AFRL-RI-RS-TR-2018-009

ANALOG INTEGRATED CIRCUIT DESIGN FOR SPIKE TIME DEPENDENT ENCODER AND RESERVOIR IN RESERVOIR COMPUTING PROCESSORS

UNIVERSITY OF KANSAS CENTER FOR RESEARCH, INC.

JANUARY 2018

FINAL TECHNICAL REPORT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

STINFO COPY

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE

■ **AIR FORCE MATERIEL COMMAND** ■ **UNITED STATES AIR FORCE** ■ **ROME, NY 13441**

NOTICE AND SIGNATURE PAGE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

This report is the result of contracted fundamental research deemed exempt from public affairs security and policy review in accordance with SAF/AQR memorandum dated 10 Dec 08 and AFRL/CA policy clarification memorandum dated 16 Jan 09. This report is available to the general public, including foreign nations. Copies may be obtained from the Defense Technical Information Center (DTIC) (http://www.dtic.mil).

AFRL-RI-RS-TR-2018-009 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

FOR THE CHIEF ENGINEER:

/ S /
GEORGE RAMSEYER
Work Unit Manager

/ S /
JOHN D. MATYJAS
Technical Advisor, Computing
Communications Division
Information Directorate

This report is published in the interest of scientific and technical information exchange, and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
JANUARY 2018	FINAL TECHNICAL REPORT	FEB 2016 – SEP 2017

4. TITLE AND SUBTITLE

ANALOG INTEGRATED CIRCUIT DESIGN FOR SPIKE TIME DEPENDENT ENCODER AND RESERVOIR IN RESERVOIR COMPUTING PROCESSORS

5a. CONTRACT NUMBER
N/A

5b. GRANT NUMBER
FA8750-16-2-0120

5c. PROGRAM ELEMENT NUMBER
62788F

6. AUTHOR(S)

Yang Yi

5d. PROJECT NUMBER
T2DN

5e. TASK NUMBER
KA

5f. WORK UNIT NUMBER
NS

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

University of Kansas Center for Research, Inc.
2385 Irving Hill Rd
Lawrence KS 66045-7552

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Air Force Research Laboratory/RITB
525 Brooks Road
Rome NY 13441-4505

10. SPONSOR/MONITOR'S ACRONYM(S)
AFRL/RI

11. SPONSOR/MONITOR'S REPORT NUMBER
AFRL-RI-RS-TR-2018-009

12. DISTRIBUTION AVAILABILITY STATEMENT
Approved for Public Release; Distribution Unlimited. This report is the result of contracted fundamental research deemed exempt from public affairs security and policy review in accordance with SAF/AQR memorandum dated 10 Dec 08 and AFRL/CA policy clarification memorandum dated 16 Jan 09

13. SUPPLEMENTARY NOTES

14. ABSTRACT

The objective of this effort was to: (a) develop novel and fundamental methodologies for data representation using hardware-based spike timing dependent encoding for neuromorphic processors; (b) build a new design of computationally efficient delay-based reservoirs that meet the requirements of high dimensionality and finite memory. This project resulted in an agile analog integrated circuit design of a spike-time encoding circuit as a signal conditioner and electronic reservoir as a dynamic processor for the reservoir computing systems. This is a multidisciplinary effort bridged high-performance computing, nanotechnology, and integrated circuits & systems.

15. SUBJECT TERMS
neuromorphic computing, neuron design, spike timing dependent encoding, reservoir computing

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			GEORGE RAMSEYER
U	U	U	UU	73	19b. TELEPHONE NUMBER (*Include area code*)

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1.0 SUMMARY

This effort is a critical part of an overall program to develop novel and fundamental methodologies for data representation using hardware-based spike timing dependent encoding for neuromorphic processors, and to build a new class of computationally efficient hardware delay-based reservoirs that meet the requirements of high dimensionality and finite memory. This project resulted in the design of an agile analog integrated circuit implementation of a spike-time encoding circuit as a signal conditioner for reservoir computing systems. This multidisciplinary effort encompassed high-performance computing, nanotechnology, integrated circuits, and integrated systems. The project's architecture was designed to be the foundation for future capabilities in signature analyses and time-series classifications. Additionally, once this design is fabricated and tested, it will be applicable to other neuromorphic computing applications. The development of neuromorphic computing devices will facilitate increased energy efficient computing.

2.0 INTRODUCTION

This project initially focused on the additional testing of the analog reservoir computing hardware [1] that had previously been fabricated at nanotechnology dimensions. We had designed and fabricated a hardware spike timing dependent encoder for neuromorphic processing that included circuit designs, design simulations, fabrication, and initial testing. Based upon that and the further testing of that encoder in this project, we have here designed, simulated, and sent to fabrication a dynamic-reservoir circuit that utilizes sensory encoding methodologies similar to those employed in biological brains.

Inspired by the neural architecture of the human brain, neuromorphic computing [2] has been proposed as a mechanism for circumventing the limitations of current generation computing systems to accommodate the ever-increasing demand for advanced computing requirements. An intrinsic learning capability of neuromorphic computing systems allows them to automatically adapt to targeted functionalities without requiring additional hardware resources. To date research in this area has been narrowly focused on the enhancement of neural network algorithms, while system implementations have primarily been confined to conventional computing units.

The straightforward hardware realization of neural networks will require a large number of memory and computing resources, resulting in high design complexities with accompanying high hardware costs. As an example, text-recognition software [3] has been designed to run on high-performance computing (HPC) clusters. In this specific work there were 70,000 processor cores that provided a massive peak computing power of 500 trillion floating-point operations per second. Algorithm enhancements and conventional hardware implementations can to some degree mitigate computation cost issues, but do not fundamentally resolve it. It is essential to design a new class of hardware that is optimized for conducting the crucial operations of neuromorphic computing, rather than relying on system implementations built upon traditional and nontraditional computer structures found in data centers.

Many types of real time neuromorphic computing applications require high-performance processing that is based upon von Neumann architecture systems. Size, Weight, and Power (SWAP) resource restrictions have ruled out many traditional high performance computing software-based approaches to many types of real-time applications, particularly outside of data processing centers. However, simulations have shown that future implementations of dedicated real-time hardware-based neuromorphic systems are ideal for pattern and signature recognitions for mobile platforms with severe (SWAP) constraints. This research project holds great promise for these and several

other important engineering and scientific applications. Systems which exploit non-traditional architectures that encompass evolutionary systems may leverage these behaviors to address specific classes of mission-critical problems that have currently not been solved by state-of-the-art complementary metal–oxide semiconductor (CMOS) digital computing.

Most reservoir computing hardware implementations are analog based. Digital implementation-based reservoir computing [4-8] offers higher computational precision, higher reliability, and is simpler to program. Synaptic weights can be stored either on or off chip. Disadvantages of digital implementations compared to analog implementations [9-13] include the necessity of relatively large circuit sizes and the consumption of higher amounts of power.

Analog implementations take advantage of fundamental electronic and physical principles to implement base functions. For example, operational amplifiers more simply perform neuron-like functions such as sigmoid transfer. Likewise, temporal integration is achieved through capacitive integrations and spatial summations based on Kirchhoff's Law. In analog computationally intensive calculations are automatically performed by the summing of currents or charges. Furthermore, analog implementations of reservoir computing systems offer significantly higher speeds, less design areas, and less energy dissipations than digital implementations.

Here we have encoded neural-type responses using different timescales and with different stimulus attributes that generate temporal inter-spike intervals of sensory information. We have compared the performances of rate codes and temporal codes, and have demonstrated the computational advantages of temporal code with inter-spike intervals. The effort included reducing the ambiguity inherent in single-scale codes, and enhancing the robustness of neural representations compared to background environmental noise.

This project began with circuit-level designs and bench-level simulations of electronic reservoirs, and proceeded to explore the effects of operational parameters. Parameters examined included the operational frequencies of the input-spike trains, varying the levels of nonlinear and chaotic dynamics, and the effects of various feedback delay and dampening parameters. The hardware implementation of nonlinear dynamics and delay feedback reservoirs were examined, which will enable networks to mimic transient neuronal responses and to project time-dependent inputs into high dimensionalities for categorizations by an outside classifier.

The overall project was organized around these two interconnected research thrusts:

- Thrust 1: A Review of and further Examination of Analog Spike Time-Dependent Encoder based upon a Leaky Integrate-and-Fire (LIF) Neuron Model.

- Thrust 2: Single Delayed-Feedback Reservoir Design with Nonlinear Node and Delay Lines.

In Thrust 1, which was nearly completed before this effort began, the following research tasks were conducted in the pursuit of a spike time dependent encoder design with the LIF Neuron Model:

Task 1.1: Comprehensive investigation of neural encoding schemes,
Task 1.2: Creating a spike time dependent encoder design with LIF model, and
Task 1.3: Encoder circuit fabrication with advanced CMOS nano-technology.

While Trust 1 was mostly completed before this effort began [1, 10], some additional testing of the fabricated encoder occurred during the early part of this effort, which established a more solid baseline for Thrust 2. Thrust 1 had produced deliverables that included an encoder circuit design, Simulation Program with the accompanying Integrated Circuit Emphasis (SPICE) circuit models, and prototypes of the circuit. As its final outcome, the Thrust 1 project provided an agile hardware implementation of spike time encoding as a signal conditioner for dynamical neural processor designs. Some of the results that were previously published are included in this report for completeness.

In Thrust 2, which is what this effort primarily focused on, the following tasks were completed as part of the research plan to design, simulate and fabricate a bench-level dynamic reservoir to serve as a computational building block within a scalable architecture:

Task 2.1: Nonlinear node design with chaotic circuits,
Task 2.2: Delay line design with CMOS transistors, and
Task 2.3: Fabrication of the designed nonlinear reservoir.

Thrust 2 produced deliverables including a dynamic reservoir circuit design, SPICE circuit models, and circuit prototypes. These designs and models were simulated, and the resultant circuit design was sent to fabrication.

The overall goal of Thrust 2 was to provide a nonlinear processor designed to exploit recent advancements in nano-technology and interconnects for a new class of computational systems based on dynamically driven architectures. This effort started in February 2016, and was planned to end in December 2018. However, due to the transfer of the Principal Investigator from the University of Kansas to the Virginia Polytechnic Institute and State University, this effort ended earlier than expected. By the end of this effort, most of the originally planned tasks were completed.

The two remaining tasks that were not completed were:

1) Testing the fabricated designed nonlinear dynamic reservoir circuit, and
2) The further optimization of the nonlinear dynamic reservoir circuitry.

In the following sections the Methods, Assumptions, Procedures, Results and Discussions for each of the aforementioned contributions are presented.

3.0 METHODS, ASSUMPTIONS, AND PROCEDURES

The overall project is summarized into two interconnected research thrusts: 1) Thrust 1: Additional Review/Comments of the Analog Spike Time-Dependent Encoder with the LIF Model, and 2) Thrust 2: Single Delayed-Feedback Reservoir Design and Simulations of a Nonlinear Node and Delay Lines. This effort utilized the previous results from Trust 1, which have been summarized here, as a basis for Trust 2.

Thrust 1: Analog Spike Time-Dependent Encoder

The following Thrust 1 tasks are summarized from a recent report [1], and include even more recent updates and summaries of the development and evaluation of a spike time dependent encoder based on the LIF model:

Task 1.1: Comprehensive investigation of neural encoding schemes

There are three general types of neural encoding schemes: rate encoding, temporal encoding by a latency code, and temporal encoding by inter-spike intervals [14]. In rate encoding the stimulus is encoded as the number of spikes within an encoding window, and the exact timings of individual spikes are not relevant [15-18]. In temporal encoding the stimulus is effected by latency, and together are encoded by the time difference of the first spike with respect to the stimulus onset. In temporal encoding by inter-spike intervals, the stimulus is encoded by the relative timing of the spikes within the encoding window, which is the time differences of successive spikes. There are lively discussions in the literature [12, 15, 18-19] concerning the benefits of temporal encoding vs. rate encoding.

These three types of neural encoding schemes are comprehensively investigated, neural activity that is patterned on multiple timescales is examined and discussed, and ways to evaluate these types of patterns are presented. The code performances for each of the encoding schemes are reviewed. The advantages afforded by temporal Inter-Spike Intervals (ISI) code for representing sensory information is shown.

Task 1.2: Spike time dependent encoder design using the LIF model

The LIF model afforded both simple structures and higher accuracy compared to other neuron models, which included the Integrate-and-Fire (IF) [20], the Hodgkin-Huxley (HH) [21], and the FitzHugh-Nagumo (FN) [22] models. Circuits designed using the LIF model have a balance between scalability and accuracy. In a LIF circuit the membrane capacitor generates the membrane current. The LIF system fires when the membrane current has charged up the membrane voltage

such that it exceeds a specific threshold voltage [23], and a reset signal resets the membrane voltage and the refractory period. The shape of the action potential is not considered. We compared the analog and digital implementations of neurons using the LIF model, and have shown how the signal-processing tasks can be modularly decomposed into elementary operators.

A robust and compact analog neuron [10] was designed and modelled using CMOS technology. Design aspects such as integration time, threshold voltage, refractory period, and encoding resolution were examined to optimize the adaptability and reconfigurability of the neuron operational parameters. Other characteristics including power consumption, the die areas required for chip fabrications, and the robustness to process variability were also reported.

Task 1.3: Encoder circuit fabrication and testing with advanced CMOS nano-technology

We had previously created an optimized circuit schematic [1], and then used the Cadence Virtuoso Analog Design Environment to generate the layout of our designed encoding circuit. This design tool was used for all aspects of designing the fully customized Integrated Circuits, including the schematic entry, the behavioral modeling (Verilog-AMS), the circuit simulation, the custom layout, the physical verification, the extraction and the back-annotation. It was also used for the analog, mixed-signal, radio frequency (RF), and standard-cell designs, and for the memory and FPGA designs. The flowchart for a layout generation is shown in Fig. 1.

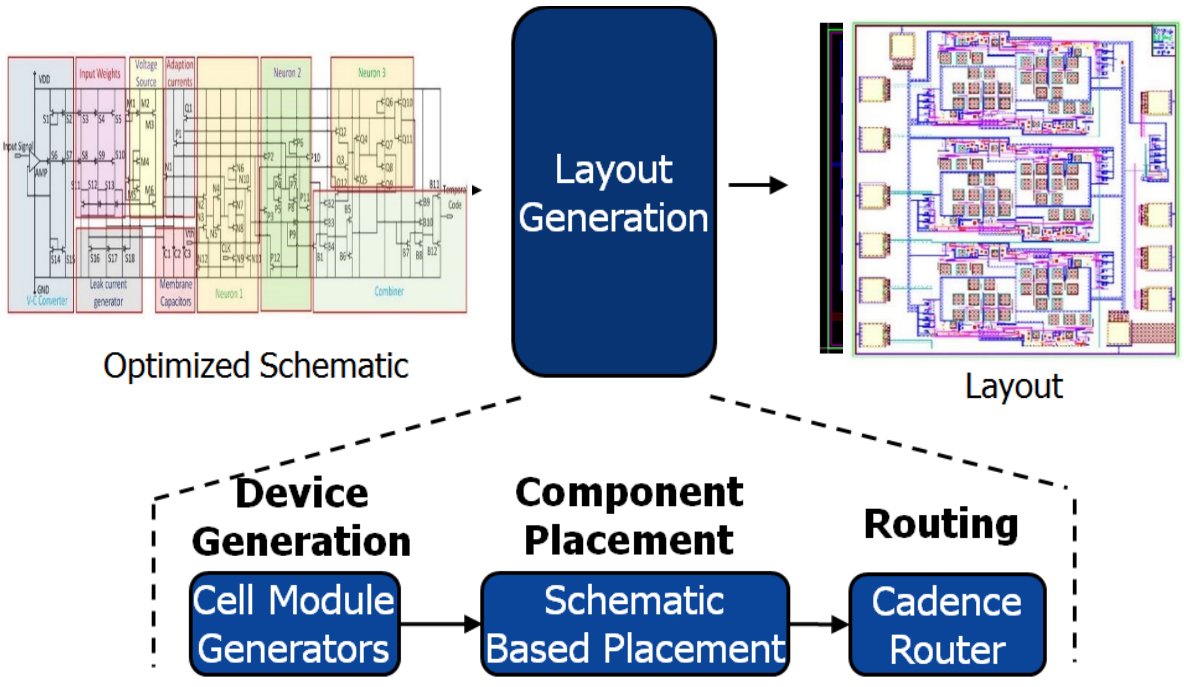


Figure 1. Flow Chart of the Layout Generation

We had previously fabricated this encoder chip [1] using the Global Foundries 180nm CMOS technology. The fabricated chip was tested at Kansas University's Information and Telecommunications Technology Center using their state-of-the-art electronic design automation (EDA) tools for circuit designs, modeling, and simulations. The Center also has specialized lab facilities for integrated circuit testing. The next step was bonding the package to the printed circuit board (PCB) board that was developed for testing. Fig. 2 shows a microscope images of a fabricated chip and its testing board, respectively.

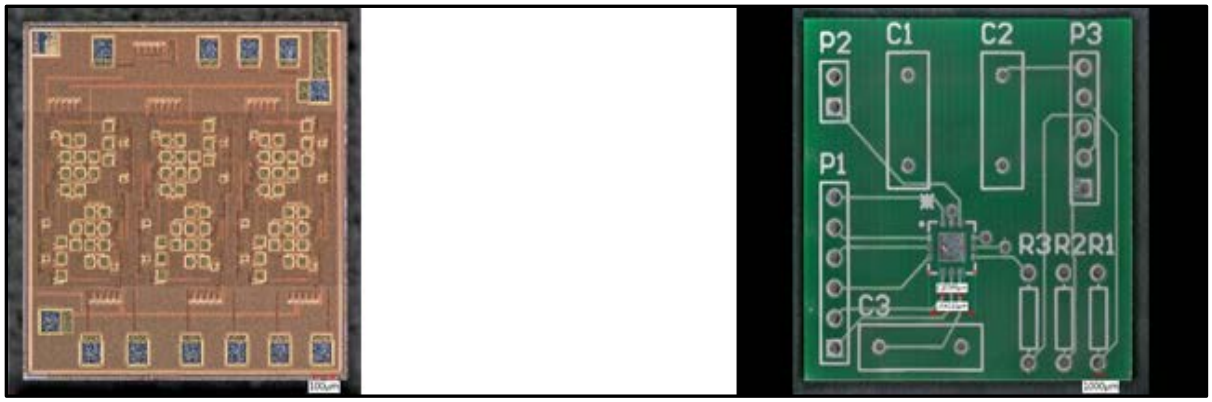
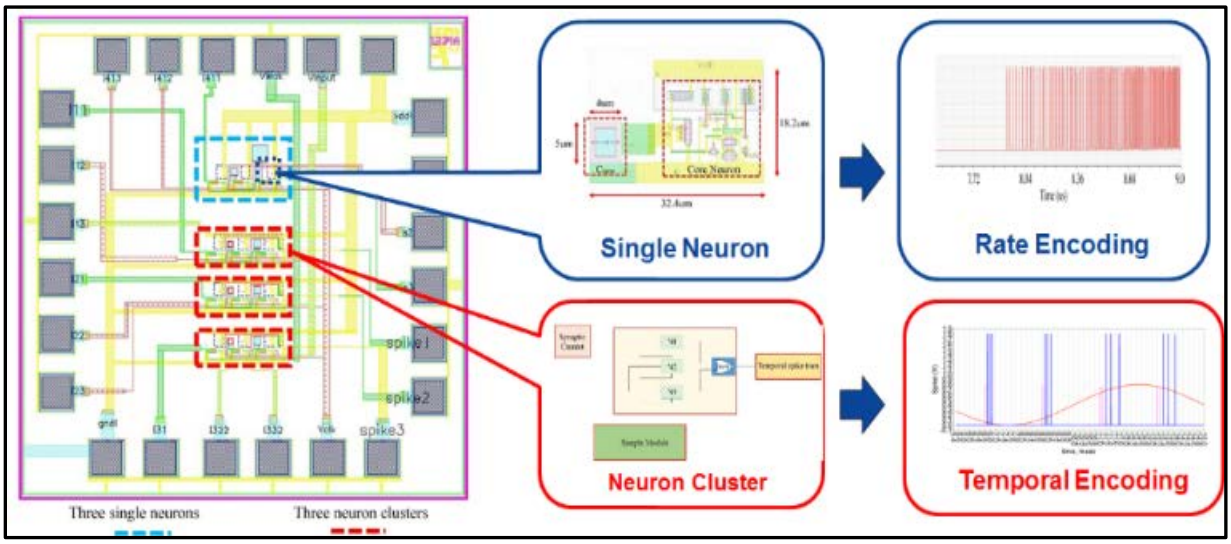


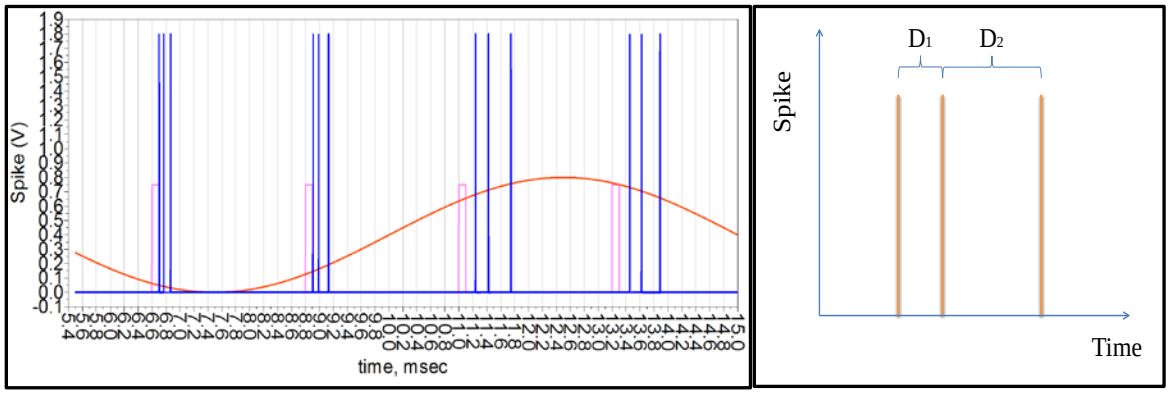
Figure 2. Optical Microscope Imagery of Encoder Chip and Testing PCB

This preliminary data provided a solid foundation for this effort. This was described in detail in our previous collaborative work with Air Force Research Laboratory (AFRL) in "Spike-Time-Dependent Encoding for Neuromorphic Processors" that was published in the *ACM Journal on Emerging Technologies in Computing Systems* [10]. This neuron circuit was the first chip to be fabricated that represented sensory data using inter-spike interval temporal encoding, in which information was encoded according to the timing of spikes with respect to each other.

The circuit that was designed and simulated for this effort was based on these previous findings. In Fig. 3a are presented high level depictions of the analog neuron design for both the rate and the temporal encoding. Depictions of simulated spike chains are presented in an enlarged view in Fig. 3b.



(a) Rate and Temporal Encoding Design



(b) Simulated Temporal and Rate Generated Spike Chains [10]

Figure 3. Analog Neuron

As shown in Fig. 3b, the spike time dependent encoder circuit will output two different inter-spike intervals within the generated spike trains. Research also focused on further reductions in the encoder circuit design area and lower power consumption. We also analyzed the relationship between the number of neurons and the number of inter-spike intervals.

Thrust 2: Nonlinear Node and Delay Lines

In Fig. 4 is shown our high level analog reservoir computing scheme with nonlinear nodes that included a delayed feedback loop. In this system the input sensory signals are encoded by a temporal encoder that is based on our previous work described in part in Thrust 1. The chaos-based nonlinear nodes were designed using chaotic circuits, while the delay feedback loop was designed with CMOS buffers, resistance circuits, and capacitance circuits. Appeltant [25] has shown that a recurrent network can be replaced by a single nonlinear node with delay by means of time-multiplexing.

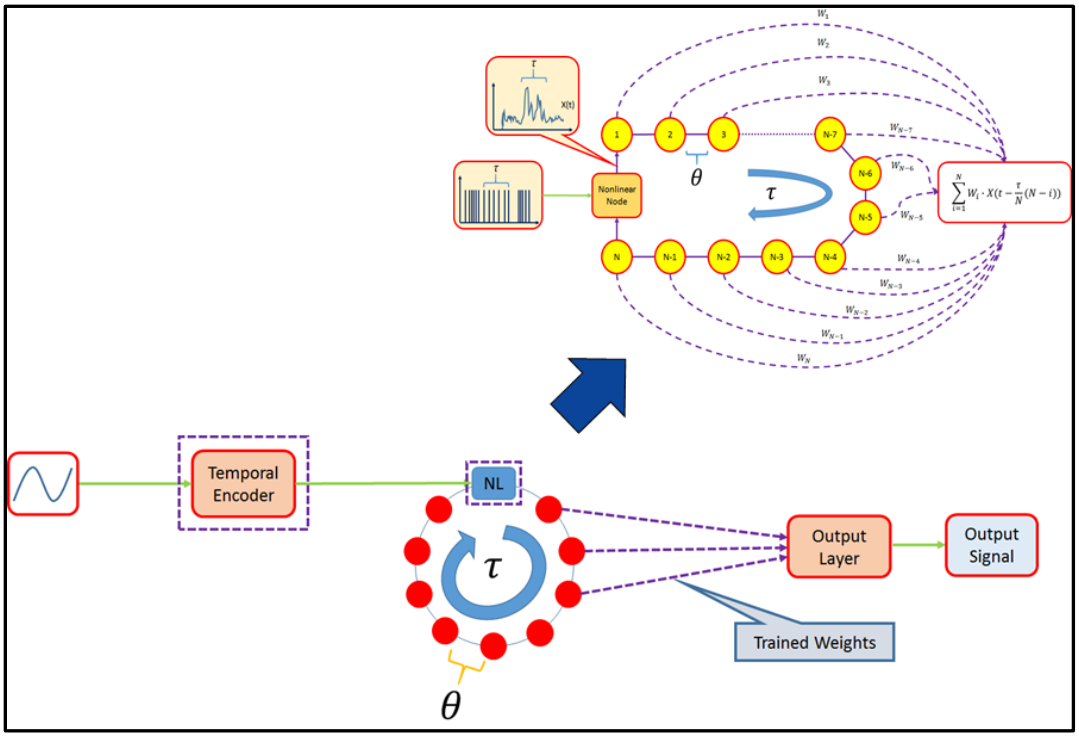


Figure 4. Delay-based Reservoir Computing System

Task 2.1: Nonlinear node design with chaotic circuits

Implementing a nonlinear node within a reservoir computing system presented a significant challenge. The chaotic system property of short period prediction was used to develop this novel reservoir structure. Both sigmoid and Mackey-Glass (MG) function circuits were investigated, and their respective performances were evaluated. A mathematical model was then developed that characterized this novel reservoir computing system design.

Task 2.2: Delay line design with CMOS transistors

Analog delay is an essential component that is required for processor-based dynamic reservoirs. Charge-coupled devices and switched-capacitor circuits are the classical methods for achieving this type of delay. Drawback to these discrete-time circuits are the necessity for clocking and aliasing effects. A continuous-time approach was attractive, particularly if the delay per section is to be controlled electronically.

Because of the temperature dependences of the timing constants in continuous-time circuits and the process-dependent values of monolithic components such as capacitors and transistors, it was necessary to develop additional circuitry to control the delay time. The designed delay-line section required that the modulus of the transfer function to be equal to unity over a broad frequency range. Also, to provide a frequency-independent group delay, the phase shift needed to be linearly dependent on the frequency.

Task 2.3: Fabrication of the designed nonlinear reservoir

The initial design was then fabricated in standard CMOS nano-technology.

Task 2.4: Initial testing of the fabricated nonlinear reservoir

Initial testing of the fabricated nonlinear reservoir was scheduled to begin in Month 22. This effort ended earlier than originally expected at Month 18[1], and so the fabricated circuits were not evaluated here. This research produced deliverables that included quarterly technical status reports for program reviews, this final technical report, as well as the following items:

1) Circuit design, fabrication, and measurement materials including SPICE (Simulation Program with Integrated Circuit Emphasis) models, circuit schematics, layout, prototypes, pre-layout and post layout simulation results, as well as testing data for both the time-dependent encoder and dynamic reservoir
2) An initial baseline study and a design trade-off report
3) Correlations of initial test results with the design report
4) Publications in archival international journals and refereed professional conferences
5) Fabricated chips that were delivered after the effort ended

[1] It should be noted that the fabricated prototype chip was delivered in October 2017 after this effort was completed, and will be subsequently tested and evaluated.

3.1 Temporal Encoders

The two types of temporal encoding are latency and Inter-Spike Interval (ISI) [10] coding are shown in Fig. 5. In latency encoding, the stimulus is encoded by the time difference of the first spike with respect to stimulus onset. In ISI encoding, the stimulus is encoded by the relative timing of spikes within an encoding window, i.e. time differences of successive spikes, as shown in the figure. ISI encoding responds to the relative time between spikes rather than the absolute time with respect to the stimulus onset. The benefits of an ISI encoding approach include the ability to rely on internal reference frames and the ability to make use of the correlations between spike times that cannot be modelled by rate modulations.

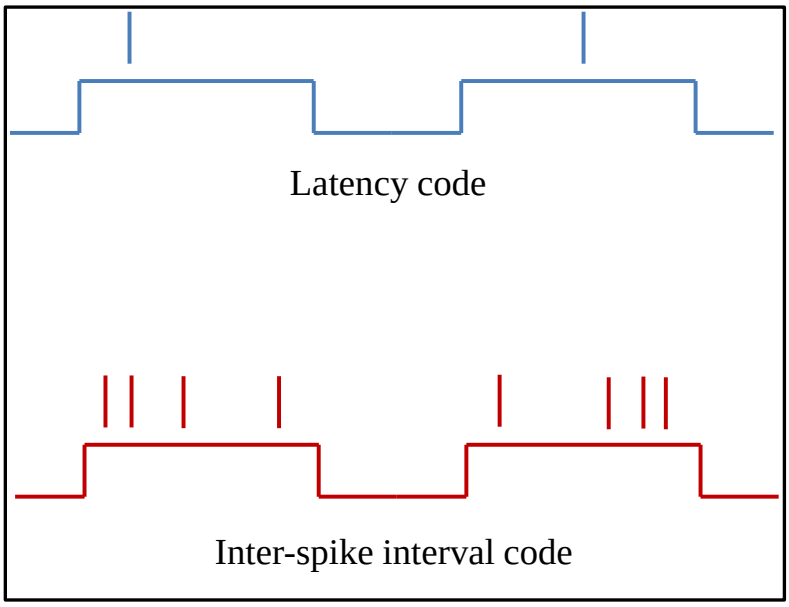


Figure 5. Latency and Inter-Spike Interval Temporal Codes

Temporal encoding implementations [15-17] have been introduced in software, and these implementations used Time-to-First-Spike (TTFS) latency encoding. We have previously developed the first temporal encoding scheme that is based upon an ISI hardware implementation [10]. Parallel structures were designed for the neural encoders, where the number of spikes was equal to the number of neurons. Here in this effort, due to an exponential correlation between the number of spikes and neurons, a larger number of intervals were utilized, which also allowed more information to be transmitted with the same number of neurons. When transforming sensory information by temporal encoding, parallel encoders process at higher speeds [14]. However, iteration encoders do provide more accurate results, which we have determined here a verification mechanism.

3.1.1 Analog vs. Digital Neurons

Since the neuron is the fundamental component in a neuromorphic system, power consumption and die size play significant roles in a neuron system design. In general, there are two implementations available: digital and analog. We compared analog and digital implementations of neurons using the LIF model and showed how signal-processing tasks can be modularly decomposed into elementary operators.

Digital implementation offers high computational precision, high reliability, and higher level programmability. Synaptic weights can be stored on or off chip. However, digital approaches also have disadvantages, since they require relatively larger circuit sized and require higher power consumptions compared to analog. Analog implementations exploit the physical characteristics of the hardware, which more closely mimics biological neurons. For example, operational amplifiers perform neuron-like functions, such as sigmoid transfer. Additionally, analog implementations afford greater flexibility for controlling the leakage current, thereby improving the model accuracy. With analog, computationally intensive calculations are performed automatically by physical processes, such as the summing of currents or charges. Finally, analog electronics are much more compact and offer high speed at low energy dissipation. However, analog implementations are more susceptible to noise, and exhibit higher sensitivity to process variables, which makes analog implementations more challenging to design.

Digital implementations and analog implementations [10] have been compared. Additional comparisons were made of power consumption, die areas, and transistor size. The results of these comparisons are summarized in Table 1, in which we normalized the data and summarized the comparison of power consumption between the analog and digital implementation.

Table 1. Normalized Analog and Digital Implementations

	Analog Implementation	Digital Implementation
Power consumption (μW)	1.15 - 52	14.3 - 100
Die area (μm^2)	74.2	1119.7
Transistor number	19.5	225

As shown in Table 1, an analog implementation requires lower power consumption, smaller die area, and less transistors when compared to a digital implementation.

3.1.2 Analog Neurons

We modeled and designed a robust and compact analog neuron that was fabricated with CMOS technology. The detailed neuron circuit schematic is shown in Fig. 6. Design aspects including integration time, threshold voltage, refractory period, and encoding resolution were examined to optimize the adaptability and reconfigurability of the neurons operational parameters. There were 11 functional modules in this circuit. The current reference module served as the current source, which provided constant current to the leak current module. The leak current module design was a current mirror with two transistors. Different leak currents were generated by adjusting the ratio of transistors M30 and M1 in the leak current module.

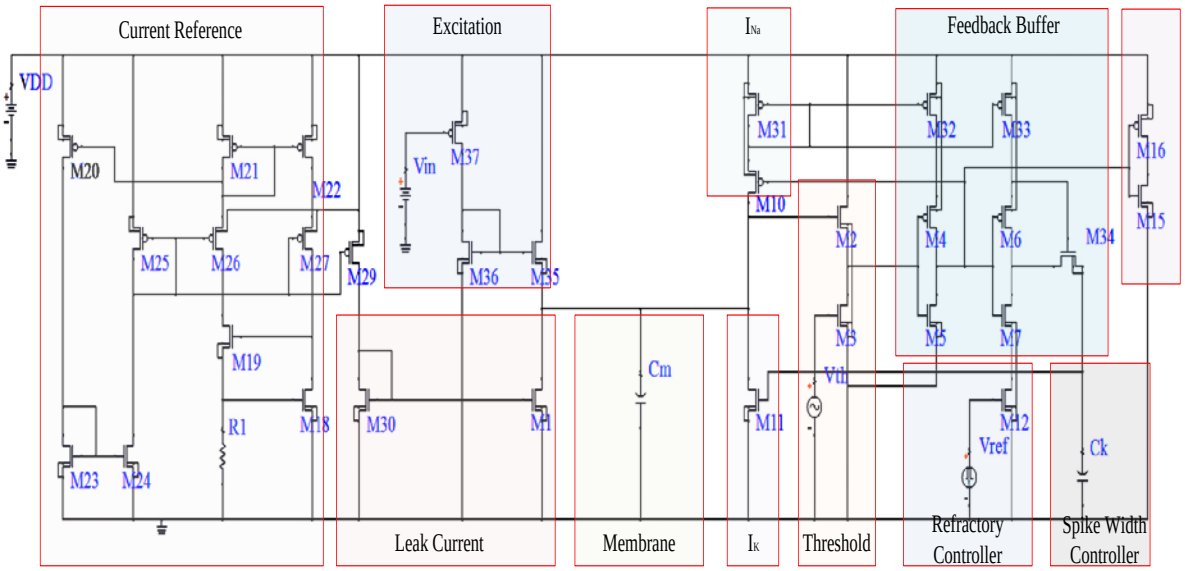


Figure 6. Neuron Circuit Schematic

The excitation module transformed the excitation voltage into current. The membrane module adopted one capacitor C_m, which modeled real biological membranes. The I_{Na} and I_K modules were applied to model charge-carrying ions Na^+ and $K^+, respectively$. A threshold module controlled the membrane threshold voltage. The feedback buffer module generated I_{Na}, I_K, and spikes. The refractory period was controlled by the refractory controller module. The spike width was tuned by adjusting capacitor C_K in the spike width controller module. Finally, the spike module generated the corresponding spike train.

The membrane capacitor C_m was charged when the excitation module was in operation. The membrane began to fire when the voltage on C_m reached the threshold voltage, which was controlled by the threshold module. The feedback buffer module, I_{Na} module, and I_K module

become operational after C_m reached the threshold voltage. Charge-carrying ion currents I_{Na} and I_K were generated, and at the same time, a spike train was generated and sent out through the spike module.

When the membrane voltage was lower than the threshold voltage, the output of the first inverter of the feedback module became high. This high voltage signal switched M10 off, which caused I_{Na} to drop to zero. At the same time, transistor M34 switched into the off region because of the low voltage output of second inverter, which caused I_K drop to zero since M11 was in an off region. At last, the leak current discharged the membrane capacitor C_m,which maintained the membrane voltage at the rest condition.

3.1.3 Temporal Encoder Structure

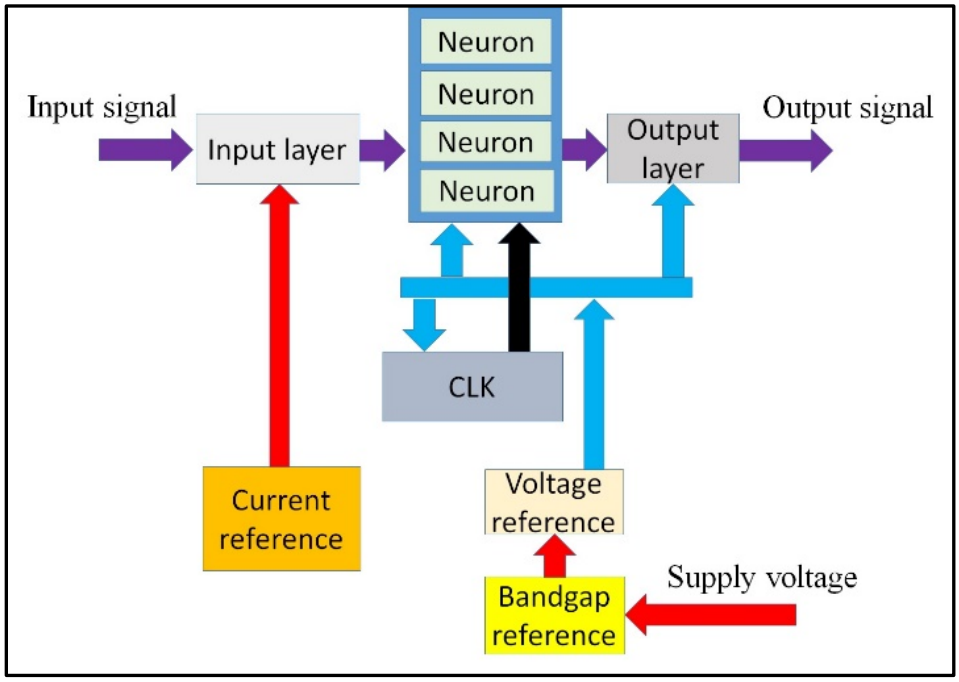


Figure 7. Structure of the Temporal Encoder

We developed an energy efficient temporal encoder which provided a general encoding scheme which could be implemented with different technologies. Low power consumption, small die area, and low cost are the three advantages of its design. In this project, the iteration encoder was designed, simulated and fabricated in standard CMOS process. The structure of the temporal encoder is shown in Fig. 7

The encoder adapted an iteration architecture in which each neuron worked on a separate clock period. The input layer served as the pre-processing unit. The iterative characteristic was achieved by the neuron pool and the output layer. As shown in the Fig. 7, there were 8 modules in this encoder, and these modules were divided into two categories. The core temporal encoder included the “input layer”, the “neuron pool”, and “output layer”. The second category was the function, which contained the “signal generator”, the “clock (CLK)”, the “voltage reference”, the “current reference”, and the “bandgap reference”..

When selecting the appropriate methodology to implement the input layer, we needed to consider the trade-off between area, power consumption, and accuracy. The resistor scheme possesses the simplest structure which is easy to implement, but, consumes the largest area which is not desirable. Although the operational amplifier scheme offers the highest accuracy, it possesses the highest power consumption. The single transistor scheme was chosen because of its adequate accuracy and moderate power consumption. A diode-structure NMOS transistor, which works in the triode region, was implemented to achieve the desired performance as the current tuner. The simplified circuit of the input layer is illustrated in Fig. 8. As shown in Fig. 8, the upper part of the input layer is a current mirror cluster. The output excitation currents, in1 to in4, would be sent to neuron pool directly. The bottom part is an input buffer for the input analog signal. Transistors, *N1* to *N4*, serve as the buffer which allow large value voltage signal.

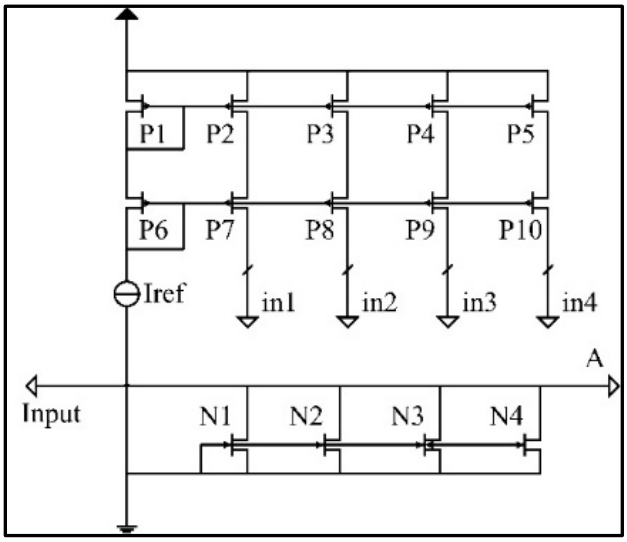


Figure 8. Simplified Input Layer Circuit

3.1.4 Neuron Pool

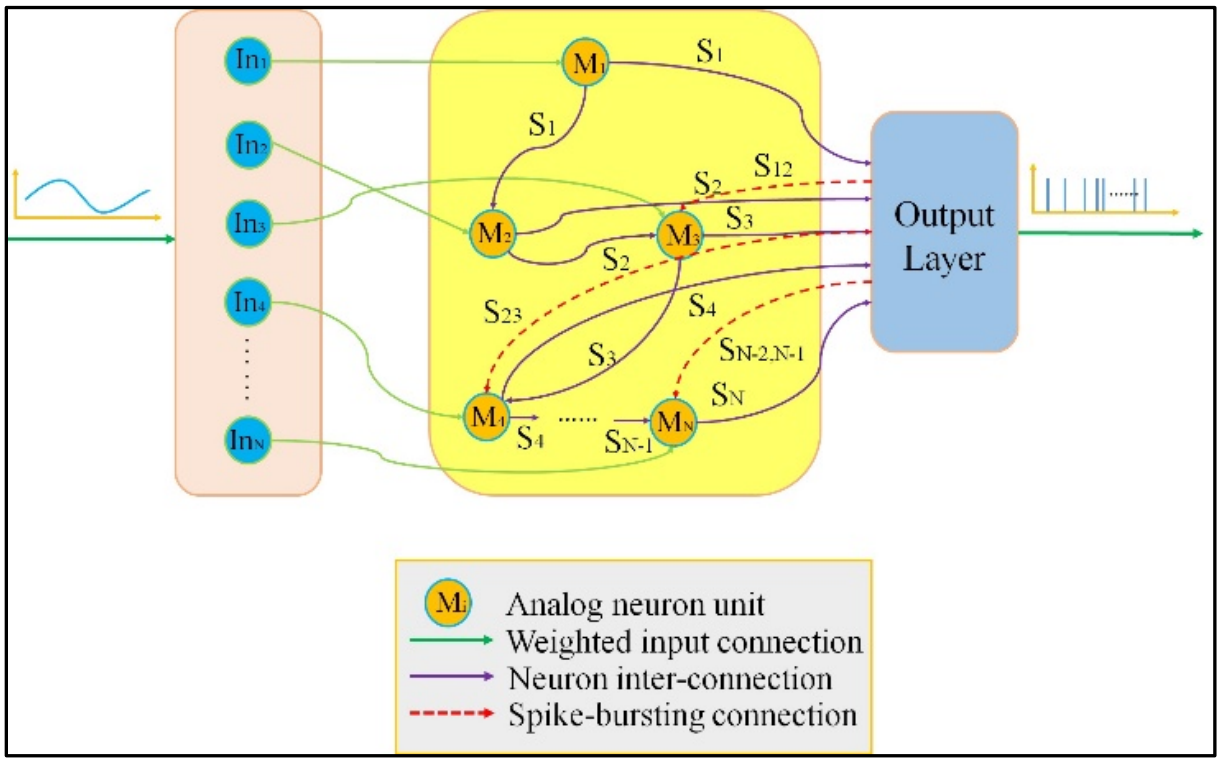


Figure 9. Neuron Pool Signal Flow Diagram

The second layer is the neuron pool, in which each neuron has its own priorities with different clock signals. In our spike time dependent temporal encoder, we assumed that every clock signal will generate only one spike with each neuron. A pool of neurons is required based upon our previously fabricated encoder to produce a burst of spikes. A general signal flow of a neuron pool is illustrated in Fig. 9. There are three kinds of signals existing in the neuron pool. In_i is the input signal that is constructed with excitation currents, and In_i is determined by V_i. S_i is the output signal of each neuron, and $S_{i,i+1}$ is the iteration signal which is generated by the output layer. The number of spikes of the temporal encoder [8] is directly proportional to the number of neurons.

The global clock CLK only controls the first neuron in the neuron cluster. When one CLK signal arrives, the first neuron will generate one spike to be sent to the second neuron and the first combiner, respectively. The output spike generated by the first neuron serves as the clock signal for the second neuron, and the output spike of the second neuron is sent to the first combiner. The output of the first combiner will be two spike signals, which are sent to the third neuron as its clock signal, and the second combiner.

An exponential relationship would greatly increase the number of spikes, which would increase the capability of containing more information with the same number of neurons than that of a linear proportional correlation. In this paper, an encoder was successfully designed to achieve the exponential relationship between the neuron number and spike number.

3.2 Encoder Scheme Analysis

Here we developed an energy efficient temporal encoder that provided a more general encoding scheme. Low power consumption, small die area, and low cost are the three advantages of its design. In this project, the iteration encoder was designed, simulated and fabricated in standard CMOS process. The temporal process is expressed as:

Neuron_1 generates S_1
for (i=2; i<N+1; i++)
{
Neuron_i<= S_i-1
Neuron_i generates S_i
Combiner_i<=[S_i-1 S_i]
Output<=Combiner_i
}

In this process, "<=" represented "sent to", and S_i represented the spike train. Thus, N neurons can generate 2^{N-1} spikes. Since the iterative structure was used, this encoder was able to produce more spikes than that of a temporal encoder which had the same number of neurons. Hence, this encoder design could carry more information. The exponential growth of the number of spikes greatly reduced the power consumptions since fewer neurons were needed.

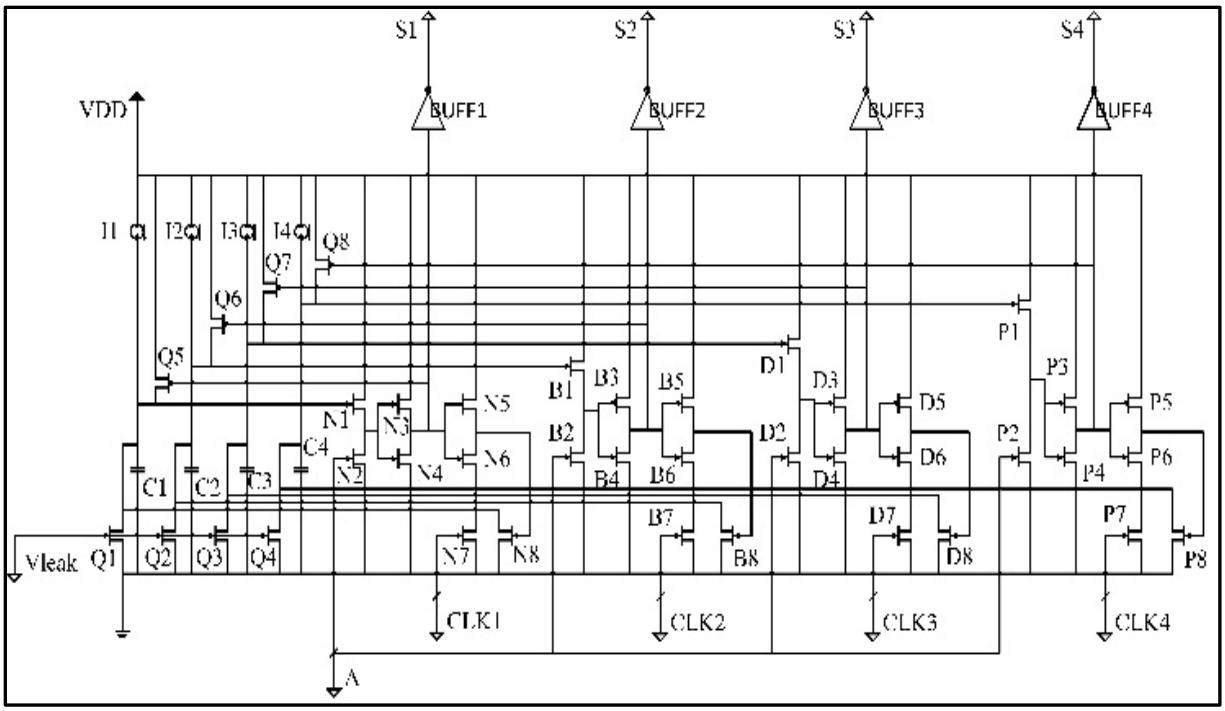


Figure 10. Simplified Neuron Pool of the Temporal Encoder

As shown in the Fig. 10, V_{leak} generates a leakage current. *CLK* represents the sampling clocks. *A* is the analog signal that is being sampled and S_i is the output signal. *CLK1* is the global sampling clock. *CLK2* to *CLK4* represents *S12* to *S34*, respectively. The excitation currents *I1* to *I4* are generated by the input layer. In this particular neuron pool, four neurons are being combined. The tags on each of the four neurons are *Ni*, *Bi*, *Di*, and *Pi*, respectively. The membrane capacitors are marked *C1* to *C4*, the leakage transistors are labeled as *Q1* to *Q4*, and the feedback transistors are labelled *Q5* to *Q8*. The numbering of these components corresponds to that of a complementary neuron.

When *I1* charges *C1*, V_{leak} forces *Q1* to the subthreshold region, which results in a constant leakage discharging current I_{in}. The voltage increases as the charging process continues, and NMOS transistor *N1* increases to the saturation region when the voltage on *C1* reaches the threshold voltage. After *N1* has transferred into the saturation region, the current between *N1* and *N2* increases. Taking the channel length effect into consideration, the current equation is expressed as Equation (1).

$$I_{ds} = K\left(V_{gs2} - V_{thn}\right)^2 (1 + \lambda V_{ds2}) \tag{1}$$

K is determined by physical parameters and the transistor dimensions, V_{gs2} is determined by the input signal in_1, V_{thn} is the threshold voltage of NMOS transistor, λ is channel length effect coefficient, which is determined by physical process, and V_{ds2} is the drain-source voltage of transistor *N2*. Since V_{gs2} is considered a constant in a short period of time, Equation (1) can be simplified to Equation (2).

$$I_{ds} = U(1 + \lambda V_{ds2}) \tag{2}$$

U is a constant value.

As shown in the Equation (2), V_{ds2} increase correspondingly when I_{ds} increases, and I_{ds} is controlled by the voltage on *C1*. Transistors *N3* and *N4* transform the high-level voltage V_{ds2} into low-level voltage. Then *CLK1* turns into an ON region. This behavior controls the gate voltage of transistor *N8*, which generates a discharge current for membrane capacitor *C1*. Such a discharging process results in *N1* becoming unsaturated, resulting in the voltage V_{ds2} dropping to a low level. Transistor *N8* has been designed as a wide NMOS transistor, which insures that the discharging process is rapid. After membrane capacitor *C1* is discharged, transistor *Q5*, *I1* and the leakage current force the membrane potential into the rest level. A spike will not be fire during the refractory period, which is controlled by the CLK1 signal. A membrane potential charging and discharging process is illustrated in Fig. 11.

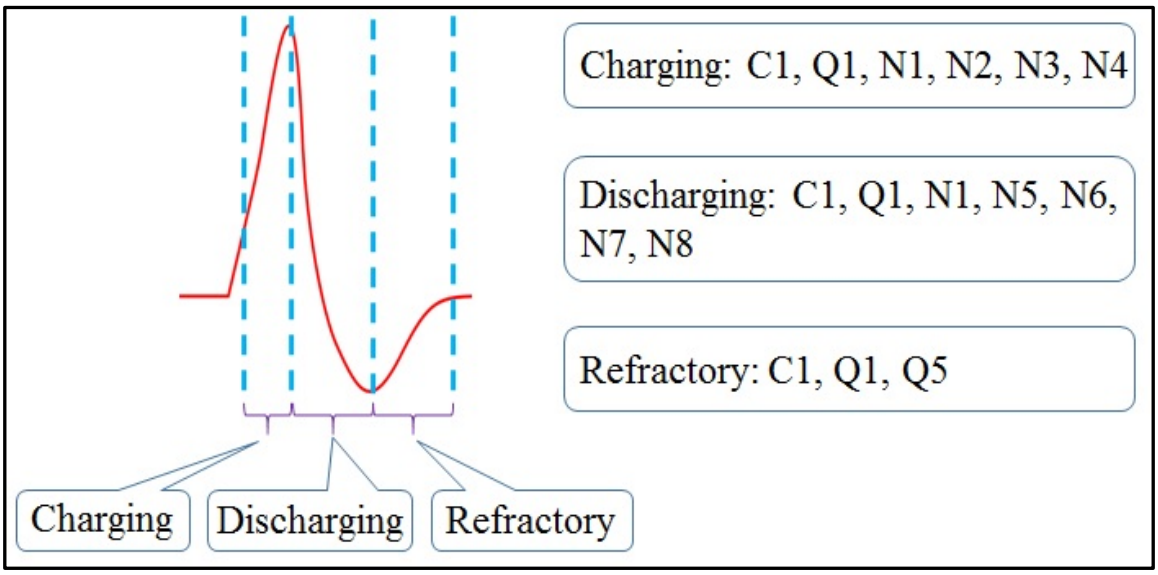


Figure 11. Membrane Potential Variation Diagram

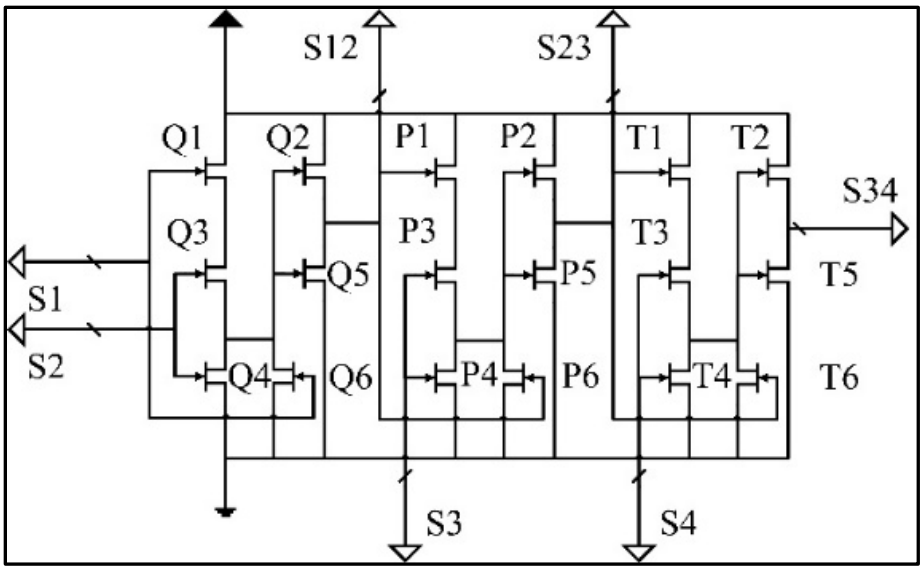


Figure 12. Simplified Output Layer Circuit

The simplified circuit of the corresponding output layer is illustrated in Fig. 12. The circuit is comprised of 4-neuron based temporal encoders and three combiners. The first combiner will be explained as to how it functions. The input spike signals are *S1* and *S2*. Only one spike can be accepted by the combiner at a time. When a spike pass through *S1, Q1* is in the cutoff region, and *Q6* is in the saturation region. Therefore, the voltage on the gates of transistor *Q2* and *Q5* is equal to the drain voltage of *Q6*, which is low voltage. In this situation, *Q5* is in the cutoff region, and *Q2* is in the saturation region. The final output from *Q2* will be at a high voltage level. In other words, this combiner accurately duplicates the input spike. The delay of each combiner is less than 10 ps, which ensures that a significant error would not be introduced in the combiner.

The encoding scheme was analyzed and demonstrated through a mathematic derivation. In the encoder, each neuron is affected by not only the input signal, but also the previous neuron's output signals, as shown in Equation (3).

$$S_n = f(S_1, S_2, \dots, S_{n-1};\ S_{n-2,n-1}; In_n) \tag{3}$$

f( ) represents the integrated scheme of these three types of signals, and the input signal is combined with both V_{thi} and I_{exi}. For a temporal encoding scheme, it is important to derive out the relationship between the inter-spike intervals and the original signal. In our design, the encoder samples the original signal without any additional systems, and encodes the input signal information of the inter-spike intervals, whereby each sampling cycle is accomplished by the neurons. Without generality, we define the relationship of the excitation current as Equation (4).

$$A_N = \beta A_{N-1} = \beta^2 A_{N-2} = \cdots = \beta^{N-1} A_1 \tag{4}$$

β is determined by the design parameters, and N is neuron amount. In our design, each neuron has the same parameters. Therefore, the firing behaviors of each neuron are determined by the excitation currents. As we have mentioned, each neuron is affected by not only the input signal, but also the previous neurons output signals. When the neuron number is determined, we can determine the relationships of each spike and the inter-spike intervals values. An output spike train with N=n, where n is the neuron number, is illustrated in Fig. 13.

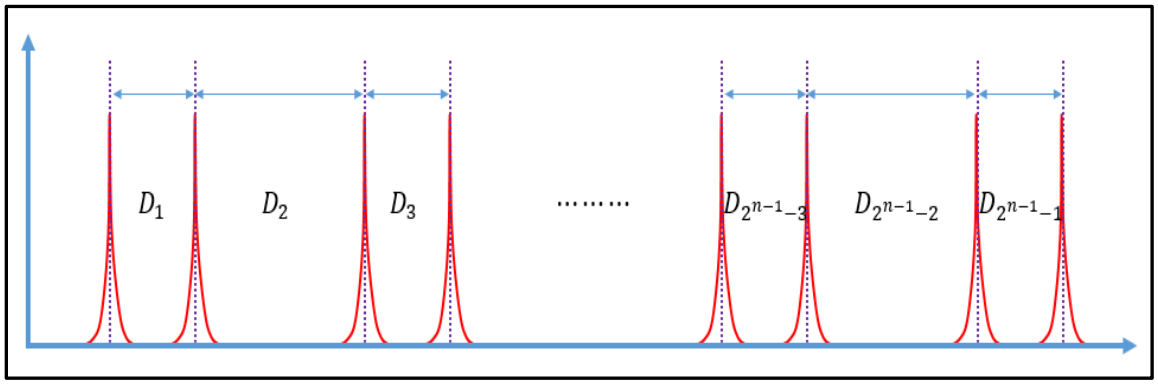


Figure 13. Output Spike Train

The inter-spike intervals in Fig. 5 is expressed in Equations (5) to (9), govern the behavior of the inter-spike intervals in a single temporal encoding train.

$$D_{2i-1} = D_{2^{n-1}-1} = \frac{a}{A_1} \cdot \frac{1}{\beta^{n-1}} \tag{5}$$

$$D_{2(2i-1)} = D_{2^{n-1}-2} = \frac{a}{A_1}\left(\frac{1}{\beta^{n-2}} - \frac{1}{\beta^{n-1}}\right) \tag{6}$$

$$D_{4(2i-1)} = D_{2^{n-1}-4} = \frac{a}{A_1}\left(\frac{1}{\beta^{n-3}} - \frac{1}{\beta^{n-2}} - \frac{1}{\beta^{n-1}}\right) \tag{7}$$

$$D_{8(2i-1)} = D_{2^{n-1}-8} = \frac{a}{A_1}\left(\frac{1}{\beta^{n-4}} - \frac{1}{\beta^{n-3}} - \frac{1}{\beta^{n-2}} - \frac{1}{\beta^{n-1}}\right) \tag{8}$$

$$\vdots$$

$$D_{2^{n-2}} = \frac{a}{A_1}\left(\frac{1}{\beta^1} - \frac{1}{\beta^2} - \frac{1}{\beta^3} - \cdots - \frac{1}{\beta^{n-2}} - \frac{1}{\beta^{n-1}}\right) \tag{9}$$

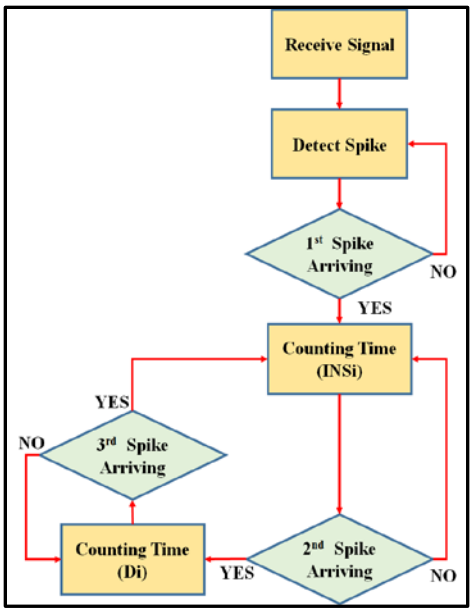


Figure 14. Verification Scheme Flow Chart

We developed verification (Fig. 14) and recovery (Fig. 15) schemes to improve the designed temporal encoder's error tolerance. When considering internal verification, the verification value serves as the criteria to inspect the working conditions of each neuron. In our design, each neuron generates a spike train that will be transferred to the neuron in the next stage, and to a verification combiner [32, 33].

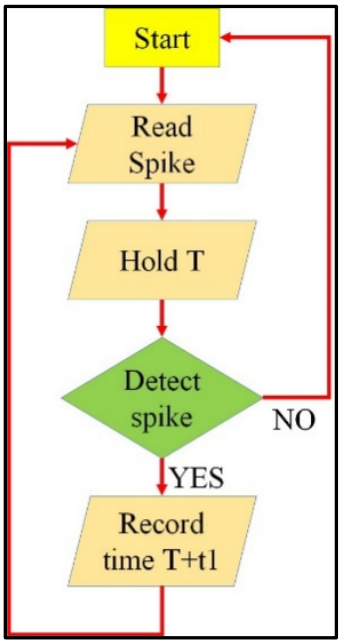


Figure 15. Temporal Spike Train Recovery Scheme

For correct temporal encode outputs, an algorithm was adopted to evaluate the spike trains, which is shown in Equation (10).

$$\forall\, i,j \in [1,N], \begin{cases} INS_i = INS_j \\ e_i = |INS_i - \dfrac{\sum_{i=1}^{N} INS_i}{N}| \\ S_i = D_i \pm 2e_i \end{cases} \tag{10}$$

N is the total account of intervals, e_i is the error of the i^{th} interval, and S_i is the final temporal encode at position i.

The flowchart of the proposed recovery scheme is presented in Fig. 15. T is defined as the minimum time slot. After the first spike is detected, the read system holds for time T. Then the recovery scheme will detect the time of the next spike. A maximum time slot T_m is defined. If a spike has been detected during T_m, the time t_1 would be recorded together with minimum time slot T. If a spike was not detected after maximum time T_m, a new read cycle would start. Additional information on this scheme is available [32].

3.3 Analog Implementation of the Delayed Feedback Reservoir

Applications of reservoir computing include chaotic dynamics predictions [34], character recognition [35, 36], speech recognition [37, 38], and the generation and prediction of chaotic time series [39]. Delay is ubiquitous to most systems, especially biological systems. Examples include the diffusion and transport of substances such as cellular metabolites, hormones, oxygen and carbon dioxide in blood, the conduction time of nerves, and the intrinsic times for synthesis, growth, and reproduction [40]. With delay embedded in these systems, they can exhibit rich dynamical behavior.

To more closely mimic mammalian brains, delay was implemented in our Delayed Feedback Reservoir (DFR) computing model. In this context reservoirs function as time-delayed recursive networks that use feedback as a short-term dynamic memory for the processing of time-series input signals. Delay systems exhibit the two prerequisites that are required for reservoir computing, high dimensionality and short-term memory. An electronic analog circuit [41] which simulated the Mackey-Glass mathematical equation and exhibited highly complex hyper-chaotic behavior was strongly dependent on the delay time.

Photonic implementations of delayed feedback systems [42-44] introduced the phenomenon of optical chaos. Complex dynamics [45] is beneficial for describing some applications, including some chronic and acute diseases. However, there is no known analog integrated circuit (IC) design in the literature for a spike time-dependent delayed feedback reservoir computing system.

One of the simplest delay systems consists of a single nonlinear node whose dynamics is influenced by its own output at time τ in the past. Such a system is simple to implement, since it is comprised of only two elements, a nonlinear node and a delay line loop. The delay line loop transverses through a number of virtual nodes. Each virtual node is separated by an equidistant delay θ. Each virtual node holds the delayed version of the previous node's output in time $\theta = \frac{\tau}{N}$, where N represents the number of virtual nodes.

The dynamic characteristics of this system can be influenced by simply changing the feedback strength or the delay interval θ and τ. A delay feedback reservoir has an approximately identical performance to a traditional reservoir computer. The output nodes are linear weighted sums of the tapped states in the delay line [25], given by Equation (11).

$$\hat{y}(t) = \sum_{i=1}^{N} w_i . x\left(t - \frac{\tau}{N}(N - i)\right) \tag{11}$$

In our design for the delayed feedback reservoir computing architecture, the input sensory information will be inputted into a temporal encoder, and then transformed to the temporal spike trains. To compensate for the loss of parallelism, signals proceed through a masking procedure before being injected into a nonlinear neural node and delay loop. We have adapted Mackey-Glass functions to design the nonlinear neural node and delay loop. Fig. 16 illustrates the architecture of our delayed feedback reservoir computing system design, which is described in additional detail in Li [46].

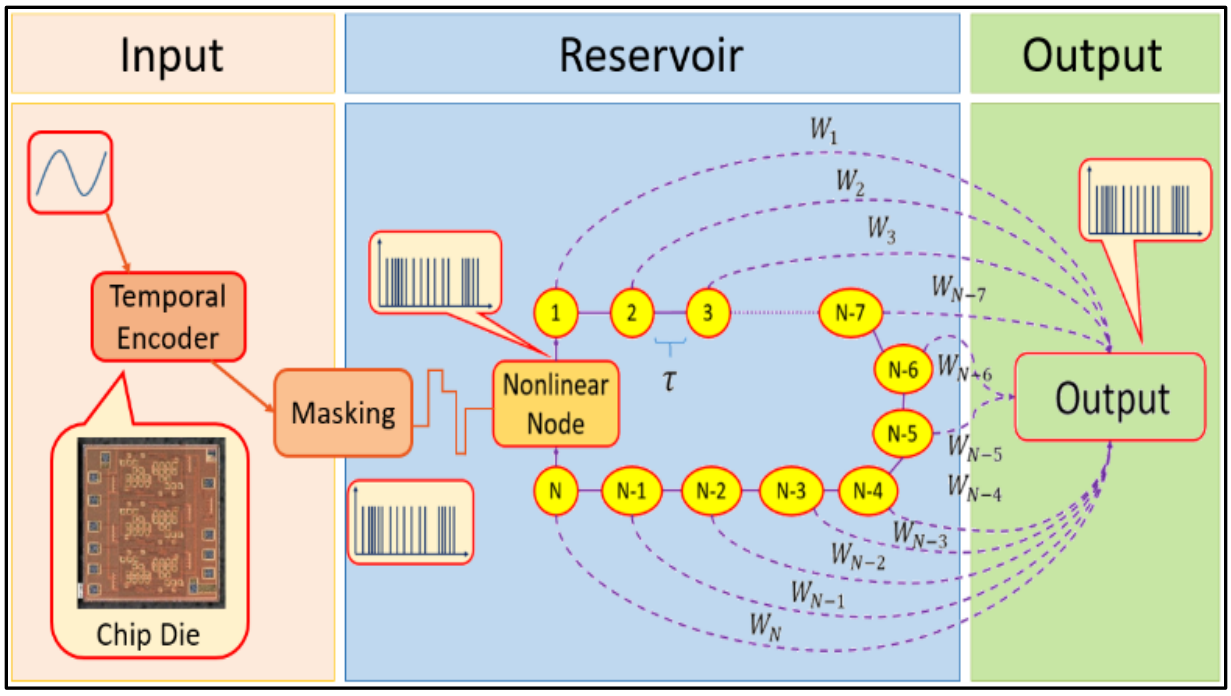


Figure 16. Design of the Delay Feedback Reservoir Computing System

The input and output relationships are mathematically described by transfer functions [47]. Step functions were one of the earliest transfer functions that expressed input-output correlations. However, for reservoir computing, nonlinear mapping of the input is required. Hence, in order to achieve such functionality, nonlinear transfer functions are employed.

Sigmoid and hyperbolic tangent functions are commonly used as transfer functions. The slope of a sigmoid function is tunable by varying its coefficients. As the slope becomes steeper, the shape of sigmoid function becomes more like a step function. However, different from a step function, a sigmoid function is a continuous function that ranges from 0 to 1, and is composed of a mixture of linear and nonlinear behaviors. Another nonlinear function that can be used is a hyperbolic tangent function tanh. Hyperbolic tangent functions are antisymmetric with respect to the origin, and converge more rapidly than the sigmoid activation functions in the neuron model.

Sigmoid and hyperbolic tangent functions are not the only nonlinear functions that are utilized as transfer functions, and other nonlinear functions that could serve as transfer functions for reservoir computing were explored. Systems were evaluated that had looped structures in which the output is fed back after a time delay τ such that the feedback influenced subsequent outputs. Such systems are described with Delay Differential Equations (DDEs). These equations are also model such things as the control of pupil light reflexes, gene regulation and population growth.

For example, in the pupil light reflex, only 20 *ms* of the 300 *ms* delay can be accounted for by the axonal conduction times [40]. A DDE equation is expressed in Equation (12).

$$\frac{dy}{dt} = f[y(t), y(t-\tau)] \tag{12}$$

f is arbitrary function, and can be either a linear or a nonlinear function depending on the application, and τ is the time delay.

The Mackey-Glass (MG) function, a type of DDE, was initially developed to describe diseases that exhibit symptom with oscillatory instabilities [45]. The dynamics of a MG function depends on both the current and the previous time, which is expressed in the following form (Equation (13)).

$$\frac{dP}{dt} = \frac{\alpha P_\tau}{1 + \beta P_\tau^n} - P \tag{13}$$

α and β are arbitrary design parameters, n is the nonlinearity exponent, and Pτ = P(t-τ). By varying the nonlinearity exponent, the nonlinearity can be tuned. Fig. 17 illustrates the functional block diagram of a MG function.

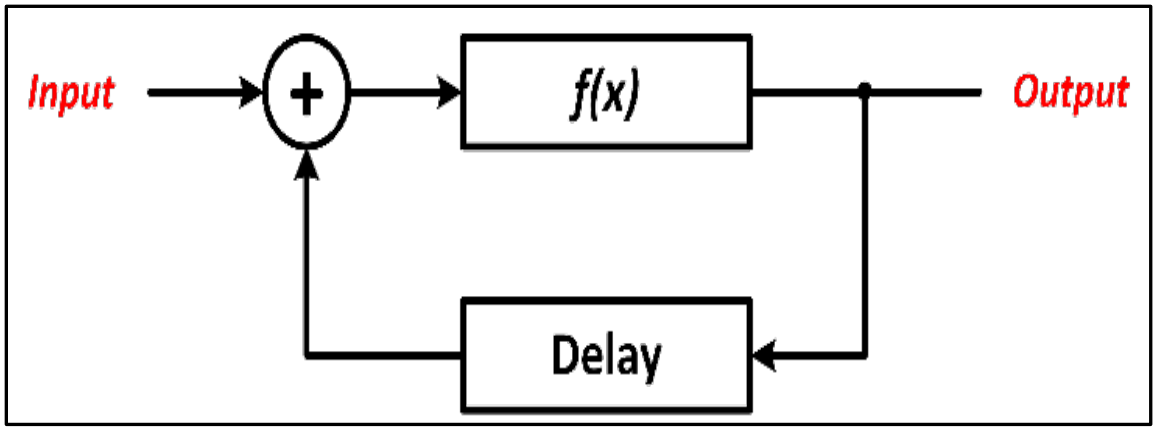
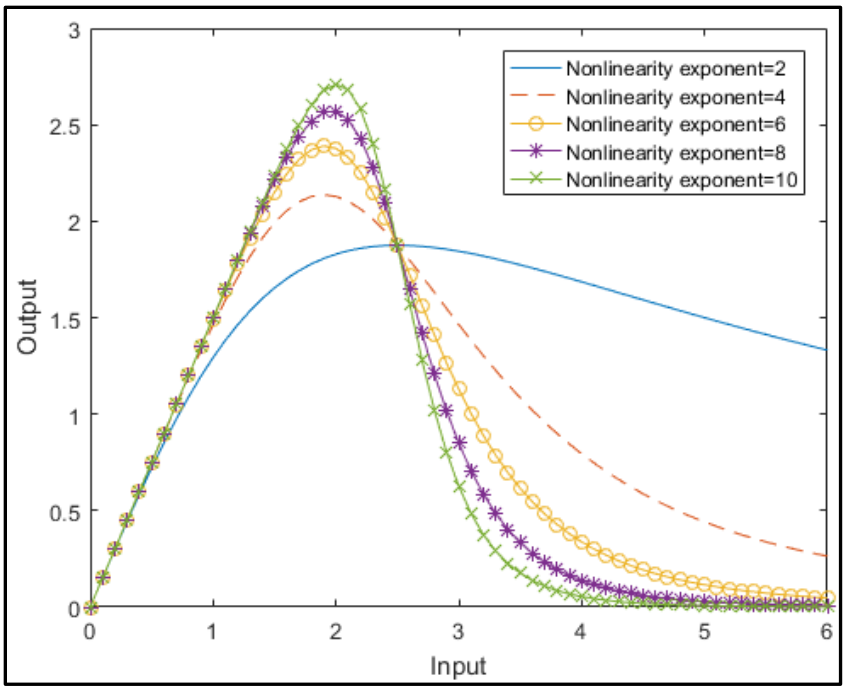
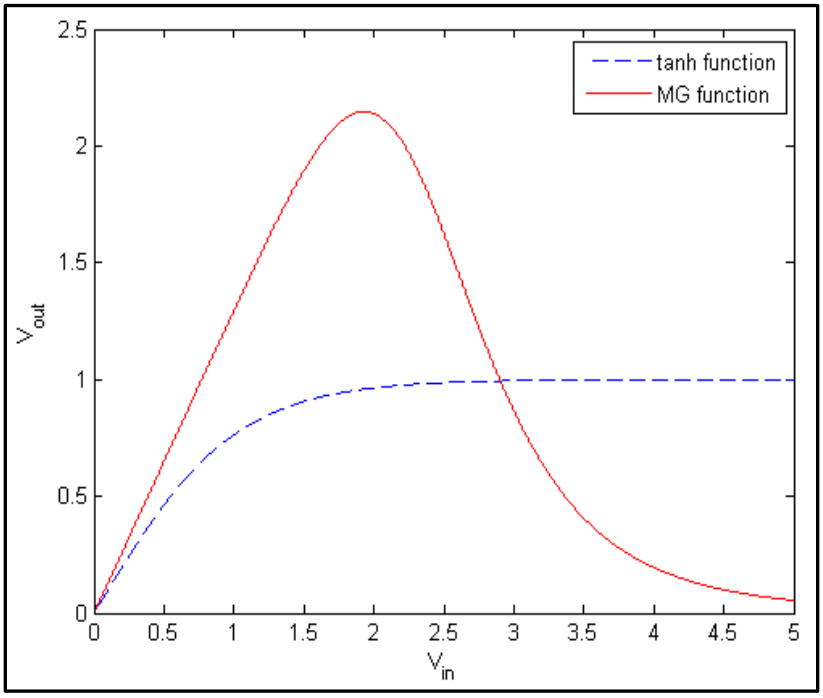


Figure 17. Functional Block Diagram of a MG Function



(a) vs. Nonlinearity Exponents



(b) vs. Hyperbolic Tangent Function

Figure 18. Mackey-Glass Function

The introduction of time delay enables MG functions to respond as time-delayed feedback structures in a way that mimics the biophysical processes in the biological neuron. In biological terms, delays between two brain neurons can arise from feedback mechanisms. This introduced delay can either stabilize or destabilize a dynamical system, which may transform dynamical behavior from stable to chaotic, or vice versa. The best computational performance occurs at the transition region between stable and chaotic regimes, which is called the "edge of chaos" [47, 48].

Neither of the two most commonly used transfer functions, the sigmoid and the hyperbolic tangent, operate at the edge of chaos. The MG function includes embedded delay in its equation, which enables it to operate at the edge. This function not only allows the nonlinear mapping that is required in reservoir computing, but it also provides a delay loop. As shown in Fig. 18 (a), the shape of Mackey-Glass function changes with the value of the nonlinearity exponent. This property of Mackey-Glass function enables the determination of an optimal regime [47]. A hyperbolic tangent and a MG function are plotted in Fig. 18 (b). Compared to the hyperbolic tangent function, the MG function exhibits higher nonlinearity.

Several research activities on the implementation of the MG model by electronic circuits were discovered in the literature [25, 41, 42, 45]. These results showed its capability of generating high-dimensional data and the potential for tuning its dynamical behavior by varying the time delay. However, all of these works focused on the time-continuous waveform of the input signals. There has been no known previous research regarding the use of analog neuron spike trains as the input signals.

A nonlinear function was modeled [50] with three fabricated analog multipliers (AD633) [51] and multiple operational amplifiers (op-amp) (AD712) [52] from Analog Devices. Moreover, as demonstrated by Namajunas [41] and Soriano [42], nonlinear functions were formed by n-type and p-type junction gate field-effect transistors (JFETs). Due to its simple structure, this electronic circuit design has been widely used to model nonlinear functions. However, the output signals from this nonlinear device are relatively small, and operational amplifiers are generally required to further increase the outgoing signals to a desired level.

Although this demonstrates the capability of nonlinear mapping with the low power CMOS technology, the output signals of these electronic circuit are usually limited to several hundred nanovolts. An op-amp with a finite gain of 10^9 is normally required to obtain sufficient output levels.

A combination of analog and digital implementation has been used to implement the MG function with electronic circuits [25]. These components have high power and large die area requirements. In our design both the spike-based nonlinear neural node and the delay loop are capable of directly processing signals in forms of spikes. We adopted neuron spike trains as the input signal, and were able to process the spike signal directly with an analog spike-based nonlinear processor.

Our nonlinear node design is comprised of input triggers (M_1 and M_2), a first order passive low-pass filter (R_1 and C_1), a nonlinear mapping transformer (R_2 and C_2), a feedback current mirror ($M_3 \sim M_5$), and an output current mirror ($M_6 \sim M_8$), as illustrated in Fig. 19.

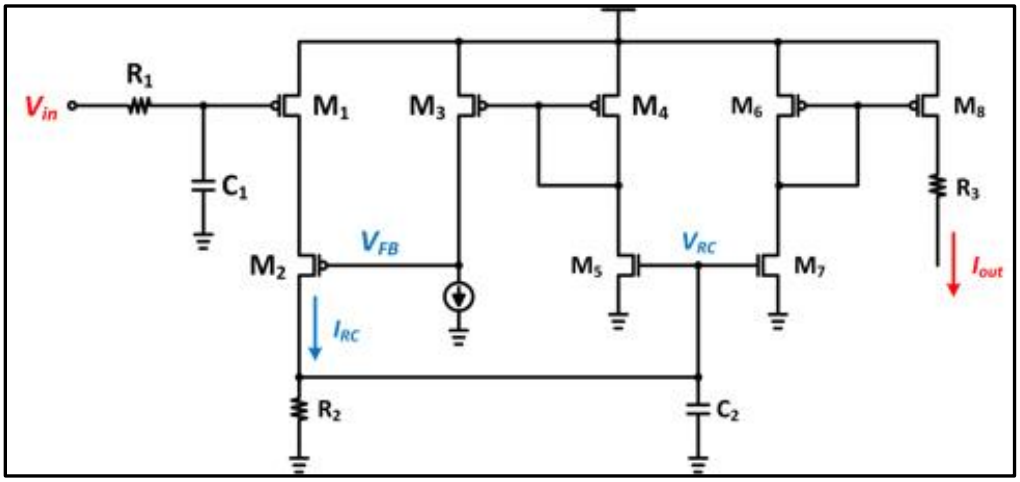


Figure 19. Simplified Schematic of the Designed Spike-based Nonlinear Node

In the reset operation, the input is charged to V_{DD}, which deactivates the input trigger of the nonlinear device, resulting in the discharge the nonlinear mapping transformer. Thus, the output current I_{out} is reset to 0 A. In the decision-making operation, the inverted spike-based input signal is first filtered by a passive low-pass filter. Once the input trigger is enabled, the input current I_{RC} charges the nonlinear mapping transformer to regulate the biasing voltage V_{RC} of M_7. Consequently, the drain-to-source voltage V_{DS} of M_7 quickly increases until it reaches its saturation level potential. While M_7 is in its sub-threshold region ($V_{RC} < V_{th7}$, where V_{th7} is the threshold voltage of M_7), the V_{DS} of M_7 is 0 V. The diode-connected structure of M_6 fully enables the output current mirror to achieve the maximum output current. Contrarily, as V_{DS} of M_7 reaches its saturation level potential, the transistor M_6 drops into the sub-threshold region, resulting in a decrease in the output current. Meanwhile, the feedback current mirror generates a high voltage at V_{FB} to disenable the input trigger. The positive feedback loop quickly reduces the output current

to 0 A, where it remains until the next process takes place. The nonlinear transformation of one input spike is completed.

The nonlinear transformation is achieved by the charging and discharging processes of the nonlinear mapping transformer. In other words, the nonlinearity of the transfer function is proportional to the time constant τ_{ND} of R_2 and C_2. To facilitate high-dimensional mapping, a large τ_{ND} is required to maintain the nonlinearity of the transfer function. However, a large τ_{ND} will results in longer charging and discharging times, which would ultimately reduce future computational efficiencies. Considering the tradeoff between accuracy and operating speed, a larger τ_{ND} was implemented to maintain the accuracy of the system.

Moreover, the *IV* characteristics of the output current mirror were utilized for nonlinear transformation in the nonlinear node design. The output current mirror was optimized such that it operated between the sub-threshold and the saturation regions, resulting in the maximum nonlinearity of the circuit.

In the delay feedback reservoir design, not only the nonlinear node but also the delay loop was designed such that it was capable of processing spike-based neuron signals. The IF-based delay unit utilized capacitor-sensing methodology based on a traditional IF neuron model, as depicted in Fig. 20. However, this is different from an IF neuron model, in that the input (excitation) current (I_{ex}) in the IF-based delay unit was a controllable current source that regulated the delay time of the circuit. During its operation the membrane capacitor C_m senses the excitation current and continues to increase its voltage potential.

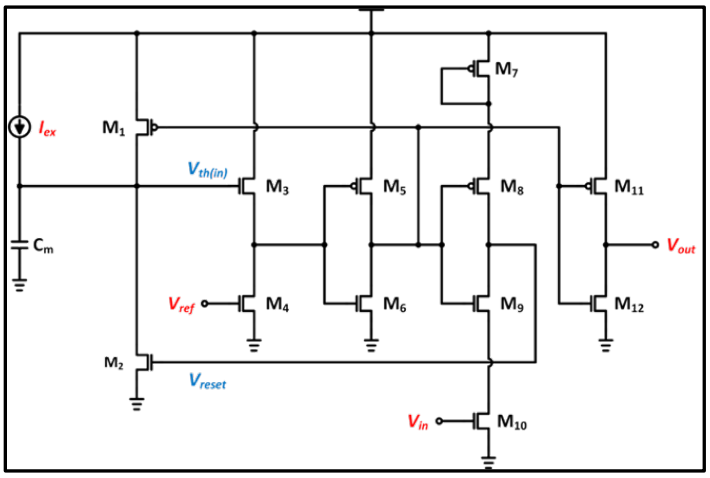


Figure 20. Simplified Schematic of the IF-based Delay Unit

When the voltage potential across the membrane capacitor *Cm* exceeds the threshold level of the input transistors (M_3 and M_4) in the Designed Spike-based Nonlinear Node, two cascading inverters ($M_5 \sim M_6$ & $M_{11} \sim M_{12}$) fire an output spike. Once the firing process takes place, a feedback loop ($M_7 \sim M_{10}$) generates a high voltage at V_{reset} to trigger the reset transistor (M_2) to discharge the membrane capacitor. As such, the firing process for one output spike is completed.

In the IF-based delay unit (Fig. 20) the delay time is regulated by the integrating time of C_m. The delay time constant τ_{delay} can be expressed as Equation (14).

$$\tau_{delay} = C_m \cdot \frac{V_{th(in)}}{I_{ex}} \tag{14}$$

$V_{th(in)}$ is the threshold voltage of the input stage and I_{ex} is the controllable excitation current. The mathematical analysis of the IF-based delay unit is similar to a traditional RC-based delay unit, since the input impedance (R_{in}) of the input stage is equivalent to $\frac{V_{th(in)}}{I_{ex}}$. Thus, the delay time constant can be rewritten as Equation (15).

$$\tau_{delay} = C_m \cdot R_{in} \tag{15}$$

However, unlike a traditional RC-based delay unit that is formed by a large capacitor, the delay time of our IF-based delay unit is regulated by the input impedance. Consequently, a larger delay time is achieved by increasing the equivalent input impedance of the delay unit, which is achieved by reducing the controllable excitation current.

The DFR for this effort was designed using a temporal encoder, nonlinear nodes and IF-based delay units. In this operation, the analog input signal is first encoded into a spike-based temporal signal by the temporal encoder, which had previously been fabricated [21], followed by the mapping process with the nonlinear node. The transformed nonlinear signal is then injected into the delay loop. Within the delay loop, the total delay time *T* is separated by N equidistant IF-based delay units, which can be expressed as Equation (16).

$$T = N \cdot \tau_{delay} \tag{16}$$

To enable the dynamic delay loop, the transformed nonlinear signal is first converted into a spike-based signal by the LIF neuron. The information of the nonlinear signal is presented as a spike train. The delay loop is constructed by connecting the output of the IF-based delay unit as a

clock trigging signal for the following stage. When a spike is generated from the LIF neuron, it resets the following delay unit. Meanwhile, the membrane capacitor of the corresponding delay unit begins to charge. Over the period of the time delay τ_{delay} the delay unit fires an output spike, which results in a nonlinear signal at the given delay time. The spike-based signal travels along the delay line until it reaches the last delay unit.

To maintain the accuracy of the signal combination, the delayed spike signal in the last stage of the delay loop is transferred into a nonlinear current, followed by a current gain adjuster to reduce the amplitude of the delayed nonlinear signal, such that the input signal will be dominant. The amplified-and-delayed nonlinear signal will then be merged with the input signal by the combiner.

3.4 Fabrication of the Reservoir Node Design

The spike-based DFR design was submitted for fabrication through the MOSIS Integrated Circuit Fabrication Service using the standard Global Foundries 130 nm CMOS technology on May-22, 2017. The chip design contained 13 DFR modules, and was separated into two sections. Each section can be biased and measured with 2 separated groups of I/O pins. Moreover, since device mismatch can occur as a result of the fabrication process, 4 different DFR design floor plans were implemented in the chip. To reduce the number of the output testing pins within the limited chip area, the outputs from all of the DFRs within the same section are shared on 1 output pin. To prevent interference between each of the output signals, a 6-to-1 multiplex was implemented, which allowed the selection of a specific DFR module. In addition to the DFR modules, an individual MG nonlinear node, and a temporal encoder are also included in this chip for individual performance testing.

Fig. 21-23 show the layouts of our spike-based DFR, our reservoir neuron node, and our delay neuron, respectively. This effort ended before the design was completely fabricated.

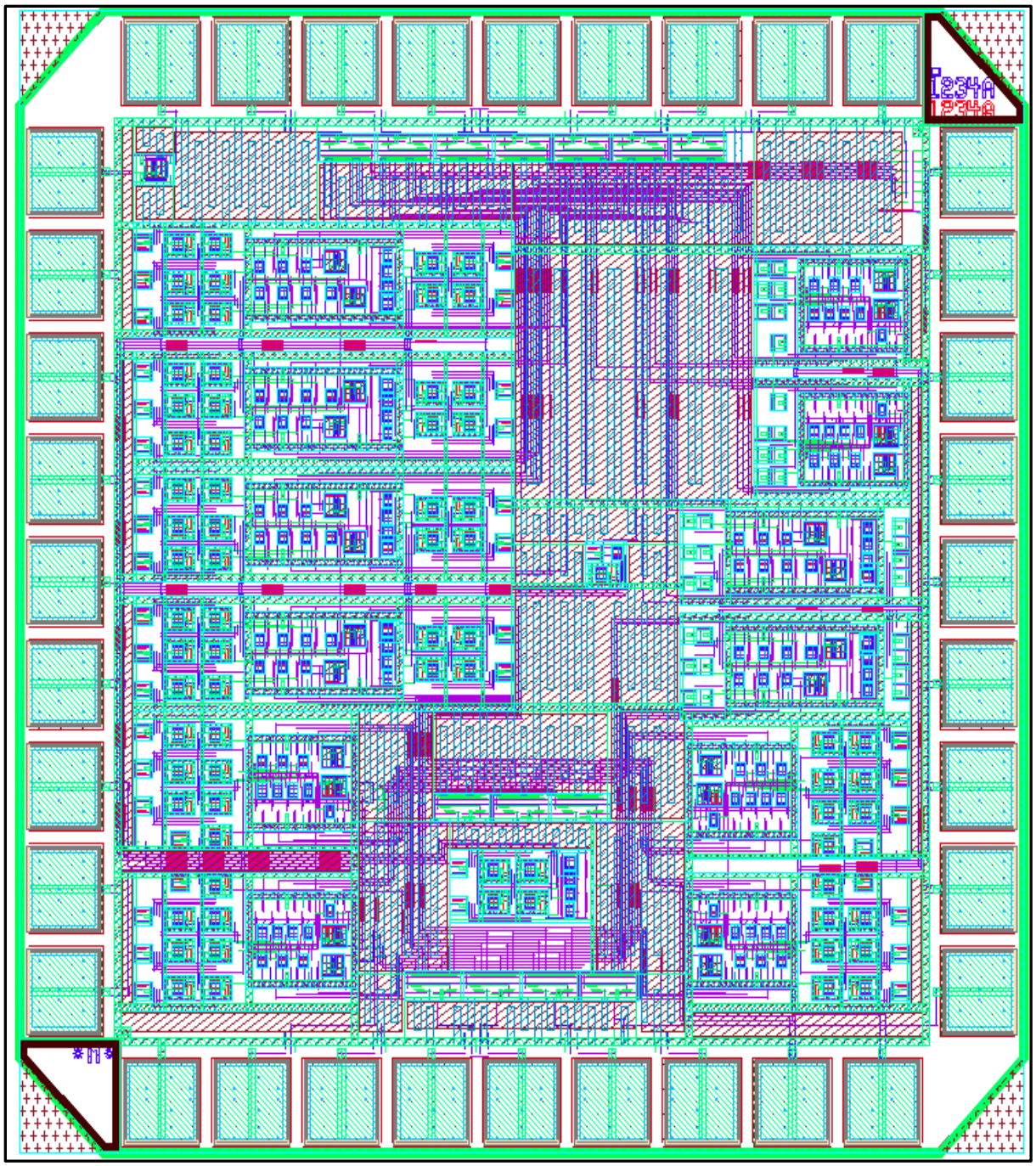


Figure 21. Layout of Our Spike-based DFR

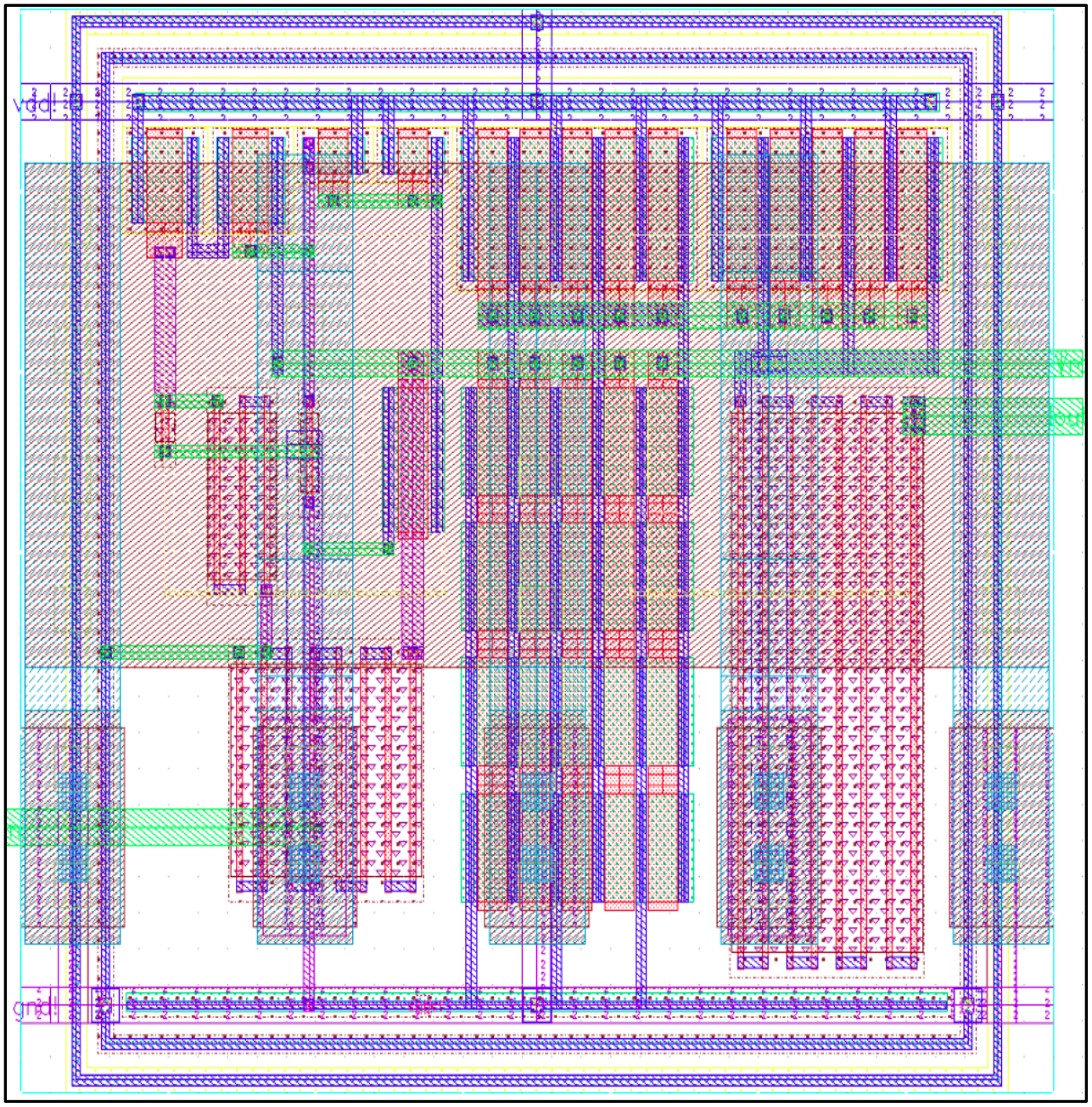


Figure 22. Layout of Our Reservoir Neuron Node

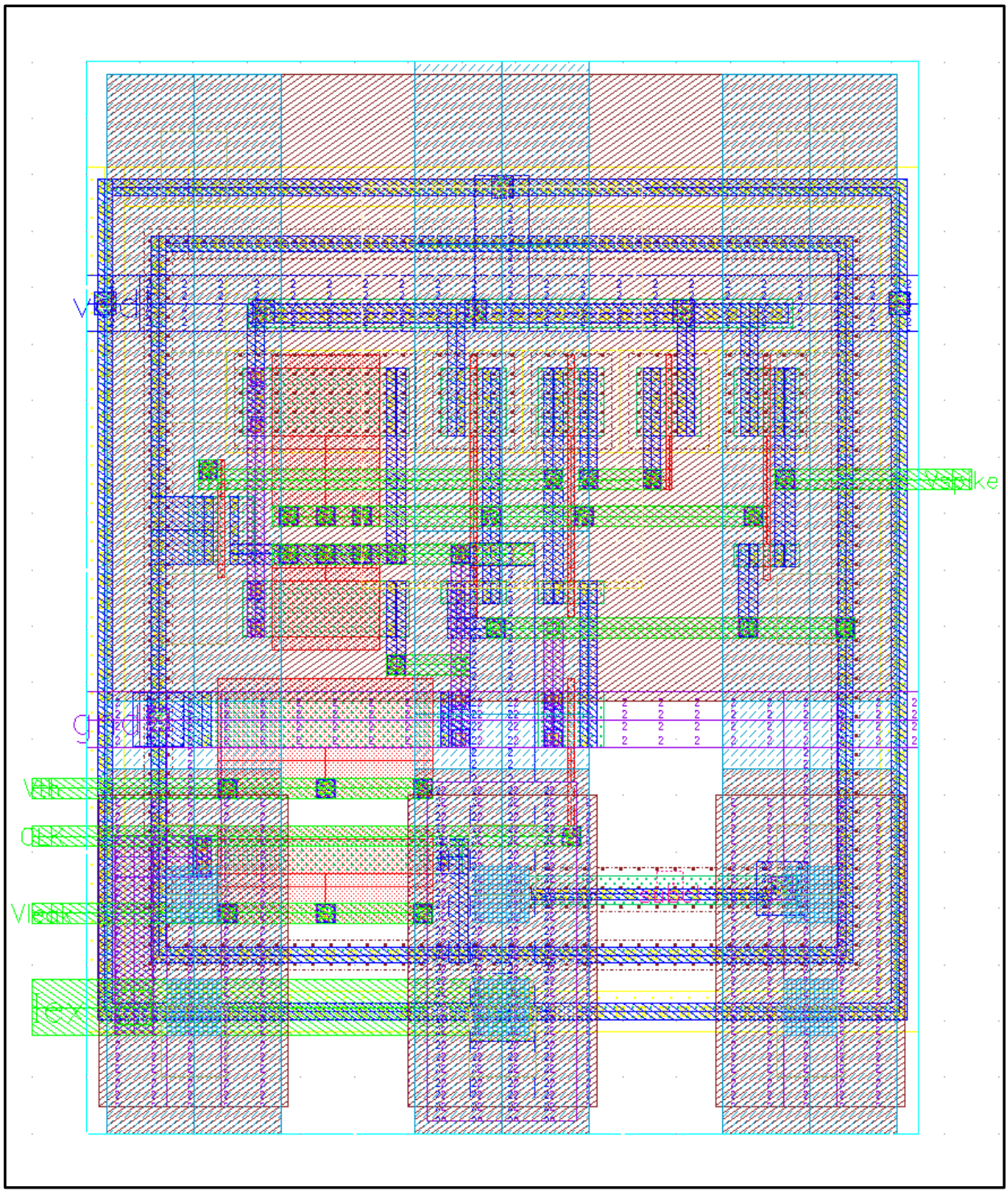


Figure 23. Layout of Our Delay Neuron

4.0 RESULTS AND DISCUSSION

In Section 4.0 is presented the results and a discussion of this effort. These results are from simulations of our circuit designs. The chip design of the temporal encoder was tested at Kentucky University's Information and Telecommunications Technology Center using their state-of-the-art electronic design automation (EDA) tools for circuit design, modeling, and simulation, as well as their specialized lab facilities for integrated circuit testing.

4.1 Simulation Results for the Temporal Encoder Design

We developed an optimized circuit schematic for our Temporal Encoder, and then used the Cadence Virtuoso Analog Design tools to generate the layout of the encoding circuit. In our circuit the leak current is generated by a current reference. While most neuron circuits simply use one transistor to provide the leak current, under these conditions the current can become unstable when the temperature changes [53].

To test the encoding capability of our design, the post layout simulation for the encoder has been carried out. The simulation result is illustrated in Fig. 24.

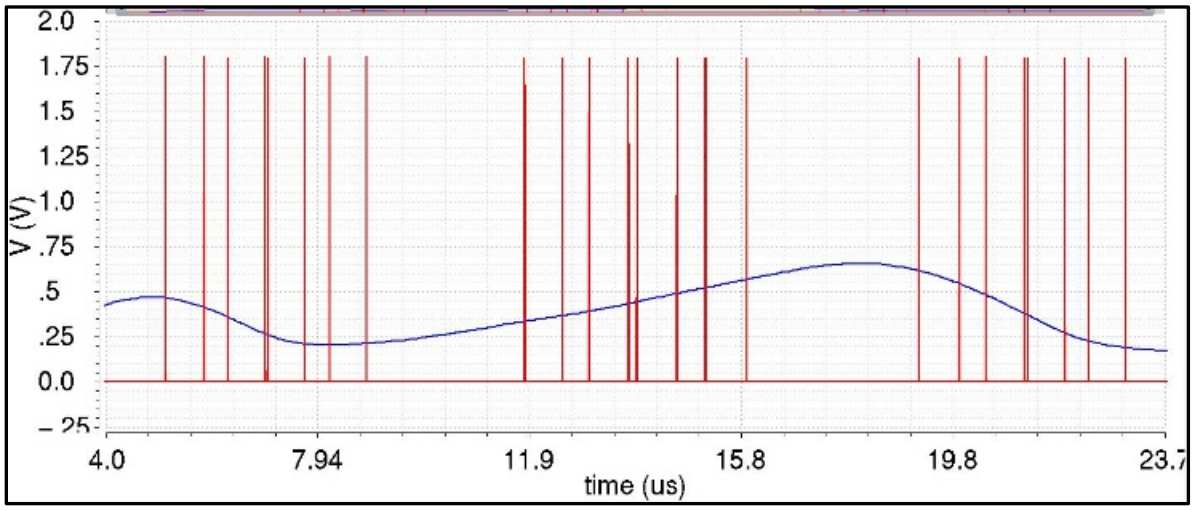


Figure 24. Simulation of the Temporal Encoder for Three Samples

The red lines in Fig. 24 represent temporal spike codes, and the blue line represents analog input signal. There are three sets of spike trains which contain 8 spikes in each sampling window.

In our encoder, 4 neurons were used to construct the encoder. The number of spikes is 8, which satisfies the design requirement.

In our designed circuit, the leak current was generated relative to a current reference. A current mirror is a circuit designed to copy or mirror the current flowing in one active device that controls the current in another active device in the same overall device circuit, keeping the output current constant regardless of the loading. Different leak current values were generated by adjusting the ratio of the current mirror in the leak current module.

Changes in leak currents were generated by changes in the ratio of the current mirror in the leak current module. The current source that was designed for our spike-time dependent encoding is an absolute temperature current source that is stable vs. temperature. For our fabrication specifications the temperature coefficient determined by modelling is the slope of the output current vs. the temperature, and is shown in Fig. 25. From the slope, the current source will have a temperature coefficient of 742ppm/℃. Comparing with [54], our current source's temperature coefficient performance fully satisfied the requirements for leak current stability.

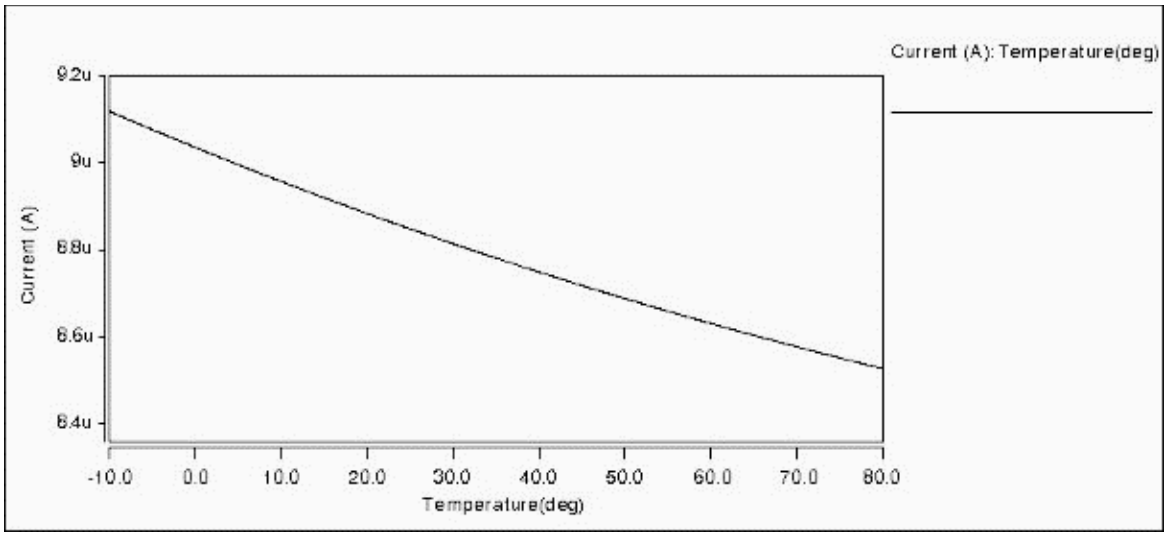


Figure 25. Modelled Temperature Coefficient of Our Design

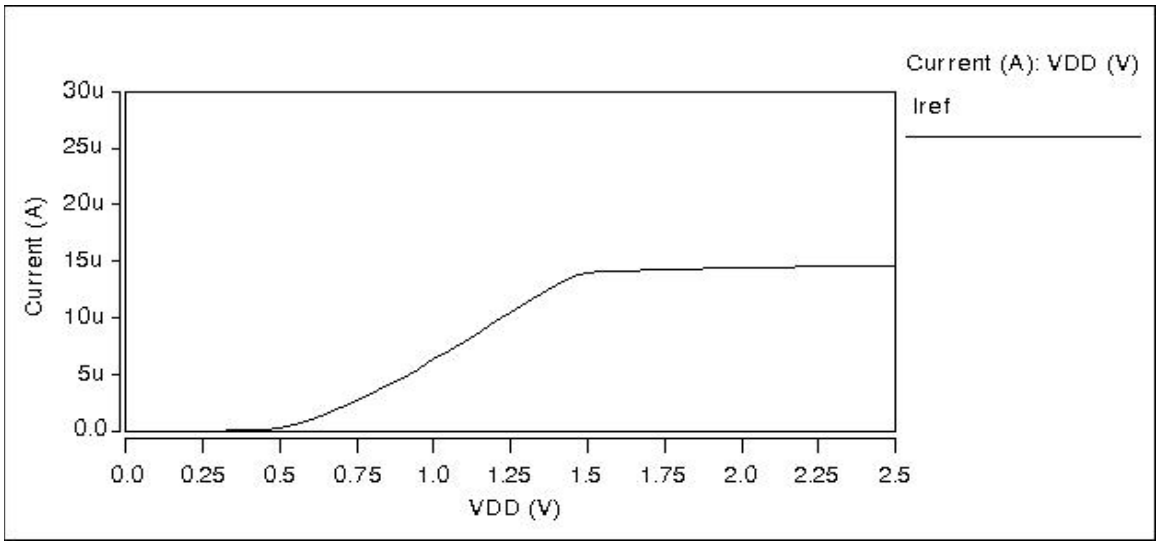


Figure 26. Power Supply Output Current vs. Voltage

As shown in Fig. 26, our power supply was determined by design simulations to have an output current of 14.0±0.3μA.

The fabricated encoder was further tested and analyzed here. As shown in Figure 27, some inspection spikes, illustrated with red dashed lines, are missing from the measured output.

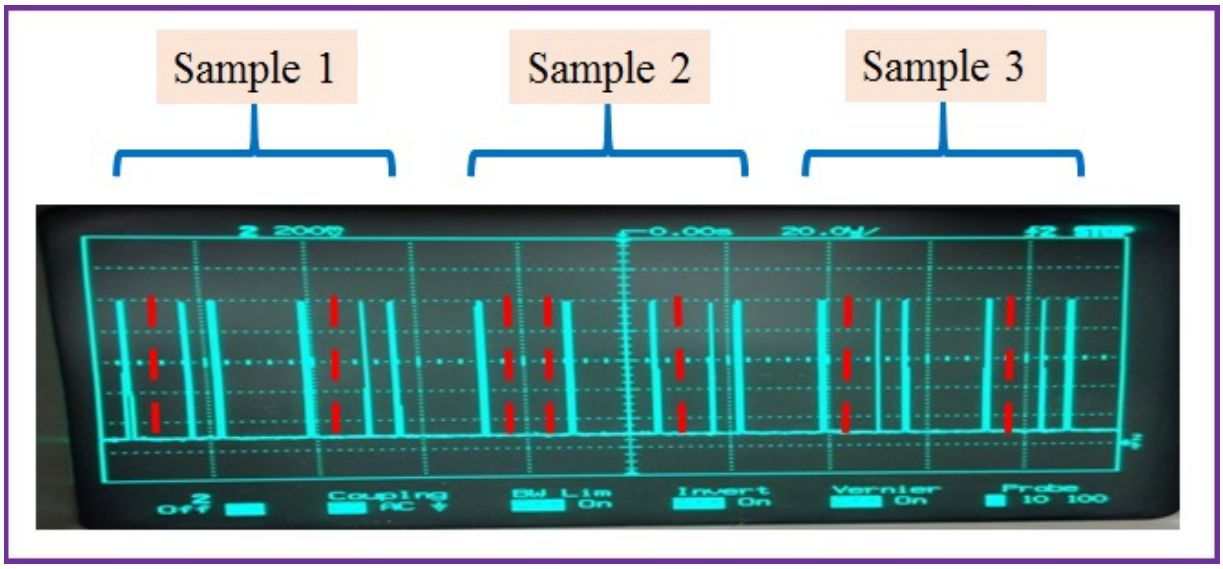


Figure 27. Output Temporal Code

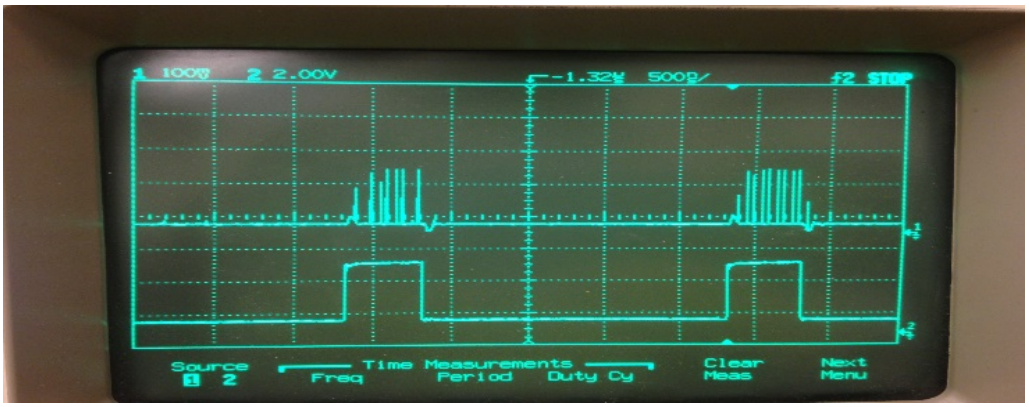


Figure 28. Measurement for Two Samples

A two-sample output temporal encodes is shown in Fig. 28. By adopting an internal verification scheme, we further tested our designed encoder's error tolerance ability. Without a loss of generality, 10 of the sampling points were chosen from the 100 sampling points, and the results showed that the difference between the maximum and minimum value was 0.057, which indicated that all of the verification values were distributed within this extremely small range. The maximum error was less than 0.034, which represents $8.16\times10^{-4}\,\mu s$. Compared to the magnitude of our sampling time of 1 μs, such an error is not detectable. Also, the difference between the maximum value and the minimum value was 0.022 μs, which was much smaller than the resolution of each single temporal encode, which was 0.1 μs.

External verification is another important aspect of the proposed encoder. In order to evaluate its performance, one segment of output temporal encode extracted from the measurement data, which is shown in Fig. 29, has been taken into account to illustrate how such kind of error-tolerance mechanism works.

Figure 29. Measurement of Temporal Encode

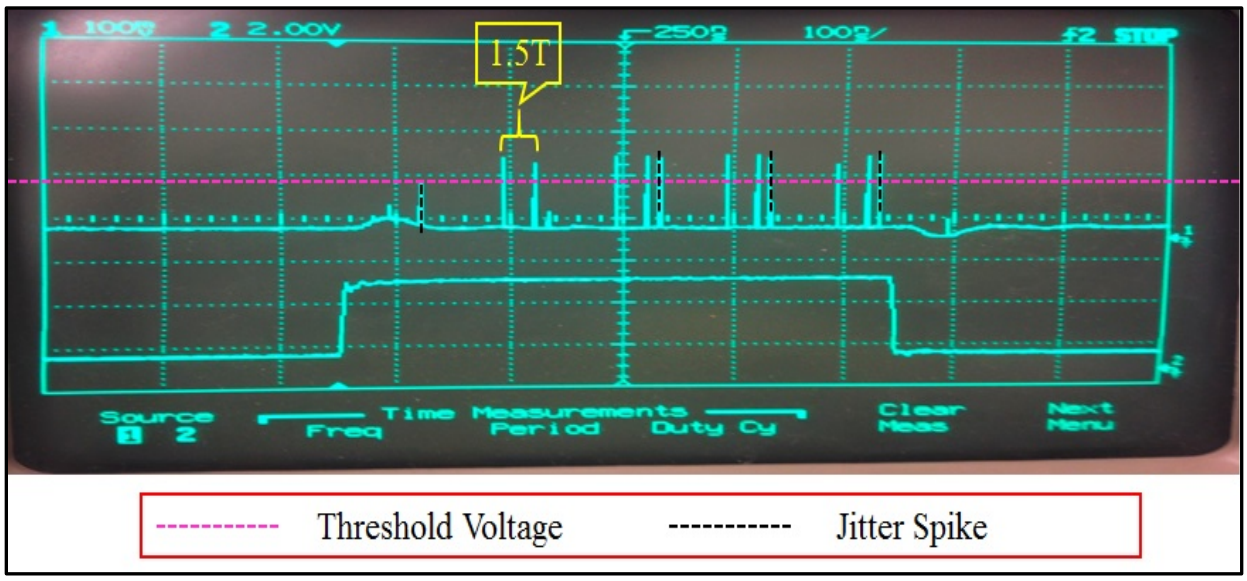


Figure 30. Temporal Spike Train with Jitter Spikes

Jitter is the deviation from true periodicity of a periodic signal. Jitter spikes were sometimes present within a sampling window. As shown in Fig. 30, there were several jitter spikes that appeared within this sampling window. Threshold voltage values were marked by the red dotted line. Spikes with amplitudes that were lower than the threshold voltage were considered noise. According to the recovery scheme, there was a minimum holding time.

4.2 Design Error

In our design, the minimum time *T* was defined as 2/3 of the smallest inter-spike interval. The distance between these spikes and the previous valid spike was much smaller than *T*. Therefore, these jitter spikes are not counted in the recovery scheme. By using this rule, the error rate is illustrated in Fig. 31.

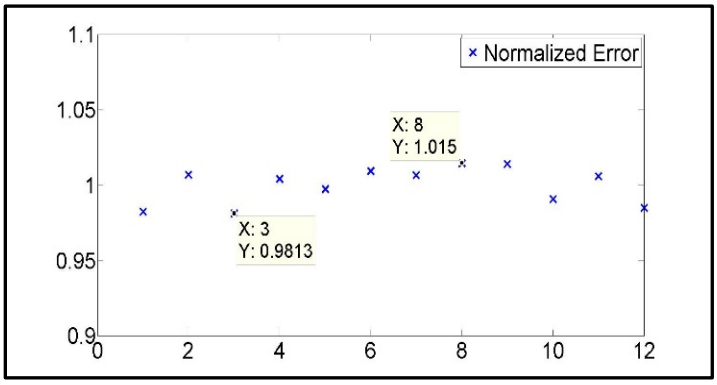


Figure 31. Inspection Error

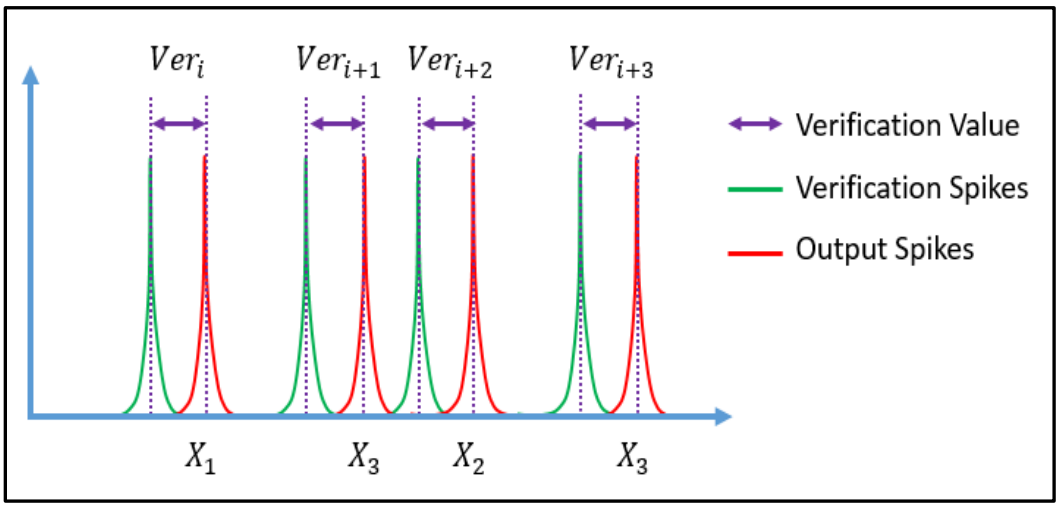


Figure 32. Spike Train Verification Spikes

With respect to the analog design of the neural encoder, a crucial component is its error tolerance. In our encoder design, internal and external verification schemes have been introduced to improve its error tolerance. The verification value Ver_i serves as the criteria to inspect the working conditions of each neuron. A spike train illustrates the verification scheme, and is shown in Fig. 32.

In our design, each neuron can generate a spike train that will be transferred to the neuron in next stage and to a verification combiner. Since our temporal encoder adopted an iteration structure, all of the neurons work together asynchronously with delays between verification code and the output code. In other words, except for the first neuron, the rest of the neurons are driven according to their order. Exception spikes can appear as soon as one neuron fails to work correctly.

By checking the verification values, it can be determined whether the temporal encoder is operating faultlessly. In our design, there are eight spikes that are generated for each sampling point, which then acquires eight verification distances V_{erij} for each cycle. In this case, the normalized value for evaluating each neurons' working conditions is expressed as Equation (16).

$$\delta_{ij} = \frac{8NVer_{ij}}{\sum_{i=1}^{N}\sum_{j=1}^{8} Ver_{ij}} \tag{16}$$

N is the sampling point. Excepting the internal verification technique, the final temporal encode output also possesses a high error-tolerance mechanism. This type of mechanism is achieved by adding additional inspection spikes, which can be used to acquire inspection intervals. This scheme is called the external verification scheme. The inspection intervals should have the same values to guarantee the output temporal code carries only the correct information.

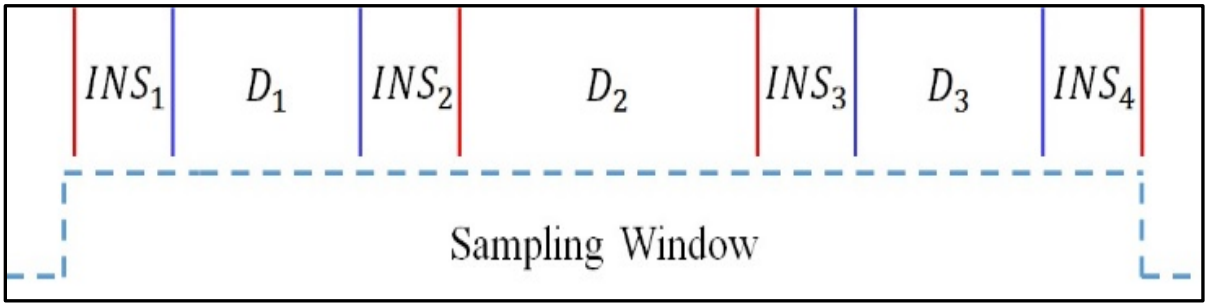


Figure 33. Inspection Scheme Overview

In Figure 33 is shown an overview of the inspection scheme. The vertical blue lines represent the inspection spikes, the inspection intervals are INS_i, and D_i represents the original temporal encoding interval and dynamic temporal encoding intervals.

4.3 Encoding Rates

4.3.1 Neural Encoding

The efficiency of neural encoding is evaluated by Information Theory. The encoding time window T_{encode} was partitioned into small time bins (Δt), and the presence or absence of spikes in each bin was represented by a binary sequence that is called the 'spike-word'. The resulting entropy, also called "self-information", characterized the encoding rate of the coding strategy. In this project we extended this concept to a more general setup to evaluate the encoding rates for the three encoding strategies.

To be specific, let Δt' be the resolution of the corresponding encoder. The encoding time window was partitioned into L= T_{encode} /Δt' bins. Let [x_1,x_2,…,x_L] be a vector of random variables where x_i is a random variable representing the number of spikes in the i^{th} bin. It is important to note that the temporal information was encoded in the corresponding vector.

Accordingly, the encoding rate (entropy) of the introduced temporal rate became H(x_1,…,x_L) bits, where H(·) was the defined entropy function b. For rate encoding, the exact value of x_i does not matter. Rather what matters is the sum of x_i, which is the total number of spikes in T_{encode}. Therefore, the encoding rate of rate encoding is expressed as H(x_1+⋯+x_L). From the chain rule of entropy, we have Equation (18).

$$\begin{aligned} & H\big((x_1, \dots, x_L), (x_1 + \cdots + x_L)\big) \\ &= H(x_1, \dots, x_L) + H\big((x_1 + \cdots + x_L) | (x_1, \dots, x_L)\big) \\ &= H(x_1 + \cdots + x_L) + H\big((x_1, \dots, x_L) | (x_1 + \cdots + x_L)\big) \end{aligned} \tag{18}$$

It is clear that $H\big((x_1 + \cdots + x_L) | (x_1, \dots, x_L)\big) = 0$, since $(x_1 + \cdots + x_L)$ was deterministically conditioned on knowing $(x_1, \dots, x_L)$, and $H\big((x_1, \dots, x_L) | (x_1 + \cdots + x_L)\big) > 0$. Therefore, from the above equation, we have $H(x_1, \dots, x_L) > H(x_1 + \cdots + x_L)$.

This means that the encoding rate of temporal coding was faster than that of the rate coding. Based on the information theoretical framework, we can evaluate the encoding rate for rate coding, parallel temporal coding, and iteration temporal coding. For the case where $T_{encode} = 1.12$ and $\Delta t' = 0.14$, which is the case when at most 8 spikes were generated for all three of the encoders, we computed that $H_{rate} = 3.17$ bits, $H_{parallel} = 4.25$ bits, and $H_{iteration} = 5.29$ bits. These results clearly indicated that the benefits of temporal coding over rate coding, and showed the

iteration based temporal coding achieved the best encoding rate across all three of the coding strategies.

4.3.2 Parallel Encoding

Encoding speed is one of the key performance parameters for neural encoder design. For the parallel encoder, the encoding time $T_p$was determined by the neuron with the largest capacitor. The encoding time T_{it} of the iteration encoder was mainly determined by the iteration times expressed as Equation (19).

$$T_{it} = \sum_{i=1}^{N} t_i \tag{19}$$

N was the total neuron number and t_i was a single neuron operation time. In order to make sure that the parallel encoder generated robust temporal spike trains, the capacitances ratio between each neuron needed to be larger than 1.2. Since the operation time of one neuron was mainly determined by the integrating time of the membrane capacitor, which was directly proportional to membrane capacitor, the encoding time is shown in Equation (20).

$$T_p = 1.2^N t_1 \tag{20}$$

t_1 was the shortest neuron's operation time. Comparing Equations (19) and (20), the parallel encoder had the faster encoding speed when the number of neurons was small, and the iteration encoder had a much faster encoding speed when the number of neurons became larger.

4.4 Feedback Reservoir Design Die Area

In traditional encoder designs, operational amplifiers and comparators are required to make a complete encoder system. The rate and temporal encoders that we have designed here were LIF neurons. This approach avoided the requirements for power consuming Analog to Digital Converters (ADC) and operational amplifiers, and resulted in a significant savings in power requirements and design area. Rate encoders required the smallest design areas, while a three-neuron based parallel encoder required the same area as that of a four-neuron iteration encoder. The membrane capacitor in the parallel encoder was different, which made the scalability of the parallel encoder very challenging. The iteration encoder adopted the same-sized neuron with the

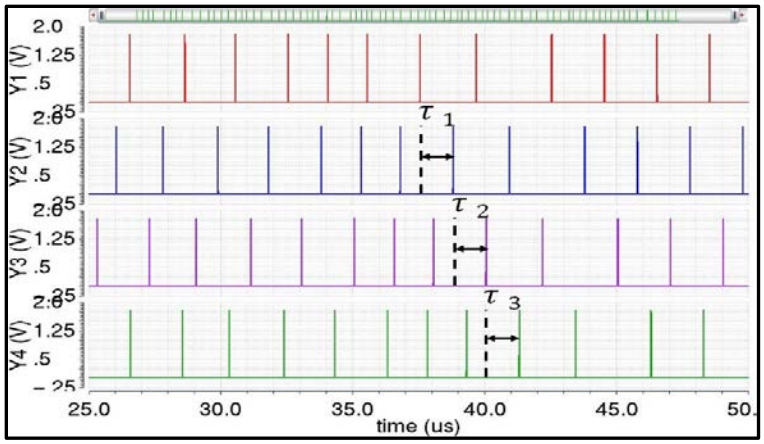
same number of membrane capacitors, which resulted in less scaling of the neuromorphic computing network as it incorporated larger numbers of neurons and synapses to more closely emulate a brain's information-processing infrastructure.

4.5 Delay Feedback Reservoir Design Performance

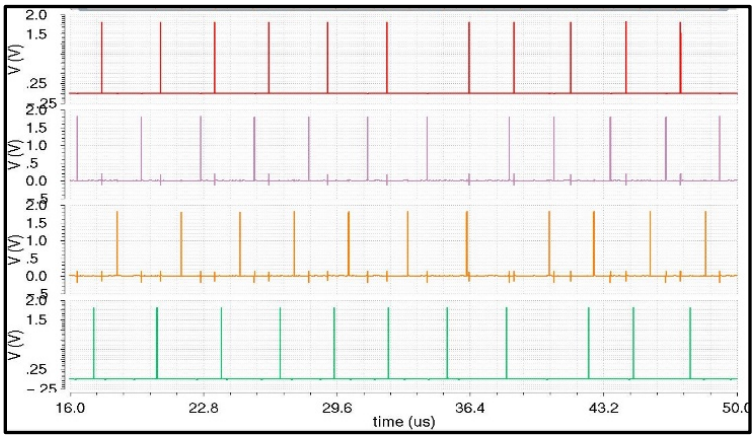
Fig. 34 shows the output spike trains for the delay times of 1.27 μs, 2.03 μs, and 3.69 μs. The time delays between each spike train were determined from these output spike train plots. These delay times were determined to be identical. In Fig. 35 are shown the phase portraits for the three different delay times. By tuning these delays, the dynamics of the system can be varied from order to the "edge of chaos". This confirms that the desired nonlinear mapping is successfully implemented with our designed spike-based nonlinear neural node.

A traditional analog delay unit is purely based on the capacitor size. In order to acquire 1 us delay, such a system requires at least a 100 KΩ resistor and a 10 pF capacitor to achieve the goal. Since the delay time is designed to be identical, the delay time can be determined by evaluating the output signals.

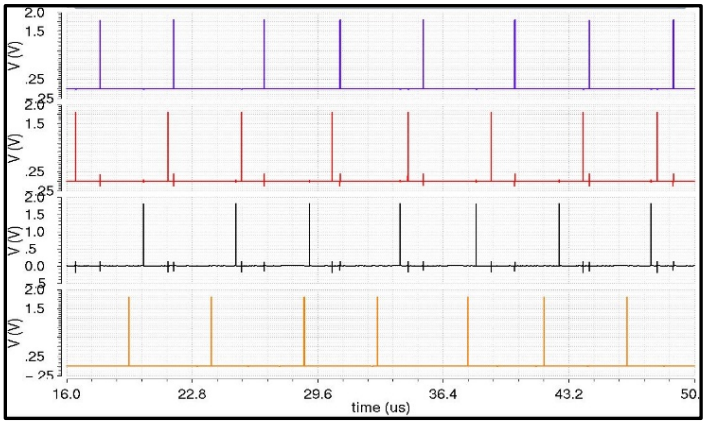
Fig. 35 showed a four-stage delay loop whereby the output spike trains are illustrated. The *Vth* is in the 1V level, and the *Iex – Ileak* is in 0.1 *uA* level, which is equivalent to a 10 MΩ resistance.



(a) 1.27 µs

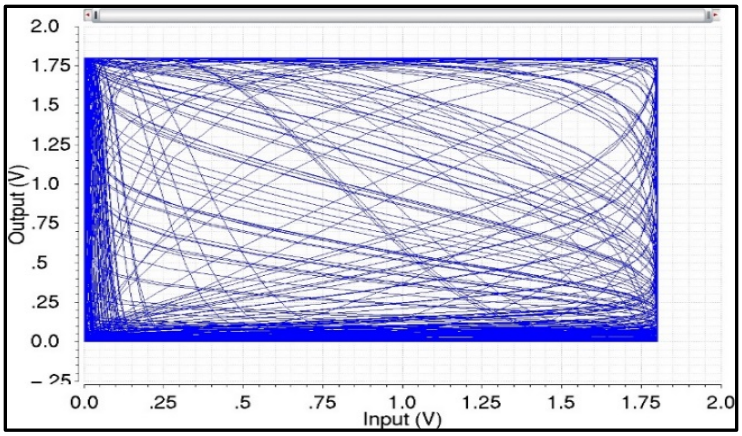


(b) 2.03 µs

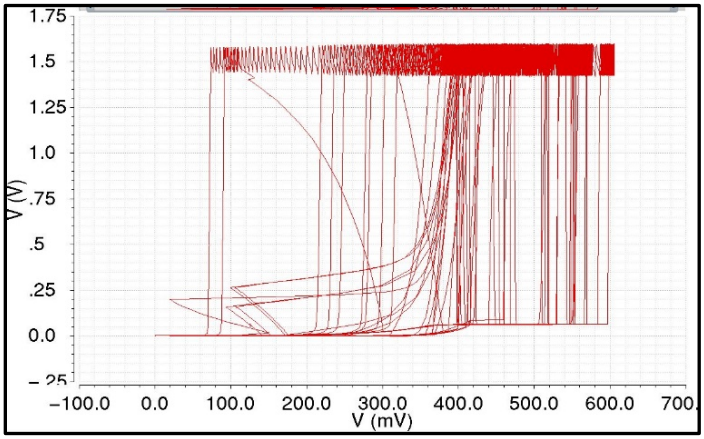


(c) 3.69 µs

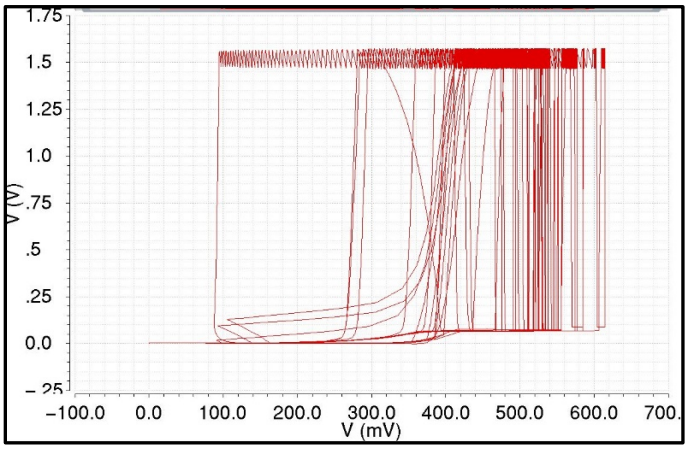
Figure 34. Output Spike Trains as a Function of Delay Times



(a) 1.27 µs



(b) 2.03 µs



(c) 3.69 µs

Figure 35. Phase Portraits as a Function of Delay Times

Therefore, the designed delay unit *d* can achieve large delay timed with very small capacitors. Hence, the dynamics of the system can be varied from order to the "edge of chaos" by tuning the delay constant with very small capacitance and resistance values.

We applied our designed dynamic MG function based neuron to serve as a class of dynamic reservoir nodes that met the requirements of high dimensionality and finite memory in reservoir computing systems. In an endeavor to reduce the complexity of RNNs, reservoir computing architectures have been proposed in the field of machine learning. For the reservoir computing, there exists three layers, input layer, reservoir, and output layer, in which the architecture of the reservoir is based on the recurrent neural network (RNN).

Unlike RNNs the connections within the reservoir would not be trained by the assignments of randomly chosen synaptic weights. The input connections serve as the scaling of the input signal, and transfer the scaled signal to the reservoir. Within the reservoir nodes are connected in a random manner whereby nonlinear mapping takes place. The outputs from the reservoir are then transferred to the output layer through the output weights. The only weight connections that are trained are the output weights for reservoir computing. Hence, the computational cost of such a computing architecture is drastically reduced when compared to RNNs.

To demonstrate the nonlinear behavior of the DFR, the nonlinear node was designed to model the nonlinear regime of MG function, as depicted in Fig. 36. Similar as the nonlinear characteristics of the MG function, it was observed that the nonlinearity of the transfer function in the nonlinear node was regulated by the time constant τ_{ND}.

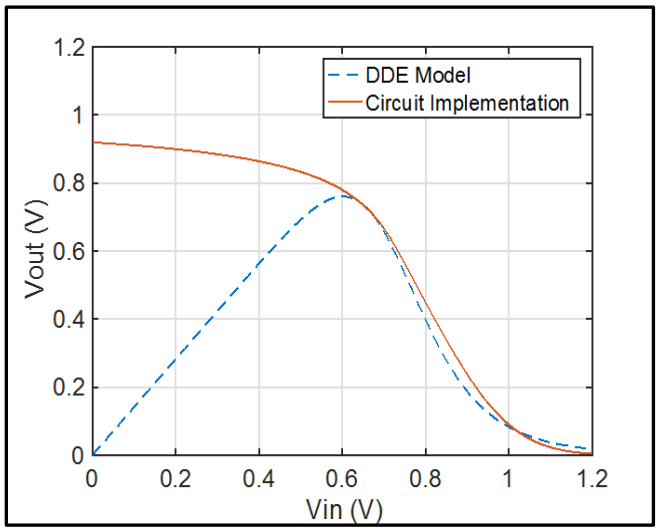
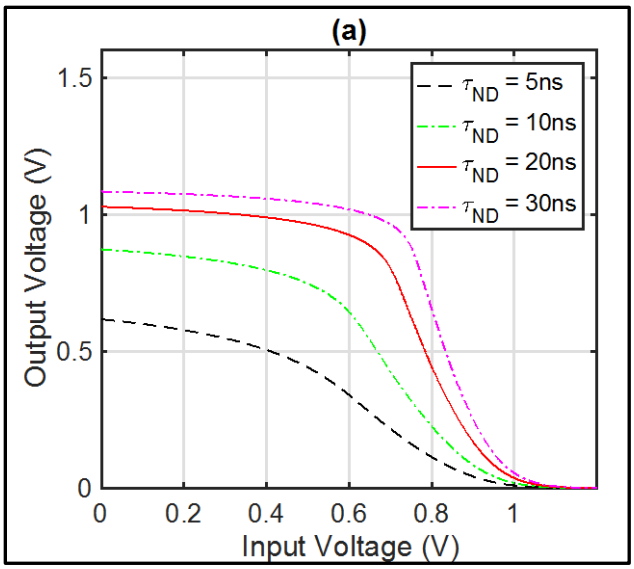
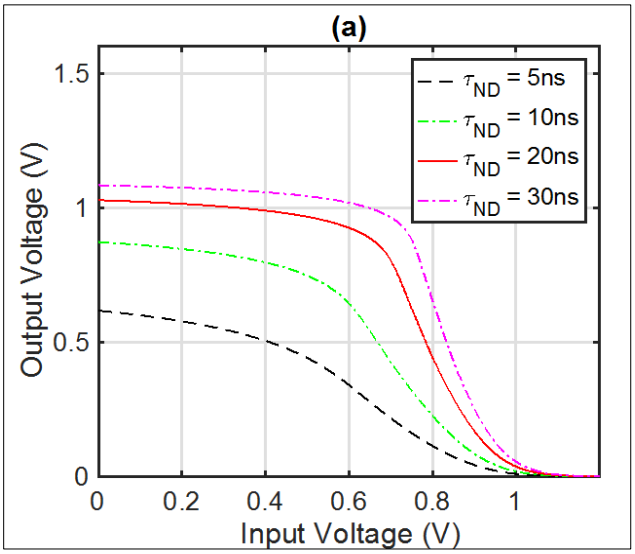


Figure 36. Nonlinear Regime of MG Function and Circuit Implementation



(a) Output of the Nonlinear Node



(b) Output of the MG Function

Figure 37. Transfer Function

To demonstrate this nonlinearity, the value of τ_{ND} was changed from 5 ns, 10 ns, 20 ns and 30 ns, and the output voltage was measured. These results are shown in Fig. 37 (a). In Fig. 37 (b) is presented the output of the MG function with four different nonlinearity exponents, n. As the exponent increased, the nonlinearity also increased. This same characteristic was observed for the circuit's nonlinear function.

In the DFR the dynamic of the system was varied from order to the "edge of chaos" by controlling the total delay time of the delay loop. To demonstrate the delay behavior of the system,

the delay time τ_{delay} of the IF-based delay unit was changed to achieve a large dynamic range of controllable delay time. The delay time of the IF-based delay unit was measured at various excitation currents. As plotted in Fig. 38, the delay time was regulated from 180 ns to 1.5 μs by changing the excitation current from 50 nA to 300 nA.

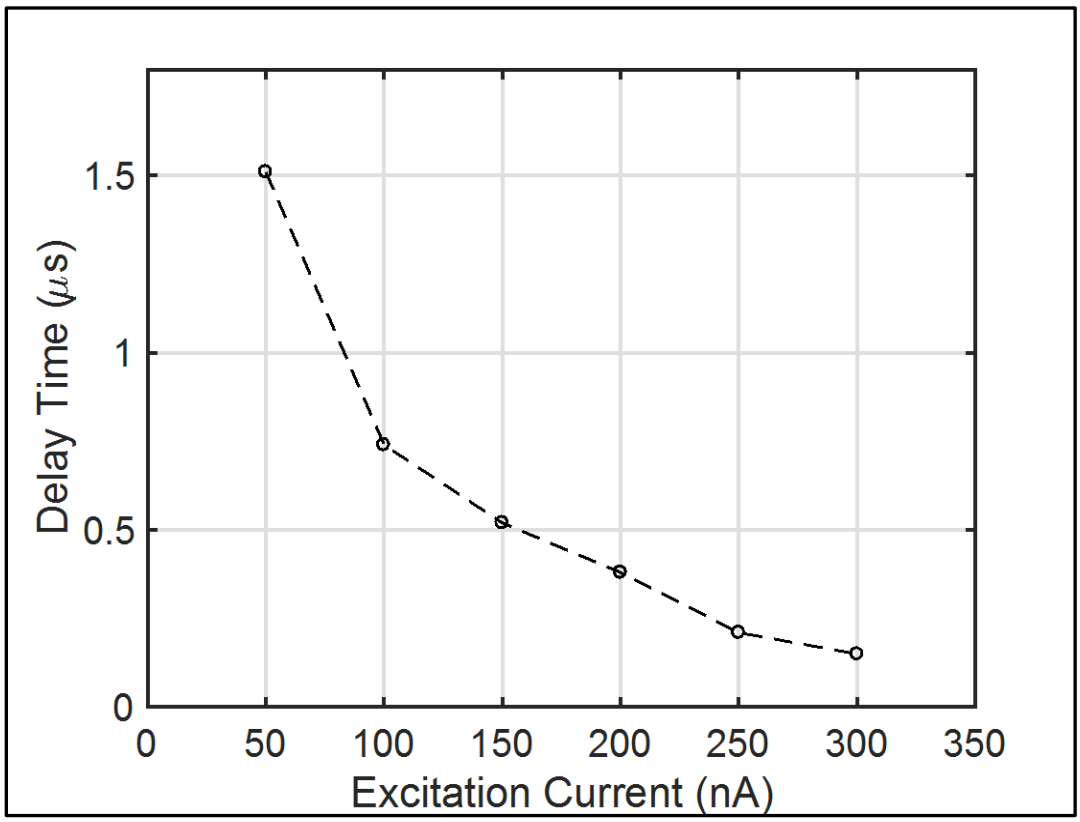


Figure 38. Controllable Delay Time

To acquire a 1.5 μs delay via a traditional RC-based delay unit, such a system required a 100 kΩ resistor and a 15-pF capacitor, which required a large chip area to implement. Our IF-based delay unit design overcame this drawback by regulating the equivalent input impedance of the circuit. By injecting 50 *nA* excitation current into the delay unit, the equivalent input impedance reached 25 MΩ. Thus, a large delay time was achieved with an extremely small capacitor. More

importantly, such a system is capable for handling spike-based signals, as presented in Fig. 39. This figure also demonstrated the output spike trains along the dynamic delay loop. The results also indicated that the time delays between each neuron were approximately identical.

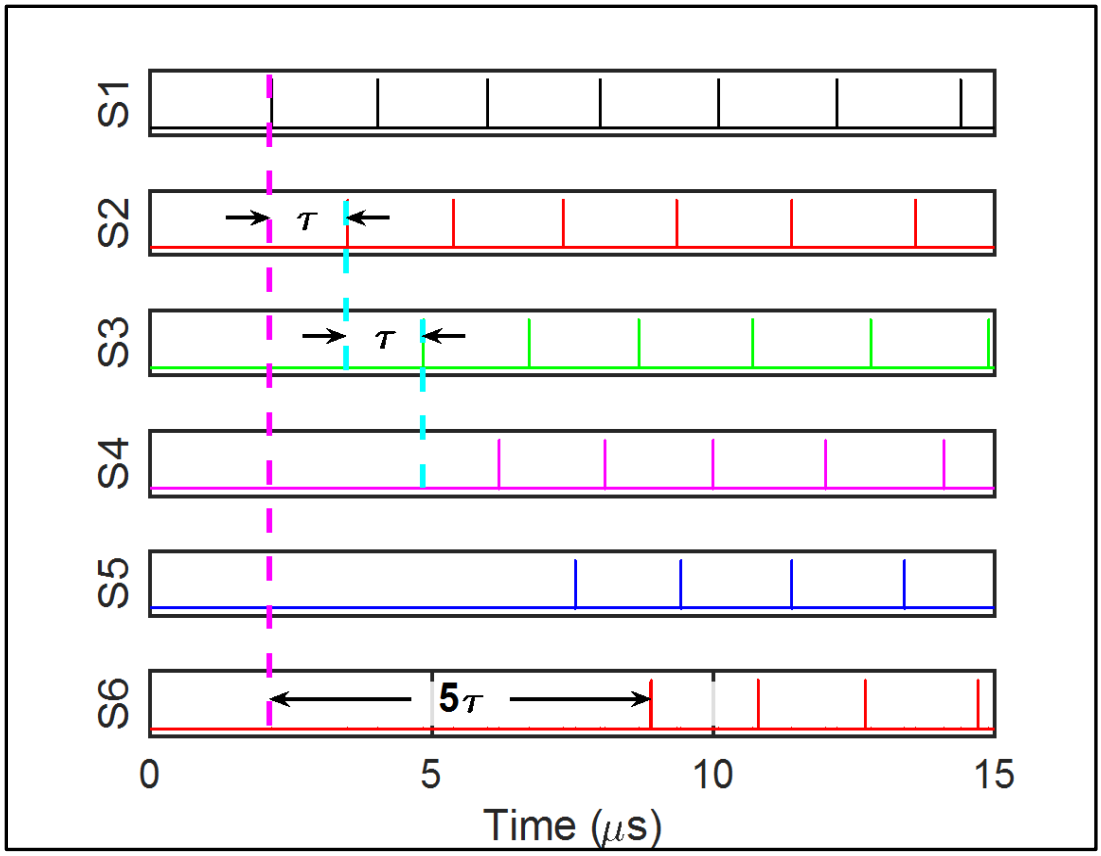
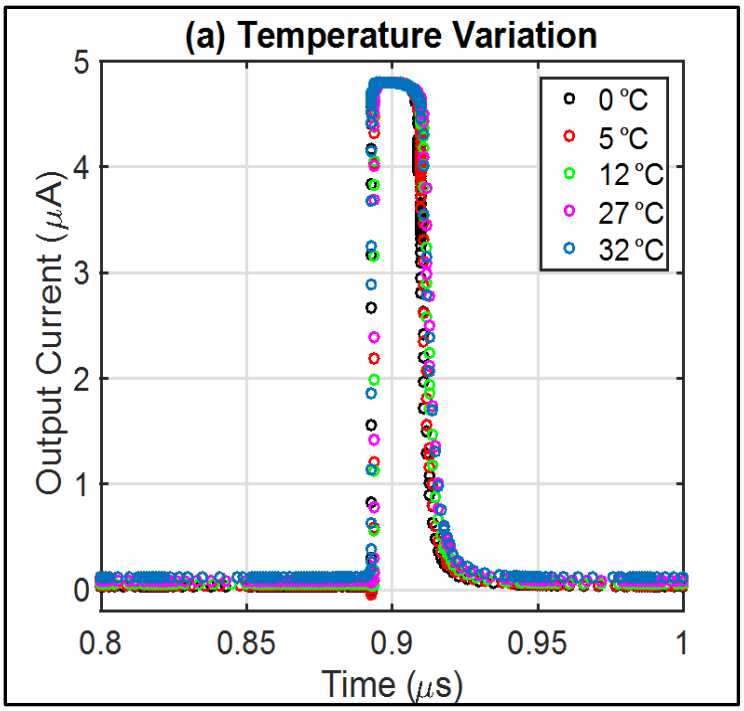


Figure 39. Output Spike Trains along the Delay Loop



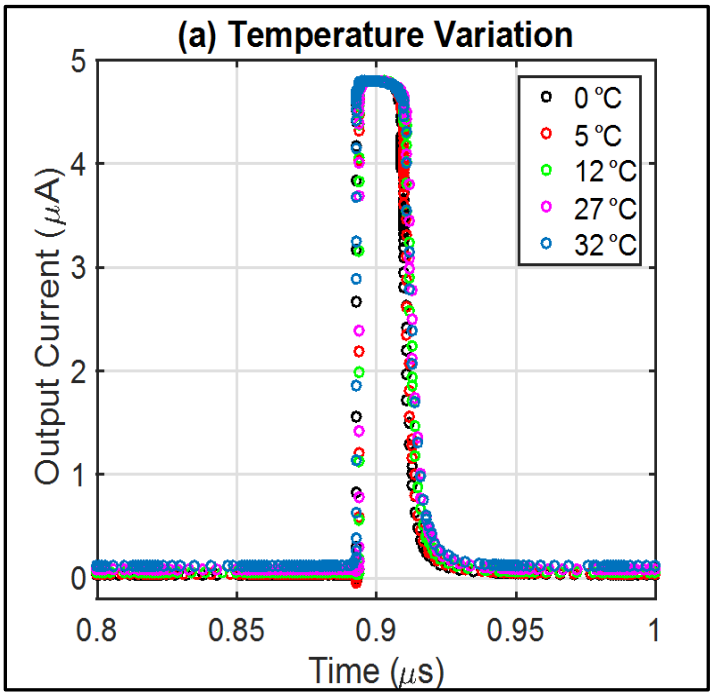


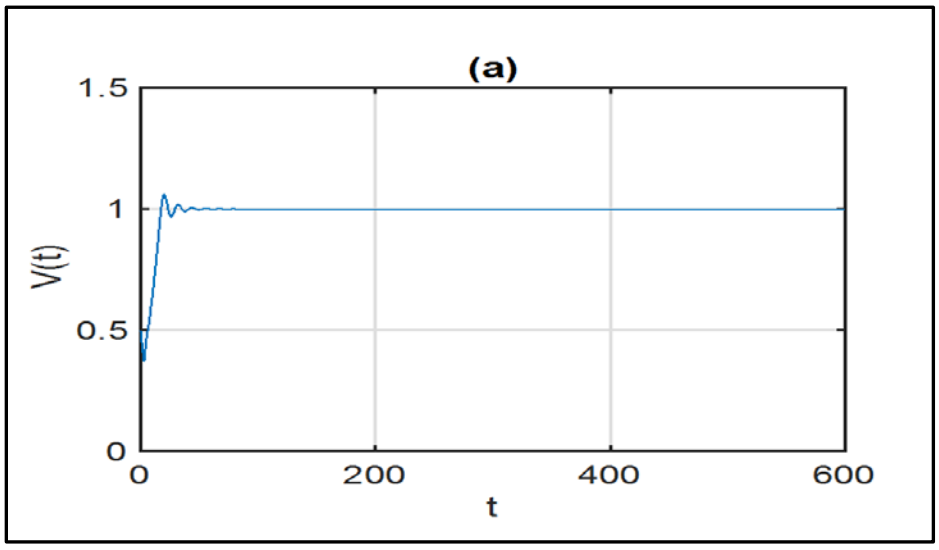
Figure 40. System Precision

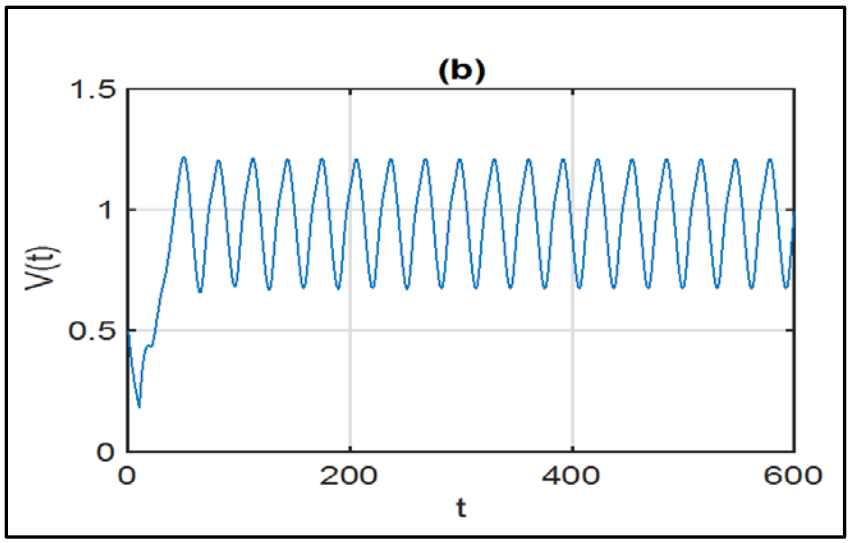
The system performance and precision were determined through Monte-Carlo simulations, where both the temperatures and the processes were varied. To demonstrate the precision of the system, the designed DFR was simulated by introducing process variations via the output current mirror of the nonlinear node. Five random samples of output current were selected to demonstrate

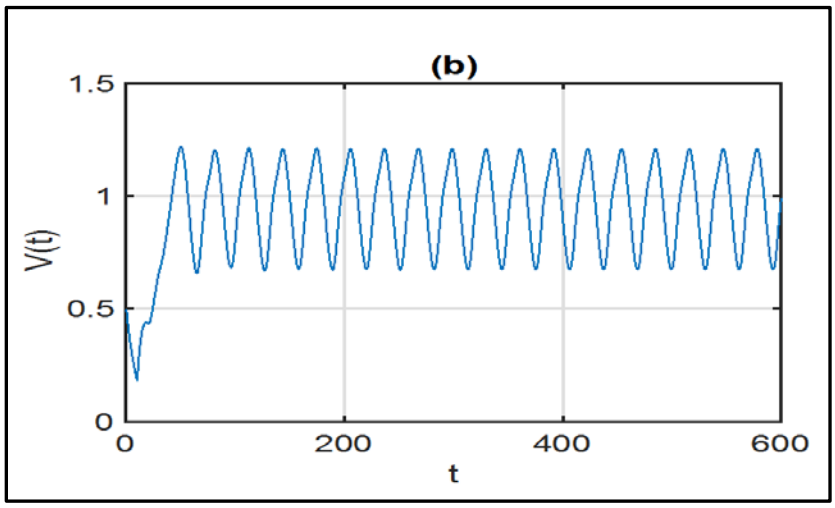
the precision of the system, as depicted in Fig. 40. The results indicated that our system had a precision of <1%, which was within the experimental temperature variation.

However, the output current mirror of the nonlinear node was optimized such that it operated between the sub-threshold and saturation region to achieve the maximum nonlinearity of the circuit. Thus, the output of the nonlinear node was sensitive to device mismatch. In the circuit implementation, the size of output current mirror of the nonlinear node was augmented to overcome this drawback.

To closely examine the dynamic behavior, solutions to the DDE equation were carried out. The dynamic behavior of the nonlinear function was modeled by the DDE with varied delay times, as demonstrated in Fig. 41. The solutions converged to an equilibrium state when the delay was small. The dynamic behavior varied accordingly as the delay increased. With increasing time delay, the dynamics changed from periodic to chaotic.







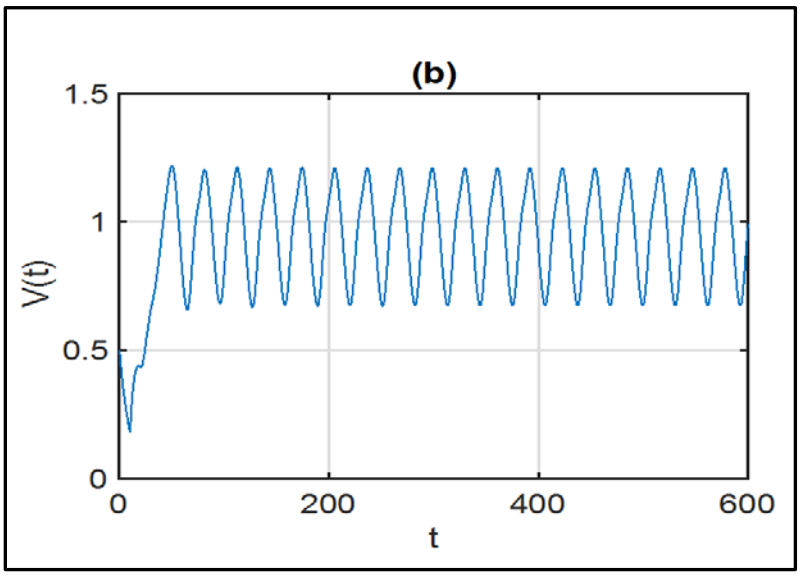


Figure 41. Dynamic Behavior of Nonlinear Functions

(a)$\boldsymbol{\tau = 3}$; (b) $\boldsymbol{\tau = 12}$; (c) $\boldsymbol{\tau = 16}$; (d) $\boldsymbol{\tau = 20}$.

The phase portrait is a representation of the solutions tracing the path of each solution. It is a graphical tool to visualize how the solutions of a given system of differential equations would behave in the long run. In other words, a phase portrait tracks the dynamic behavior of a system solution. By varying the time delay of dynamic systems, the phase portraits are illustrated as Fig. 44.

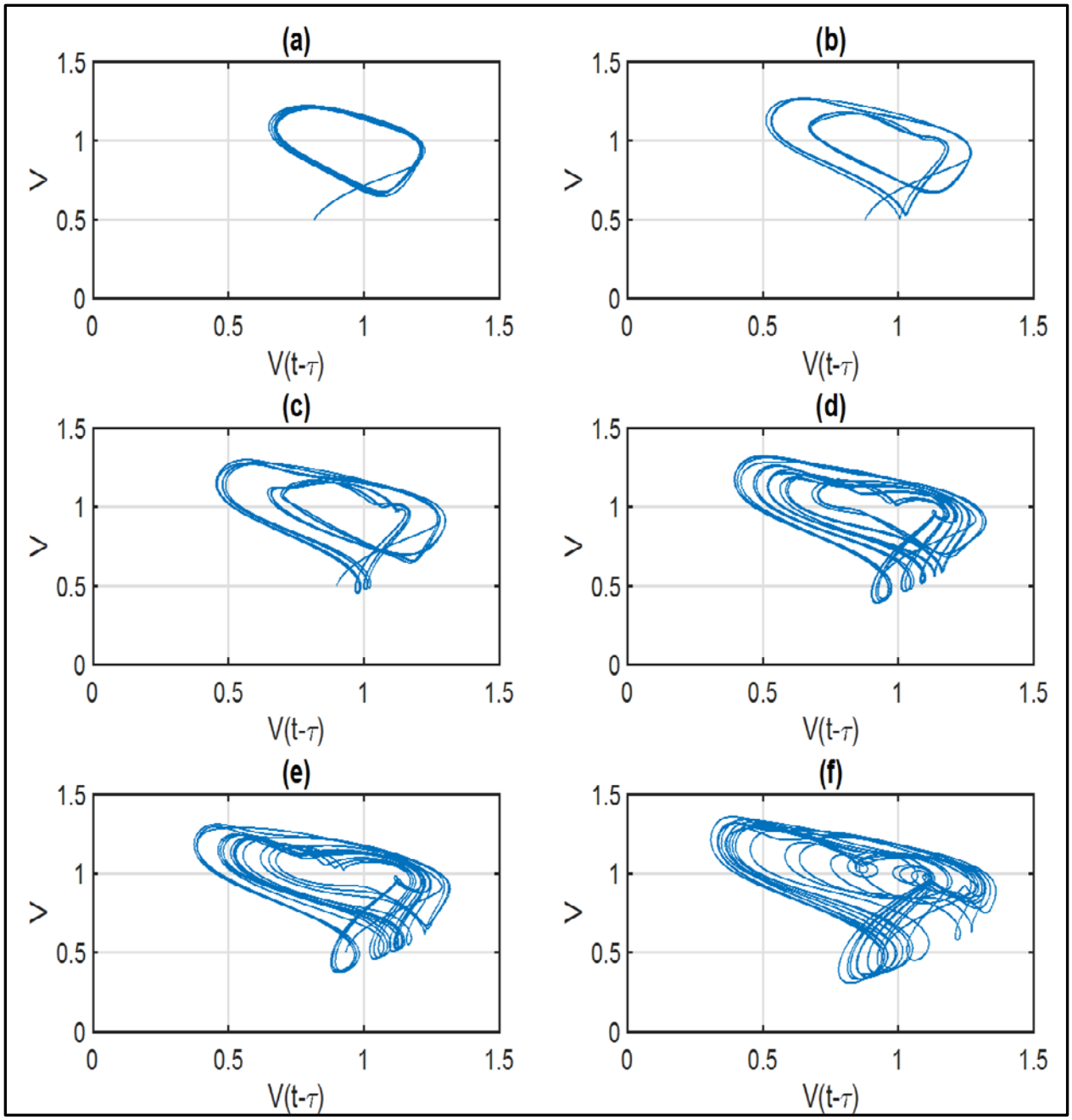


Figure 42. Phase Portraits of Dynamic Systems

(a) $\boldsymbol{\tau = 12}$; (b) $\boldsymbol{\tau = 14}$; (6) $\boldsymbol{\tau = 16}$; (d) $\boldsymbol{\tau = 18}$; (e) $\boldsymbol{\tau = 20}$; (f) $\boldsymbol{\tau = 22}$.

4.6 Delay Feedback Reservoir Design Summary

Table 2 provides a summary of the design of the Delay Feedback Reservoir system chip that is being fabricated.

Table 2. Delay Feedback Reservoir System Summary

Technology	130 nm
Implementation	Integrated Circuit
Supply Voltage	1.2 V
Delay Structure	IF (spike)-based Delay
Power	175 μW
Design Area	0.0098 mm2

5.0 CONCLUSIONS

We have encoded neural responses using different timescales with different stimulus attributes to generate temporal inter-spike intervals of sensory information. The performances of rate codes and temporal codes were compared, and the computational advantages of temporal code with inter-spike intervals was demonstrated. This reduced the ambiguity inherent in single-scale codes and enhanced the robustness of neural representations compared to environmental noise.

While most implementations of reservoir computing are embodied in software, efficient hardware implementations of these concepts provided numerous advantages. Hardware implementations are capable of exploiting the full potential of the intrinsic parallelism of neural networks. Dedicated hardware implementations for specific tasks also offer advantages over software implementations whenever either low power consumption or high processing speeds are a priority.

More specifically, we have designed and tested an analog delay-based reservoir node. This advanced neuron circuit design for a delay feedback MG function bears a much closer resemblance to the behavior of neural networks than that of the tanh and sigmoid functions. In order to ensure the real-time operation, the digital signals are required to interface with the analog world, which leads to the addition of digital-to-analog and analog-to-digital converters. Our analog implementation has the advantage of implicit real-time operation, resulting in a small design area and lower power. Furthermore, our dynamic delay based neuron could perform nonlinear transformation and map input signals to higher dimensional state, which makes it a potential suit for reservoir computing.

Our design has made three main contributions:

- An analog spike-based nonlinear processor directly processes spike signals.
- Our delay loop for the reservoir node is the first type of spike-based delay loop.
- The power consumption is greatly reduced since components including analog-to-digital converters (ADCs) and operational amplifiers (Op-AMPs) are not necessary.

This project resulted in the design of an agile analog integrated circuit implementation of a spike-time encoding circuit as a signal conditioner and electronic reservoir as a dynamic processor for the reservoir computing systems. This effort bridged high-performance computing, nanotechnology, and integrated circuits & systems.

We completed the optimized circuit schematic and then use the Cadence Virtuoso platform to generate the layout of our designed reservoir circuit. This research produced deliverables including a dynamic reservoir circuit design, SPICE circuit models, and circuit prototypes. We provided a nonlinear processor designed to exploit recent advancements in nano-technology and interconnects for a new class of computational systems based on dynamically driven architectures. This effort started in February 2016, and was planned to end in December 2018. However, due to the transfer from KU to Virginia Polytechnic Institute and State University, this effort is ended earlier than expectation. By the end of this effort, we have accomplished most of the originally planned tasks. The remaining tasks that need to be completed in future effort included testing the fabricated dynamic reservoir circuit, and incorporating these results into an additional iteration of the circuit.

In summary, we have completed:

- Comprehensive investigation of sensory information mapping in the neocortex
- Spiking time dependent encoder design and analysis with multiple inter-spike intervals
- Chaotic circuit (serving as a pseudorandom time series generator) design and optimization for the sampling clock input of an encoder circuit
- Encoder circuit fabrication with advanced CMOS nano-technology was in-process when the effort was completed.

A delayed feedback reservoir was designed, and that design was optimized. The reservoir was designed to have rich dynamics, and a wide range of tasks should result from using linear readout neurons to extract relevant information from the reservoir.

6.0 REFRENCES

[1] Y. Yi, “Neuron Design in Neuromorphic Computing Systems and Its Application in Wireless Communications,” AFRL-RI-RS-TR-2017-057, The University of Kansas Center for Research, Inc., Lawrence, Kansas, March 2017.

[2] D. Monroe, “Neuromorphic Computing Gets Ready for the (Really) Big Time,” Communications of the ACM, 57(6), pp. 13-15 (2014).

[3] Q. Qiu, Q. Wu, M. Bishop. R.E. Pino and R.W. Linderman, “A Parallel Neuromorphic Text Recognition System and Its Implementation on a Heterogeneous High-Performance Computing Cluster,” IEEE Transactions on Computers, 62(5), pp. 886-899 (2012).

[4] S.B. Eryilmaz, S. Joshi, E. Neftci, W. Wan, G. Cauwenberghs, and H.-S.P. Wong. "Neuromorphic Architectures with Electronic Synapses," *Proceedings IEEE 17th International Symposium on Quality Electronic Design (ISQED)*, pp. 118-123 (2016).

[5] I.E. Ebong, and P. Mazumder, “CMOS and Memristor-Based Neural Network Design for Position Detection,” Proceedings of the IEEE, 100(6), pp. 2050-2060 (2012).

[6] Y. Kim, Y. Zhang, and P. Li, “A Digital Neuromorphic VLSI Architecture with Memristor Crossbar Synaptic Array for Machine Learning,” Proceedings of the IEEE International System-on-Chip (SOC) Conference (SOCC), pp. 328-333 (2012).

[7] J-s. Seo and M. Seok, “Digital CMOS Neuromorphic Processor Design Featuring Unsupervised Online Learning,” Proceedings of the IFIP/IEEE International Conference on Very Large Scale Integration (VLSI-SoC), pp. 49-51 (2015).

[8] C. Piguet, Low-Power CMOS Circuits: Technology, Logic Design and CAD Tools, CRC Press, Boca Raton, Florida (2004).

[9] L.S. Smith, "Neuromorphic Systems: Past, Present and Future," in **Brain Inspired Cognitive Systems 2008**, A. Hussain, I. Aleksander, S.L. Smith, K.A. Barros, R. Chrisley and V. Cutsuridis, eds., pp. 167-182, New York, NY: Springer New York (2010).

[10] C. Zhao, B.T. Wysocki, Y. Liu, C.D. Thiem, N.R. McDonald and Y. Yi, "Spike-Time-Dependent Encoding for Neuromorphic Processors," *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, 12(3), Article 23 (2015).

[11] K. Ramanaiah and S. Sridhar, “Hardware Implementation of Artificial Neural Networks,” Manager's Journal on Embedded Systems, 3(4), pp. 31-34 (2015).

[12] A. Joubert, B. Belhadj, O. Temam, and R. Héliot, "Hardware Spiking Neurons Design: Analog or Digital?," Proceedings of the 2012 International Joint Conference on Neural Networks, pp. 1-5 (2012).

[13] A. Basu, and P.E. Hasler, “Nullcline-Based Design of a Silicon Neuron,” IEEE Transactions on Circuits and Systems I: Regular Papers, 57(11), pp. 2938-2947 (2010).

[14] S. Panzeri, N. Brunel, N.K. Logothetis and C. Kayser, "Sensory Neural Codes using Multiplexed Temporal Scales," Trends in Neurosciences, 33(3), pp. 111-120 (2010).

[15] D.S. Reich, F. Mechler, K.P. Purpura and J.D. Victor, “Interspike Intervals, Receptive Fields, and Information Encoding in Primary Visual Cortex,” Journal of Neuroscience, 20(5), pp. 1964-1974 (2000).

[16] R. Brasselet, S. Panzeri, N.K. Logothetis, and C. Kayser, “Neurons with Stereotyped and Rapid Responses Provide a Reference Frame for Relative Temporal Coding in Primate Auditory Cortex,” Journal of Neuroscience, 32(9), pp. 2998-3008 (2012).

[17] L. Shao, X. Zhen, D. Tao, and X. Li, “Spatio-Temporal Laplacian Pyramid Coding for Action Recognition,” IEEE Transactions on Cybernetics, 44(6), pp. 817-827 (2014).

[18] C. Zhao, J. Li, and Y. Yi, "Making Neural Encoding Robust and Energy Efficient: An Advanced Analog Temporal Encoder for Brain-Inspired Computing Systems," Proceedings of the 35th International Conference on Computer-Aided Design (ICCAD), pp. 1-6 (2016).

[19] Y. Yi, Y. Liao, B. Wang, X. Fu, F. Shen, H. Hou and L. Liu, "FPGA Based Spike-Time Dependent Encoder and Reservoir Design in Neuromorphic Computing Processors," Microprocessors and Microsystems, 46(PB), pp. 175-183 (2016).

[20] L.F. Abbott, “Lapicque’s Introduction of the Integrate-and-Fire Model Neuron (1907),” Brain Research Bulletin, 50(5/6), pp. 303-304 (1999).

[21] 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, 117(4), pp. 500-544 (1952).

[22] R. FitzHugh, “Impulses and Physiological States in Theoretical Models of Nerve Membrane,” Biophysical Journal, 1(6), pp. 445-466 (1961).

[23] Y.-H. Liu, and X.-J. Wang, “Spike-Frequency Adaptation of a Generalized Leaky Integrate-and-Fire Model Neuron,” Journal of Computational Neuroscience, 10(1), pp. 25-45 (2001).

[24] C. Zhao, B.T. Wysocki, C.D. Thiem, N.R. McDonald, J. Li, L. Liu and Y. Yi, "Energy Efficient Spiking Temporal Encoder Design for Neuromorphic Computing System," IEEE Transactions on Multi-Scale Computing Systems, 2(4), pp. 265-276 (2016).

[25] L. Appeltant, M.C. Soriano, G. Van der Sande, J. Danckaert, S. Massar, J. Dambre, B. Schrauwen, C.R. Mirasso and I. Fischer, "Information Processing using a Single Dynamical Node as Complex System," Nature Communications, 2:468, pp. 1-6 (2011).

[26] M. Eyal, "Predicting the Computational Performance of Neural Circuits Modeled with Simple Digital Neurons", Thesis, University of Wisconsin-Madison, URL: http://digital.library.wisc.edu/1793/73349, last modified May 15, 2015. Accessed January 10, 2018.

[27] G. Indiveri, "A Low-Power Adaptive Integrate-and-Fire Neuron Circuit," Proceedings of the 2003 International Symposium on Circuits and Systems, 4, IV-820-IV-823 (2003).

[28] A. Joubert, B. Belhadj and R. Héliot, "A Robust and Compact 65 nm LIF Analog Neuron for Computational Purposes," Proceedings of the *2011 IEEE 9th International New Circuits and Systems Conference (NEWCAS),* pp. A1L-A3 (2011)

[29] R. Wojtyna and T. Talaska, "Transresistance CMOS Neuron for Adaptive Neural Networks Implemented in Hardware," Bulletin Polish Academy: *Technical Sciences,* 54(4), pp. 443-448 (2006).

[30] J.H.B. Wijekoon and P. Dudek, "Integrated Circuit Implementation of a Cortical Neuron," Proceedings of the *IEEE International Symposium on Circuits and Systems*, pp. 1784-1787 (2008).

[31] S. Millner, A. Grubl, K. Meier, J. Schemmel and M.-O. Schwartz, "A VLSI Implementation of the Adaptive Exponential Integrate-and-Fire Neuron Model." *Advances in Neural Information Processing Systems*, pp. 1-9 (2010).

[32] C. Zhao, Y. Yi, J. Li, X. Fu and L. Liu, "Inter-Spike Intervals (ISI) based Analog Spike-Time-Dependent Encoder for Neuromorphic Processors," *IEEE Transactions on Very Large Scale Integration (TVLSI) Systems,* 25(8), pp. 2193-2205 (2017).

[33] C. Zhao, J. Li, H. An and Y. Yi, "Energy Efficient Analog Spiking Temporal Encoder with Verification and Recovery Scheme for Neuromorphic Computing Systems," *18th International Symposium on Quality Electronic Design (ISQED),* pp. 138-143 (2017).

[34] H. Jaeger and H. Haas, "Harnessing Nonlinearity: Predicting Chaotic Systems and Saving Energy in Wireless Communication," Science, 304(5667), pp. 78-80 (2004).

[35] A. Goudarzi, M.R. Lakin and D. Stefanovic, "Reservoir Computing Approach to Robust Computation using Unreliable Nanoscale Networks", **Unconventional Computation and Natural Computation, 8553 Lecture Notes in Computer Science**, Springer, pp. 164-176 (2014).

[36] Y. Jin, Q. Zhao, H. Yin and H. Yue, "Handwritten Numeral Recognition utilizing Reservoir Computing Subject to Optoelectronic Feedback," 11th International Conference on Natural Computation (ICNC), pp. 1165-1169 (2015).

[37] X. Hinaut and P.F. Dominey, "On-Line Processing of Grammatical Structure Using Reservoir Computing," 22nd International Conference on Artificial Neural Networks, 755, pp. 596-603 (2012).

[38] A. Ghani, T.M. McGinnity, L. Maguire, L. McDaid and A. Belatreche, "Neuro-Inspired Speech Recognition Based on Reservoir Computing," **Advances in Speech Recognition,** N. Shabtai (Editor), InTech**,** Chapter 2, pp. 7-36 (2010).

[39] F. Wyffels, B. Schrauwen and D. Stroobandt, "Using Reservoir Computing in a Decomposition Approach for Time Series Prediction," Proceedings of the European Symposium on Time Series Prediction (ESTSP), pp. 149–158 (2008).

[40] J.G. Milton, “Time Delays and the Control of Biological Systems: An Overview,” IFAC-PapersOnLine, 48(12), pp. 87-92 (2015).

[41] A. Namajūnas, K. Pyragas, and A. Tamaševičius, "An Electronic Analog of the Mackey-Glass System," Physics Letters A, 201(1), pp. 42-46 (1995).

[42] M.C. Soriano, S. Ortin, L. Keuninckx, L. Appeltant, J. Danckaert, L. Pesquera and G. van der Sande, “Delay-Based Reservoir Computing: Noise Effects in a Combined Analog and Digital Implementation,” IEEE Transactions on Neural Networks and Learning Systems, 26(2), pp. 388-393 (2015).

[43] C. Koch and I. Segev, “The Role of Single Neurons in Information Processing,” Nature Neuroscience Supplement, 3, pp. 1171-1177 (2000).

[44] C.L. Giles and T. Maxwell. “Learning, Invariance, and Generalization in High-Order Neural Networks,” Applied Optics, 26(23), pp. 4972-4978 (1987).

[45] M. Mackey and L. Glass, “Oscillation and Chaos in Physiological Control Systems,” Science, 197(4300), pp. 287-289 (1977).

[46] J. Li, C. Zhao, K. Hamedani and Y. Yi, "Analog Hardware Implementation of Spike-Based Delayed Feedback Reservoir Computing System," IEEE International Joint Conference on Neural Networks (IJCNN), pp. 3439-3446 (2017).

[47] M. Tateno and A. Uchida, "Nonlinear Dynamics and Chaos Synchronization in Mackey-Glass Electronic Circuits with Multiple Time-Delayed Feedback," Nonlinear Theory and Its Applications, IEICE 3(2), pp. 155-164 (2012).

[48] S. Haykin, **Neural Networks: A Comprehensive Foundation**, 2nd Edition, Prentice Hall, Upper Saddle River, NJ (1998).

[49] J. Li, L. Liu, C. Zhao, K. Hamedani, R. Atat, and Y. Yi, "Enabling Sustainable Cyber Physical Security Systems through Neuromorphic Computing," IEEE Transactions on Sustainable Computing, PP(99), (2017).

[50] P. Amil, C. Cabeza and A.C. Marti, “Exact Discrete-Time Implementation of the Mackey–Glass Delayed Model,” IEEE Transactions on Circuits and Systems II: Express Briefs, 62(7), pp. 681-685 (2015).

[51] Analog Devices, “Low Cost Analog Multiplier, AD633 datasheet,” (2015) (Revised Feb. 2017).

[52] Analog Devices, “Precision, Low Cost, High Speed BiFET Dual Op Amp, AD712 datasheet,” (2010) (Revised Feb. 2017).

[53] W. Himmelbauer and A.G. Andreou, "Log-Domain Circuits in Subthreshold MOS," IEEE Proceedings of the 40th Midwest Symposium on Circuits and Systems, 1, pp. 26-30 (1997).

[54] K. Ueno, T. Hirose, T. Asai and Y. Amemiya, "A 1-μW 600-ppm /°C Current Reference Circuit Consisting of Subthreshold CMOS Circuits," IEEE Transactions on Circuits and Systems II: Express Briefs, 57(9), pp. 681-685 (2010).

PUBLICATIONS

[1] C. Zhao, B.T. Wysocki, Y. Liu, C.D. Thiem, N.R. McDonald, and Y. Yi, "Spike-Time-Dependent Encoding for Neuromorphic Processors," ACM Journal on Emerging Technologies in Computing Systems (JETC), 12(3), pp. 23-46 (2015).

[2] C. Zhao, J. Li, and Y. Yi, "Making Neural Encoding Robust and Energy-Efficient: An Advanced Analog Temporal Encoder for Brain-Inspired Computing Systems," IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp. 1-6 (2016).

[3] C. Zhao, B. Wysocki, C. Thiem, N. McDonald, J. Li, and Y. Yi, "Energy Efficient Spiking Temporal Encoder Design for Neuromorphic Computing Systems," IEEE Transactions on Multi-Scale Computing Systems (TMSCS), 2(4), pp. 265-276 (2016).

[4] C. Zhao, J. Li, L. Liu, L.S. Koutha, J. Liu and Y. Yi, "Novel Spike Based Reservoir Node Design with High Performance Spike Delay Loop," Proceedings of the 3rd ACM International Conference on Nanoscale Computing and Communication, p. 14 (2016).

[5] C. Zhao, J. Li, and Y. Yi, "Analog Spiking Temporal Encoder with Inter-spike Intervals with Verification and Recovery Scheme for Neuromorphic Computing Systems," Proceedings of IEEE International Symposium on Quality Electronic Design (ISQED) (2017).

[6] C. Zhao, Y. Yi, J. Li, and L. Liu, "Inter-Spike Intervals (ISI) based Analog Spike-Time-Dependent Encoder for Neuromorphic Processors," IEEE Transactions on Very Large Scale Integration Systems (2017).

[7] J. Li, R. Atat, L. Liu, and Y. Yi "Enabling Sustainable Cyber Physical Security Systems through Neuromorphic Computing," IEEE Transactions on Sustainable Computing (2017).

MEETINGS AND PRESENTATIONS

[1] IEEE/ACM Design Automation Conference (DAC) Work-in-Progress Session in Jun. 8th, 2016.

[2] Air Force Research Lab kick off meeting in Jun 23rd, 2016.

[3] “Computationally Efficient Temporal Neural Coding for Neuromorphic Computing,” Computing & Communications Division, Air Force Research Lab, Rome, NY, Aug. 2016.

[4] "Building Brain-Like Computers – Small, Cool, and Robust: A Novel Paradigm for Analog Neuron Circuit Design," Iowa State University (ISU), Ames, IA, Sept. 2016.

[5] “Making Neural Encoding Robust and Energy Efficient: an Advanced Analog Temporal Encoder for Brain-inspired Computing Systems,” IEEE/ACM International Conference on Computer Aided Design (ICCAD), Austin, TX, Nov. 2016.

[6] “Brain-Inspired Computing: Grand Challenges, Hardware Designs, and Emerging Applications,” Virginia Tech, Blacksburg, VA, Mar. 2016.

LIST OF SYMBOLS, ABBREVIATIONS, AND ACRONYMS

ADCs	Analog-to-Digital Converters
AFRL	Air Force Research Lab
CLK	Clock
CMOS	Complementary Metal-Oxide-Semiconductor
DACs	Digital-to-Analog Converters
DDE	Delay Differential Equation
DFR	Delayed Feedback Reservoir
EDA	Electronic Design Automation
FN	Fitzhugh-Nagumo
FPGA	Field-programmable Gate Array
HH	Hodgkin-Huxley
HPC	High-performance Computing
IC	Integrated Circuit
IF	Integrate-and-Fire
ISI	Inter-Spike Intervals
KU	University of Kansas
LIF	Leaky Integrate-and-Fire
MG	Mackey-Glass
Op-AMPs	Operational Amplifiers
PCB	Printed Circuit Board
PTSG	Pseudorandom Time Series Generator
RF	Radio Frequency
RNN	Recurrent Neural Network
SNR	Signal-to-Noise Ratio
SPICE	Simulation Program with Integrated Circuit Emphasis
SWAP	Size, Weight, and Power
TTFS	Time-to-First-Spike