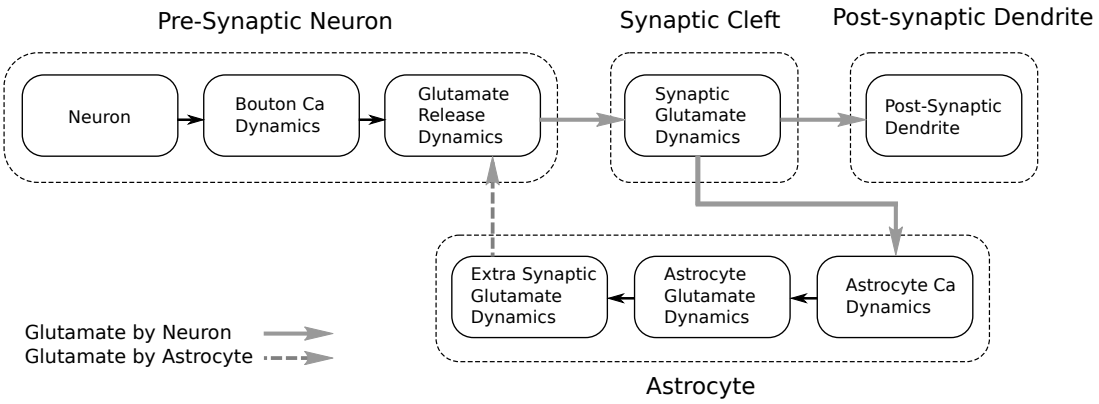


Figure 3.1: Arriving an action potential in the pre-synaptic terminal increases Ca^{2+} in the pre-synaptic bouton. This will activate the release of the glutamate into synaptic cleft. The glutamate in the synaptic cleft binds to the post-synaptic terminal receptors which may contribute to a spiking post-synaptic neuron. The glutamate could also bind to astrocyte receptors and modulations of Ca^{2+} dynamics in the asrtocyte. In a similar mechanism to pre-synaptic bouton, increasing Ca^{2+} concentration will result in extra glutamate release by astrocyte. Such release will affect the pre-synaptic bouton Ca^{2+} dynamics and regulates the synaptic transmission.

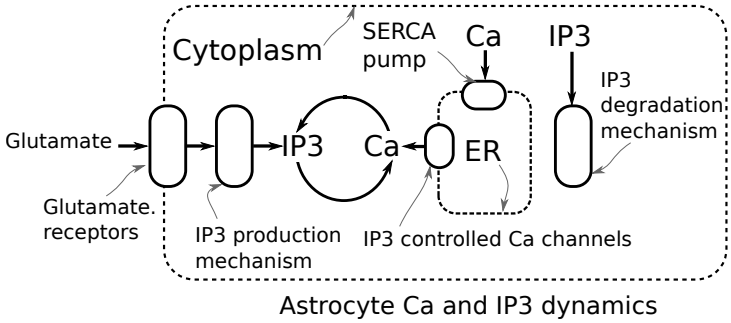


Figure 3.2: Binding glutamate to the astrocyte receptors activates IP3 production mechanism. Increasing IP3 opens the IP3 controlled calcium channels which results in flowing Ca^{2+} from ER into cytoplasm. More Ca^{2+} ions, leads to even more IP3 production. The cycle continues until the action of Ca^{2+} release, reverses at high Ca^{2+} concentration where SERCA pump quickly draws back the excess cytoplasmic Ca^{2+} into ER. Subsequently, extra IP3 will also be removed by IP3 degradation mechanisms. If glutamate stimulation remains high enough, this process repeats and results in Ca^{2+} and IP3 oscillation in the astrocyte.

3.2.2 Astrocyte Ca^{2+} Oscillation

The glutamate in synaptic cleft may also spread and bind to receptors in processes of neighbouring astrocytes. With activation of these receptors, cytosolic IP3 concentration in astrocyte slightly increases. In turn, this increase, triggers a rise in the level of Ca^{2+} in the cell cytoplasm. More Ca^{2+} ions, will result in even more IP3 production. The cycle continues until action of Ca^{2+} release, reverses at high Ca^{2+} concentration. At this point, IP3 production is deactivated and SERCA pump quickly draws back the excess cytoplasmic Ca^{2+} into Endoplasmic Reticulum (ER).

The intracellular Ca^{2+} concentration consequently recovers toward basal value and suppresses IP3 production. If glutamate stimulation continues, intracellular IP3 remains high enough to repeat the cycle into oscillations of Ca^{2+} and IP3 ions. This process is depicted in the Fig. 3.2. The mathematical equations that describe Ca^{2+} and IP3 dynamics in astrocyte are presented in Eq. 3.1 to 3.8. The parameters values for these equations for both AM and FM cases are presented in the Table 3.1. For

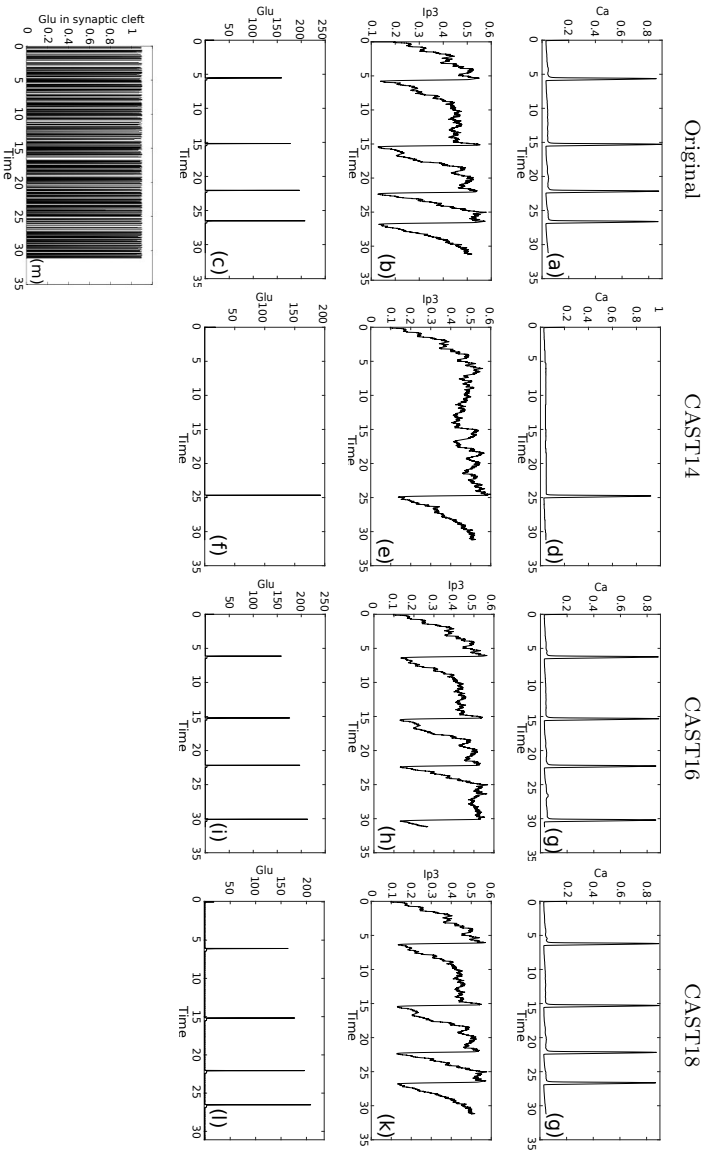


Figure 3.3: Simulation of the astrocyte and glutamate production mechanism of the original and CORDIC models. First column shows the simulation result for the original model, second column *CAST14*, third column *CAST16* and last column *CAST18* . As it can be seen from this figures, *CAST14* does not follow the original model while *CAST16* has closer behaviour. Finally, *CAST18* has exact output waveform as the original astrocyte model. The glutamate stimulation that applied to all models is shown in (m).

more details and description of parameters, please refer to [25].

$$\begin{aligned}\frac{dI_{p3}}{dt} &= \nu_\beta Hill(1,\gamma,K_R(1+\frac{K_p}{K_r}Hill(1,Ca,K_\pi))) \\ &+ \frac{\nu_\delta}{1+\frac{I_{p3}}{k_\delta}}Hill(2,Ca,K_{PLC\delta}) \\ &- \nu_{3K}Hill(4,Ca,K_D)Hill(1,I_{p3},K_3) - r_{5P}I_{p3}\end{aligned} \tag{3.1}$$

$$\begin{aligned}\frac{dCa}{dt} &= (r_C m_\infty^3 n_\infty^3 h(i)^3 + r_L)(C_{ER}-Ca) \\ &- \nu_{ER}Hill(2,Ca,K_{ER})\end{aligned} \tag{3.2}$$

$$\frac{dh}{dt} = \frac{(h_\infty - h(i))}{\tau_h} \tag{3.3}$$

$$\tau_h = \frac{1}{\alpha_2(Q_2+C_a)} \tag{3.4}$$

$$Q = \frac{(d_2 I + d_1)}{(I+d_3)} \tag{3.5}$$

$$h_\infty = \frac{Q}{(Q_2+C_a)} \tag{3.6}$$

$$m_\infty = Hill(1,I,d1) \tag{3.7}$$

$$n_\infty = Hill(1,C_a,d5) \tag{3.8}$$

where C_a and I_{p3} are the concentration of Ca^{2+} and IP3 in the astrocyte cytoplasm, respectively. The Hill equation [56] is one of the common and useful equations in biochemistry which is given by:

$$Hill(n,x,y) = \frac{x^n}{x^n+y^n} \tag{3.9}$$

The power of the first Hill function in IP3 original model was 0.7 but here for simplifying power 1 was used.

3.2.3 Astrocyte Glutamate Production

There is a debate over the exact mechanism of transmitter release in astrocyte. However, it is widely believed that glio-transmitter release is similar to that of neurotransmitters in neurons. The mathematical model for glutamate release is given in

Eq. 3.10 to 3.15. To read more about these equations and their parameters please refer to [57]. The Ca^{2+} ions must bind to three independent gates (O_1, O_2 *and* O_3) for possible transmitter release.

$$\frac{dO_1}{dt} = k_1^+ Ca - (k_1^+ Ca + k_1^-)O_1 \tag{3.10}$$

$$\frac{dO_2}{dt} = k_2^+ Ca - (k_2^+ Ca + k_2^-)O_2 \tag{3.11}$$

$$\frac{dO_3}{dt} = k_3^+ Ca - (k_3^+ Ca + k_3^-)O_3 \tag{3.12}$$

$$\frac{dR_a}{dt} = \frac{I_a}{\tau_{rec}^a} - \mathcal{H}(Ca - Ca^{th}) f_r^a R_a \tag{3.13}$$

$$\frac{dR_a}{dt} = -\frac{E_a}{\tau_{inc}^a} - \mathcal{H}(Ca - Ca^{th}) f_r^a R_a \tag{3.14}$$

$$\frac{dG_a}{dt} = n_a^V g_a^V E_a - g_a^C G_a \tag{3.15}$$

$$f_r^a = O_1 O_2 O_3 \tag{3.16}$$

$$I_a = 1 - R_a - E_a \tag{3.17}$$

where $\mathcal{H}$ is Heaviside function defined as:

$$\mathcal{H}(x) = \begin{cases} 1, & \text{if } x \geq 0. \\ 0, & \text{otherwise.} \end{cases} \tag{3.18}$$

and G_a is glutamate released by astrocyte in the synaptic cleft. The values for these equations parameters are presented in the Table 3.2.

3.3 CORDIC Astrocyte Model

In this section, astrocyte and glutamate release equations are modified for efficient digital implementation. Afterwards, the proper function and accuracy of the modified model is investigated through computer simulations.

Table 3.1: The parameters used for simulation of astrocyte. The original parameters are scaled for simulation and hardware implementation.

Parameter	Value	Parameter	Value
r_C	6	a_2	0.2
r_L	0.11	v_δ	AM=0.02 FM=0.05
C_0	2	$K_{PLC\delta}$	0.1
c_1	0.185	k_δ	1.5
v_{ER}	0.9	K_π	0.6
k_{ER}	AM=0.1 FM=0.05	r_{5p}	AM=0.04 FM=0.05
d_1	0.13	v_{3k}	2
d_2	1.049	K_D	0.7
d_3	0.9434	K_3	1
d_5	0.08234	v_β	AM=0.2 FM=0.5
K_R	1.3	K_p	10

Table 3.2: The parameters used for simulation of equations describing glutamate release (Eq. 3.10 to 3.15). The parameters are scaled for simulation and hardware implementation.

Parameter	Value	Parameter	Value	Parameter	Value
k_1^+	0.0375	k_2^+	0.0250	k_3^+	0.125
k_1^-	0.0040	k_2^-	0.01	k_3^+	0.1
τ_{rec}^a	80	Ca^{th}	0.1967	g_a^v	20
τ_{inc}^a	0.3	n_a^v	12	g_a^c	100

3.3.1 CORDIC Based Astrocyte and Glutamate Release

What make hardware implementation of biological systems challenging and cost expensive are nonlinear terms describing the biochemical reactions in the cells. For instance, calculation of Hill function in Eq. 3.9 requires $2n$ times multiplication and also a division. Computation of a new value for Ca^{2+}, IP3 and gating variables (Eq. 3.1 to 3.8 and Eq. 3.10 to 3.15) involve several times calculation of Hill and other nonlinear functions.

To overcome this problem, CORDIC algorithm was used to compute these non-

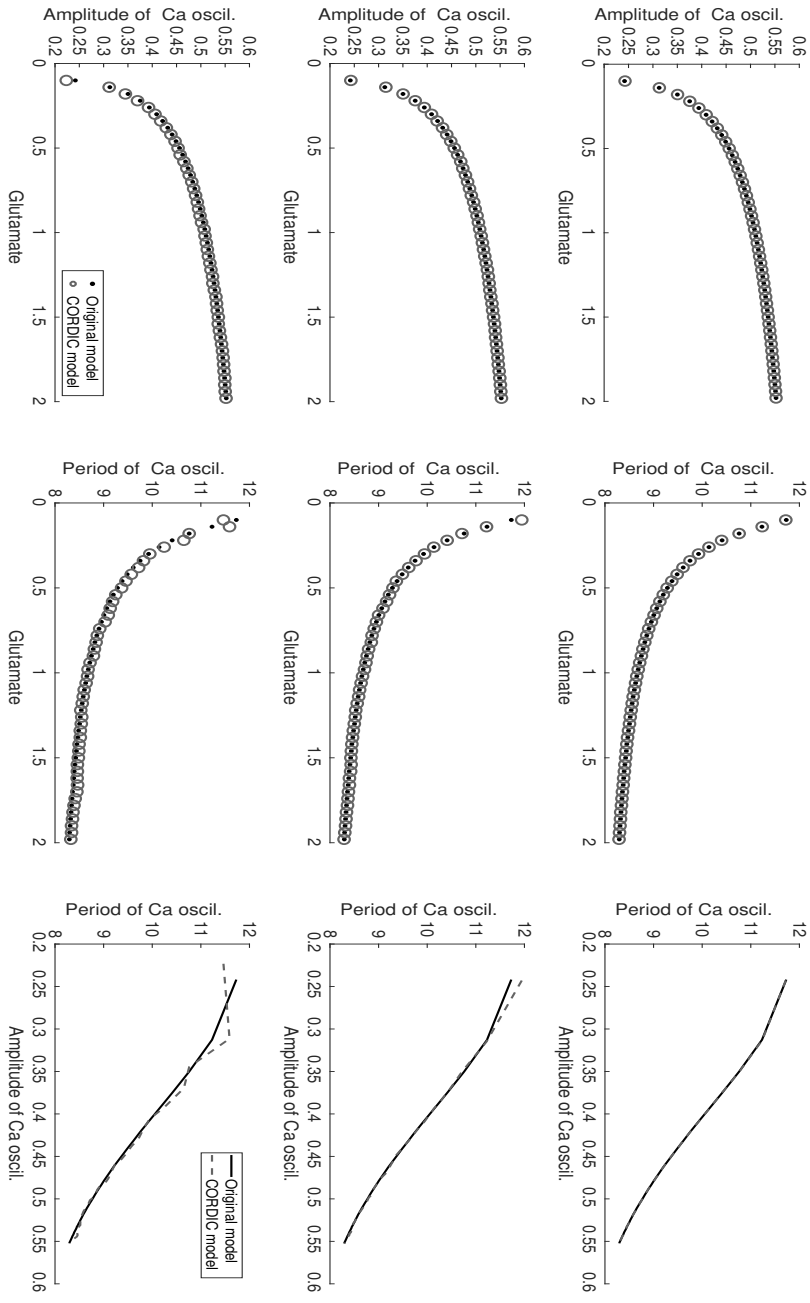


Figure 3.4: Ca^{2+} oscillation's period, amplitude and the ratio of these two for AM CORDIC astrocyte (dots-solid lines) and original (circles-dashed lines) models are shown in this figure. The first row is corresponding to data for *CAST14*, the second is for *CAST16* and the last *CAST18*. As it is evident from the figure, *CAST18* follows the original model with high accuracy while other models have deviations.

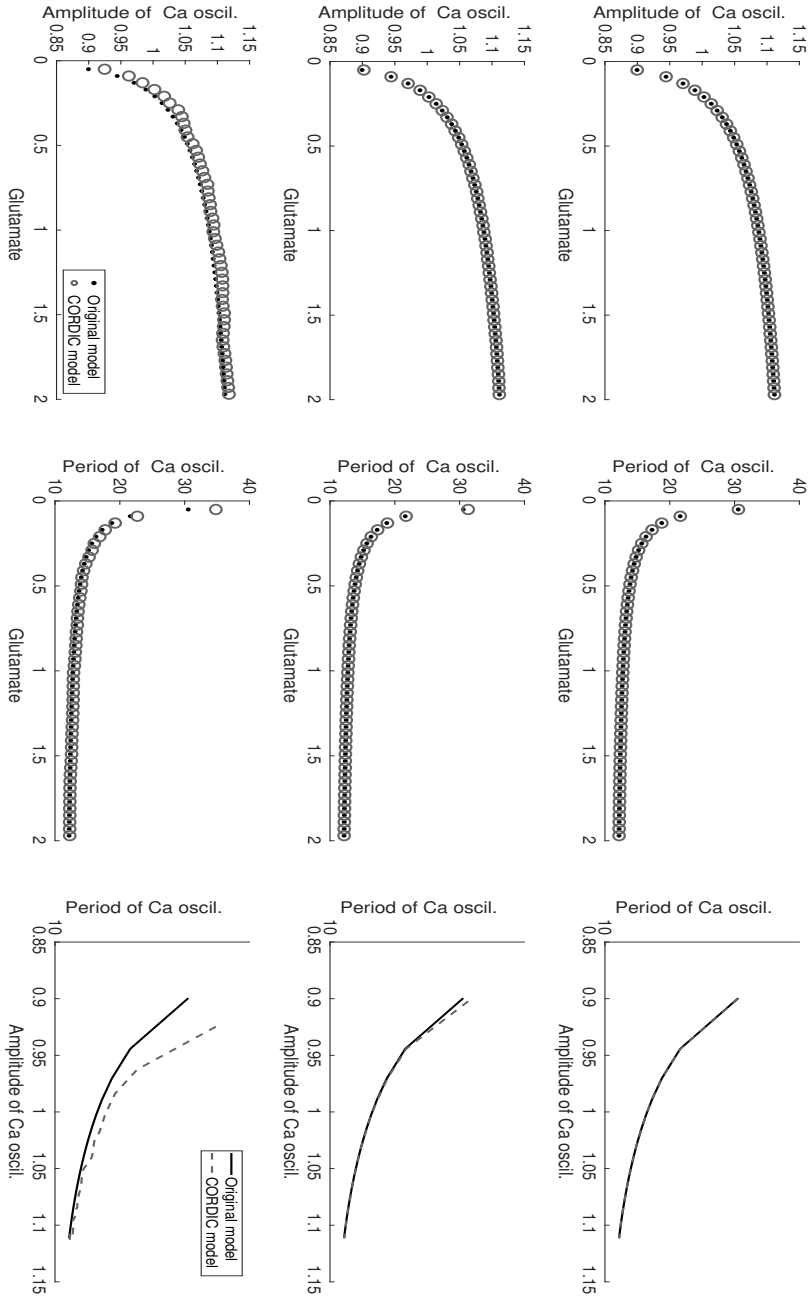


Figure 3.5: Ca^{2+} oscillation's period, amplitude and the ratio of these two for FM CORDIC astrocyte (dots-solid lines) and original (circles-dashed lines) models are shown in this figure. The first row is corresponding to data for *CAST14*, the second is for *CAST16* and the last *CAST18*. As it is evident from the figure, *CAST18* follows the original model with high accuracy while other models have deviations.

linear terms. This method was preferred and selected primarily for three reasons. First, this technique can be used for calculation of many nonlinear functions such as trigonometric, exponential, square roots, etc. Second, since it only requires shift and addition operations, could be effectively implemented on hardware. Third, CORDIC has a very high precision comparing to other methods such as PWL. Considering those, a CORDIC core was developed to perform nonlinear terms including Hill function. Comparing with conventional calculation of these operations, CORDIC core generates n-bit result for a given two n-bit inputs, instead of first calculating 2×n-bit and thereafter truncating results to n-bit. Thereby, it decreases the implementation cost to half. As for precision, experimental results show that, despite having simpler structure, CORDIC core has considerably low error from floating-point results.

CORDIC is an iterative algorithm and n is presumed to be number of iterations required to calculate nonlinear functions. More number of iterations, reduces the error from floating-point results but it also increases the total computational time. Therefore, choosing n is a trade-off between computation complexity and precision. To investigate this trade-off, we used three different values of $n = 14$, $n = 16$ and $n = 18$ to calculate astrocyte equations. This contributes to comparing results in the terms of variance from original model and also hardware cost and speed. We refer to these models as *CAST14*, *CAST16* and *CAST18* according to the number of their iterations.

3.3.2 Simulation Results

In this section, the accuracy of CORDIC based models are investigated through computer simulations. For this purpose, first, Maximum Deviation (MD) and Normalized Root Mean Square Deviation (NRMSD) errors were calculated to measure differences between the results of CORDIC and original operations. These errors are defined as:

$$MD = Max(|COR(x,y) - OR(x,y)|) \tag{3.19}$$

$$RMSD = \sqrt{\frac{\sum_{k=1}^{n} (COR(x,y) - OR(x,y))^2}{n}} \tag{3.20}$$

For $0.01 \leq x, y \leq 20$, these errors were calculated and presented in Table 3.3. It

Table 3.3: MD and NRMSD errors to measure difference between CORDIC and original multiplication and division.

Iterations	Error	Multiplication	Division
n=14	RMSD	13×10^{-3}	11×10^{-4}
	MD	15×10^{-4}	38×10^{-7}
n=16	RMSD	33×10^{-4}	28×10^{-5}
	MD	95×10^{-6}	23×10^{-8}
n=18	RMSD	81×10^{-5}	8870×10^{-6}
	MD	59×10^{-7}	2314×10^{-9}

is evident from the table data that CORDIC units have a very high precision. Next, new astrocyte differential equation was formed using CORDIC units. To evaluate similarity of CORDIC to original astrocyte model, both were stimulated with same glutamate input and Ca^{2+} oscillation were observed. The input was taken from simulation of glutamate in the synaptic cleft. Fig. 3.3 shows Ca^{2+} and IP3 oscillations alongside released glutamate by astrocyte for the proposed and original model. As this figure indicates, *CAST14* model does not follow the original model while *CAST16* has closer behaviour. However, *CAST18* has a output waveform exactly the same as the original astrcoyte model.

To further study the behaviour of the CORIDC model, for both AM and FM cases, period and amplitude of Ca^{2+} oscillation are plotted against different values of the glutamate in the Fig. 3.4 and 3.5. For the AM case (Fig. 3.4), the amplitude and frequency change diagrams of *CAST14* and *CAST16* differs from that of the original model. On the other hand, those for *CAST18* are identical to original model diagrams. The similar behaviour also was observed for the FM case as shown in the Fig. 3.5. Considering results of these simulations, *CAST18* model was selected for digital implementation. Nevertheless, depending on the application, one can choose smaller number of iterations to increase the speed of the hardware or choose larger number to achieve higher precision.

Lastly, to verify the accuracy of CORDIC astrocyte model and also studying the

astrocyte behaviour, a triangle wave of glutamate were applied to the both original and CORDIC models. Fig. 3.6 shows Ca^{2+}, IP3 and gating variable in the AM astrocyte as function of glutamate in the synaptic cleft. As it can be seen from this figure, the amplitude of Ca^{2+} oscillations increases with the amount of the glutamate, indicating the amplitude modulation. Moreover, Fig. 3.7 shows how frequency of Ca^{2+} oscillations is modulated with glutamate. It worth noting that in either cases, both AM and FM exist simultaneously and modulation is rather FAM. However, effect of one of these two is more prevalent. The results of these simulations also confirm that CORDIC model closely follows the original astrocyte model.

3.4 Hardware Implementation

This section presents hardware design and FPGA implementation of CORDIC and DSP based models.

3.4.1 Hardware Design

As the first step, astrocyte and glutamate release differential equations (3.1-3.6 and 3.10-3.17) were discretized using Euler method. As discussed before, numerical calculation of these equations require a large number of multiplication and division operations. In order to design the hardware, these operations must be properly scheduled in time for the hardware to use minimum number of these units while having shortest possible delay. This makes the scheduling of these operations crucial. Selecting number of steps and arithmetic units is a trade off between delay and area. More number of steps results in lower cost but at the same time, it increases the total delay and vise versa. Here, similar to previous section, we put different constraints on the number of multiplication and division in each step and developed three models to compare the results:

- *ASTRO2*: One multiplication and one division in a step
- *ASTRO3*: Two multiplications and one division in a step

- *ASTRO5*: Three multiplications and two divisions in a step

Considering those constraints, we tried to decrease number of steps as much as possible. The scheduling diagram for implementation of *ASTRO5* is shown in the Fig. 3.8 (a). The CORDIC multiplication and division are shown as $c*$ and $c/$, respectively. To further optimize the design, where it was accurate and efficient, multiplication in, or division to constants were performed with shift and add operations. To do this, the constants were broken into sum of powers of two as $c = \sum_{-n}^{m} 2^i$. Thereafter, multiplication performed as:

$$c * x = \sum_{-n}^{m} x2^i \tag{3.21}$$

This operation is shown in the Fig. 3.8 (a) with "<<" symbol. Fig. 3.8 (b) presents the hardware for implementation of CORDIC division and multiplication. As can be seen from the figure, hardware has simple structure. Nevertheless, it could perform division and multiplication fast with high precision. The operation mode can change with *SL* signal.

Considering the fact that reducing the hardware area and improving its speed was the primary goal of this work, the fixed-point arithmetic was used in the design. Next, the word length of the design was specified considering the range and resolution of the variables. Moreover, reserve bits added to to prevent underflow and overflow during the shift operations. Taking into account these requirements, 13 and 17 bits were dedicated for the fraction and integer parts, respectively.

Similar method was also used for discretization and scheduling of glutamate release equations. However, number of multiplication and division operations is smaller than that of the astrocyte and, therefore, it was scheduled in six steps with one multiplier and one divider in each step. Further in this paper, we refer to combined astrocyte and glutamate release mechanism as *GASTRO*.

3.4.2 FPGA Implementation

To implement the designs, Finite State Machines (FSM) were developed and described using VHDL hardware description language. After that, developed VHDL

codes were simulated in Modelsim software to check their validity and thereafter synthesized by Xilinx ISE XST and downloaded into Zynq-XC7Z020-CLG484 FPGA on the Zedboard development board. The on-FPGA data were converted to analog using digital to analog converter of the VGA port. Fig. 3.9 shows Oscilloscope photos of the on-FPGA Ca^{2+}, IP3 and gating variable oscillation with glutamate input of *ASTRO5* for both AM and FM. As this figure shows, CORDIC astrocyte model implemented on FPGA well resembles the computer simulation results. For better measurement of difference between the implementation and computer simulation results, a UART transmitter and receiver core was implemented on FPGA. Furthermore, using UART to USB bridge, data were transferred to PC through USB port and recovered. Fig. 3.10 compares a Ca^{2+} spike from computer simulation with first, a spike from CORDIC and then, with one from DSP based implementation. As it can be seen from the figure, for the same amount of the glutamate input, the Ca^{2+} spike of CORDIC model is very closer to computer simulation comparing to DSP based implementation. Since both designs are identical except for CORDIC and DSP units, this error is probably induced by truncation of the results.

3.5 Results and Discussion

In this section, to evaluate performance of CORDIC units, and also comparing area and latency of different scheduling strategies, all three models of *GASTRO2*, *GASTRO3* and *GASTRO5* were implemented on FPGA, first using CORDIC units (*C_GASTRO*) and thereafter with FPGA embedded DSP multipliers and dividers (*D_GASTRO*). The results are discussed in following:

Area: Table 3.4 presents the resources used to implement these models alongside their corresponding speed. As it can be seen from data, the amount of slice LUTs that is needed for the implementation of *C_ASTRO2* is almost double compared to that of *D_ASTRO2*. Beside that, *D_ASTRO2* uses 4 DSP units of FPGA while *C_ASTRO2* uses none. The amount of resources needed for the implementation of *C_GASTRO*, *D_GASTRO* is also compared in this table. With more number of the multipliers and

Table 3.4: Resource utilization and speed for FPGA implementation of CORDIC models (*C_ASTRO*, *C_GASTRO*) and DSP based models (*D_ASTRO*, *D_GASTRO*) multipliers and dividers.

Model	Slice Registers	Slice LUTs	DSPs	MRUP	Frequency (MHz)	Delay(μs)	Throughput(Mb/s)
C_ASTRO2	1318	1156	0	%2.17	342	1.02	29.6
C_ASTRO3	1498	1293	0	%2.43	342	0.66	45.2
C_ASTRO5	1876	1455	0	%2.73	385	0.33	90.1
C_GASTRO5	3124	2540	0	%4.77	385	1.33	22.5
D_ASTRO2	1064	2188	4	%4.11	22	2.63	11.4
D_ASTRO3	1171	2220	8	%4.17	22	1.81	16.5
C_ASTRO5	1446	3454	12	%6.49	22	1.81	16.5
D_GASTRO5	2320	4104	20	%9.09	22	5.2	5.8

dividers in each step, the utilization of FPGA resources increases, both for CORDIC and DSP based models. However, such increase for DSP based models is much higher.

Table 3.5: Number of *ASTRO* and *GASTRO* models that can be implemented in different FPGA devices

Model	Spartan 6 XC6lx75	Virtex 6 XC6VHX380T	Artix 7 XC7A200T
C_GASTRO5	18	94	52
D_GASTRO5	6	58	32

Model	Kinetix 7 XC7K160T	Virtex 7 XC7VX980T
C_GASTRO5	39	240
D_GASTRO5	24	149

The number of *GASTRO* models that could be implemented on FPGA is limited by amount of available slice registers, slice LUTs or in the case of *D_GASTRO* models, by number of DSP blocks. Regarding this, we calculated the Resource Utilization Percentage (RUP) for each resource as:

$$\%RUP = \frac{Utilized\ Resources}{Available\ Resources} \times 100 \tag{3.22}$$

Afterwards, we determined the Maximum Resource Utilization Percentage (MRUP) as:

$$MRUP = Max\ RUP\ of\{DSPs,\ LUTs,\ Registers\} \tag{3.23}$$

Having MRUP, one can easily calculate how many number of *GASTRO* models can be implemented on FPGA. These numbers for *C_GASTR5* and *D_GASTRO* for different FPGA devices are presented in the Table 3.5. As data indicate, number of *C_GASTRO* models that could be implemented on FPGA, is higher than that of *D_GASTRO* models. Despite FPGAs that DSP blocks are already part of device whether one use them or not, when it comes to ASIC design, cost of adding a multiplier or divider will be considerable in the term of area, power and delay. Therefore, this ratio for the same die area in a ASIC hardware is expected to be much higher.

Speed: Using the DSP embedded multipliers and dividers considerably reduces the frequency of the hardware on FPGA. The corresponding data are presented in

Table 3.6: Reported device utilization and spreed for some of recent published works that implemented similar astrocyte model on FPGA

Reference	Slice Registers	Slice LUTs	DSPs	Frequency (MHz)
Karim et al [50]	-	21000	230	-
Martin et al [51]	16305	16182	-	-
Lui et al [22]	11666	11394	42	200
	Delay(μs)	Throughput(Mb/s)	Device	
	-	-	AlteraEP4SGX530KH40C2	
	-	-	Virtex-7 XC7VX485T	
	12.5	-	Virtex-7 XC7VX485T	

the Table 3.4. FPGA frequency for CORDIC based models is much higher. However, CORDIC is an iterative algorithm and requires several clock cycles to be completed. Total delay to calculate CORDIC multiplications or division operations could be calculated as:

$$D_{md} = \frac{1}{f} \times (n + 1) \tag{3.24}$$

Where n is the number of iterations (As discussed in section II) and f is frequency of the hardware. One extra clock was considered to transfer data between registers in FPGA. Furthermore, additional 7 clock cycles are required in the design to perform addition and shift operations. The total delay to calculate *C_ASTRO* could be defined as:

$$D = n_s \times D_{md} + 1/f \times 7 \tag{3.25}$$

Using a similar method, the delay of the astrocyte model with DSP units could be determined by considering $D_{md} = 2$. Since the word length of design is 30 bits, the throughput calculated as:

$$Throughput = \frac{30}{D} \tag{3.26}$$

The throughput and delay for the different astrocyte models implemented using CORDIC and DSP units are presented and compared in the Table 3.4. As it is evident from data, the astrocyte models implemented using CORDIC units perform approximately 4 times faster than the astrocyte model with DSP multipliers.

Furthermore, Table 3.6 presents reported device utilization and speed in works that implemented biological-plausible astrocyte models on FPGA. As it is evident from the table, the proposed models are far more faster and area efficient. It is also worth noting that implementing the same design on different FPGAs, will result in different area utilization and speeds. Therefore, on high end devices, the proposed design will run faster. Besides, higher number than that reported in the Table 3.5, could be implemented on these FPGAs.

3.6 Conclusion

In this paper, a digital hardware was proposed to implement biological-plausible models of astrocyte and glutamate release mechanism. The design demonstrated high accuracy in replicating the behaviour of aforementioned biological cells in hardware. Furthermore, implementation results indicated that the design is much more area efficient and faster comparing to recently published works. This new design, allows researchers to implement large number of these biological-plausible cells on FPGAs. This is most important because, unlike high level models, simulation of these models due to high biological details requires long time and computational power and will fall behind the real time easily. This hardware is most useful to replicate the tripartite synapse and its components. Such hardware could also be scaled to study brain diseases and information processing algorithms.

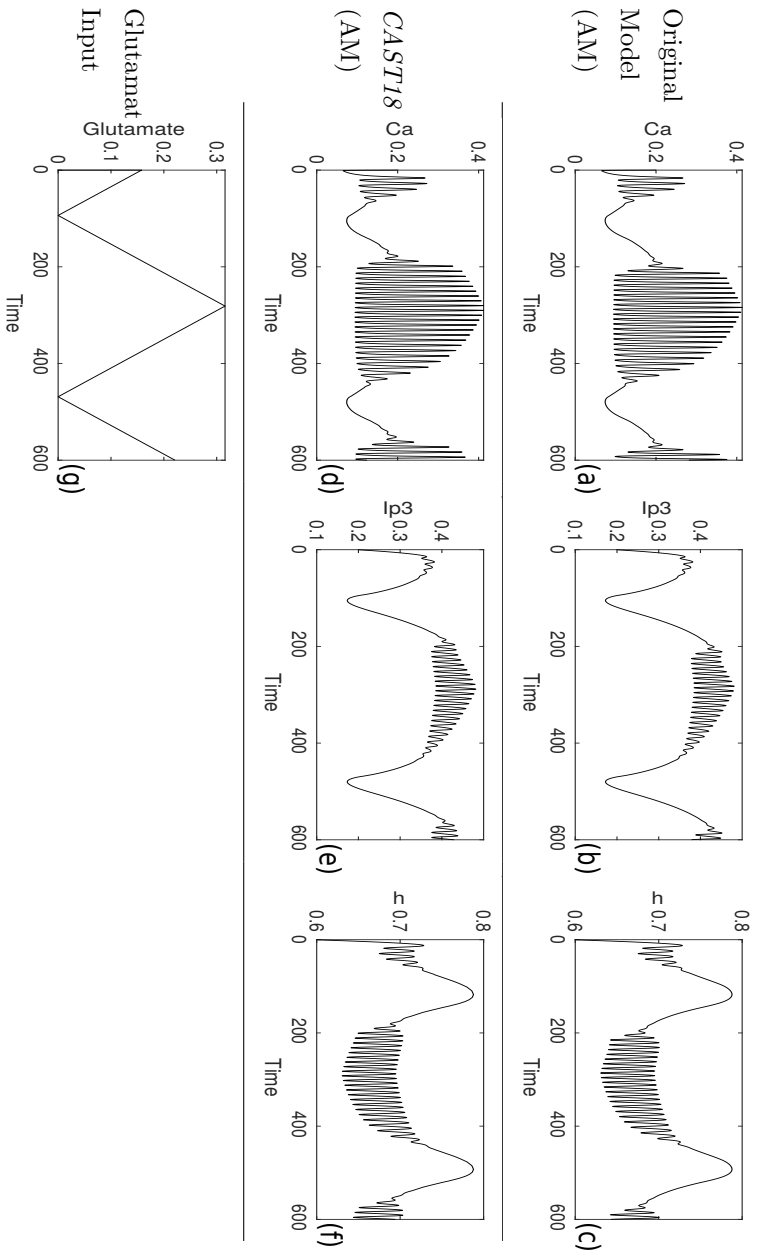


Figure 3.6: Ca^{2+}, IP3 and gating variable as function of a triangle wave of glutamate in AM astrocyte. The first row shows the result for the original and the second row for *CAST18* model. The glutamate input is depicted in the (g). As this figure indicates, amplitude of Ca^{2+} oscillation is modulated by level of glutamate in the synaptic cleft. The result of the second row also verifies that *CAST18* closely follows the original model.

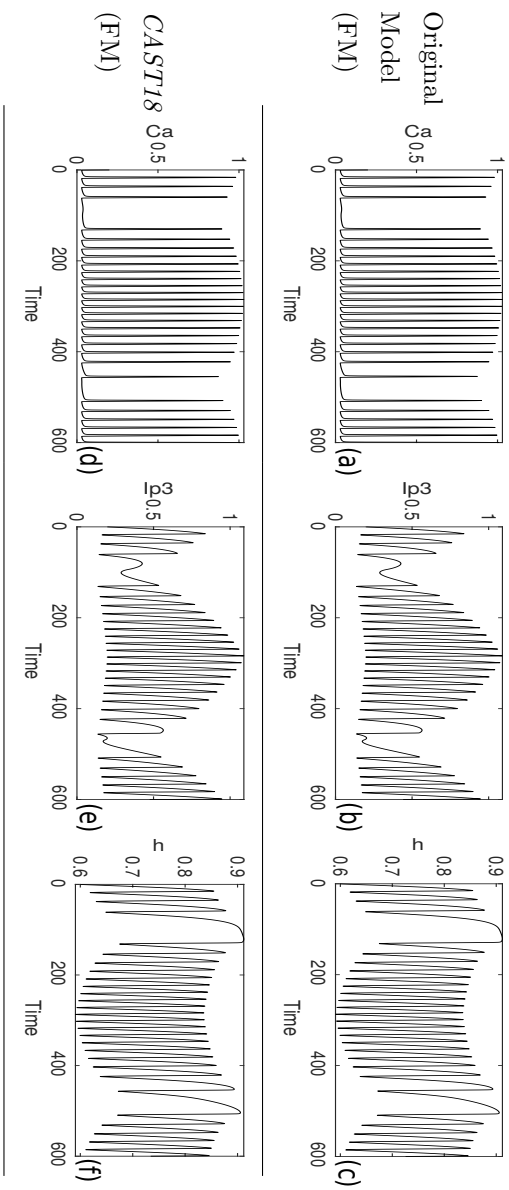


Figure 3.7: Ca^{2+}, IP3 and gating variable as function of a triangle wave of glutamate in FM astrocyte. The first row shows the result for the original and the second row for *CAST18* model. The glutamate input is as depicted in the Fig. 3.7 (g). As this figure indicates, frequency of Ca^{2+} oscillation is modulated by level of glutamate in the synaptic cleft. The result of the second row also verifies that *CAST18* closely follows the original model.

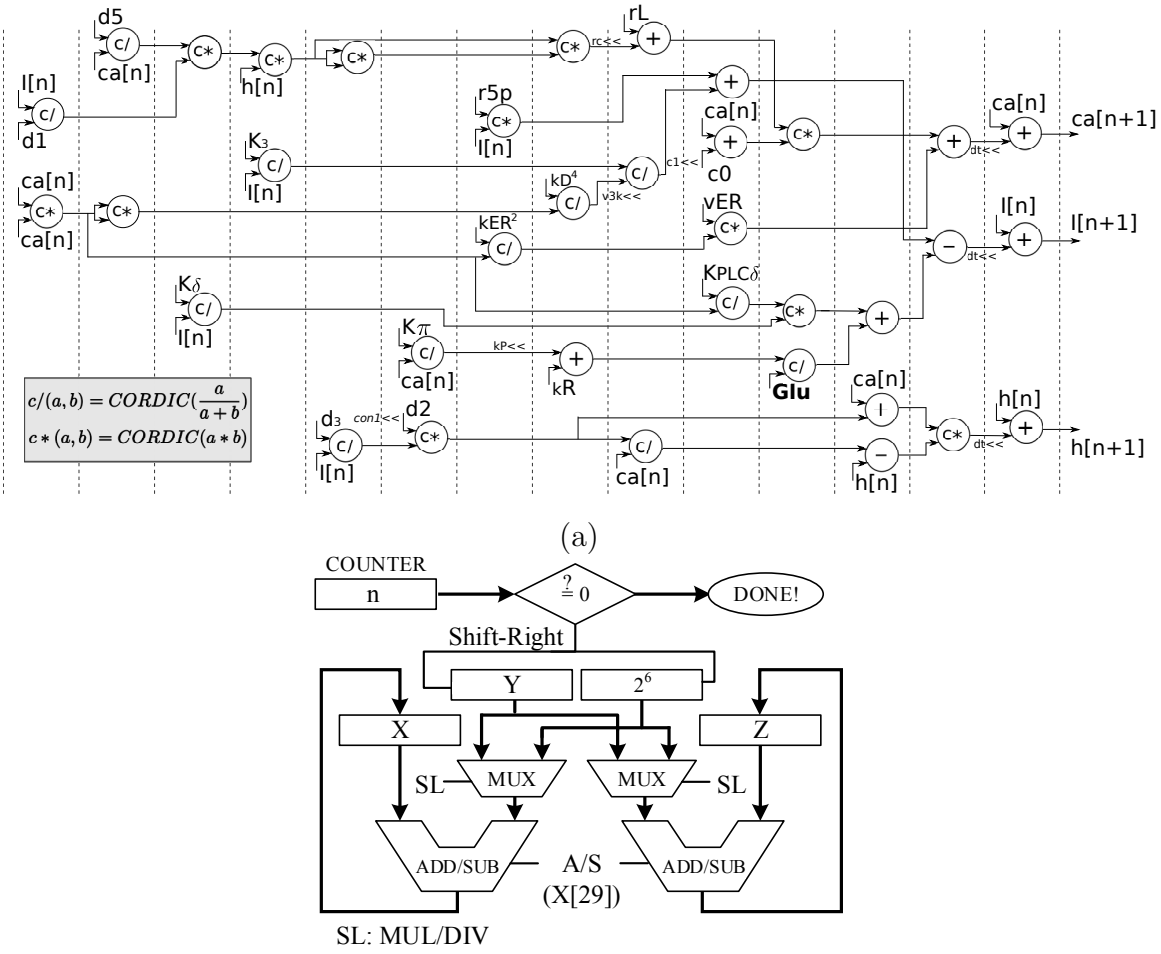


Figure 3.8: (a) Scheduling diagram for digital implementation of *ASTRO5* model. With three multipliers and two dividers in each step, the diagram takes 8 steps to execute which is also number of steps of critical path. The input of system is glutamate, shown in the diagram as **Glu** and con1=d2×(d1-d3)/d3. (b) Hardware implementation of CORDIC division and multiplication. At start, register x,y,z will be loaded with operand 1, operand 2 and zero, respectively. In addition, counter will be loaded with number of iterations (n) and the shift register with 2^6. At each iteration, the counter decreases by one and if it is not zero, it activates shift Right signal. The mode of operation (division or multiplication) can be changed with *SL*.

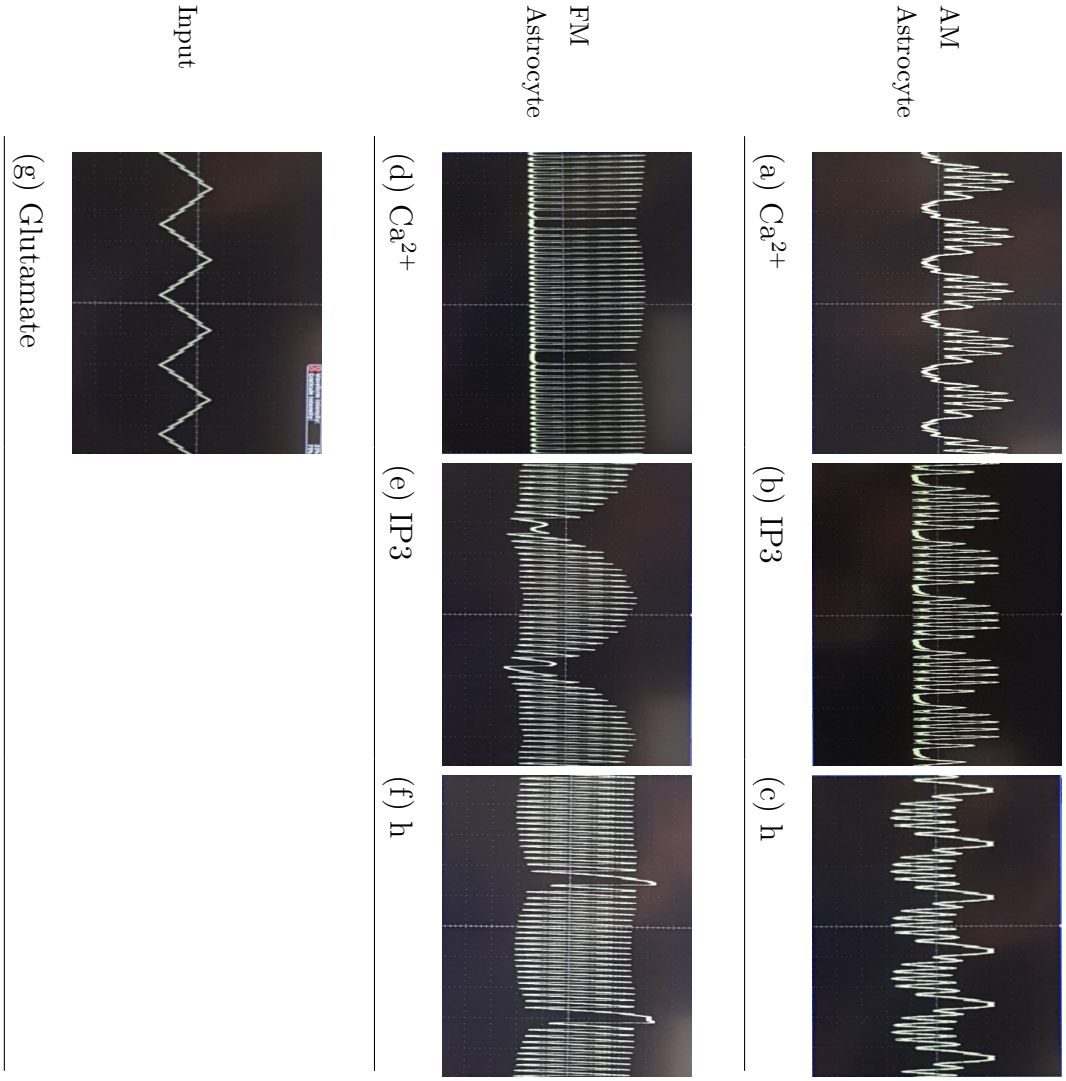


Figure 3.9: Oscilloscope photos of the on-FPGA Ca^{2+}, IP3 and gating variables oscillations against the glutamate input for *C_ASTRO5*. First row shows the amplitude modulation of Ca^{2+} with level of glutamate. Second row implies the frequency modulation with glutamate. Data were converted to analog using digital to analog converter of the VGA port.

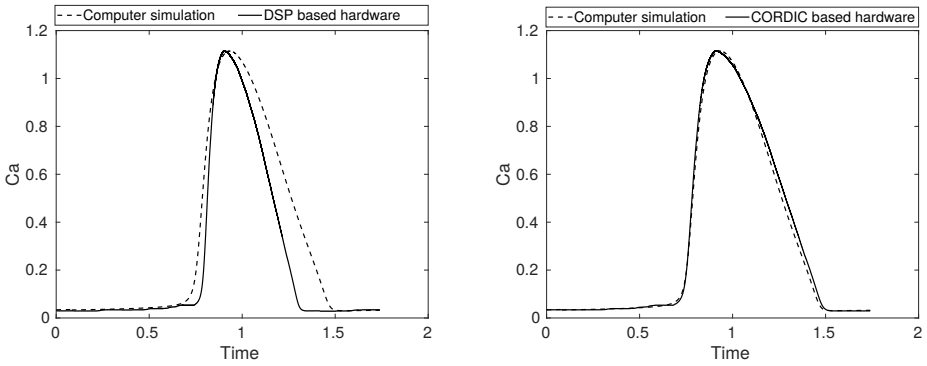


Figure 3.10: The CORDIC and DSP based astrocyte on-FPGA data, were transferred to PC through UART-USB port and plotted versus computer simulation results. For the same glutamate input, the Ca^{2+} spike of CORDIC model is very closer to the computer simulation results.

References

[1] E. Capecci, F. C. Morabito, M. Campolo, N. Mammone, D. Labate, and N. Kasabov, *A Feasibility Study of Using the NeuCube Spiking Neural Network Architecture for Modelling Alzheimer's Disease EEG Data*. Cham: Springer International Publishing, 2015, pp. 159–172.

[2] S. Ghosh-Dastidar and H. Adeli, "A new supervised learning algorithm for multiple spiking neural networks with application in epilepsy and seizure detection," *Neural Networks*, vol. 22, no. 10, pp. 1419–1431, Dec 2009.

[3] I. Wallach, M. Dzamba, and A. Heifets, "Atomnet: A deep convolutional neural network for bioactivity prediction in structure-based drug discovery," *CoRR*, vol. abs/1510.02855, Oct 2015.

[4] Y. Li and C. S. Nam, "Collaborative brain-computer interface for people with motor disabilities," *IEEE Computational Intelligence Magazine*, vol. 11, no. 3, pp. 56–66, Aug 2016.

[5] F. O. Carreon, J. G. G. Serna, A. M. Rendon, N. G. Franco, A. M. P. Jimenez, and J. H. Gomez, "Induction of emotional states in people with disabilities through film clips using brain computer interfaces," *IEEE Latin America Transactions*, vol. 14, no. 2, pp. 563–568, Feb 2016.

[6] X. Zeng, T. Song, X. Zhang, and L. Pan, "Performing four basic arithmetic operations with spiking neural p systems," *IEEE Transactions on NanoBioscience*, vol. 11, no. 4, pp. 366–374, Dec 2012.

[7] K. E. Friedl, A. R. Voelker, A. Peer, and C. Eliasmith, "Human-inspired neurorobotic system for classifying surface textures by touch," *IEEE Robotics and Automation Letters*, vol. 1, no. 1, pp. 516–523, Jan 2016.

[8] J. A. Wall, L. J. McDaid, L. P. Maguire, and T. M. McGinnity, "Spiking neural network model of sound localization using the interaural intensity difference," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 4, pp. 574–586, Apr 2012.

[9] J. H. Lee, T. Delbruck, M. Pfeiffer, P. K. J. Park, C. W. Shin, H. Ryu, and B. C. Kang, "Real-time gesture interface based on event-driven processing from stereo silicon retinas," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 12, pp. 2250–2263, Dec 2014.

[10] N. Srinivasa and Y. Cho, "Self-organizing spiking neural model for learning fault-tolerant spatio-motor transformations," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 10, pp. 1526–1538, Oct 2012.

[11] M. Samie, G. Dragffy, A. M. Tyrrell, T. Pipe, and P. Bremner, "Novel bio-inspired approach for fault-tolerant vlsi systems," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 21, no. 10, pp. 1878–1891, Oct 2013.

[12] M. D. Pittà, N. Brunel, and A. Volterra, "Astrocytes: Orchestrating synaptic plasticity?" *Neuroscience*, vol. 323, pp. 43–61, May 2016.

[13] M. De Pittà and N. Brunel, "Modulation of synaptic plasticity by glutamatergic gliotransmission: a modeling study," *Neural plasticity*, vol. 2016, Feb 2016.

[14] S. Nadkarni and P. Jung, "Modeling synaptic transmission of the tripartite synapse," *Physical biology*, vol. 4, no. 1, pp. 1–9, Jan 2007.

[15] V. Volman, E. Ben-Jacob, and H. Levine, "The astrocyte as a gatekeeper of synaptic information transfer," *Neural computation*, vol. 19, no. 2, pp. 303–326, Feb 2007.

[16] G. Perea, M. Navarrete, and A. Araque, "Tripartite synapses: astrocytes process and control synaptic information," *Trends in Neurosciences*, vol. 32, pp. 421–431, Jul 2009.

[17] M. M. Halassa, T. Fellin, H. Takano, J.-H. Dong, and P. G. Haydon, "Synaptic islands defined by the territory of a single astrocyte," *Journal of Neuroscience*, vol. 27, no. 24, pp. 6473–6477, Jun 2007.

[18] C. Steinhäuser, M. Grunnet, and G. Carmignoto, "Crucial role of astrocytes in temporal lobe epilepsy," *Neuroscience*, vol. 323, pp. 157–169, May 2016.

[19] J. Rodríguez-Arellano, V. Parpura, R. Zorec, and A. Verkhratsky, "Astrocytes in physiological aging and alzheimer's disease," *Neuroscience*, vol. 323, pp. 170–182, May 2016.

[20] D. Do-Ha, Y. Buskila, and L. Ooi, "Impairments in motor neurons, interneurons and astrocytes contribute to hyperexcitability in als: Underlying mechanisms and paths to therapy," *Molecular Neurobiology*, vol. 55, no. 2, pp. 1410–1418, Feb 2018.

[21] T. Fellin, J. Ellenbogen, M. De Pittà, E. Ben-Jacob, and M. Halassa, "Astrocyte regulation of sleep circuits: experimental and modeling perspectives," *Frontiers in Computational Neuroscience*, vol. 6, p. 65, Aug 2012.

[22] J. Liu, J. Harkin, L. McDaid, D. M. Halliday, A. M. Tyrrell, and J. Timmis, "Self-repairing mobile robotic car using astrocyte-neuron networks," in *2016 International Joint Conference on Neural Networks (IJCNN)*, July 2016, pp. 1379–1386.

[23] D. Postnov, L. Ryazanova, and O. Sosnovtseva, "Functional modeling of neural–glial interaction," *Biosystems*, vol. 89, no. 1–3, pp. 84–91, May 2007.

[24] S. Nadkarni and P. Jung, "Modeling synaptic transmission of the tripartite synapse," *Physical Biology*, vol. 4, no. 1, p. 1, Jan 2007.

[25] M. De Pittà, M. Goldberg, V. Volman, H. Berry, and E. Ben-Jacob, "Glutamate regulation of calcium and ip3 oscillating and pulsating dynamics in astrocytes," *Journal of Biological Physics*, vol. 35, no. 4, pp. 383–411, Oct 2009.

[26] V. Volman, E. Ben-Jacob, and H. Levine, "The astrocyte as a gatekeeper of synaptic information transfer," *Neural Comput.*, vol. 19, no. 2, pp. 303–326, Feb 2007.

[27] J. J. Wade, L. J. McDaid, J. Harkin, V. Crunelli, and J. A. S. Kelso, "Bidirectional coupling between astrocytes and neurons mediates learning and dynamic coordination in the brain: A multiple modeling approach," *PLOS ONE*, vol. 6, no. 12, pp. 1–24, Dec 2011.

[28] M. De Pittà, V. Volman, H. Levine, and E. Ben-Jacob, "Multimodal encoding in a simplified model of intracellular calcium signaling," *Cognitive Processing*, vol. 10, no. 1, p. 55, Nov 2008.

[29] R. Brette, "Adaptive exponential integrate-and-fire model as an effective description of neuronal activity," *Journal of Neurophysiology*, vol. 94, no. 5, pp. 3637–3642, Nov 2005.

[30] E. M. Izhikevich, "Simple model of spiking neurons," *IEEE Transactions on Neural Networks*, vol. 14, no. 6, pp. 1569–1572, Nov 2003.

[31] M. Graupner and N. Brunel, "Calcium-based plasticity model explains sensitivity of synaptic changes to spike pattern, rate, and dendritic location," *Proceedings of the National Academy of Sciences*, vol. 109, no. 10, pp. 3991–3996, Mar 2012.

[32] E. Jokar and H. Soleimani, "Digital multiplierless realization of a calcium-based plasticity model," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 64, no. 7, pp. 832–836, July 2017.

[33] F. A. Azevedo, L. R. Carvalho, L. T. Grinberg, J. M. Farfel, R. E. Ferretti, R. E. Leite, W. J. Filho, R. Lent, and S. Herculano-Houzel, "Equal numbers of neuronal and nonneuronal cells make the human brain an isometrically scaled-up primate brain," *The Journal of Comparative Neurology*, vol. 513, no. 5, pp. 532–541, Apr 2009.

[34] "Nest simulator — the neural simulation tool," 2019. [Online]. Available: http://www.nest-simulator.org

[35] "The human brain project," 2019. [Online]. Available: https://www.humanbrainproject.eu

[36] K. Cheung, S. R. Schultz, and W. Luk, "Neuroflow: A general purpose spiking neural network simulation platform using customizable processors," *Frontiers in Neuroscience*, vol. 9, p. 516, Jan 2016.

[37] R. Brette and D. F. M. Goodman, "Simulating spiking neural networks on gpu," *Network: Computation in Neural Systems*, vol. 23, no. 4, pp. 167–182, Dec 2012.

[38] L. Maguire, T. McGinnity, B. Glackin, A. Ghani, A. Belatreche, and J. Harkin, "Challenges for large-scale implementations of spiking neural networks on fpgas," *Neurocomputing*, vol. 71, no. 1, pp. 13–29, Dec 2007.

[39] A. S. Cassidy, J. Georgiou, and A. G. Andreou, "Design of silicon brains in the nano-cmos era: Spiking neurons, learning synapses and neural architecture optimization," *Neural Networks*, vol. 45, pp. 4–26, Sep 2013.

[40] "Brainscales," 2019. [Online]. Available: https://brainscales.kip.uni-heidelberg.de/

[41] "Spinnaker," 2019. [Online]. Available: http://apt.cs.manchester.ac.uk/projects/SpiNNaker/

[42] "Ibm truenorth," 2019. [Online]. Available: http://bluebrain.epfl.ch

[43] B. V. Benjamin, P. Gao, E. McQuinn, S. Choudhary, A. R. Chandrasekaran, J. M. Bussat, R. Alvarez-Icaza, J. V. Arthur, P. A. Merolla, and K. Boahen, "Neurogrid: A mixed-analog-digital multichip system for large-scale neural simulations," *Proceedings of the IEEE*, vol. 102, no. 5, pp. 699–716, May 2014.

[44] R. K. Lee and A. C. Parker, "A cmos circuit implementation of retrograde signaling in astrocyte-neuron networks," in *2016 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, Oct 2016, pp. 588–591.

[45] A. Ahmadi and M. Heidarpur, "An integrated astrocyte-adaptive exponential (aadex) neuron and circuit implementation," in *2016 24th Iranian Conference on Electrical Engineering (ICEE)*, May 2016, pp. 1545–1550.

[46] Y. Irizarry-Valle, A. C. Parker, and J. Joshi, "A cmos neuromorphic approach to emulate neuro-astrocyte interactions," in *The 2013 International Joint Conference on Neural Networks (IJCNN)*, Aug 2013, pp. 1–7.

[47] J. Liu, J. Harkin, L. P. Maguire, L. J. McDaid, J. J. Wade, and G. Martin, "Scalable networks-on-chip interconnected architecture for astrocyte-neuron networks," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 63, no. 12, pp. 2290–2303, Dec 2016.

[48] A. P. Johnson, D. M. Halliday, A. G. Millard, A. M. Tyrrell, J. Timmis, J. Liu, J. Harkin, L. McDaid, and S. Karim, "An fpga-based hardware-efficient fault-tolerant astrocyte-neuron network," in *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, Dec 2016, pp. 1–8.

[49] J. Liu, J. Harkin, L. Maguire, L. McDaid, J. Wade, and M. McElholm, "Self-repairing hardware with astrocyte-neuron networks," in *2016 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2016, pp. 1350–1353.

[50] S. Karim, J. Harkin, L. McDaid, B. Gardiner, J. Liu, D. M. Halliday, A. M. Tyrrell, J. Timmis, A. Millard, and A. Johnson, "Assessing self-repair on fpgas with biologically realistic astrocyte-neuron networks," in *2017 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, July 2017, pp. 421–426.

[51] G. Martin, J. Harkin, L. J. McDaid, J. J. Wade, J. Liu, and F. Morgan, "Astrocyte to spiking neuron communication using networks-on-chip ring topology," in *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, Dec 2016, pp. 1–8.

[52] H. Soleimani, M. Bavandpour, A. Ahmadi, and D. Abbott, "Digital implementation of a biological astrocyte model and its application," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 1, pp. 127–139, Jan 2015.

[53] M. Heidarpur, A. Ahmadi, and N. Kandalaft, "A digital implementation of 2d hindmarsh–rose neuron," *Nonlinear Dynamics*, vol. 89, no. 3, pp. 2259–2272, Aug 2017.

[54] M. Heidarpour, A. Ahmadi, and R. Rashidzadeh, "A cordic based digital hardware for adaptive exponential integrate and fire neuron," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 63, no. 11, pp. 1986–1996, Nov 2016.

[55] J. E. Volder, "The cordic trigonometric computing technique," *Electronic Computers, IRE Transactions on*, vol. EC-8, no. 3, pp. 330–334, Sep 1959.

[56] S. Goutelle, M. Maurin, F. Rougier, X. Barbaut, L. Bourguignon, M. Ducher, and P. Maire, "The hill equation: a review of its capabilities in pharmacological modelling," *Fundamental and Clinical Pharmacology*, vol. 22, no. 6, pp. 633–648, Dec 2008.

[57] S. G. Tewari and K. K. Majumdar, "A mathematical model of the tripartite synapse: astrocyte-induced synaptic plasticity," *Journal of biological physics*, vol. 38, no. 3, pp. 465–496, May 2012.

Digital Implementation of a Biological-Plausible Model For Astrocyte Ca^{2+} Oscillations

4.1 Introduction

Understanding the information processing algorithms in the brain can help to develop more intelligent, efficient, fault tolerant and much faster computing devices [1, 2, 3]. Moreover, studying biological reactions in the brain, plays a key role to find the mechanisms underlying neurological and psychiatric diseases [4, 5]. Additionally, it could be the initial step toward building a platform to test the drugs made for brain diseases prior to test on live animals [6]. To study brain, first, we need to replicate its components and the way they are interconnected and interact with each other. However, the complexity of the brain ,due to the large number of cells [7] and their communication pathways, make this a challenging task. The path to this goal commences by proper modelling and building a simulation platform.

This work focuses on the astrocytes, a type of glial cells which were believed to have only nutritional and supportive roles for neurons. New experimental results ,however, shows that astrocytes participate in neuronal plasticity [8, 9, 10, 11, 12, 13], development of neuronal pathologies [4, 5], sleep functions [14] and brain self-repair

ability [15]. Although, astrocytes do not generate action potentials the way neurons do but instead, propagate Ca^{2+} waves and release transmitters over a large area and long time span.

Several mathematical models are presented for astrocytes [8, 9, 16, 17, 10]. All of these models (except the model in [8] where a high level mathematical specifications are utilized) are biologically-plausible models which describe cellular phenomena and physiological parameters. Comparing these two types of models, the abstract models are simpler to understand and cheaper to simulate whereas they can not investigate dynamics, complexity and emergent nonlinear coherence that arise when large number of neurons and glial cells are coupled [18]. But in general, the choice of the model depends on the scale, purpose and cost of the simulation.

As for simulation platform, it could be computer based or Very Large Scale Integration (VLSI) dedicated hardware. Computers are flexible and available for every one which make them best choice for small networks. But for large networks, VLSI systems are more efficient and affordable in comparison with supercomputers. Field Programmable Gate Arrays (FPGA) provide a viable platform to simulate neural networks [19, 20, 21, 22]. However, limited resources is one of main challenges of large scale neural network FPGA implementations [23].

This work presents an approach to simulate the biologically-plausible astrocyte model presented in the [9] on FPGA. The advantage of this model is incorporation of glutamate regulation of Ca^{2+} waves. The model equations include nonlinear terms as division, non-integer roots, multiplication, quadratic, cubic, quartic etc which make the FPGA implementation difficult. This work uses linearization techniques to design a hardware capable of replicating the behavior of astrcocyte on FPGA. In designing process, search algorithms were utilized to find the most efficient parameters for linearization that both reduce the implementation cost and maintain the accuracy of the model.

Researchers already presented circuits for simplified and abstract models of astrocytes [24, 25] while this on-FPGA hardware is capable of simulating the astrocyte down to ions levels. This is very useful to study calcium dynamics and calcium-based

learning algorithms that are getting a a lot of attentions recently [13, 26, 27, 28]. The FPGA implementation results is further compared with another work that also implemented a biological realistic model on FPGA [29].

The rest of this Chapter is organized as follows. The astrocyte model is described in Section 4.2. Section 4.3 presents linearized models and evaluation of the accuracy of proposed models. Design and hardware implementation are discussed in Section 4.4 and 4.5. The paper concludes in Section 4.6.

4.2 Background

In general, when an action potential arrives at the pre-synaptic terminal, neurotransmitters release into synaptic cleft as shown in Fig. 4.1. Among them, glutamate is the major excitatory neurotransmitter in the nervous system and is critically involved in many functions [30]. Glutamate in the synaptic cleft quickly binds to glutamate receptors on the post-synaptic terminal. Activation of these glutamate receptors leads to the depolarization of the post-synaptic terminal and eventually could result in the excitatory post-synaptic potential. The released glutamate may spill over the synaptic cleft and bind to the extra-cellular part of astrocytic Metabotropic GLUtamate Receptors (mGluRs). Binding of glutamate to mGluRs promotes opening of a few Inositol Trisphosphate (IP3) channels. As a consequence, intracellular Ca^{2+} slightly increases. Since the opening probability of IP3 channels nonlinearly increases with Ca^{2+} concentration, such an initial amount boosts the opening probability of neighboring channels which In turn, leads to a further increase of Ca^{2+}.

Action of Ca^{2+} release, reverses at high cytoplasmic Ca^{2+} concentrations, when inactivation of IP3R channels takes place and SERCA pumps, which its activity increases with cytoplasmic Ca^{2+}, quickly pumps back excess cytoplasmic Ca^{2+} into the Endoplasmic Reticulum (ER). The intracellular Ca^{2+} concentration consequently recovers toward basal value which suppresses IP3 channels activation. If the glutamate stimulation continues, intracellular IP3 level remains high enough to repeat the cycle into oscillations of Ca^{2+} and IP3 ions.

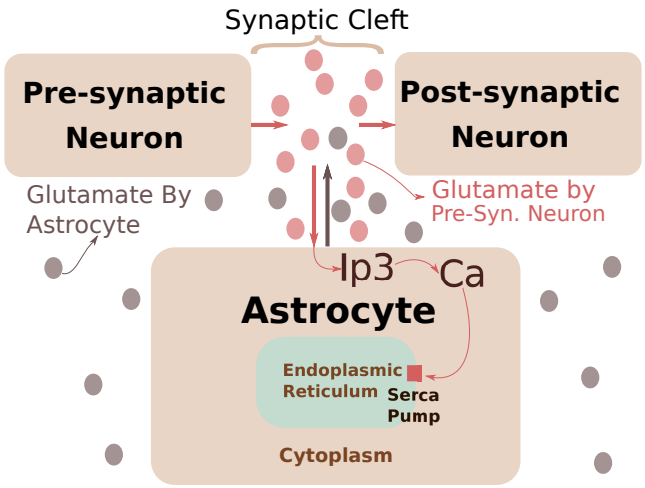


Figure 4.1: Mechanism of glial regulation of synaptic transmission. (1-) Release of glutamate (Glu) from pre-synaptic terminal activates astrocyte receptors (2-) evoking an increase in IP3 and consequently Ca^{2+} levels (3-) and release of glutamate from glia.

Glutamate in astrocyte, acts as modulator of the calcium oscillations which modulates the frequency (FM), amplitude (AM) or combination of both (AFM) [9]. Such oscillations further trigger the release of glutamate by astrocyte which effect the same or other synapses considering it's comparatively larger size. The G-ChI model that describe calcium and IP3 dynamics is shown in equations 4.1 to 4.8. The parameters values and descriptions are available in [9].

$$\begin{aligned}\frac{dI}{dt} =& \nu_\beta Hill(0.7, \gamma, K_R(1 + \frac{K_p}{K_r} Hill(1, Ca, K_\pi))) \frac{\nu_\delta}{1 + \frac{I}{k_\delta}} Hill(2, Ca, K_{PLC\delta}) \\ & - \nu_{3K} Hill(4, Ca, K_D) Hill(1, I, K_3) - r_{5P} I\end{aligned} \tag{4.1}$$

$$\frac{dCa}{dt} = (r_C m_\infty^3 n_\infty^3 h(i)^3 + r_L)(C_{ER} - Ca) - \nu_{ER}.Hill(2, Ca, K_{ER}) \tag{4.2}$$

$$\frac{dh}{dt} = \frac{h_\infty - h(i)}{\tau_h} \tag{4.3}$$

$$\tau_h = \frac{1}{\alpha_2(Q_2 + C_a)} \tag{4.4}$$

$$n_\infty = hill(1, C_a, d5) \tag{4.5}$$

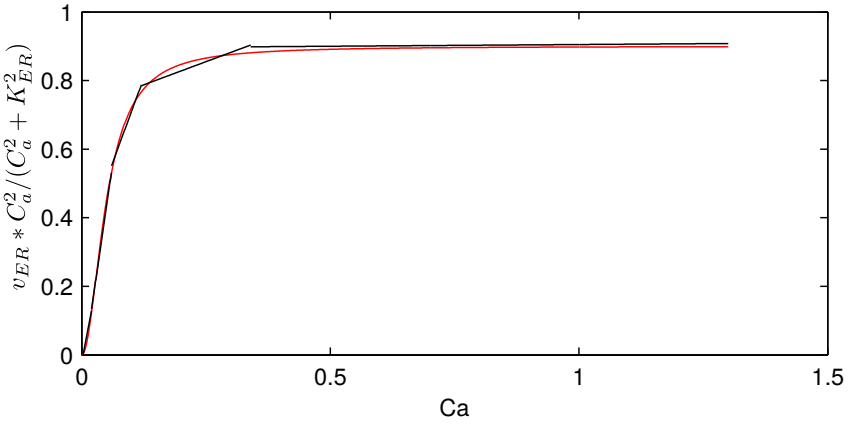


Figure 4.2: Computer simulation of the term $v_{ER} * C_a^2/(C_a^2 + K_{ER}^2)$ (red line) and it's linear equivalent term (black line) as described in eq. 4.11, calculated with $\epsilon = 0.02$.

$$m_\infty = hill(1, I, d1) \tag{4.6}$$

$$Q = \frac{(d_2 I + d_1)}{(I + d_3)} \tag{4.7}$$

$$h_\infty = \frac{Q}{(Q_2 + C_a)} \tag{4.8}$$

The Hill equation [31] is one of common and useful equations in bio-chemistry which is given by:

$$Hill(n, x, y) = \frac{x^n}{x^n + y^n} \tag{4.9}$$

4.3 Modified Model

In this section, the astrocyte model is modified for digital hardware implementation and simulated to ensure its accuracy. The main difficulty for the circuit implementation of the G-ChI model lies in nonlinearity of the expressions which describe the biochemical reactions. To bypass the problem, these nonlinearities were substituted with equivalent linear terms. The objective is to find a sequence of linear functions, L, so that for domain of nonlinear function N:

$$|L - N| \leq \epsilon \tag{4.10}$$

where ϵ is maximum acceptable error. For one variable function N , L functions is as follows:

$$L = \alpha x + \beta \tag{4.11}$$

and for two variable function L is:

$$L = \alpha x + \beta y + \delta \tag{4.12}$$

The algorithm for finding α, β and δ for each segment of two variable L described in follows:

> For given numbers of $\{\epsilon,\ f(0,0),\ f(0,1),\ f(0,1),...\}$ find real numbers $\alpha, \beta, \delta, m, n$ such that:
>
> $$|f(i,j) - (\alpha i + \beta j + \delta)| \leq \epsilon \ for\ 0 \leq i \leq m \ and\ 0 \leq j \leq n \tag{4.13}$$
>
> and $n * m$ is maximal.

The similar algorithm was used for one variable L. The algorithm begins by selecting the largest possible restricted plate (longest possible single line) which satisfies the error restriction and repeats the process by selecting restricted plates (lines) of the maximum area (length) from the left-hand sides (side) towards remaining of the function to be approximated. The number of linear segments depends on the value of maximum tolerable error ϵ. For the larger values, there are less number of segments which results in more deviation from the original term and vice versa. To compare the results, the nonlinear equations were linearized using three values of ϵ=0.04, 0.03 and 0.02. Fig. 4.2 and shows the simulation of one variable term:

$$N1 = \frac{v_{ER}\, C_a^2}{C_a^2 + K_{ER}^2} \tag{4.14}$$

and Fig. 4.3 displays simulation of the nonlinear terms:

$$\begin{cases} N2 = \dfrac{d2\ (I + d_1)\ (1 - h)}{(I + d_3)} \\ N3 = \dfrac{v_{3k}\, C_a^4\, I}{(C_a^4 + k_D^4)\ (I + k_3)} \end{cases} \tag{4.15}$$

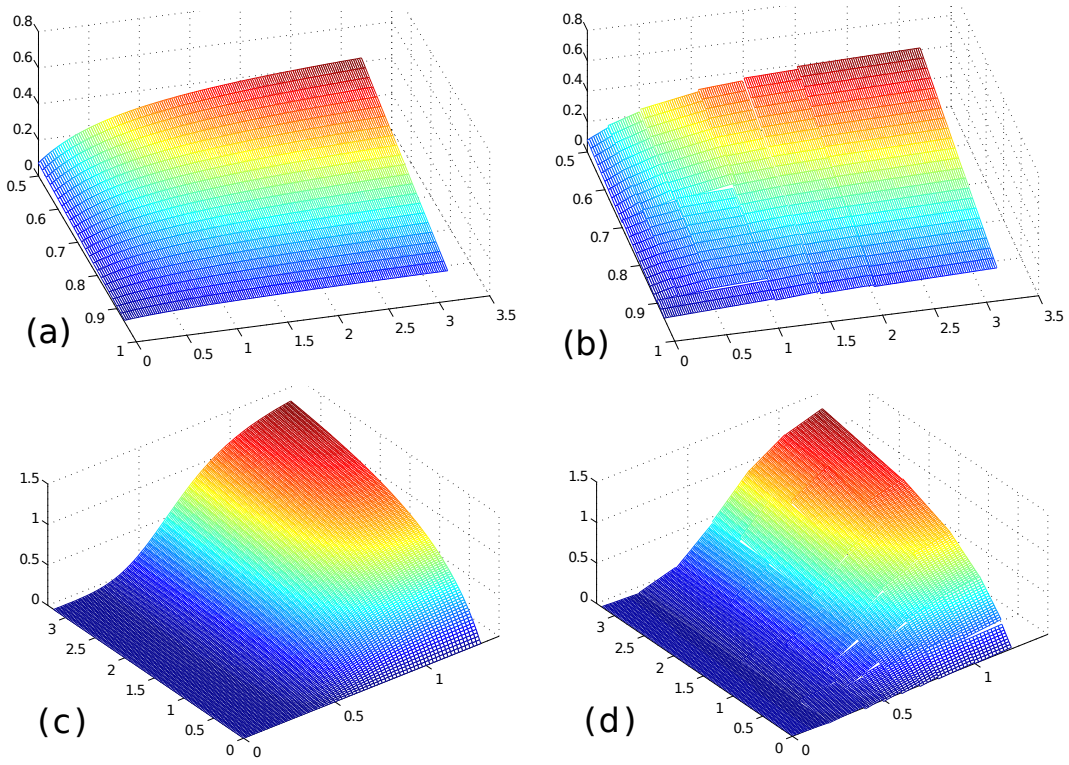


Figure 4.3: Computer simulation of the nonlinear and their linear equivalent term as described in eq. 4.14 and 4.15, obtained with $\epsilon = 0.02$. (a) Simulation of the term $d2 * (I + d_1)/(I + d_3) * (1 - h_a)$ and (b) it's linear substitute. (c) Simulation of the term $v_{3k} Hill(4, C_a, k_D) Hill(1, I, k_3)$ and (d) it's linear substitute.

Table 4.1: NRMSE calculation of nonlinear terms in equations 4.14, 4.15 and their corresponding linear term.

	$\epsilon = 0.02$		$\epsilon = 0.03$		$\epsilon = 0.04$	
	no. l.s.	NRMSE	no. of l.s.	NRMSE	no. of l.s.	NRMSE
$N1$	5	0.008	4	0.033	3	0.039
$N2$	9	0.006	6	0.009	4	0.011
$N3$	16	0.0080	11	0.0122	8	0.0213

no. of l.s. : number of the linear segments

and their respected linearized equivalent for $\epsilon = 0.02$. As these figures indicate, despite using much simpler computational units, the linearized equivalent terms follow their respected nonlinear equations closely. For a quantitative comparison, normalized Root Mean Square Deviation (NRMSE) error[32] was calculated for the nonlinear term N and linearized term L as:

$$NRMSE = \frac{\sqrt{\frac{\sum_{i=0}^{n}(L-N)^2}{n}}}{N_{max} - N_{min}} \tag{4.16}$$

where N_{max} and N_{min} are minimum and maximum of the N and n is number of points that this error is calculated. Table 4.1 presents the values of this error for nonlinear terms in equations 4.14 and 4.15. The small values denote the resemblance of the suggested linear substitute terms. Furthermore, the nonlinear terms in the equations 4.1 to 4.8 were substituted by their respective linear terms to develop a linearized model for astrocyte. Fig. 4.4 shows the simulation of the proposed and the original model for different modes and values of glutamate. As it can be seen in the figure, the responses of the models are very similar.

4.4 Hardware Implementation

The linearized model presented in the previous section, make it possible to implement the nonlinear astrocyte differential equations on FPGA. In this section an efficient fixed point hardware for astrocyte is designed and physically implemented. The

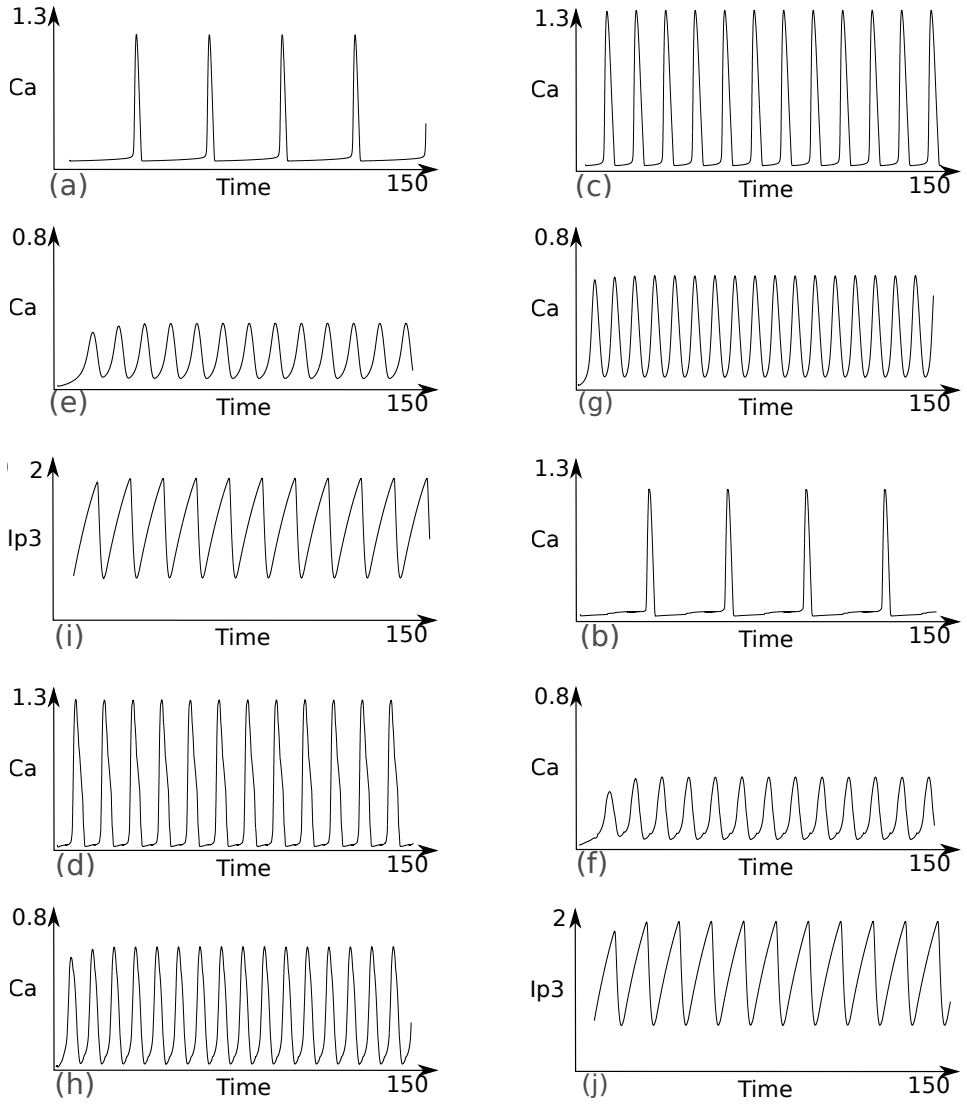


Figure 4.4: Computer simulation of the calcium and Ip3 oscillations in original (first row) and linearized (second row) models, correspond to different values of glutamate: (a,b) calcium oscillations for FM mode and glutamate=0.05, (c,d) calcium oscillations for FM mode and glutamate=1.5, (e,f) calcium oscillations for AM mode and glutamate=0.15, (g,h) calcium oscillations for AM mode and glutamate=1.5, (i,j) Ip3 oscillations for FM mode and glutamate=0.15.

reason that we use a fixed point instead of a floating point hardware, is that despite lower range and precision, it is in general faster and cheaper.

To design the hardware, in the first step, the Eulers method was used to numerically solve the equations 4.1 to 4.8. Small step size (1/256) was selected to ensure minimum error in the method. The scheduling diagram for execution of this operations are shown in Fig. 4.5. Since nonlinear terms was already linearized, the operations that hardware will perform only includes addition, subtraction, comparison and arithmetic shifts. There are three major blocks correspond to three differential equations 4.1, 4.2 and 4.3. The α, β and δ are parameters of the linear segments in equation 4.11 and 4.12 for $\epsilon = 0.02$. Multiplication in these and other constants, was performed with shift and add operations. To do this, first, constants were represented with sequence of $+$, $-$ and 0 symbols (+-0) which each position representing and addition or subtraction. For instance, multiplication in 15 was performed by $2^3 * x - x$, which in hardware computed with three arithmetic right shift x minus x.

Since each nonlinear term was replaced by a sequence of linear terms, α, β and δ change with the range of the function. Therefore, a control unit is required to coordinate these parameters with range of ca, $Ip3$ and h. This unit for $\alpha 1$, $\beta 1$ and $\delta 1$ is shown in the Fig 4.6. In this figure, based on the current value of ca and $Ip3$, the comparators generate the address for selecting the $\alpha 1$, $\beta 1$ and $\delta 1$ from the memory $M1$. These values will further be decoded into addition/subtraction operations. Further, word size of the functional units was determined to prevent over/under flow and preserving the precision. For this purpose, factors such as range of variables and coefficients as well as number of right/left shift operations were taken into account. With those concerns, 34 word size was selected, 24 bit for the fraction and 10 bit for the integer part.

4.5 Implementation Results

To prove validation of the proposed architecture, it was first described as a Hardware Description Language (HDL) code (Verilog) using a Finite State Machine (FSM).

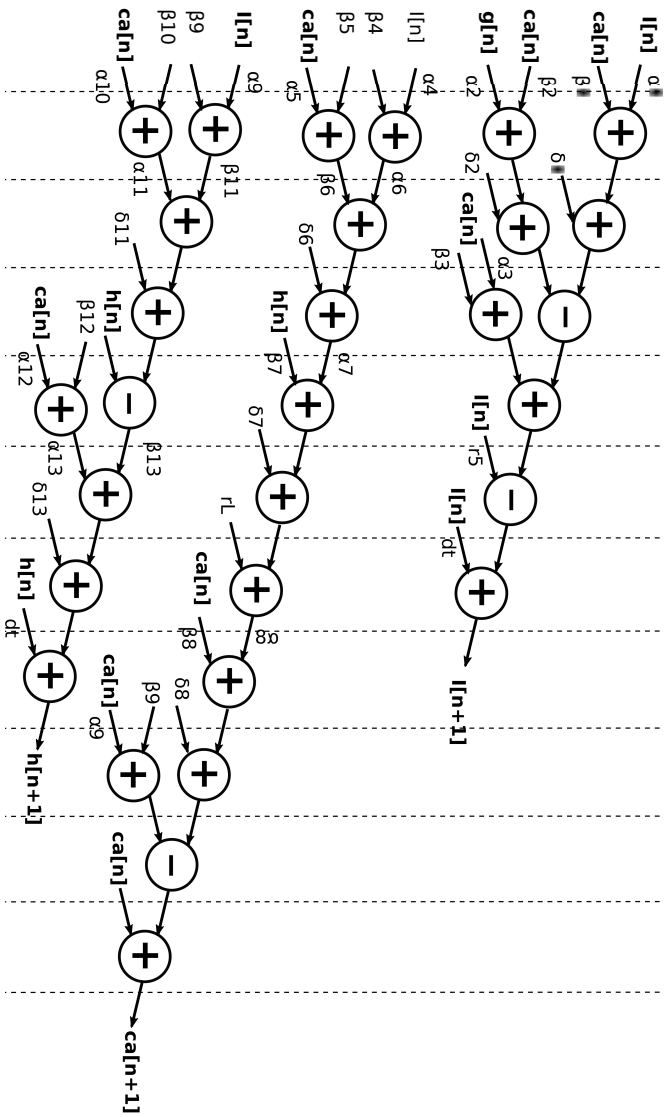


Figure 4.5: As Soon As Possible (ASAP) scheduling diagram for the implementation of the linearized model.

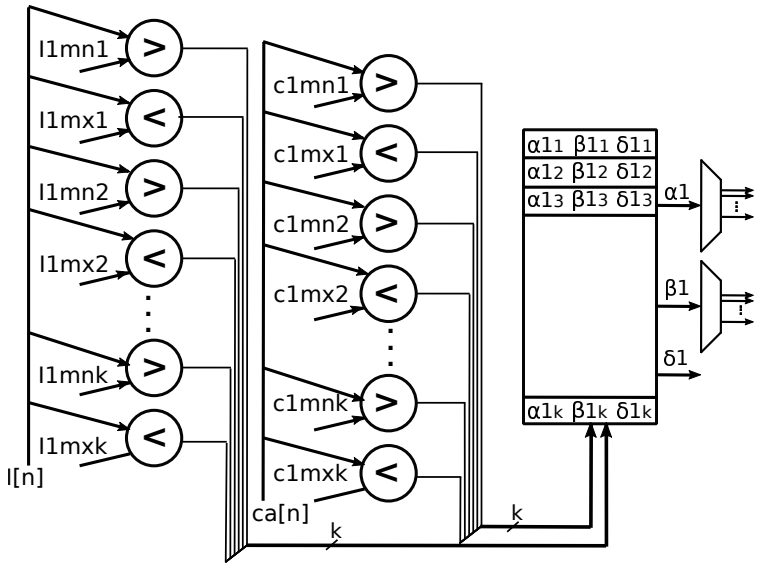
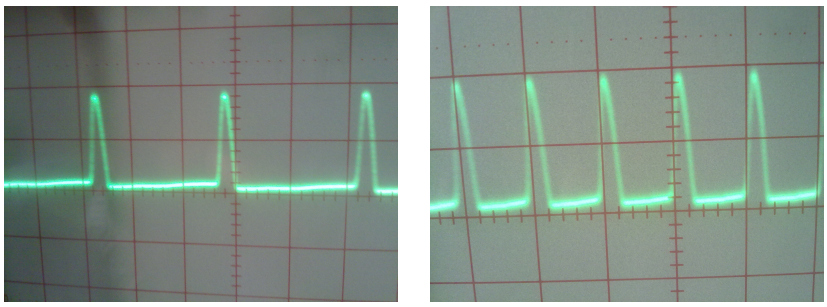


Figure 4.6: The logic unit for selecting α, β and δ based on the value of ca, $Ip3$ for each linear segment.

Table 4.2: Device utilization of the XILINX Spartan 6 Lx75

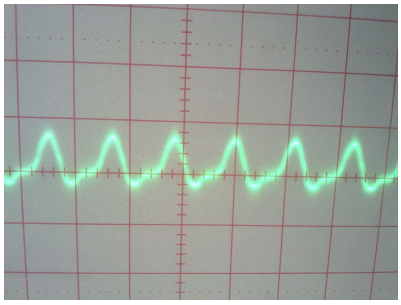
	4In.LUT	Frequency	DSPs
[29]	$\approx$ 1400	NA	$\approx$ 180
This work(FM)	13667	89.35 MHz	0
This work(AM)	13119	88.62 MHz	0

Moreover, Verilog code was simulated and then implemented on the Spartan-6 XC6SLX75T FPGA. The on-FPGA values of ca, $Ip3$ and h were converted to analog and observed on oscilloscope which for Ca^{2+} oscillation is shown in the Fig. 4.7. Further, table 4.2 compares implementation resources and operation frequency of the proposed hardware with another work that implements a biological realistic astrocyte model. In this work no DSP multiplier is used while [29] reports extensive use of that units. Being multiplier-less gives this work two advantages. First is higher speed and throughput. Multiplication is a slow arithmetic operation. Using such number of multipliers significantly degrade the operation frequency and throughput of the system. However, comparing with this work, the proposed method in [29] will have lower error and deviation from the software floating point simulation.

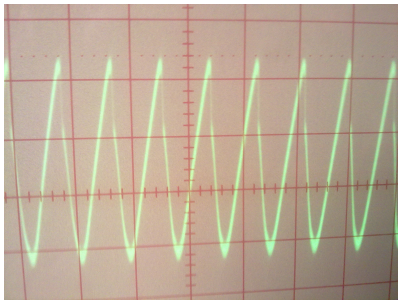


(a) v.sc.=0.1v, t.sc=0.2 ms

(b) v.sc.=0.1v, t.sc=0.2 ms



(c) v.sc.=50mv, t.sc.=0.2 ms



(d) v.sc.=50mv, t.sc=0.2 ms

Figure 4.7: Oscilloscope photos of Spartan-6 XC6LX75 on-FPGA Ca^{2+} oscillation in astrocyte for different modes and values of glutamate. (a) FM, glutamate=0.2. (b) FM, glutamate=2 (c) AM, glutamate=0.2. (c) AM, glutamate=2.

Second advantage is area efficiency. Number of DSP multipliers are limited in FPGAs. For instance, the FPGA that we used in this work has 132 DSP fast multipliers. As result, not even one DSP based implementation can be fitted into this FPGA. Nevertheless, in the proposed design, almost %28 of the FPGA LUTs is used and more number could be implemented on the high-end FPGAs.

The benefit of a multiplier-less design relative to the one presented in [29] will show more on ASIC implementation. As FPGA is a already fabricated device with a number of multipliers. Implementing proposed hardware on silicon will have considerably less cost and have better performance.

4.6 Conclusion

A biological G-ChI astrocyte model for Ca^{2+} oscillation was modified for hardware implementation. Simulation data reveals that this models follow the original model with an acceptable accuracy. The simplicity of the models, which only consist of add/sub and shift operations, made it possible to implement the nonlinear astrocyte equations effectively on hardware. The HDL code describing the hardware was first simulated and further implemented on FPGA as proof of concept.

References

[1] K. E. Friedl, A. R. Voelker, A. Peer, and C. Eliasmith, "Human-inspired neurorobotic system for classifying surface textures by touch," *IEEE Robotics and Automation Letters*, vol. 1, no. 1, pp. 516–523, 2016.

[2] J. A. Wall, L. J. McDaid, L. P. Maguire, and T. M. McGinnity, "Spiking neural network model of sound localization using the interaural intensity difference," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 4, pp. 574–586, 2012.

[3] J. H. Lee, T. Delbruck, M. Pfeiffer, P. K. J. Park, C. W. Shin, H. Ryu, and B. C. Kang, "Real-time gesture interface based on event-driven processing from stereo silicon retinas," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 12, pp. 2250–2263, 2014.

[4] E. Capecci, F. C. Morabito, M. Campolo, N. Mammone, D. Labate, and N. Kasabov, *A Feasibility Study of Using the NeuCube Spiking Neural Network Architecture for Modelling Alzheimer's Disease EEG Data*. Cham: Springer International Publishing, 2015, pp. 159–172.

[5] S. Ghosh-Dastidar and H. Adeli, "A new supervised learning algorithm for multiple spiking neural networks with application in epilepsy and seizure detection," *Neural networks*, vol. 22, no. 10, pp. 1419–1431, 2009.

[6] I. Wallach, M. Dzamba, and A. Heifets, "Atomnet: A deep convolutional neural network for bioactivity prediction in structure-based drug discovery," *arXiv preprint arXiv:1510.02855*, 2015.

[7] F. A. Azevedo, L. R. Carvalho, L. T. Grinberg, J. M. Farfel, R. E. Ferretti, R. E. Leite, W. J. Filho, R. Lent, and S. Herculano-Houzel, "Equal numbers of neuronal and nonneuronal cells make the human brain an isometrically scaled-up primate brain," *The Journal of Comparative Neurology*, vol. 513, no. 5, pp. 532–541, 2009.

[8] D. Postnov, L. Ryazanova, and O. Sosnovtseva, "Functional modeling of neural-glial interaction," *Biosystems*, vol. 89, no. 1–3, pp. 84 – 91, 2007, selected Papers presented at the 6th International Workshop on Neural CodingNeural Coding 20056th International Workshop on Neural Coding.

[9] M. De Pittà, M. Goldberg, V. Volman, H. Berry, and E. Ben-Jacob, "Glutamate regulation of calcium and ip3 oscillating and pulsating dynamics in astrocytes," *Journal of Biological Physics*, vol. 35, no. 4, pp. 383–411, 2009.

[10] J. J. Wade, L. J. McDaid, J. Harkin, V. Crunelli, and J. A. S. Kelso, "Bidirectional coupling between astrocytes and neurons mediates learning and dynamic coordination in the brain: A multiple modeling approach," *PLOS ONE*, vol. 6, no. 12, pp. 1–24, 2011.

[11] S. G. Tewari and K. K. Majumdar, "A mathematical model of the tripartite synapse: Astrocyte-induced synaptic plasticity," *Journal of Biological Physics*, vol. 38, no. 3, pp. 465–496, 2012.

[12] M. Nedergaard, B. Ransom, and S. A. Goldman, "New roles for astrocytes: Redefining the functional architecture of the brain," *Trends in Neurosciences*, vol. 26, no. 10, pp. 523 – 530, 2003.

[13] M. D. Pittà, N. Brunel, and A. Volterra, "Astrocytes: Orchestrating synaptic plasticity?" *Neuroscience*, vol. 323, pp. 43 – 61, 2016, dynamic and metabolic astrocyte-neuron interactions in healthy and diseased brain.

[14] T. Fellin, J. Ellenbogen, M. De Pittà, E. Ben-Jacob, and M. Halassa, "Astrocyte regulation of sleep circuits: Experimental and modeling perspectives," *Frontiers in Computational Neuroscience*, vol. 6, p. 65, 2012.

[15] J. Liu, J. Harkin, L. McDaid, D. M. Halliday, A. M. Tyrrell, and J. Timmis, "Self-repairing mobile robotic car using astrocyte-neuron networks," in *2016 International Joint Conference on Neural Networks (IJCNN)*, 2016, pp. 1379–1386.

[16] S. Nadkarni and P. Jung, "Modeling synaptic transmission of the tripartite synapse," *Physical Biology*, vol. 4, no. 1, p. 1, 2007.

[17] V. Volman, E. Ben-Jacob, and H. Levine, "The astrocyte as a gatekeeper of synaptic information transfer," *Neural Comput.*, vol. 19, no. 2, pp. 303–326, 2007.

[18] W. Gerstner, R. Kempter, J. L. van Hemmen, and H. Wagner, "A neuronal learning rule for sub-millisecond temporal coding," *Nature*, vol. 383, no. 6595, pp. 76–78, 1996.

[19] M. Heidarpour, A. Ahmadi, and R. Rashidzadeh, "A cordic based digital hardware for adaptive exponential integrate and fire neuron," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 63, no. 11, pp. 1986–1996, 2016.

[20] T. Matsubara, H. Torikai, and T. Hishiki, "A generalized rotate-and-fire digital spiking neuron model and its on-fpga learning," *Circuits and Systems II: Express Briefs, IEEE Transactions on*, vol. 58, no. 10, pp. 677–681, 2011.

[21] J. Liu, J. Harkin, L. Maguire, L. McDaid, J. Wade, and M. McElholm, "Self-repairing hardware with astrocyte-neuron networks," in *2016 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2016, pp. 1350–1353.

[22] M. Heidarpur, A. Ahmadi, and N. Kandalaft, "A digital implementation of 2d hindmarsh-rose neuron," *Nonlinear Dynamics*, vol. 89, no. 3, pp. 2259–2272, Aug 2017.

[23] L. Maguire, T. McGinnity, B. Glackin, A. Ghani, A. Belatreche, and J. Harkin, "Challenges for large-scale implementations of spiking neural networks on {FPGAs}," *Neurocomputing*, vol. 71, no. 1–3, pp. 13 – 29, 2007, dedicated Hardware Architectures for Intelligent SystemsAdvances on Neural Networks for Speech and Audio Processing.

[24] E. Jokar and H. Soleimani, "Digital multiplierless realisation of a calcium based plasticity model," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. PP, no. 99, pp. 1–1, 2016.

[25] A. Ahmadi and M. Heidarpur, "An integrated astrocyte-adaptive exponential (aadex) neuron and circuit implementation," in *2016 24th Iranian Conference on Electrical Engineering (ICEE)*, 2016, pp. 1545–1550.

[26] M. Graupner and N. Brunel, "Calcium-based plasticity model explains sensitivity of synaptic changes to spike pattern, rate, and dendritic location," *Proceedings of the National Academy of Sciences*, vol. 109, no. 10, pp. 3991–3996, 2012.

[27] E. Jokar and H. Soleimani, "Digital multiplierless realisation of a calcium based plasticity model," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. PP, no. 99, pp. 1–1, 2016.

[28] M. Falcke, "Reading the patterns in living cells -the physics of ca2+ signaling," *Advances in Physics*, vol. 53, no. 3, pp. 255–440, 2004.

[29] S. Karim, J. Harkin, L. McDaid, B. Gardiner, J. Liu, D. M. Halliday, A. M. Tyrrell, J. Timmis, A. Millard, and A. Johnson, "Assessing self-repair on fpgas with biologically realistic astrocyte-neuron networks," in *2017 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, July 2017, pp. 421–426.

[30] S. R. McIver, M. Faideau, and P. G. Haydon, *Astrocyte-Neuron Communications*. Boston, MA: Springer US, 2013, pp. 31–64.

[31] S. Goutelle, M. Maurin, F. Rougier, X. Barbaut, L. Bourguignon, M. Ducher, and P. Maire, "The hill equation: A review of its capabilities in pharmacological modelling," *Fundamental and Clinical Pharmacology*, vol. 22, no. 6, pp. 633–648, 2008.

[32] R. J. Hyndman and A. B. Koehler, "Another look at measures of forecast accuracy," *International Journal of Forecasting*, vol. 22, no. 4, pp. 679 – 688, 2006.

Time Step Impact on Performance and Accuracy of Izhikevich Neuron: Software Simulation and Hardware Implementation

5.1 Introduction

Spiking neural networks (SNNs) are the third generation of neural networks, which neurons communicate through sequences of spikes. Comparing to previous generations, SNNs are faster, smaller in size, more energy efficient and biologically realistic [1, 2, 3]. There exist a number of mathematical models, with different levels of biological detail, that describe spiking neurons [4, 5, 6].

Simulation of SNNs is computationally complex because of nonlinear expressions in neuron models, size of the network and various communication pathways. Researchers have used variety of simulation platforms including: PCs [7], supercomputers [8], analog [9], digital [10] and mixed analog-digital application specific hardware

Table 5.1: Reported time step (ms) in some of published works that implemented spiking neurons and astrocytes on FPGA.

Reference	Time step	Reference	Time step
Grassia et al. [18]	0.015625	Matsubara et al. [19]	0.5
Soleimani et al. [20]	2^{-11}	Yang et al. [21]	0.125
Farsa et al. [22]	10^{-3}	Johnson et al. [23]	2^{-10}
Chen et al. [24]	10^{-4}	Heidarpur et al. [25]	2^{-7}

[11] and FPGAs [12]. Each of these platforms has its advantages and disadvantages. Nevertheless, in all platforms, optimization of SNN is important to increase performance and reduce cost of implementation. Some of the techniques to optimize neurons as main components of SNNs are presented in [13, 14, 15, 16].

Euler's method [17] is one of the most common methods used in previous works to numerically solve Ordinary Differential Equations (ODEs) describing neurons. A factor that has major impact on speed and computational cost of this method is time step. Smaller time steps preserve accuracy with the cost of higher computational cost.

Table 5.1 presents time steps used in some of published works that implemented spiking neurons and astrocytes on FPGA. As data indicates, employed time steps are different, even for the same model. We found no work explaining their criteria for selecting time steps. Indeed, in most works, authors just indicated that time step needs to be small enough and even they did not report its value [26, 27, 28]. Time step is an important factor, particularly in neuromorphic hardware, where it can significantly impact cost and speed of the system. Reference [29] investigated this impact for the software simulation of Izhikevich neuron. In [30], Computational Cost Factor (CCF) and Global Performance Factor (GPF) were calculated and compared for different spiking neurons firing at different rates.

In this paper, we investigated how different values of the time step affect the behaviour of the Izhikevich model. Further, we increased time step and obtained the threshold that neuron becomes unstable for different input currents. It was uncovered for the first time that instability threshold is an exponential function of neuron's

input current. Moreover, the neuron model was synthesized and implemented on the FPGA. Accuracy and hardware performance metrics such as speed, area and power consumption were compared for different time steps. Results indicated that larger time steps than those that used in the previous works could be employed which significantly improves performance and cost of the design.

The rest of this Chapter is organized as follows. Section 5.2 reviews Izhikevich neuron and presents the accuracy analysis for different time steps. Performance measurement for software simulation and hardware implementation are provided in the Section 5.3. Section 5.4 discusses achieved results. Finally, paper concludes in the Section 5.5.

5.2 Accuracy Analysis

In this work we used Izhikevich neuron as a case study model to analyze effects of changing time step on its accuracy and performance. Izhikevich neuron is a biologically inspired model in the form of two coupled differential equations as:

$$\frac{dv}{dt} = 0.04v^2 + 5v + 140 - u + I \tag{5.1}$$

and a reset equation as:

$$if\ v > \ 30\ mv \quad then \quad \begin{cases} v \rightarrow v_r \\ u \rightarrow w_r = u + d. \end{cases} \tag{5.2}$$

Where v is membrane potential, w is recovery variable and I is applied current. Other parameters are model constants.

5.2.1 Impact on Software Simulation

Fig. 5.1 shows the simulation of a tonic spiking and a tonic bursting Izhikevich neuron for different time steps (all time steps are in ms). A rectangular current pulse with maximum level of 4 was applied to the neurons as depicted in this figure with a red line. First, Izhikevich neuron was simulated with small time step of 0.001, and

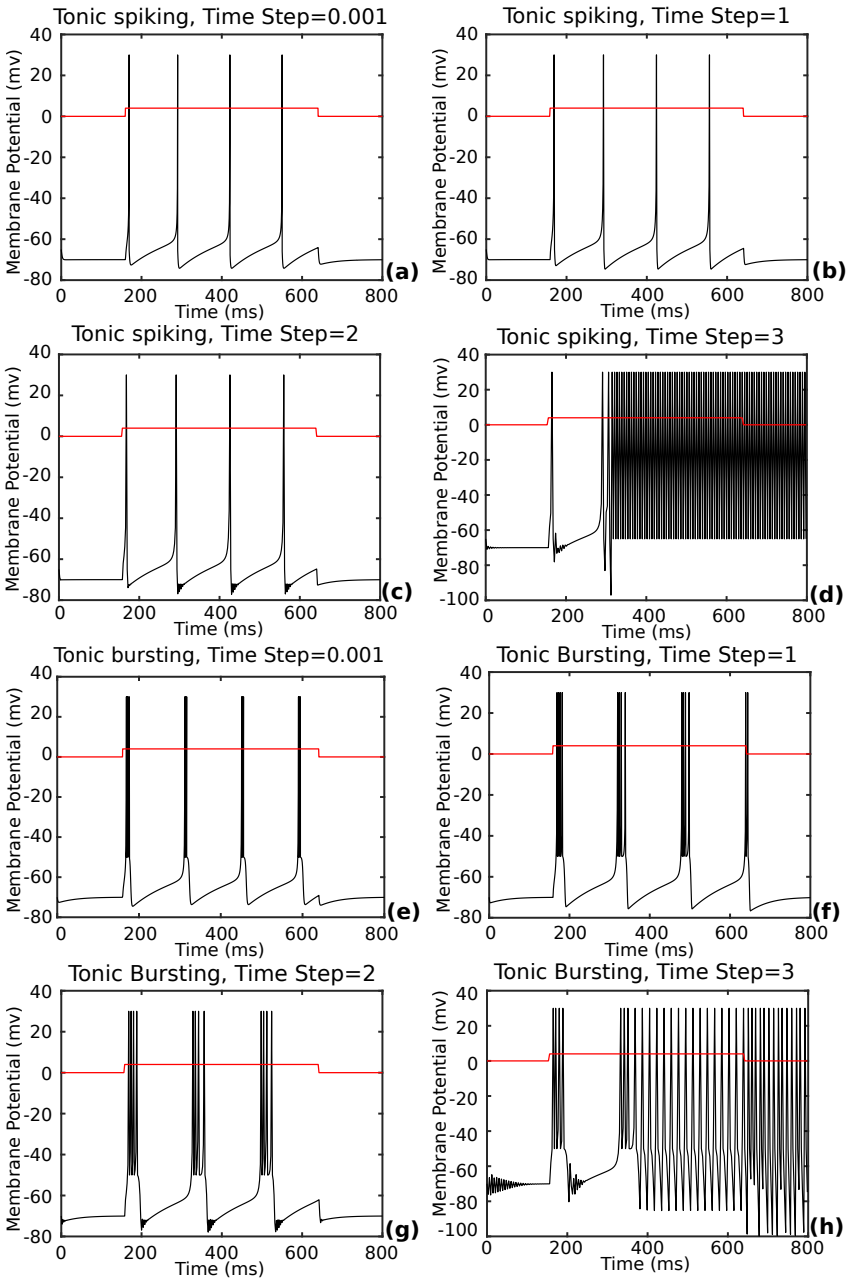


Figure 5.1: Simulation of the Izhikevich neuron for different time steps. First and second row show the result for case of tonic spiking and tonic bursting respectively. Input current is specified with a red color.

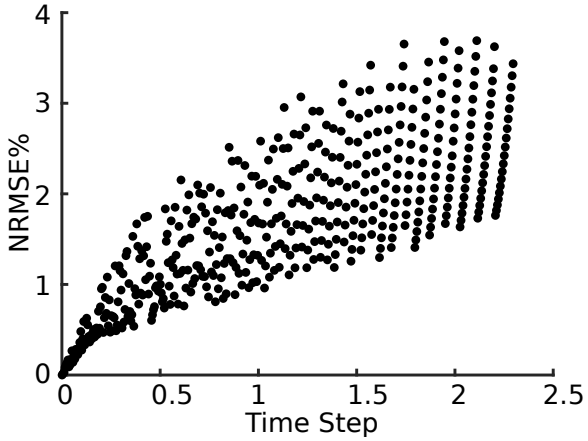


Figure 5.2: NRMSD error between waveform of the Izhikevich neuron with reference time step (0.001) and those of with larger time steps.

subsequently, the simulation repeated with larger time steps. As Fig. 5.1 indicates, for a time step up to 1, still the response of the model is similar to that of with the small time step of 0.001. The Izhikevich neuron with a time step of 2 showed additional unnecessary dumped oscillations in membrane potential waveform, but nonetheless, results resemble the neuron with time step of 0.001. The neuron with a time step of 3 is unstable and is not following the Izhikevich neuron anymore.

To quantitatively measure the error, the Normalized Root Mean Square Deviation (NRMSD) was calculated for each time step [31]. This error is defined as :

$$NRMSD = \frac{\sqrt{\sum_{i=1}^{n}(v(n) - v_{ts}(n))^2}}{\sqrt{n}\ (v_{max} - v_{min})} \tag{5.3}$$

where v and v_{ts} are waveforms of the izhikevich neuron with the reference (0.001) and larger time steps. v_{max} and v_{min} are the maximum and minimum values of v in the span of measuring the error. NRMSD error as function of the time step is plotted in Fig. 5.2. As this figures illustrates, with increasing the time step, NRMSD shows an increasing trend. Another factor that contributes to this error is the input current of the neuron. By increasing the input current, the spike frequency of the neuron increases. Therefore, for a specific time step (sampling time), higher input currents result in a larger error. In addition, neurons simulated with higher input current tend

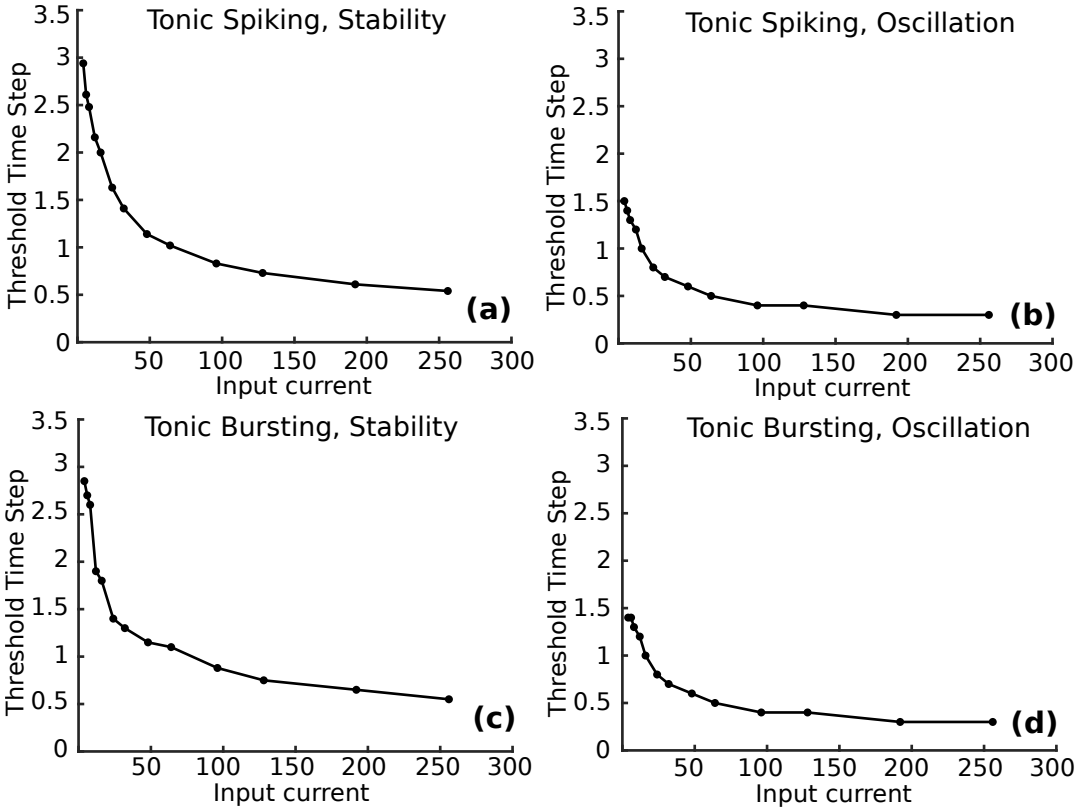


Figure 5.3: Threshold time steps that dumped oscillation starts in Izhikevich neuron and thresholds that it becomes unstable for two models of tonic spiking and tonic bursting.

to become unstable for a smaller time steps compared to those with lower currents.

The threshold time step that triggers dumped oscillation and the threshold that neuron becomes unstable are plotted against the input current in Fig. 5.3. As this figure shows, the threshold time steps are decreasing exponentially with the input current. A function approximation to the measured data in Fig. 5.3 (a) is:

$$ts_{th} = 5.76\ I^{-0.22} - 1.26 \tag{5.4}$$

Where ts_{th} is the stability threshold and I is the input current.

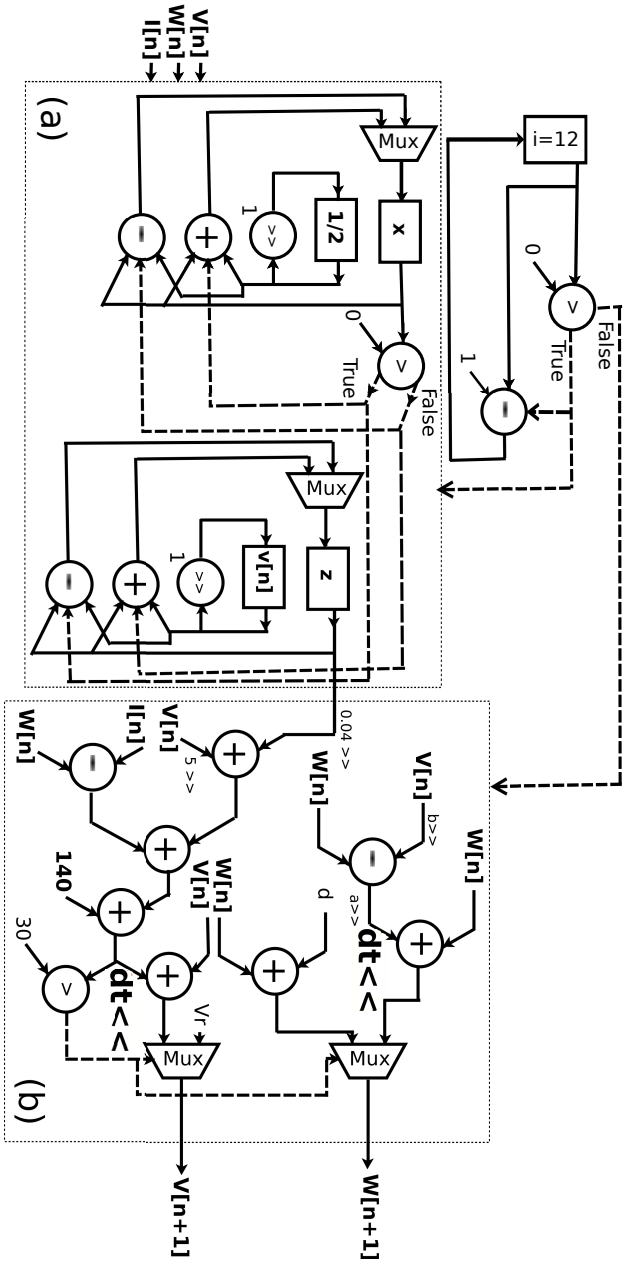
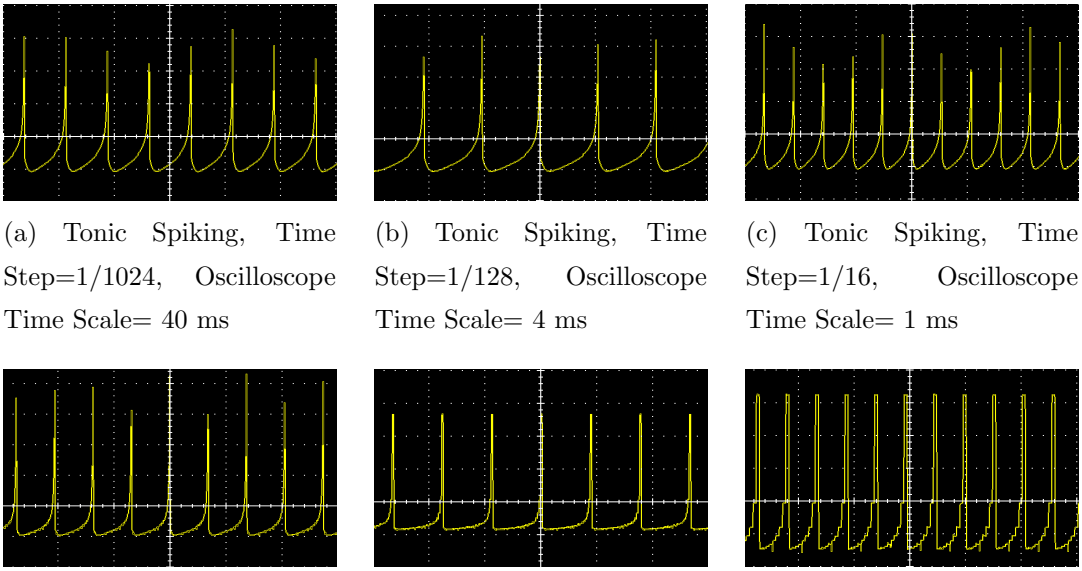


Figure 5.4: Control data flow graph for neuron's hardware. (Figure is taken from [32])

Table 5.2: Total memory required for 2 second simulation of Izhikevich neuron on Matlab software for different time steps.

Time Step	0.001	0.01	0.05	0.5	1
Memery	45.8 MB	4.2 MB	920 KB	94 KB	47 KB



(a) Tonic Spiking, Time Step=1/1024, Oscilloscope Time Scale= 40 ms

(b) Tonic Spiking, Time Step=1/128, Oscilloscope Time Scale= 4 ms

(c) Tonic Spiking, Time Step=1/16, Oscilloscope Time Scale= 1 ms

(d) Tonic Spiking, Time Step=1/2, Oscilloscope Time Scale= 100 μs

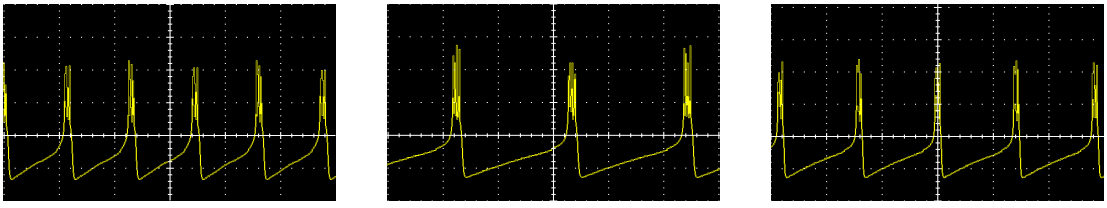
(e) Tonic Spiking, Time Step=1, Oscilloscope Time Scale= 40 μs

(f) Tonic Spiking, Time Step=2, Oscilloscope Time Scale= 20 μs

Figure 5.5: Oscilloscope photos of FPGA implementation of a tonic spiking Izhikevich neuron for various time steps. The input current for tonic spiking neuron is 12.

5.2.2 Impact on Hardware Implementation

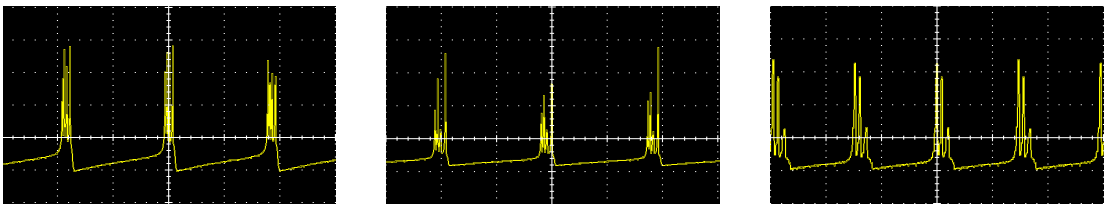
This section presents the behaviour of the on-FPGA Izhikevich neuron for various time steps. Fig. 5.4 shows the hardware data flow graph for FPGA implementation of the neuron. This design described in VHDL, synthesized using XILINX XST synthesizer and implemented on XILINX Spartan-6 XC6SLX75. For efficient digital implementation, a fixed point arithmetic was used. Coefficients in the model were approximated with the power of two numbers to reduce multiplication in constants to shift and add operations. The square term in Izhikevich neuron equation was



(a) Tonic Bursting, Time Step=1/1024, Oscilloscope Time Scale= 100 ms

(b) Tonic Bursting, Time Step=1/128, Oscilloscope Time Scale= 10 ms

(c) Tonic Bursting, Time Step=1/16, Oscilloscope Time Scale= 2 ms



(d) Tonic Bursting, Time Step=1/2, Oscilloscope Time Scale= 200 μs

(e) Tonic Bursting, Time Step=1, Oscilloscope Time Scale= 100 μs

(f) Tonic Bursting, Time Step=2, Oscilloscope Time Scale= 40 μs

Figure 5.6: Oscilloscope photos of FPGA implementation of a tonic bursting Izhikevich neuron for various time steps. The input current for tonic bursting neuron is 6.

calculated using COordinate Rotation DIgital Computer (CORDIC) technique. For more information about the design please refer to [32].

Fig. 5.5 and Fig. 5.6 shows oscilloscope photos of FPGA implementation of a tonic bursting and a tonic spiking Izhikevich neuron for different time steps. One can see in these figures that for relatively larger values of time step such as 0.5 or 1, the shape of the spikes and response of the neuron is still close to that with very small time step (1/1024). Even for time step of 2, the form of spikes is still preserved.

Table 5.3: FPGA frequency and resource utilization for the Izhikevich neuron with different time steps.

Time step	1/1024	1/128	1/16	1/2
LUTs	458	420	380	380
Slice Registers	253	229	205	205
Frequency (MHz)	177.5	182.7	185.9	185.9

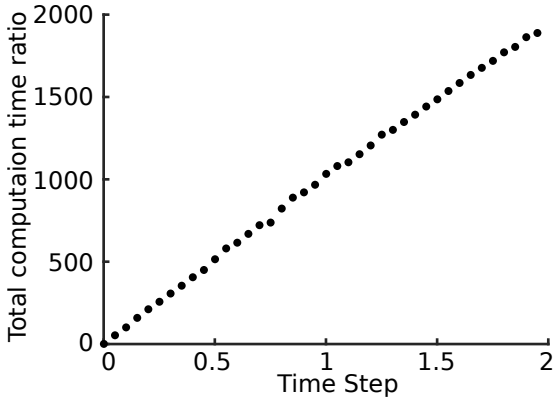


Figure 5.7: The ratio of computational time for larger time steps to that for 0.001. The results are obtained for 2 seconds Matlab simulation of this neuron.

5.3 Performance analysis

5.3.1 Impact on Software Simulation

The number of points that the neuron function is evaluated for a specific simulation time has a negative linear relation with the time step. Fig. 5.7 shows computation time required for a 2 second simulation of a single Izhikevich neuron on the Matlab as function of the time step. As this figure denotes, a decreasing time step will result in a corresponding linear increase in computation time. Furthermore, simulations with smaller time steps increase the memory usage. Table 5.2 compares the memory required for a 2 second simulation of the Izhikevich neuron on Matlab software for various time steps. According to this table, software simulation with a time step of 0.5 required 500 times less memory than that with a time step of 0.001. High memory

Table 5.4: On FPGA power and total number of clock cycles required for each design to generate 5 spikes.

Time Step	1/1024	1/128	1/16	1/2
Dynamic power (mW)	90	87	85	85
Number of clocks	13839040	1729880	216235	27030

usage could become a critical issue for large scale simulations.

5.3.2 Impact on Hardware Implementation

Area: Fig. 5.4 shows the points at which variables are shifted by the time step (indicated with 'dt' in Fig. 5.4 in the hardware . Changing the time step does not require any extra computational unit. However, small values of the time step require larger word length to prevent underflow. This, in turn, slightly increases the FPGA utilization of the design. Table 5.3 shows the implementation of the FPGA resource utilization for the Izhikevich neuron with different time steps. As this Table indicates, designs with smaller time steps require more resources. No extra bit is required for time steps smaller than 1/16.

Speed: Table 5.3 presents the frequency of neurons implemented with different time steps. Changing the time step does not directly result in a considerable change in the frequency and, therefore, in the throughput of the hardware. However, the spiking frequency linearly changes with the time step. For instance, in Fig. 5.5 (a), the period between two spikes for the neuron with time step of 0.001 is approximately 0.8×40ms=32ms. This time for the neuron with a time step of 1, as shown in Fig. 5.5 (e), is roughly $0.9\times40\mu$s=36μs. In other words, neurons with a time step of 1 are roughly 1125 times faster than those with time step of 1/1024.

Power: Table 5.4 compares the on-FPGA power for the neurons with different time steps. As this table shows, designs with smaller time steps consume marginally less power. However, energy consumption is a function of the total time taken to complete a task. Table 5.4 presents the number of clock cycles required for each design to generate 5 spikes. According to this table, generating 5 spikes with a time

step of 1/1024 consumes $64 \times (90/85) = 67.8$ times more power compared to those with time step of 1/16.

5.4 Discussion

As discussed in the previous sections, for the Izhikevich neuron there is a threshold time step in which unnecessary dumping oscillations appear in the output waveform of the neuron. If we continue to increase the time step, at a certain point the neuron becomes unstable. Further, we showed that the time step linearly affects the performance of the software simulations and hardware implementation. The question to raise is what would be the optimum time step for simulation of Izhikevich neuron? The answer is dependant on the maximum input current. For instance, in a network by knowing the number of the inputs and the maximum weight of each one, the maximum input to the neuron and accordingly the appropriate time step could be determined using Fig. 5.3.

5.5 Conclusion

In this work, the relation between the threshold time step that neuron produces stable output and its input current was uncovered. Total computational time and memory required for simulation of a single Izhikevich neuron with different time steps were measured and compared. Further, the model was implemented on the FPGA and design with different time steps were compared in the terms of speed, area and power consumption.

References

[1] M. Pfeiffer and T. Pfeil, "Deep learning with spiking neurons: Opportunities and challenges," *Frontiers in Neuroscience*, vol. 12, p. 774, 2018.

[2] M. Davies, N. Srinivasa, T.-H. Lin, G. Chinya, Y. Cao, S. H. Choday, G. Dimou, P. Joshi, N. Imam, S. Jain *et al.*, "Loihi: A neuromorphic manycore processor with on-chip learning," *IEEE Micro*, vol. 38, no. 1, pp. 82–99, 2018.

[3] J. Vreeken, "Spiking neural networks, an introduction," 2003.

[4] A. L. Hodgkin and A. F. Huxley, "A quantitative description of membrane current and its application to conduction and excitation in nerve," *The Journal of physiology*, vol. 117, no. 4, p. 500, 1952.

[5] E. M. Izhikevich, "Simple model of spiking neurons," *IEEE Transactions on neural networks*, vol. 14, no. 6, pp. 1569–1572, 2003.

[6] W. Gerstner and W. M. Kistler, *Spiking neuron models: Single neurons, populations, plasticity*. Cambridge university press, 2002.

[7] "Nemo: High-performance spiking neural network simulator," 2019. [Online]. Available: http://nemosim.sourceforge.net/

[8] "The human brain project," 2019. [Online]. Available: https://www.humanbrainproject.eu

[9] E. Covi, S. Brivio, M. Fanciulli, and S. Spiga, "Synaptic potentiation and depression in al: Hfo2-based memristor," *Microelectronic Engineering*, vol. 147, pp. 41–44, 2015.

[10] F. Akopyan, J. Sawada, A. Cassidy, R. Alvarez-Icaza, J. Arthur, P. Merolla, N. Imam, Y. Nakamura, P. Datta, G.-J. Nam *et al.*, "Truenorth: Design and tool flow of a 65 mw 1 million neuron programmable neurosynaptic chip," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, no. 10, pp. 1537–1557, 2015.

[11] B. V. Benjamin, P. Gao, E. McQuinn, S. Choudhary, A. R. Chandrasekaran, J. M. Bussat, R. Alvarez-Icaza, J. V. Arthur, P. A. Merolla, and K. Boahen, "Neurogrid: A mixed-analog-digital multichip system for large-scale neural simulations," *Proceedings of the IEEE*, vol. 102, no. 5, pp. 699–716, May 2014.

[12] D. Neil and S. Liu, "Minitaur, an event-driven fpga-based spiking network accelerator," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 22, no. 12, pp. 2621–2628, Dec 2014.

[13] T. Naka and H. Torikai, "A novel generalized hardware-efficient neuron model based on asynchronous ca dynamics and its biologically plausible on-fpga learnings," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 66, no. 7, pp. 1247–1251, July 2019.

[14] S. M. Kueh and T. Kazmierski, "A dedicated bit-serial hardware neuron for massively-parallel neural networks in fast epilepsy diagnosis," in *2017 IEEE Healthcare Innovations and Point of Care Technologies (HI-POCT)*, Nov 2017, pp. 105–108.

[15] M. Heidarpur, A. Ahmadi, and N. Kandalaft, "A digital implementation of 2d hindmarsh–rose neuron," *Nonlinear Dynamics*, vol. 89, no. 3, pp. 2259–2272, Aug 2017.

[16] R. Brette and W. Gerstner, "Adaptive exponential integrate-and-fire model as an effective description of neuronal activity," *Journal of neurophysiology*, vol. 94, no. 5, pp. 3637–3642, 2005.

[17] D. F. Griffiths and D. J. Higham, *Euler's Method*. London: Springer London, 2010, pp. 19–31.

[18] F. Grassia, T. Levi, T. Kohno, and S. Saïghi, "Silicon neuron: digital hardware implementation of the quartic model," *Artificial Life and Robotics*, vol. 19, no. 3, pp. 215–219, Nov 2014.

[19] T. Matsubara and H. Torikai, "Asynchronous cellular automaton-based neuron: Theoretical analysis and on-fpga learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 24, no. 5, pp. 736–748, May 2013.

[20] H. Soleimani, A. Ahmadi, and M. Bavandpour, "Biologically inspired spiking neurons: Piecewise linear models and digital implementation," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 59, no. 12, pp. 2991–3004, Dec 2012.

[21] S. Yang, J. Wang, S. Li, B. Deng, X. Wei, H. Yu, and H. Li, "Cost-efficient fpga implementation of basal ganglia and their parkinsonian analysis," *Neural Networks*, vol. 71, pp. 62 – 75, 2015.

[22] E. Z. Farsa, A. Ahmadi, M. A. Maleki, M. Gholami, and H. N. Rad, "A low-cost high-speed neuromorphic hardware based on spiking neural network," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 66, no. 9, pp. 1582–1586, Sep. 2019.

[23] A. P. Johnson, D. M. Halliday, A. G. Millard, A. M. Tyrrell, J. Timmis, J. Liu, J. Harkin, L. McDaid, and S. Karim, "An fpga-based hardware-efficient fault-tolerant astrocyte-neuron network," in *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, Dec 2016, pp. 1–8.

[24] Q. Chen, J. Wang, S. Yang, Y. Qin, B. Deng, and X. Wei, "A real-time fpga implementation of a biologically inspired central pattern generator network," *Neurocomputing*, vol. 244, pp. 63 – 80, 2017.

[25] M. Heidarpour, A. Ahmadi, and R. Rashidzadeh, "A cordic based digital hardware for adaptive exponential integrate and fire neuron," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 63, no. 11, pp. 1986–1996, Nov 2016.

[26] H. Soleimani and E. M. Drakakis, "A compact synchronous cellular model of non-linear calcium dynamics: Simulation and fpga synthesis results," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 11, no. 3, pp. 703–713, June 2017.

[27] T. Matsubara, H. Torikai, and T. Hishiki, "A generalized rotate-and-fire digital spiking neuron model and its on-fpga learning," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 58, no. 10, pp. 677–681, Oct 2011.

[28] E. Jokar, H. Abolfathi, and A. Ahmadi, "A novel nonlinear function evaluation approach for efficient fpga mapping of neuron and synaptic plasticity models," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 13, no. 2, pp. 454–469, April 2019.

[29] S. Valadez-Godínez, H. Sossa, and R. Santiago-Montero, "The step size impact on the computational cost of spiking neuron simulation," in *2017 Computing Conference*, July 2017, pp. 722–728.

[30] S. Valadez-Godínez, H. Sossa, and R. Santiago-Montero, "On the accuracy and computational cost of spiking neuron implementation," *Neural Networks*, 2019.

[31] R. J. Hyndman and A. B. Koehler, "Another look at measures of forecast accuracy," *International journal of forecasting*, vol. 22, no. 4, pp. 679–688, 2006.

[32] M. Heidarpur, A. Ahmadi, M. Ahmadi, and M. Rahimi Azghadi, "Cordic-snn: On-fpga stdp learning with izhikevich neurons," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 66, no. 7, pp. 2651–2661, July 2019.

Conclusion and Future Work

6.1 Summary

In this dissertation, several hardware developed for efficient digital implementation of spiking neural network on FPGAs. To take full advantage of available resources on FPGAs, one needs to develop a rich library of different neural network building blocks and primitives to be used in a high level synthesizer for large scale implementation of SNNs. This research could be considered as step toward this objective.

In Chapter 2, a novel hardware was presented based on the CORDIC method for on-FPGA online STDP learning. This hardware proved to be accurate while requiring less FPGA resources and having higher speed compared to the original models and state-of-the-art designs. The CORDIC method was utilized because of the simplicity of its structure, since it only uses add and shift operations which could be cheaply implemented on hardware. In order to implement the proposed learning system, first, the CORDIC method was used to implement Izhikevich neurons and its accuracy was analyzed. Second, the STDP algorithm was adopted for online learning and modified using the CORDIC algorithm to improve hardware efficiency. Furthermore, error analysis was performed on computer simulation data to ensure the accuracy of the implemented CORDIC models. Consequently, hardware was designed, described in VHDL, and simulated for both neuron and learning mechanism. Finally, the models were implemented on FPGA to form a spiking neural network composed of Izhikevich

neurons and STDP synapses to demonstrate competitive Hebbian learning.

In Chapter 3, a digital hardware was proposed to implement biological-plausible models of astrocyte and glutamate release mechanism. The design demonstrated high accuracy in replicating the behaviour of aforementioned biological cells in hardware. Furthermore, implementation results indicated that the design is much more area efficient and faster comparing to recently published works. This new design, allows researchers to implement large number of these biological-plausible cells on FPGAs. This is most important because, unlike high level models, simulation of these models due to high biological details requires long time and computational power and will fall behind the real time easily. This hardware is most useful to replicate the tripartite synapse and its components. Such hardware could also be scaled to study brain diseases and information processing algorithms.

In Chapter 4, a biological G-ChI astrocyte model for Ca^{2+} oscillation was modified for hardware implementation. Simulation data reveals that these models follow the original model with an acceptable accuracy. The simplicity of the models, which only consist of add/sub and shift operations, made it possible to implement the nonlinear astrocyte equations effectively on hardware. The HDL code describing the hardware was first simulated and further implemented on FPGA as proof of concept.

In Chapter 5, the relation between the threshold time step that neuron produces stable output and its input current was uncovered. Total computational time and memory required for simulation of a single Izhikevich neuron with different time steps were measured and compared. Further, the model was implemented on the FPGA and design with different time steps were compared in the terms of speed, area and power consumption.

6.2 Conclusion

In this dissertation, hardware were presented based on CORDIC and linearization method to implement spiking neural networks on FPGA.

Using linearization technique, nonlinear terms in the neuron and astrocyte differ-

ential equations were replaced with a sequence of linear segments. This modification, resulted in a small deviation in behaviour of astrocyte model. NRMSE error was calculated to numerically measure this error. It was observed that for a higher number of linear segments, this error tends to be smaller. As an example, for the term N2 in the Table 4.1, this error was 0.011 for 4 linear segments, 0.009 for 6 linear segments and 0.006 for 9 linear segments. Different models was proposed with different number of linear segments. Furthermore, computer simulations and error analysis were performed to ensure proper function of the modified astrocyte model.

In another technique, CORDIC method was used to calculate nonlinear terms in the differential equations describing spiking neurons and astrocytes. Comparing with previous method, this algorithm had considerably higher precision and lower deviation from the original model. For instance the NRMSD error for the CORDIC Izhikevich measured as 0.0034 for 6 iterations, 0.0006 for 8 iterations and 0.0001 for 10 iterations. Several errors including NRMSD, MD and timing error was calculated to compare the results. Afterwards, the original and CORDIC based models were simulated using Matlab software both as a single cell and population of cells. Simulation results confirmed that CORCID models follow the original model with a very small deviation.

Further, implementation results confirmed the proper functioning of the proposed CORDIC and linearization based SNNs on FPGA. These results also show that the proposed design can lead to more efficient and faster FPGA-based SNNs compared to implementation of the original models and other available implementations in the literature. As an example, the Izhikevich neuron in the chapter 2, while working in higher frequency, uses less resources comparing to other implementations in the literature, almost half and even one third in some cases. Despite being more efficient, the implemented model had considerably smaller NRMSD error of 0.003% comparing to other works where the minimum reported error was 3.7%. In chapter 3 a biological astrocyte model is implemented on the FPGA. As a result of using CORDIC algorithm, the implemented model also had a very small error. While being accurate, implementation results indicated that the design is more area efficient and faster comparing to other works. The minimum resources reported in the literature for implementation

of a similar astrocyte model are 11394 LUTs, 11666 slice registers and 42 DSP slices while the proposed hardware uses 1156 LUTs, 1380 slice registers and no DSP slices. This results indicate more than ten times optimization of the circuits. In the terms of speed, delay from input to output for the proposed hardware is 1.02 μs while this delay is reported in another work as 12.5 μs which indicates that the CORDIC based model is approximately 12 times faster. Using the CORDIC method, the area were decreased to one tenth while being almost 12 times faster.

Overall, this dissertation contributes to design and implementation of low-cost and high-speed large-scale digital neuromorphic systems. It is important to note that FPGA devices utilize more resources for hardware implementation than that of ASICs. Implementing such hardware on silicon would have considerably less cost and have better performance.

6.3 Suggested Future Work

The work in this dissertation could be continued as following :

• In this dissertation the CORCID algorithm was used to implement nonlinear terms in differential equations describing biological neurons and astrocytes. This algorithm has a very high precision , however, it is an iterative algorithm and requires a certain number of the iteration to complete. Author believe that this algorithm could be optimized to compensate the delay correspond to the iterations since the the derivative of the membrane potential is have a limited value.

• So far, using the CORDIC algorithm, the models describing astrocyte and its glutamate release mechanism were implemented on a FPGA. This work could be continued by implementing other biological detailed models such Hodgkin-Huxley neuron and eventually building a biological detailed neuro-processor.

• The CORDIC algorithm was used to calculate nonlinear terms in neuron and astrocyte ODEs. Author believe that a CORDIC algorithm could be developed to calculate all differential equations which make the implementation very cheaper and faster.

VITA AUCTORIS

Moslem Heidarpur received the B.Sc. degree in electrical engineering and M.Sc. degree in electronic engineering from the Department of Electrical Engineering, Razi University, Kermanshah, Iran, in 2012 and 2014. He is now Ph.D. student in the University of Windsor, Canada. His research interests include analog and digital electronic circuit design and optimization, bio-inspired computing, neuromorphic and integrated circuit design.