5-2014

An Analog VLSI Deep Machine Learning Implementation

Junjie Lu
University of Tennessee - Knoxville, jlu9@vols.utk.edu

To the Graduate Council:

I am submitting herewith a dissertation written by Junjie Lu entitled "An Analog VLSI Deep Machine Learning Implementation." I have examined the final electronic copy of this dissertation for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, with a major in Electrical Engineering.

Jeremy Holleman, Major Professor

We have read this dissertation and recommend its acceptance:

Benjamin J. Blalock, Itamar Arel, Xiaopeng Zhao

Accepted for the Council:

Carolyn R. Hodges

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

An Analog VLSI Deep Machine Learning Implementation

A Dissertation Presented for the

Doctor of Philosophy

Degree

The University of Tennessee, Knoxville

Junjie Lu

May 2014

Acknowledgement

I would like to express my sincere gratitude to my advisor, Dr. Jeremy Holleman, for his support, guidance and encouragement. His profound knowledge and rigorous attitude toward research inspires me to grow and will benefit me in my future professional and personal life.

I am also deeply grateful to Dr. Benjamin J. Blalock, Dr. Itamar Arel and Dr. Xiaopeng Zhao for serving as my Ph.D. committee member. Their valuable suggestions help me to improve my research and dissertation.

I would like to thank Dr. Itamar Arel and Mr. Steven Young for their great help and support in the analog machine learning project. Their expertise in machine learning is essential to this project from architecture definition to testing and data processing.

I would like to thank my colleagues in ISiS lab at the University of Tennessee, Mr. Tan Yang and Mr. M. Shahriar Jahan, for their help and friendship.

Last but also the most important, I offer my deepest gratitude and love to my parents, Minghua Lu and Huijun Wang, and my wife, Yang Xue, for their unconditional love, support and confidence in me.

Abstract

Machine learning systems provide automated data processing and see a wide range of applications. Direct processing of raw high-dimensional data such as images and videos by machine learning systems is impractical both due to prohibitive power consumption and the "curse of dimensionality," which makes learning tasks exponentially more difficult as dimension increases. Deep machine learning (DML) mimics the hierarchical presentation of information in the human brain to achieve robust automated feature extraction, reducing the dimension of such data. However, the computational complexity of DML systems limits large-scale implementations in standard digital computers. Custom analog signal processing (ASP) can yield much higher energy efficiency than digital signal processing (DSP), presenting a means of overcoming these limitations.

The purpose of this work is to develop an analog implementation of DML system.

First, an analog memory is proposed as an essential component of the learning systems. It uses the charge trapped on the floating gate to store analog value in a non-volatile way. The memory is compatible with standard digital CMOS process and allows random-accessible bi-directional updates without the need for on-chip charge pump or high voltage switch.

Second, architecture and circuits are developed to realize an online k-means clustering algorithm in analog signal processing. It achieves automatic recognition of underlying data pattern and online extraction of data statistical parameters. This unsupervised learning system constitutes the computation node in the deep machine learning hierarchy.

Third, a 3-layer, 7-node analog deep machine learning engine is designed featuring online unsupervised trainability and non-volatile floating-gate analog storage. It utilizes massively parallel reconfigurable current-mode analog architecture to realize efficient computation. And

algorithm-level feedback is leveraged to provide robustness to circuit imperfections in analog signal processing. At a processing speed of 8300 input vectors per second, it achieves 1×10^{12} operation per second per Watt of peak energy efficiency.

In addition, an ultra-low-power tunable bump circuit is presented to provide similarity measures in analog signal processing. It incorporates a novel wide-input-range tunable pseudo-differential transconductor. The circuit demonstrates tunability of bump center, width and height with a power consumption significantly lower than previous works.

Keywords: analog signal processing, deep machine learning, floating gate memory, current mode computation, k-means clustering, power efficiency

Table of Contents

List of Tables

List of Figures

Chapter 1 Introduction

This chapter introduces the background and motivation of this work. It first discusses some basic ideas of machine learning and deep machine learning systems. Then the advantages of analog signal processing are analyzed, justifying the purpose of the analog deep machine learning implementation. The structure and organization of the dissertation is given in the last part.

1.1 Introduction to Machine Learning

1.1.1 Machine Learning: Concepts and Applications

Learning covers a broad range of activity and process and therefore is difficult to define precisely. In general, it involves acquiring new, or modifying and reinforcing existing knowledge, behaviors, skills, values, or preferences and may involve synthesizing different types of information [1]. Learning is first studied as a subject of psychologists and zoologists on humans and animals. And it is arguable that many techniques in machine learning are derived from the learning process of human or animals.

Machine learning is generally concerned with a machine that automatically changes its structure, program, or data based on its inputs or in response to external information to improve its performance. The "changes" might be either enhancements to already performing systems or synthesis of new functions or systems.

In the past, machines are programmed to perform a certain task in the first place. The reasons behind the need for a "learning machine" are manifold.

First, the environments in which the machines are used are often hard to define at the time of programming and changes over time. Machine learning methods can be used for on-the-job improvement of existing design and adaptation to a changing environment, therefore reduce the need for constant redesign.

Second, machine learning provides means of automated data analysis, which is especially important in the face of the deluge of data in our era. It is possible that hidden among large piles of data are important relationships and correlations. For example, Wal-Mart handles more than 1M transactions per hour and has databases containing more than 2.5 petabytes (2.5×10^{15}) of information [2]. From it, a machine learning algorithm can extract purchase patterns of people from different demographics profiles and make customized buying recommendation to them [3]. Machine learning methods used to extract these relationships are called "data mining".

Another reason is that new knowledge about tasks is constantly being discovered. There is a constant stream of new events in the world and it is impractical to continuously redesign the systems to accommodate new knowledge. However, machine learning methods are able to keep track of these new trends. Since the methods are data-driven, the learning-based algorithms are often more accurate than the stationary algorithm when facing the ever-changing world.

Apart from the above mentioned reasons, there are many more reasons why machine learning has become a heated area of research in recent years. Moreover, it is not merely a research topic, but has penetrated to the people's lives and become a powerful and indispensable tool in a wide variety of applications:

- To predict if patient will respond to particular drug/therapy based on microarray profiles in bioinformatics
- To categorize text to filter out spam emails.

- In banking and credit card institution to detect fraud.
- Optical character recognition
- Machine vision, face detection and recognition
- Natural language processing in automatic translation
- Market segmentation
- Robot control
- Classification of stars and galaxies
- Weather or stock market price forecast
- Electric power load prediction

1.1.2 Three Types of Machine Learning

Machine learning is usually divided into three main types [2].

In the *supervised learning* approach, an output or label is given to each input in the training data set, and the machine learning system learns the mapping from inputs to outputs. The simplest training input can be a multi-dimensional vector of numbers, representing the feature of the data being learned. In general, however, the input can have a complex structure, representing an image, a sentence, a time sequence, etc. The output of the system can be either a categorical label or a real-valued variable. When the output is a label, the problem is referred to as classification or pattern recognition. And when the output is a real-valued scalar, the problem is called regression.

The second type of machine learning is the *unsupervised learning*, where only the inputs are given and the learning system finds underlying patterns in the data. The problem of unsupervised learning is less well-defined compared to supervised learning, because the system is free to look

for any patterns, and there is no obvious error metric to correct the current perception. However, it is arguably more typical of human and animal learning, because we got most of our knowledge without being told what the right answers are. Unsupervised learning is also more widely applicable because it does not require a human expert to manually label the data.

There is a third type of machine learning, known as *reinforcement learning*. The machine learns how to act or behave based on occasional external signals. In some applications, the output of the system is a sequence of actions. The machine is given occasional reward or punishment signals based on the goodness of the actions, and the goal is to learn how to act or behave to maximize the award and minimize the punishment. Reinforced learning is employed in applications where a single move is not so important as the rule or policy of the behavior, for example, game playing or robot navigating.

1.1.3 DeSTIN - A Deep Learning Architecture

1.1.3.1 The Curse of Dimensionality

A machine learning system usually processes observations in a multi-dimensional space. When the dimension of the observations is large, such as that from an image or video, a phenomenon called “curse of dimensionality” [4] arises. This phenomenon stems from the fact that as the dimensionality increases, the volume of the space increases exponentially and as a result, the available data become sparse. This sparsity reduces the predictive power of machine learning systems. In order to obtain a statistical sound and reliable result, the amount of data and computational power needed to support the result often grows exponentially with the dimensionality.

1.1.3.2 Deep Machine Learning

When dealing with high dimensional data such as images or videos, it is often necessary to pre-process the data to reduce its dimensionality to what can be efficiently processed, while still preserving the "essence" of the data. Such dimensionality reduction schemes are often referred to as feature extraction techniques.

The most effective feature extraction engine we know might be our brain. The human brain can process information with an efficiency and robustness that no machine can compare with. They are exposed to a sea of sensory data every second and able to capture the critical aspects of them in a way that allows for future use in a concise manner. Therefore, mimicking the performance of the human brain has been a core goal and challenge in machine learning research. Recent neuroscience findings have provided insight into information representation in the human brain. One of the key findings has been that the sensory signals propagate through a complex hierarchy of modules that, over time, learn to represent observations based on the regularities they exhibit. This discovery motivated the emergence of the subfield of deep machine learning, which focuses on computational models for information representation that exhibit similar characteristics to that of the neocortex [5].

1.1.3.3 Deep Spatiotemporal Inference Network (DeSTIN)

The deep learning architecture adopted in this work is based on the Deep Spatiotemporal Inference Network (DeSTIN) architecture, first introduced in [6]. DeSTIN consists of multiple instantiations of identical functional unit called cortical circuits (nodes); each node is a parameterized models which learns by means of an unsupervised learning process. These nodes are arranged in layers and each node is assigned children nodes from the layer below and a parent node from the layer above as shown in Figure 1-1. Nodes at the lowest layer receive raw sensory data while nodes at all other layers receive the belief states, or outputs, from their children nodes as input. Each node attempts to capture the salient spatiotemporal regularities contained in its input and continuously update a belief state meant to characterize the input and the sequences thereof. The beliefs formed throughout the architecture can then be used as rich

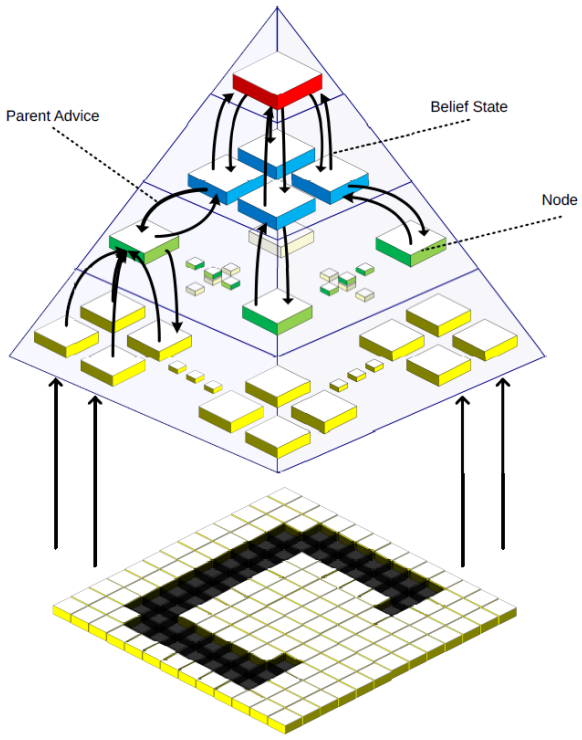


Figure 1-1: The DeSTIN hierarchical architecture [6].

features for a classier that can be trained using supervised learning. Beliefs extracted from the lower layers will characterize local features and beliefs from higher layers will characterize global features. Thus, DeSTIN can be viewed as an unsupervised feature extraction engine that forms features from data based on regularities it observes. In this framework, a common cortical circuit populates the entire hierarchy, and each of these nodes operates independently and in parallel to all other nodes. This solution is not constrained to a layer-by-layer training procedure, making it highly attractive for implementation on parallel processing platforms. Its simplicity and repetitive structure facilitates parallel processing platforms and straightforward training [5].

1.2 Analog Deep Machine Learning Engine - the Motivation

Deep layered architectures offer excellent performance attributes. However, the computation requirements involved grow dramatically as the dimensionality of the input space increases. Compositional deep layered architectures compose multiple instantiations of a common cell and the computation is performed concurrently. In CPU based platforms, however, processing is performed sequentially, thereby greatly increasing execution time [7]. Therefore, many recent efforts research focus on implementing DML systems on GPUs. While GPUs have advantages over CPU-based realizations in computation time and cost/performance ratio, they are power hungry, making such schemes impractical in energy-constraint environments and limiting the scale of these systems.

Custom analog circuitry presents a means of overcoming the limitation of digital VLSI technology. By fully leveraging the computational power of transistors, exploiting the inherent tolerance to inaccuracies of the learning algorithm and performing computation in a slow but massively parallel fashion, the proposed analog deep machine learning engine promises to

largely improve the power efficiency of digital DML systems to take full advantage the scaling potential of DeSTIN.

1.2.1 Analog versus Digital - the Neuromorphic Arguments

It is meaningful to compare the energy efficiency between the brain and the digital computer. The human brain is estimated to perform roughly 10^{15} synapse operations at about 10 impulse/sec. The total energy consumption of our brain is about 25 watts [8]. This yields an energy consumption of about 10^{15} operation per joule. Today's super computer can perform 8.2 billion megaflops with a power consumption of 9.9 million watts, enough to power 10000 houses [9]. Its energy efficiency is thus about 8.3×10^{9} operation per joule, more than 6 orders of magnitude lower than the human brain.

The great discrepancy in energy efficiencies of neurobiology and electronics suggests that there are fundamental differences in the ways they do computation. One significant difference is the state variables they use. Digital computers employ only two state variables while ignoring all the values in the middle to achieve noise immunity at the expense of dynamic range. The neurons, on the other hand, present and process information in analog domain: the firing rates are continuous variables; and each neuron resembles a lossy integrator with the leakage controlled by fluctuating number of ion channels. Analog signaling allows a single wire to carry multi-bit information, therefore largely increasing power and area efficiency. It also interfaces naturally with the analog computation primitives, as discussed below.

The other important trait leading to enormous efficiency of neurological systems is their clever exploitation of the physics they are built with. The nervous system does basic aggregation of information using the conservation of charge. Kirchhoff's current law implements current summing, and this current is integrated with respect to the time by the node capacitance. In the

neuron tissue, ions are in thermal equilibrium and their energies are Boltzmann distributed. If an energy barrier exists and is modulated with the applied voltage, the current through the barrier will be an exponential function of that applied voltage [10]. This principle is used to create active devices and compute complex nonlinear functions in neural computation. The principle of operation of the transistors in the integrated circuit can be surprisingly similar to that of the nervous operation: in weak inversion, the energy barrier for the carrier to travel from source to drain is modulated by the gate voltage; therefore the drain current is exponentially dependent on the gate voltage. However, digital computers completely disregard these inherent computation primitives in the device physics, and only use two extremes of the operation points: on and off states, therefore represent the information with 0 or 1 only. This also confines us to a set of very limited elementary operations: NOT, NOR, OR or their equivalences. This is in contrast with how the neuron does computation and can cause a factor of 10^4 efficiency penalty [10]. Analog circuits provide a means to reclaim this efficiency loss: by exploiting the computational primitives inherent in the device and physics like our brains do, operations can be naturally carried out with much higher efficiency.

The scaling of CMOS technology reduces the power of digital systems. However, this scaling trend is slowing down and seeing its end due to physical limitations such as the thickness of gate oxide [11]. In addition, the power in digital system does not scale as fast as the feature size due to the saturation of threshold and supply voltage scaling in order to keep down subthreshold leakage [12]. On the other hand, analog system can also benefit from the technology scaling. The improved subthreshold slop in FinFET improves the transconductance efficiency in weak inversion, and improves the computation efficiency [13]. And the reduced wiring parasitic capacitance improves computation throughput.

1.2.2 Analog Advantages

Analog signal processing makes use of the physics of the devices: physical relations of transistors, capacitors, resistors, Kirchhoff's current and voltage laws and so on. It also represents the information with multi-bit encoding. Therefore it can be far more efficient than digital signal processing. For example, addition of two numbers takes only one wire in analog circuit by using Kirchhoff's current law, whereas it takes about 240 transistors in static CMOS digital circuits to implement an 8-bit adder. Similarly, an 8-bit multiplication in the analog domain using current-mode operation takes 4 to 8 transistors, whereas a parallel 8-bit digital multiplier has approximately 3000 transistors [14]. Another example is the bump circuit, shown in Figure 1-2, which computes the derivative of *tanh(·)*. The bump circuit simultaneously provides a measure of similarity between two inputs and the *tanh(·)* of their difference. The bump function can also be used as a probability distribution, as it peaks with zero difference and saturates to zero for large differences. The bump circuit illustrates the power advantage that analog computation holds over digital methods. A bump circuit, biased at 200 fA, can evaluate the similarity between a stored value at about 200 observations per second, according to simulations using 0.24 μm transistors. A single inverter consumes about four times that much when switching at 200 Hz, and about half that much statically, without switching at all. To perform a comparable computation digitally would require dozens more transistors and one to two orders of magnitude more current [15].

1.2.3 Inaccuracies in Analog Computation

However, the analog computation does have some disadvantages when compared to digital, and they are caused by the very same reasons that make it more efficient. The analog systems are much more sensitive to noise and offset than digital systems. While the digital systems use restoring logic at every computational step to obtain good noise immunity, the use of continuous signal variables prevents analog systems from having any restoring mechanism. Thus, the noise accumulation in analog systems becomes severe as the system scales up. It is found in [14] that the cost for precision for analog system increases faster than digital: the power consumption is a polynomial function of the required signal to noise ratio (SNR) in analog system; in digital system, however, it is a logarithm function. Therefore, analog computation is cheaper at low values of accuracy but more expensive at high accuracy.

However, in certain cases the feedback inherent to the learning algorithms naturally compensates for inaccuracies introduced by the analog circuits. Similarly, this lack of accuracy in analog signal processing can also be found in neural computers. The brain is known to be built from noisy, inaccurate neurons. For example, many behavioral responses, such as a fly making a course correction after a disturbance, occur over a period of around 30 ms [16]. Neural signals

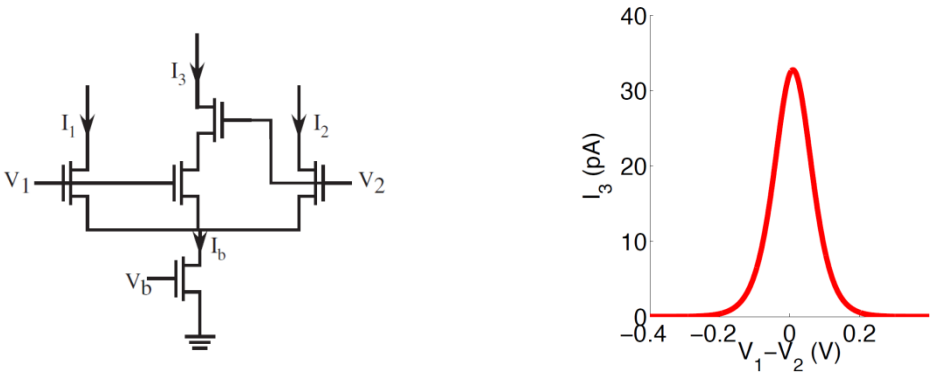


Figure 1-2: Bump circuit, which computes tanh(V1−V2) and its derivative simultaneously [15].

integrated over comparable time windows typically exhibit a signal-noise ratio (SNR) in the range of 1-10 [16], [17], [18], much lower than what can be easily achieved in moderate precision analog electronics. Comparisons between the noise and power tradeoffs in analog and digital circuits and biological systems have also been explored in [14]. The low SNR and outstanding power efficiency of neural systems suggests that relaxed accuracy requirements for electronic computational primitives could allow aggressive optimization for area and power consumption.

1.2.4 Analog versus Digital – Parallel Computation

The power efficiency of a computational system can be expressed by its delay-power product. The delay-power product of a single stage can be approximated to be

$$t_d \cdot P_D = V_{DD} C_P I_D / g_m \tag{1.1}$$

where V_{DD} is the supply voltage, C_P is the equivalent parasitic capacitances associated with the internal nodes, I_D is the current consumption, and the g_m is the equivalent trans-conductance of the transistors. The V_{DD} and C_P can be scaled down with the technology, and I_D / g_m is minimized when the transistors are biased in weak inversion. Therefore (1.1) indicates that an efficient computational system can be built by slow but massively parallel computational elements biased in weak inversion (or sub-threshold).

Subthreshold digital designs are difficult in that the high susceptibility to process variability in subthreshold region causes timing errors [19]. For high performance applications low-threshold devices must be used and leakage becomes a significant problem [20]. In a massively paralleled system, the subthreshold leakage can consume a large portion of total power without any contribution to the computation throughput.

The error in analog systems behaves more benignly than that in digital systems: an error in digital causes the complete loss of information (unless error correction is implemented), while the errors in analog has much smaller magnitude and cause graceful degradation of performance and if static, can be compensated by the feedback inherent to the learning algorithms. Moreover, the leakage is no longer a problem: the subthreshold channel current in analog circuit is used to carry information and perform operation, instead of being deemed as wasted in digital computer.

1.3 Original Contributions

In this work, an analog signal processing system implementing DeSTIN, a state-of-art deep machine learning algorithm is proposed. The original contribution of this work is summarized below:

- Characterized a floating gate device in 0.13 μm standard digital CMOS process.
- Designed and tested a novel floating gate analog memory with random-accessible bidirectional sigmoid updates in 0.13 μm standard digital CMOS.
- Proposed novel architecture and circuits to realize an analog online k-means clustering circuit with non-volatile storage, first reported in the literature.
- Designed an analog deep machine learning engine to implement DeSTIN, first reported in the literature. Proposed techniques to greatly increase power and area efficiency.
- Presented an ultra-low-power tunable bump circuit to provide similarity measures widely applicable in analog signal processing, incorporating a novel wide-input-range tunable pseudo-differential transconductor.

1.4 Dissertation Organization

The remaining chapters of this proposal will cover the design of components, circuits and architectures to implement the analog deep machine learning engine, in a bottom-up way.

Chapter 2 provides the implementation of the analog non-volatile memory, which is an essential component in the learning system.

Chapter 3 describes the design of an analog k-means clustering circuit, the key building block in the deep machine learning engine.

Chapter 4 presents the proposed analog deep machine learning engine, including its architecture and circuit designs. And the techniques to greatly improve the energy and area efficiency are presented.

Chapter 5 develops the ultra-low-power tunable bump circuit, which can have wide application in analog signal processing systems.

Chapter 6 concludes the dissertation and proposes potential future works.

Chapter 2 A Floating-Gate Analog Memory with Random-Accessible Bidirectional Sigmoid Updates

Memory is an essential component in a computation system. Modern digital memory can afford very high read/write speed and density [21]. However, most digital memories are volatile: DRAM requires constant refreshing, and SRAM requires a minimum V_{DD} for state retention. This volatility precludes their use in intermittently powered devices such as those utilizing harvested energy.

Non-volatile digital memories such as flash memory [22] require special process. FRAM (Ferroelectric RAM) is reported to be embeddable using two additional mask steps during conventional CMOS process [23], and has been proven to be commercially viable [24]. Recent researches in this area have proposed other types of memory such as ReRAM (Resistive RAM) [25], and MRAM (Magnetoresistive RAM) [26]. However, these technologies are still new and not commercially available, and all require special processing.

Another major challenge using digital memory in analog signal processing systems is that A/D/A conversion is needed to interface the memories to other circuits. This is especially problematic in distributed-memory architectures, where the A/D/A cannot be shared among the memory cells, and this leads to prohibitive area and power overhead.

In this work, I propose a floating-gate current-output analog memory which interfaces naturally with the current-mode analog computation system, and allows random-accessible control of bidirectional updates, described in [27]. The update scheme avoids the use of charge pump, minimizes interconnection and pin count, and is compatible with standard digital process. The update rule is sigmoid-shaped, which is a smooth, monotonic and bounded function. Implemented in a commercially available 0.13μm single-poly digital CMOS process using thick-

oxide IO FETs, the memory cell achieves small area and low power consumption, and is suitable for integration into systems that exploit the high-density digital logic available in modern CMOS technology.

2.1 Overview of Floating Gate Device

2.1.1 Principles of Operation

Floating gate (FG) device utilizes the charge trapped on the isolated gate to store analog or digital values in a non-volatile way. The cross-section of a typical FG NFET in a bulk CMOS process is shown in Figure 2-1 [28]; note that a double-poly process is used to obtain the control gate. The earliest research on this device can be dated back to the 1960's [29], and the modern EEPROM and Flash memory are both based on FG devices. Due to the excellent insulation from the thermally-grown SiO_2 surrounding the floating gate, the electron trapped on the gate can have a retention time of more than 10 years [30]. And the memory can be programmed by two mechanisms: Fowler–Nordheim tunneling and hot-electron injection.

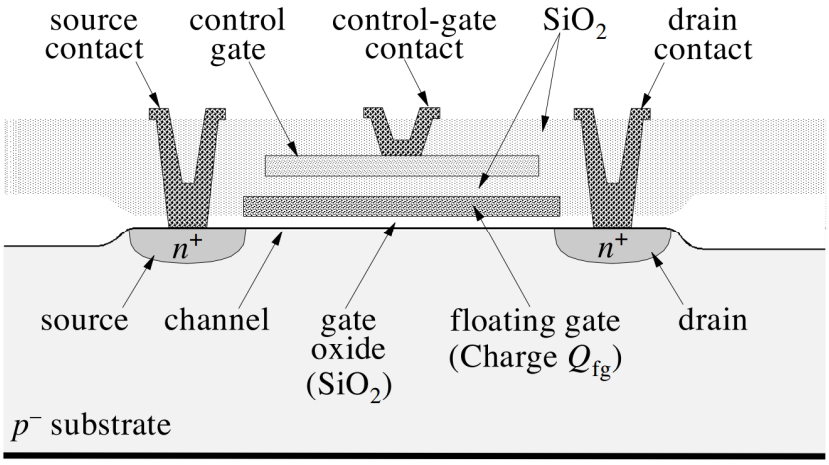


Figure 2-1: Cross-section of a typical FG NFET in a bulk CMOS process [28].

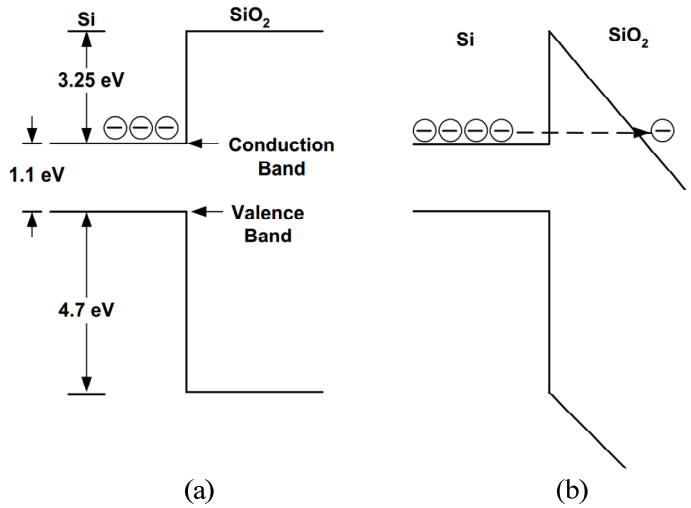


Figure 2-2: Energy band diagram of Si/SiO2 interface (a) with and (b) without applied field [31].

2.1.2 Fowler–Nordheim Tunneling

Fowler–Nordheim (FN) tunneling is used to remove electrons from the floating gate. The potential difference applied across the poly-SiO_2-Si structure reduces the effective thickness of the gate-oxide barrier, facilitating electron tunneling from the floating gate, through the SiO_2 barrier, into the oxide conduction band. This is illustrated by Figure 2-2, showing the energy band diagrams of the Si/SiO2 interface with and without applied field [31]. At sufficiently high field, the width of the barrier becomes small enough for electrons to tunnel through the silicon conduction band into the oxide conduction band. This phenomenon is first described by Fowler and Nordheim in electrons tunneling through the vacuum barrier, and the FN-tunneling is found in SiO_2 in 1969 [32].

2.1.3 Hot Electron Injection

Hot electron injection is used to add electrons to the floating gate. In a PFET, the carrier holes are accelerated by the lateral field applied between its drain and source. Near the drain

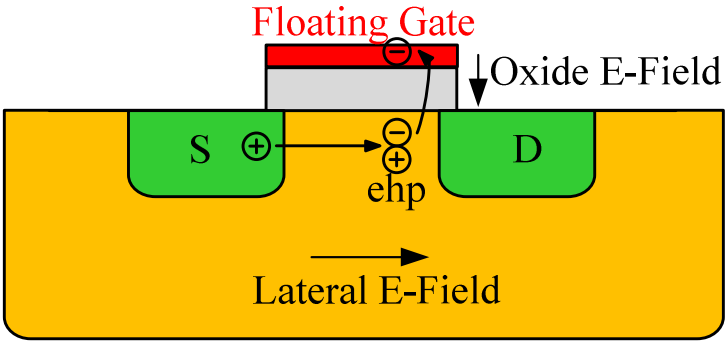


Figure 2-3: Hot electron injection in PFET.

terminal, the kinetic energy of the holes is large enough to liberate electron-hole pairs (ehp) when they collide with the silicon lattice. Electrons scattered upwards with energy larger than 3.2 eV will be able to go over the Si–SiO2 work-function barrier into the oxide conduction band. These electrons are then swept over to the floating gate by the oxide electric field. This process is illustrated in Figure 2-3.

2.2 Literature Review on Floating Gate Analog Memory

Although FG devices are usually associated with digital memories such as EEPROM or Flash memory to store binary values, they are intrinsically an analog device, because the charge on the FG can be modified in a continuous way. A floating-gate analog memory uses the charge trapped on the isolated gate to store analog variables in a non-volatile way. It has been widely used in analog reconfigurable, adaptive and neuromorphic systems, such as electronic potentiometer [33], precision voltage reference [34], offset-trimmed opamp [30], pattern classifier [35], silicon learning networks [36], and adaptive filter [37].

Without direct electrical connections, the stored value of the memory is updated by depositing electrons to the floating gate by hot-electron injection, or removing them by Fowler–Nordheim tunneling. Compared to injection, tunneling selectivity is harder to obtain because it

often involves controlling a high voltage (HV) on chip. Therefore, many previous works [35], [36] use tunneling as the global erase, and injection to program individual memory to its target value. However, in an online adaptive system as this work, a bidirectional update is preferable because the stored values need to vary with the inputs. Previous works have proposed approaches to achieve selective tunneling. In [38], the selected memory is tunneled by pulling up the tunneling voltage and pulling down the control gate voltage simultaneously. This approach requires a number of tunneling control pins equal to the number of rows in the memory array, which is not desirable for large-scale systems. In [33], a HV switch is built with lightly-doped-drain nFETs. This device is not compatible with standard digital processes and consumes static power because it cannot be completely turned off. In [37], a charge pump is used to generate a local HV for the selected memory. A simple charge pump provides limited voltage boost, while a more complex one consumes larger area and/or requires multi-phase clocks.

Another important performance metric of analog memory is the update rule. The dynamic of the single-transistor FG memory [38] leads to exponential and value-dependent update, which, in general, affects the stability of the adaptation [37]. A linear update can be obtained by fixing the FG node voltage during update with a capacitive feedback loop around a differential [33] or single-ended amplifier [37].

2.3 Proposed Floating Gate Analog Memory

2.3.1 Floating-Gate Analog Memory Cell

2.3.1.1 Circuit Description

The schematic of the proposed FG analog memory cell is shown in Figure 2-4. The gate of MP1-MP3 and the top plate of C_f form the FG. The stored charge can be modified by the injection transistor MP2 and the tunneling transistor MP3. The two MUXs at the sources of MP1 and MP2 control the tunneling and injection of the FG, which will be discussed later. The transconductor g_m converts voltage V_{out} to output current I_{out}. *Vref* determines the nominal voltage of V_{out} during operation.

The negative feedback loop comprising the inverting amplifier MP1/MN1 and C_f keeps the FG voltage V_{fg} constant, ensuring a linear update of V_{out}. Tunneling or injection to the FG node changes the charge stored in C_f, therefore changes the output of the amplifier by $\Delta V_{out} = \Delta Q / C_f$.

The transconductor is implemented with a differential pair MN2/MN3 and a cascode current mirror MP4-MP7, depicted in Figure 2-5(a). Biased in deep sub-threshold region, the

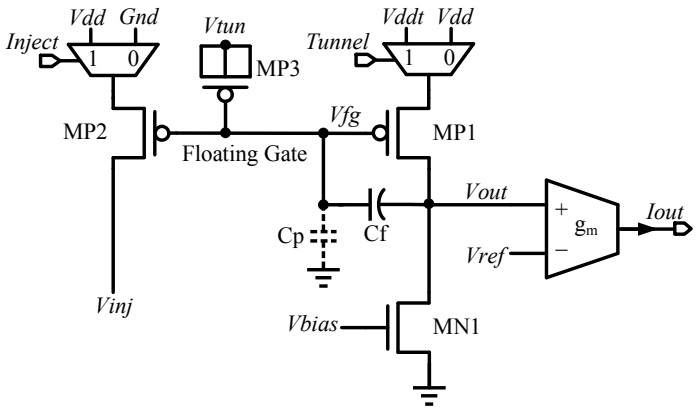
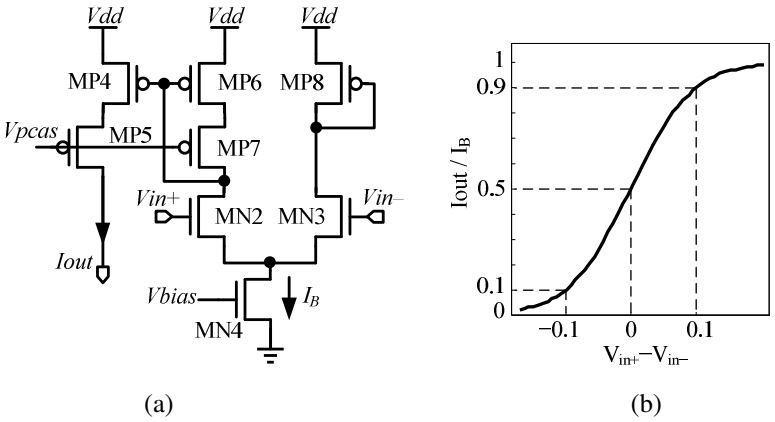


Figure 2-4: Schematic of the proposed floating-gate analog memory cell.



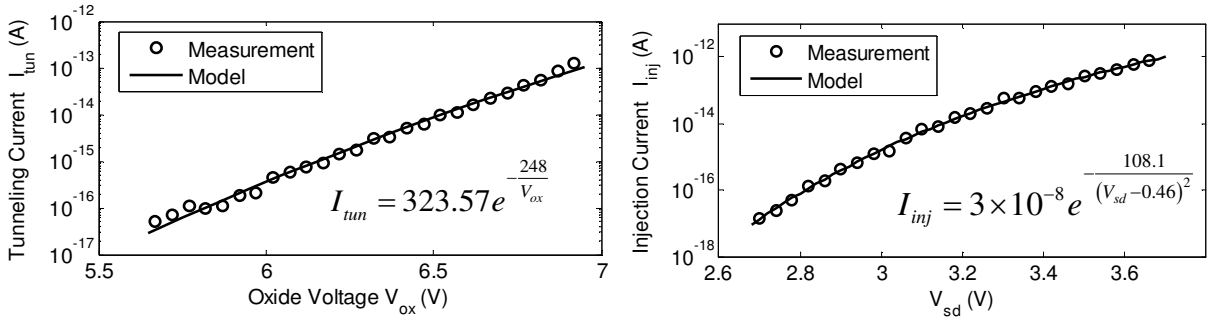
(a) (b)

Figure 2-5: (a) Schematic of the transconductor and (b) its transfer function.

transconductor exhibits a transfer curve resembling a *tanh* function, plotted in Figure 2-5(b). The mild nonlinearity is smooth, monotonic and bounded. From Figure 2-5(b), a ΔV_{in} of 0.2 V is enough to cause a change of I_{out} from $0.1I_B$ to $0.9I_B$, this reduced swing requirement further improves the update linearity, and enables the selective tunneling.

2.3.1.2 Floating Gate Charge Modification Modeling

The proposed analog memory uses Fowler–Nordheim tunneling to remove the electrons from the FG and decrease the memory value. The tunneling current I_{tun} can be expressed by the empirical model [39] as



(a) (b)

Figure 2-6: (a) Tunneling current versus oxide voltage Vox. (b) Injection current versus drain-to-source voltage of the injection transistor.

$$I_{tun} = I_{tun0} \exp\left(-\frac{V_f}{V_{ox}}\right), \tag{2.1}$$

where V_{ox} is the voltage across the tunneling transistor gate oxide, I_{tun0} and V_f are process dependent constants determined by measurements. Figure 2-6(a) shows the measured tunneling current I_{tun} versus the oxide voltage V_{ox}, and the fitted model.

Hot-electron injection is employed to increase the stored value of the memory. The injection current I_{inj} depends on the source current and the drain-to-source voltage of MP2. A simplified empirical model derived from [39] approximates I_{inj} as

$$I_{inj} = \alpha I_s \exp\frac{\beta}{\left(V_{sd} + \delta\right)^2}, \tag{2.2}$$

where I_s is the injection transistor's source current, V_{sd} is the drain-to-source voltage, and α, β, δ are fit constants. In our memory cell, I_s is set by the biasing current and the aspect ratios between MP1 and MP2. Figure 2-6(b) shows the measured I_{inj} versus V_{sd}, and the fitted model.

The extracted models above can be used in the future designs as well as to improve programming convergence, as will be described in Section 2.3.3.

2.3.1.3 Selective and Value-Independent Update Scheme

The proposed tunneling scheme exploits the steep change of tunneling current with regard to V_{ox} to achieve a good isolation between selected and unselected memories. The operation of this scheme can be described by Figure 2-7, showing the memory cell omitting components irrelevant to tunneling process. To show how V_{ox} is changed, typical nodal voltages are annotated. The negative feedback keeps the FG voltage at

$$V_{fg} = Vdd - V_{SG,P} \approx Vdd - 0.4, \tag{2.3}$$

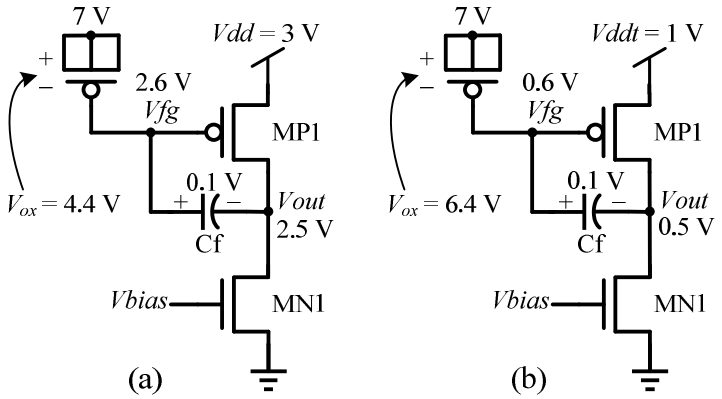
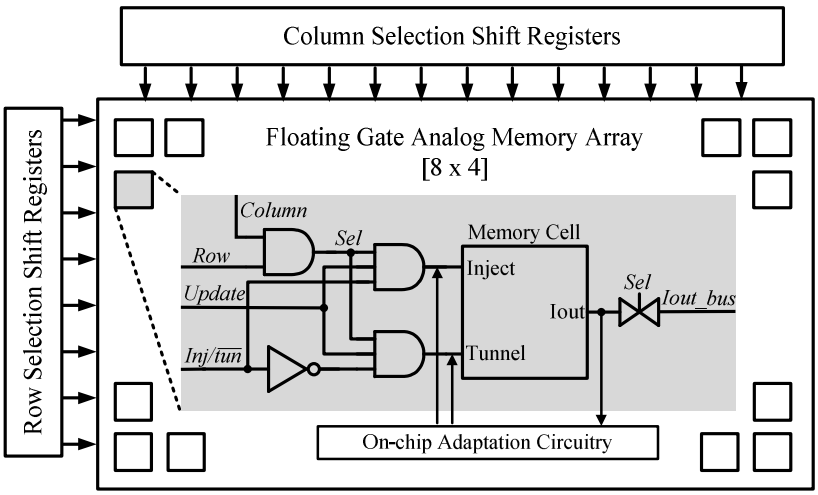


Figure 2-7: Simplified schematics and typical nodal voltages of memory cells (a) not selected. (b) selected for tunneling.

where $V_{SG,P}$ is the source to gate voltage of MP1. Therefore, reducing supply voltage of the selected memory effectively reduces V_{fg} and increases V_{ox}. In our design, the power supply is switched from 3 V *Vdd* to a 1 V *Vddt*, resulting in isolation over 7 orders of magnitude according to (2.1). In practice, the leakage at lower V_{ox} may be degraded by direct tunneling, which is a weaker function of the applied field [40], and parasitic coupling. Isolation of 83.54 dB is observed in measurement. The condition that MN1 stays in saturation during tunneling can be satisfied by choosing a proper V_{ref} and using the proposed transconductor to reduce the V_{out} swing.

Injection selectivity is achieved by switching the source voltage of the injection transistor MP2. The source of MP2 in the unselected memory is connected to ground while the one in the selected cell is connected to *Vdd*, enabling injection. Therefore, the injection is also selective and value-independent.



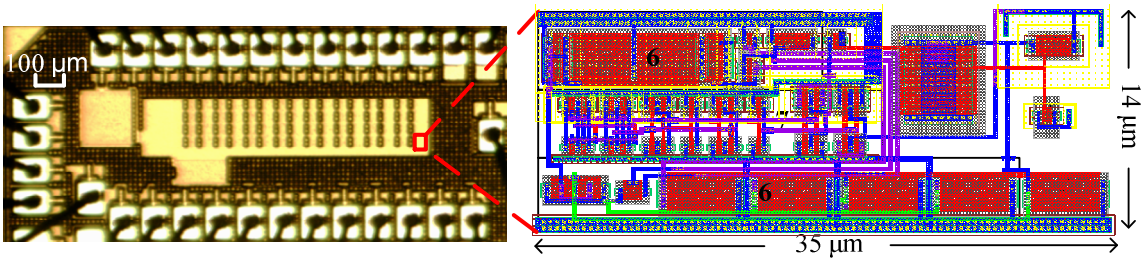
Selected	Update	Inj/$\overline{\text{tun}}$	Mode
No	–	–	On-chip Adaptation
Yes	0	–	Read
Yes	1	0	Tunneling
Yes	1	1	Injection

Figure 2-8: Block diagram of the FG analog memory array, and a table showing control signal settings for different operation modes of the cells.

2.3.2 Floating Gate Memory Array

32 proposed FG analog memory cells are connected to form a memory array. They are organized in two dimensions and can be randomly accessed (selected) for read and write operations by setting both *column* and *row* inputs to high. The block diagram is shown in Figure 2-8 with the cell symbolized. The cells are augmented by digital logics controlling their operation modes. The list of digital control combinations and their corresponding operation modes is shown in Figure 2-8.

Once selected, a transmission gate connects the output of that cell to off-chip through *Iout_bus* for read-out during programming. The $Inj/\overline{tun}$ signal sets the direction of memory writing. The magnitude of writing is controlled by the pulse width of *Update* signal. When a cell is not selected, it maintains its value and can be read or written by on-chip circuits to implement



(a) (b)

Figure 2-9: (a) Chip micrograph of the memory array together with on-chip adaptation circuitry and (b) layout view of a single memory cell.

adaptive algorithms. The proposed architecture is scalable because all signals and inter-connections are shared among the cells, and the pin count does not increase with the size of the array.

2.3.3 Measurement Results

The proposed FG memory array has been fabricated in a 0.13µm single-poly standard digital CMOS process using thick-oxide IO FETs. The die micrograph is shown in Figure 2-9. Due to extensive metal fills in this process, details of the circuits cannot be seen. So the Virtuoso layout view is also presented.

The area of a single memory cell is 35×14 μm^2. It operates at 3 V power supply and

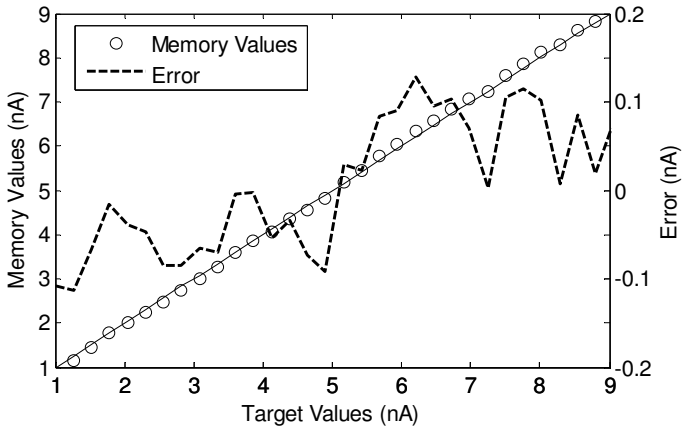


Figure 2-10: Analog memory programming accuracy of 30 linearly spaced values.

consumes 15 nA with an output range of 0-10 nA. The biasing current is tunable and allows the designer to balance between range, speed and power consumption.

The test setup is built around a National Instruments data acquisition (DAQ) card and a host PC. The programming procedure is controlled by a Labview program in the host PC and based on the models in Section 2.3.1.2 to achieve fast convergence. The average number of iterations required to achieve a 0.5% error is 5-6. Figure 2-10 demonstrates 30 memory cells programmed to values between 1 and 9 nA. The standard deviation of programming error is 76 pA, limited by the external circuits and equipment, indicating a 7-bit programming resolution. The memory output noise is 20.5 pA_{rms} over 10 KHz bandwidth from simulation, indicating a 53.8 dB dynamic range.

To show the update rule, a memory is first ramped up then ramped down with fixed pulse width of 1 ms. The corresponding V_{out} and I_{out} are plotted in Figure 2-11. Both injection and tunneling is linear to V_{out}, and the current output has a smooth sigmoid update rule. During the same test, the stored values of the other 31 unselected cells are monitored to measure the writing crosstalk. The crosstalk from the injection and tunneling of the selected cell to the unselected ones are plotted in Figure 2-12. There is no observable injection crosstalk. Tunneling crosstalk is

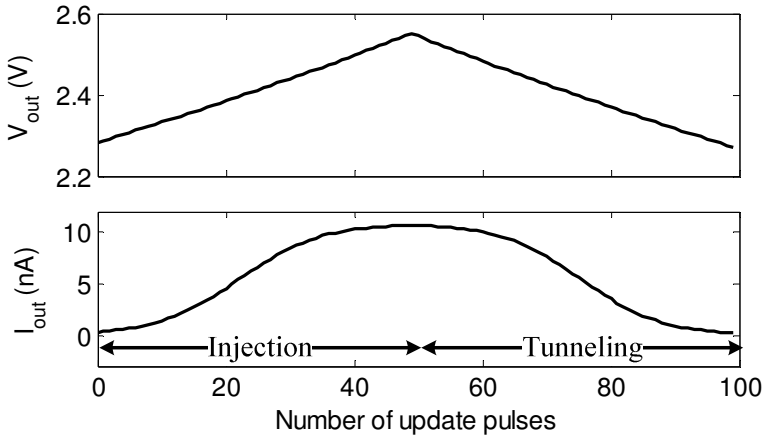


Figure 2-11: Ramping of the memory value, showing the update rules.

very small, comparable to the noise floor of the measurement system. By averaging the values among 31 cells, a 471 fA tunneling crosstalk is approximated with a 10 nA writing magnitude in the selected cell, corresponding to an 86.5 dB isolation. The retention of the proposed memory cells were tested by continuously monitoring their outputs for 2 days at room temperature. During these 48 hours, no observable leakage was seen after the initial relaxation period during which the electrons trapped in the oxide are released. This is sufficient for general adaptive and neuromorphic applications. The measured performance is summarized in Table I.

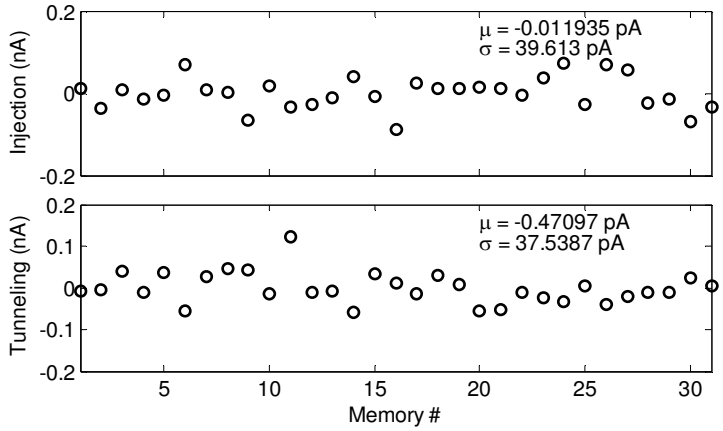


Figure 2-12: Crosstalk among the 31 unselected cells when a selected cell is injected or tunneled with a magnitude of 10 nA.

TABLE I. PERFORMANCES SUMMARY OF THE FLOATING GATE MEMORY

Parameter	*Value*
Technology	1P8M 0.13-μm CMOS
Area	35×14 μm^2
Power supply	3 V
Power consumption	45 nW
Output range	0 - 10 nA
Programming resolution	7 bits
Dynamic range	53.8 dB
Programming isolation	86.5 dB

Chapter 3 An Analog Online Clustering Circuit in 0.13 μm CMOS

As described in Chapter 1, the nodes in the DML architecture learn by an unsupervised online clustering algorithm. In this chapter, an analog online clustering circuit is presented [41]. It is capable of inferring the underlying patterns and extracting the statistical parameters from the input vectors, as well as providing measures of similarity based on both mean and variance.

An 8-dimension 4-centroid prototype was fabricated in a 0.13 μm standard CMOS process. Measurement results demonstrate vector classification at 16 kHz, and unsupervised online clustering at 4 kHz with a power consumption of 15 μW.

3.1 Introduction and Literature Review of Clustering Circuit

This chapter describes the implementation of an analog signal processing (ASP) system realizing an online k-means clustering algorithm, widely-used in feature extraction, pattern recognition, data compression, and other applications. It infers the underlying data patterns by capturing the regularity of it [42]. A vector quantizer (VQ) searches a set of stored centroids (templates) for the one nearest to the input vector. The proposed system enhances VQ with online construction and adaptation of templates to yield optimal performance under changing input statistics. While this algorithm is expensive in digital domain, it can be realized in ASP with relatively low cost in terms of power and area by exploiting the inherent computational primitives [14]. Analog or mixed-mode VQ processors have been developed in [43], [44]. The lack of learning capability requires explicit programming. VQs with learning capability are presented in [45], [46], the centroids are stored in volatile capacitors or digital memories. The

non-volatile memory used in this work enables intermittent powering, and the fully analog operation avoids the power and area overhead of internal A/D/A conversion.

I present a novel analog online k-means clustering circuit which performs unsupervised learning in real time. Parameters are stored in non-volatile analog memories compatible with standard digital CMOS. Confidence scores are constructed and passed to the higher hierarchical layers in the deep learning architecture. The architecture and circuit design are optimized for scalable low-power fully-autonomous computation applications.

3.2 Architecture and Algorithm

The architecture of the clustering circuit is shown in Figure 3-1. The signal processing is implemented in current mode to allow efficient arithmetic operations and wide linear range. The

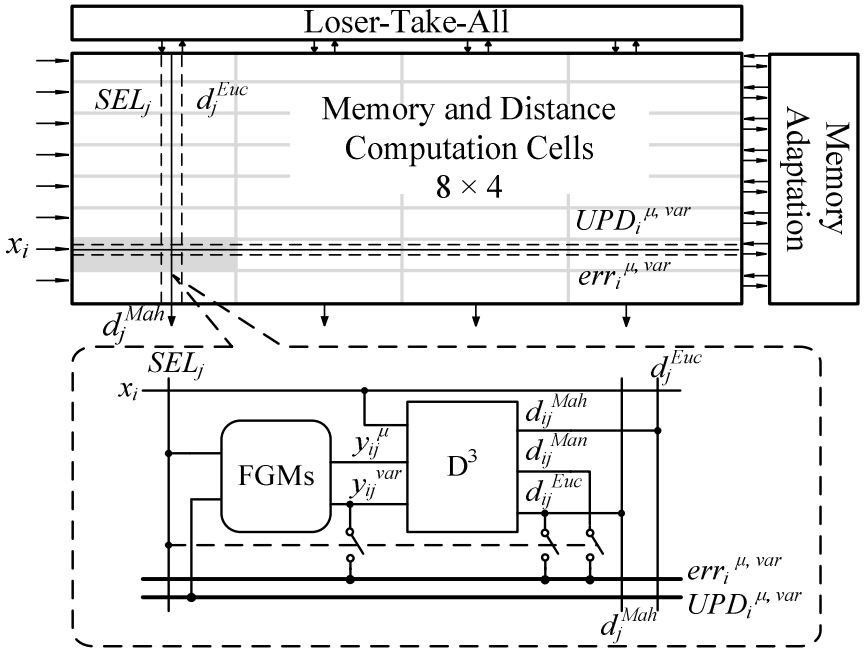


Figure 3-1: The architecture of the proposed analog online clustering circuit, with the details of the memory and distance computation cell.

core of the prototype is an array of memory and distance computation cells (MDCs). The 4 columns form 4 centroids, each with 8 dimensions. The MDC consists of two analog memories (FGMs) and a distance computation block (D^3). The FGM stores the centroid mean and variance, and is accessible and programmable from off-chip in test mode. The D^3 block provides 3 distance metrics between the input vector and the local centroid, necessary for different operation modes. Memory adaptation circuits common to each row and loser-take-all circuits common to each column perform memory adaptation and classification, respectively.

The online k-means clustering algorithm is similar to that used in the training of the analog deep learning engine. Therefore, the details of this algorithm will be discussed in Chapter 4.

3.3 Circuit Implementation

3.3.1 Floating-Gate Analog Memory

The design utilizes the proposed floating gate memory for non-volatile analog storage. A detailed description is presented in Chapter 2.

3.3.2 Distance Computation (D^3) Block

The schematic of the D^3 block is depicted in Figure 3-2. The output devices of the preceding stages are shown in grey. A1 is built with a differential pair with current mirror load and the comparator with cascade of single-ended amplifiers. *Vs* biases M8, M9 in saturation. The arrow indicates the translinear loop. The two output currents from preceding stages (centroid mean y_{ij}^{μ} and input x_i) are summed with opposite polarities at its input node, and then rectified by M1-M4 and amplifier A1 to yield the unsigned output. The absolute value circuit is modified from [47]. Improvement is made by introducing a virtual ground using A1 to mitigate the error due to finite

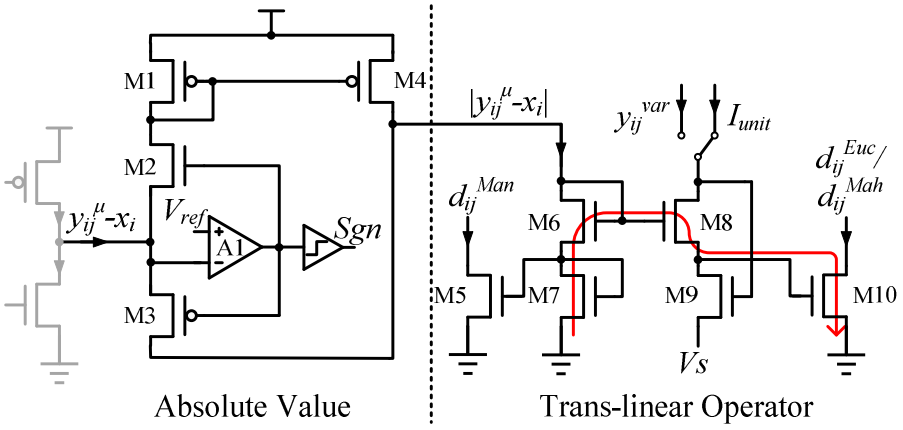


Figure 3-2: The schematic of the D[3] block.

drain resistances, and improve the speed. A comparator provides the polarity of input current, *Sgn*, used in the adaptation phase.

In the translinear operator circuit M5-M10, M5 copies the current from the absolute value circuit to get the Manhattan distance $|y_{ij}^{\mu} - x_i|$, M9 forces the current into the drain of M8 by modulating its source voltage, and M6-8 and M10 form the translinear loop [48]. The translinear circuits exploits the exponential relationship between the drain current and gate to source voltage of subthreshold transistors to implement efficient arithmetic functions. For transistors M6-M8 and M10 in Figure 3-2, their gate to source voltages form a loop, and according to Kirchhoff Voltage Law:

$$V_{GS6} + V_{GS7} = V_{GS8} + V_{GS10}, \tag{3.1}$$

due to the exponential relationship (neglecting body effect for simplicity):

$$I_D = I_{D0} e^{V_{GS}}, \tag{3.2}$$

where I_{D0} is the pre-exponential constant, and assuming that M6-8 and M10 are matched, the drain current of M10 is given by

$$I_{d,M10} = I_{d,M6}^2 / I_{d,M8} = (y_{ij}^{\mu} - x_i)^2 / I_{d,M8}. \tag{3.3}$$

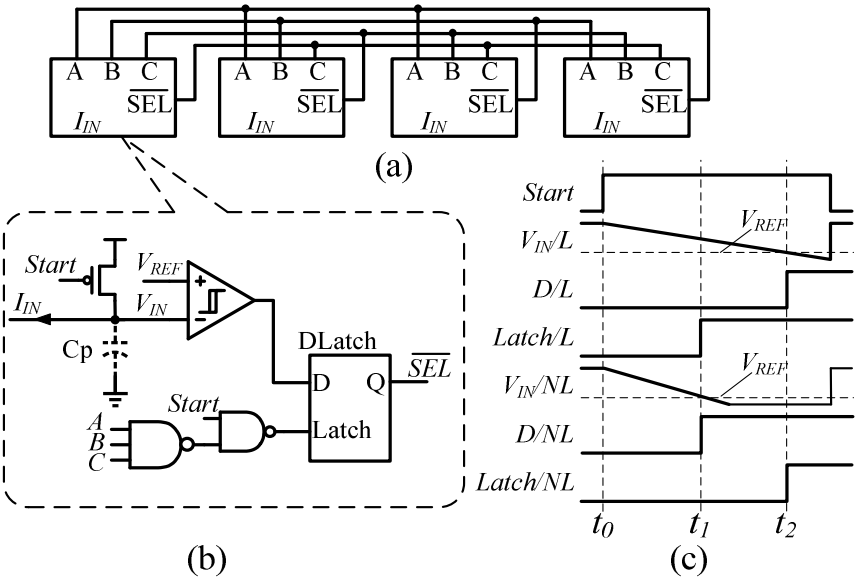


Figure 3-3: The simplified schematic of (a) the LTA network, (b) one cell of the LTA, (c) typical timing diagrams.

The Euclidean distance d_{ij}^{Euc} can be obtained by supplying M8 with a unit current I_{unit}, and the Mahalanobis distance d_{ij}^{Mah} realized by connecting the variance memory output y_{ij}^{var} to M8.

3.3.3 Time-Domain Loser-Take-All (TD-LTA) Circuit

The LTA circuit receives the Euclidean distances d_j^{Euc}, and searches for the centroid with th smallest distance. It consists of 4 LTA cells interconnected as shown in Figure 3-3 (a). The LTA cell shown in Figure 3-3(b) operates in time domain and exploits the dense digital blocks in modern process. The typical timing diagram of the "loser" and a "non-loser" cell is plotted in Figure 3-3 (c), where the signals in the loser cell is suffixed with /L, and non-loser cell with /NL. The capacitor Cp is initially precharged to *Vdd*, and is discharged by the input current when *Start* goes high (t_0). For the loser cell, the threshold crossing of the comparator (t_2) is the latest among the 4 cells, so the data input *D* of its D-latch is low when *Latch* goes high. For the "non-loser" cell, *D* is high when *Latch* goes high (t_1). Therefore, the output of the "loser" is latched to low

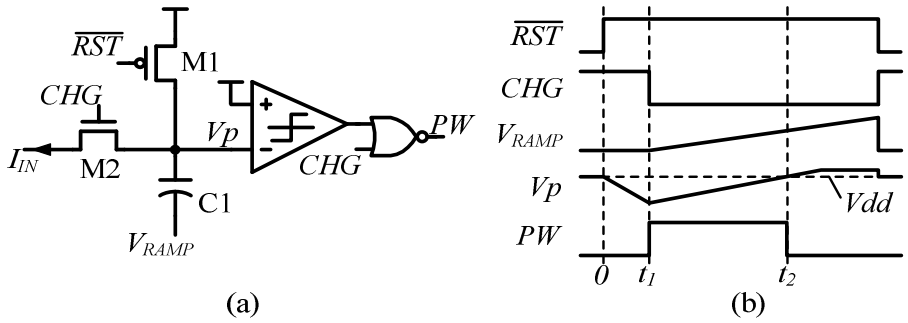


Figure 3-4: (a) The simplified schematic and (b) timing diagram of the MA circuit.

while those of the others latched to high. At the end of LTA phase, *Start* latches all the cells regardless of V_{IN}. Additional logic, omitted from Figure 3-3 for clarity, prevents the selection of multiple losers.

Compared to a continuous time (CT) implementations similar to [49], the proposed TD-LTA can potentially yield lower power-delay product if Cp is realized with the parasitic capacitance at the input node.

3.3.4 Memory Adaptation (MA) Circuit

The error currents between the input and the best-matching centroids' memory values are passed to the MA circuits. Each row of the MDC cells shares two MA circuits, for mean and variance memory respectively. The simplified schematic and timing diagram is shown in Figure 3-4. The MA circuit utilizes the charging and discharging of a capacitor to realize current-to-pulse-width conversion. The voltage *Vp* is first discharged from *Vdd* by the input current for a fixed period of t_1, then ramped up by the external voltage V_{RAMP} at the bottom plate of C1, until *Vp* crosses *Vdd* at t_2. The update pulse is defined by t_2-t_1, and is proportional to the input error current, allowing the memory values to adapt to the moving averages.

3.4 Measurement Results

The proposed clustering circuit has been fabricated in a 0.13 µm CMOS process using thick-oxide IO FETs, occupying 0.18 mm^2 of active area including the programming registers and biasing circuits. The prototype has 8 input dimensions and 4 centroids, and consumes 15 µW with 3 V supply.

The classification test was performed by programming the centroids to fixed positions and disabling memory adaptation. The inputs are equally spaced and randomly presented to the circuit. To allow easier visual interpretation, only 2 out of 8 dimensions of input vectors are shown. The results are color-coded and the measured decision boundaries show good matching with the ideal boundaries, illustrated in Figure 3-5. In this plot, the 4 centroids are shown in diamond shapes. The circuit assigns the input data to different centroids based on the Euclidean distances. The measured decision boundaries are shown as solid lines and ideal boundaries as dashed lines. The prototype circuit runs classification at a speed of 16 kHz, limited by the

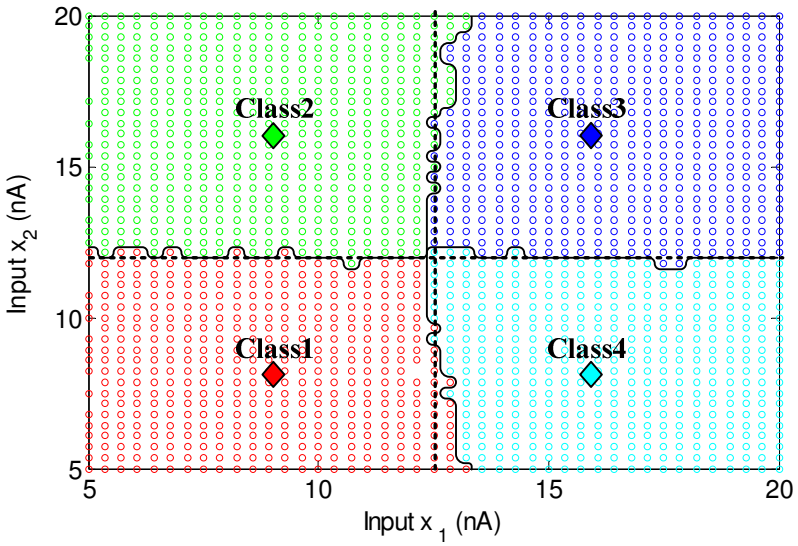


Figure 3-5: Classification test results.

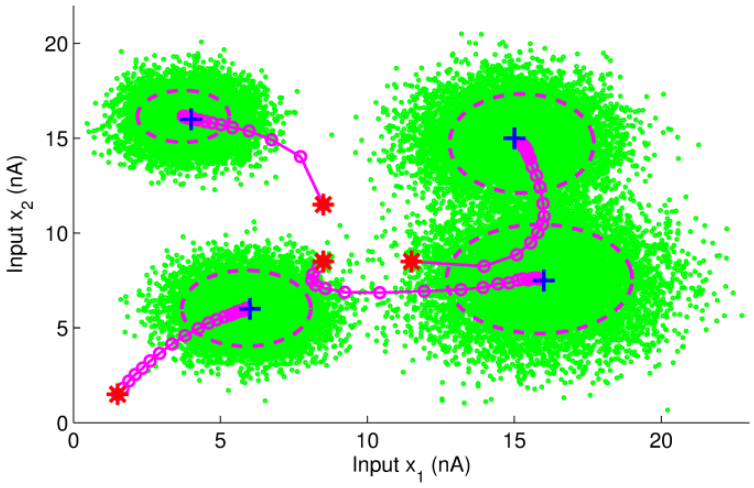


Figure 3-6: Clustering test result.

settling time of the input current.

I demonstrated the full functionality of the prototype by solving a clustering problem. 40000 8-dimensional vectors were generated as the inputs to the circuit. The dataset contains 4 underlying clusters, each drawn from Gaussian distributions with different means and variances. Initially the centroids were programmed to separate means marked with red stars and equal variance. During the test, the centroid means were read out every 0.5 s, plotted with circles connected by lines on top of the data scatter in Figure 3-6, and shown together is the learned variance values at the end of test plotted with dashed ellipses. The centroids adapt accurately to

TABLE II. PERFORMANCE SUMMARY OF THE CLUSTERING CIRCUIT

Parameter	*Value*
Technology	1P8M 0.13 μm CMOS
Total Area	0.9×0.2 mm^2 (8×4 array)
MDC Cell Area	90×30 μm^2
Power consumption	15 μW @ 3V
Classification Speed	16 kHz
Clustering Speed	4 kHz

centers of the input data clusters marked with blue crosses despite the clusters' overlapping, and the extracted variances match with the true values, both confirming a robust learning performance. The task takes 10 s at 4 kHz; higher speed is possible at the cost of lower learning rate. The measured performance is summarized in Table II.

Chapter 4 Analog Deep Machine Learning Engine

In this chapter, the design of an analog deep machine learning engine (ADE) implementing DeSTIN is presented. The hierarchical architecture consists of 3 layers of nodes; each is an evolved version of the clustering circuit discussed in Chapter 3, with greatly increased power and area efficiency and additional functionality. The ADE extensively utilizes the floating gate memory described in Chapter 2, with improvement, for distributed non-volatile analog storage.

The online clustering circuit is a key component in the ADE system. The previous implementation in Chapter 3 put more focus on realizing the algorithm for proof of concept. Although its power and area consumption is small, a large scale learning system requires even

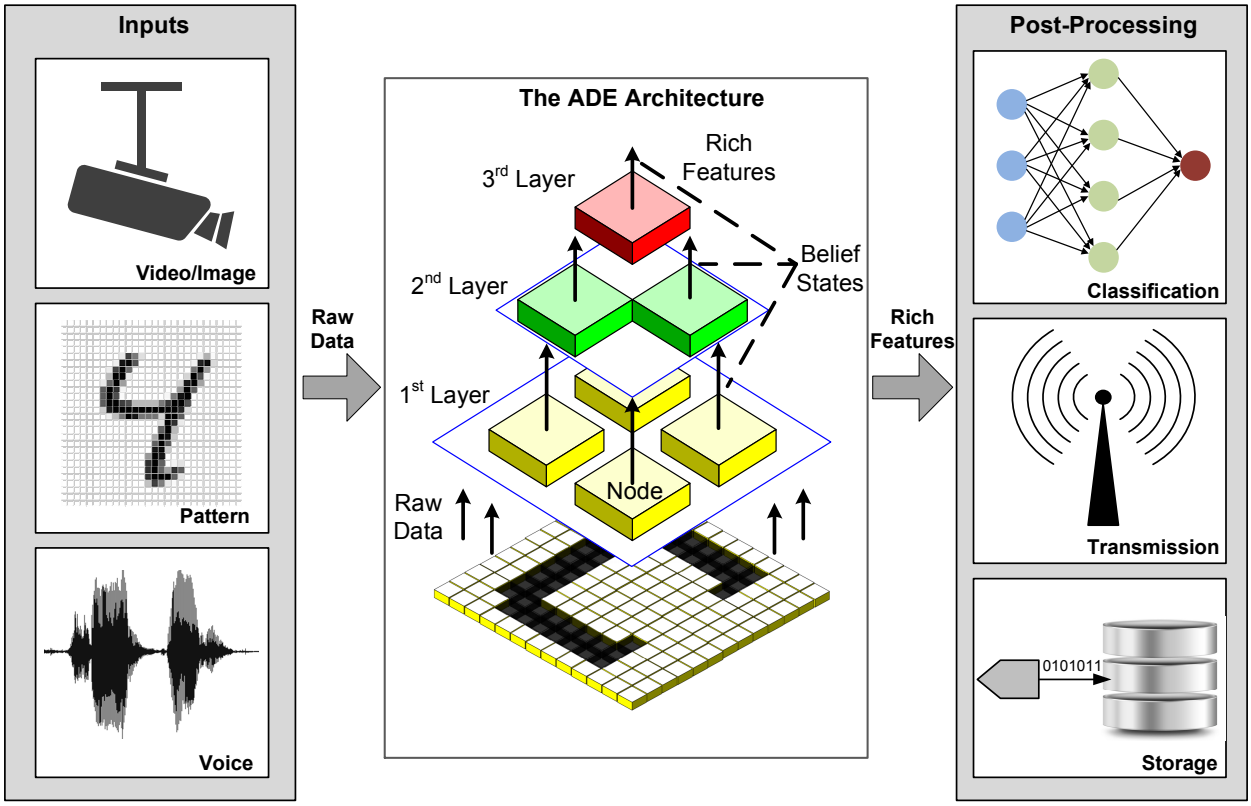


Figure 4-1: The architecture of the analog deep machine learning engine and possible application scenarios.

higher efficiency. Therefore, this chapter will focus on the power and area efficiency and propose various design techniques to greatly improve these performances for analog signal processing systems.

4.1 Introduction and Literature Review

Machine learning systems provide automated data processing. It sees a wide range of applications from computer vision, data mining, natural language processing, to economics and biology [50]. When a machine learning system is used to process high-dimensional data such as raw images and videos, the difficulty of "curse of dimensionality" [4] arises. Therefore, when dealing with such high dimensional data, it is often necessary to pre-process the data to reduce its dimensionality to what can be efficiently processed, while still preserving the essence of it, a technique known as feature extraction. Deep machine learning (DML) architectures have recently emerged as a promising bio-inspired framework, which mimics the hierarchical presentation of information in the human brain to achieve robust automated feature extraction [5].

While these deep layered architectures offer excellent performance attributes, the computation requirements involved grow dramatically as the dimensionality of input increases. GPU based platforms have been proposed to provide the required parallel computation [51], but they are prohibitively power hungry, making them impractical in power-constrained environments and limiting their large-scale implementations. Custom analog circuitry presents a means of overcoming the limitation of digital VLSI technology. By exploiting the computational primitives inherent in the physics of the devices, and presenting the information with multi-bit encoding, analog signal processing (ASP) systems have the potential to achieve much higher energy efficiency compared to their digital counterparts [14]. Therefore, analog and mixed-mode

signal processing is widely employed in ultra-low-power circuits and systems such as vision processors [52], adaptive filters [37], and biomedical sensors [53]. In [54], [55], [56], analog circuits are embedded in digital systems to perform efficient non-linear functions. The other advantage of ASP is that it interfaces directly with sensors. By performing pre-processing and compression of the sensory data at the front-end, the accuracy and bandwidth requirement of subsequent blocks can be relaxed, increasing the overall system efficiency [57].

ASP has been successfully applied to build machine learning systems [35], [43], [45], [46], [58]. But many of them do not have on-chip learning capability, and a software emulation session is needed to generate the parameters which will then be programmed in the chip [35], [43], [58]. This limits the system to the specific task or dataset it was pre-programmed to process. An on-chip trainable machine learning system is described in [46]. It is based on supervised learning and relies on a human expert to label the input data during training. An unsupervised learning system that is able to learn from the data continuously without any external assistance is more desirable in many applications.

The other important component of a learning system is the memory, which stores the previous learned knowledge. Digital memory requires A/D/A conversions to interface with analog circuits, consuming area and power headroom [46], [54], [55], [56], especially in a system with distributed memories where the data converters cannot be shared. Capacitors can be used for analog storage [45], but require constant refreshing and are prone to long-term drift due to the leakage current, notably large in deep-sub-micron processes. In addition, both the digital and capacitor storage discussed above are volatile, and lose their states without power. This precludes their use in intermittently powered devices such as those depending on scavenged power, where blackout is common [59].

The purpose of this work is to develop an analog implementation of a deep machine learning system [60]. It features unsupervised online trainability driven by the input data only. Unsupervised learning is arguably more difficult than supervised one, as there is no obvious error metric to correct the current perception. But it is more widely applicable because it eliminates the need for manually labeling the data. This ability to learn from the input data in real time without external intervention is essential for fully-autonomous systems. The proposed ADE utilizes floating-gate memory to provide non-volatile storage, facilitating the operation with harvested energy. The memory has analog current output, interfacing naturally with the rest of the system, and is compatible with standard digital CMOS process, And the architecture is designed for scaling. To maximize energy efficiency, several strategies are pursued at the system level. 1) The architecture adopts massively parallel computation, and the power-delay product is minimized by biasing transistors deep in weak inversion. 2) The feedback inherent in the learning algorithm is exploited to desensitize the system to inaccuracy such as mismatch, allowing aggressive area and bias current scaling-down with negligible performance penalty. 3) Current mode circuits are extensively employed to realize efficient arithmetic, such as current wire-summing and translinear multiplication/division. 4) Distributed memories are kept local to the computational elements, minimizing their access energy. 5) System power management applies power gating to the inactive circuits.

4.2 Architecture and Algorithm

The analog deep machine learning engine (ADE) implements deep spatiotemporal inference network (DesTIN) [61], a state-of-art compositional DML framework, the architecture of which is shown in Figure 4-1. Seven identical cortical circuits (nodes) form a 4-2-1 hierarchy. Each node captures the regularities in its inputs through an unsupervised learning process. The lowest

layer receives the raw data (e.g. the pixels of an image), and continuously constructs belief states that characterize the sequence observed. The inputs of nodes on the 2nd and 3rd layers are the belief states of nodes at their respective lower layers. Beliefs extracted from the lower layers characterize local features and beliefs from higher layers characterize global features. From bottom to top, the abstraction level of the information increases while the dimensionality reduces. The beliefs formed at the top layer are then used as rich features for post-processing.

The node learns through an online k-means clustering algorithm, which extracts the salient features of the inputs by recognizing spatial density patterns (clusters) in the input data. Each recognized cluster is represented by a centroid, which is characterized by the estimated center of mass (centroid mean $\hat{\mu}$) and spreads (centroid variance $\hat{\sigma}^2$). The architecture of the node is shown in Figure 4-2(a). It incorporates an 8×4 array of reconfigurable analog computation cell (RAC), grouped into 4 centroids, each with 8-dimensional input. The centroids parameters $\hat{\boldsymbol{\mu}}$ and $\hat{\boldsymbol{\sigma}}^{\boldsymbol{2}}$are stored in their respective floating gate memories (FGM). The input to the node is an 8-D observation vector sequence $\boldsymbol{o}[\boldsymbol{n}]$, presented row-parallel to the RAC array.

A training cycle begins with the classification phase (Figure 4-2 (b)). The RAC in the *i*-th element of centroid *j* calculates the 1-D Euclidean distance from its own centroid mean to the input D_{ij}^{EUC}. The Euclidean distance from $\boldsymbol{o}$ to each centroid is obtained by wire-summing all the RAC output currents along the column:

$$\mathrm{D}_j^{EUC} = \sum_i D_{ij}^{EUC} = \sum_i (o_i - \hat{\mu}_{ij})^2. \tag{4.1}$$

Then a winner-take-all (WTA) network in the distance processing unit (DPU) searches for the best-matching centroid *k* with the minimum Euclidean distance to $\boldsymbol{o}$:

$$k = \arg\min_j (D_j^{EUC}), \tag{4.2}$$

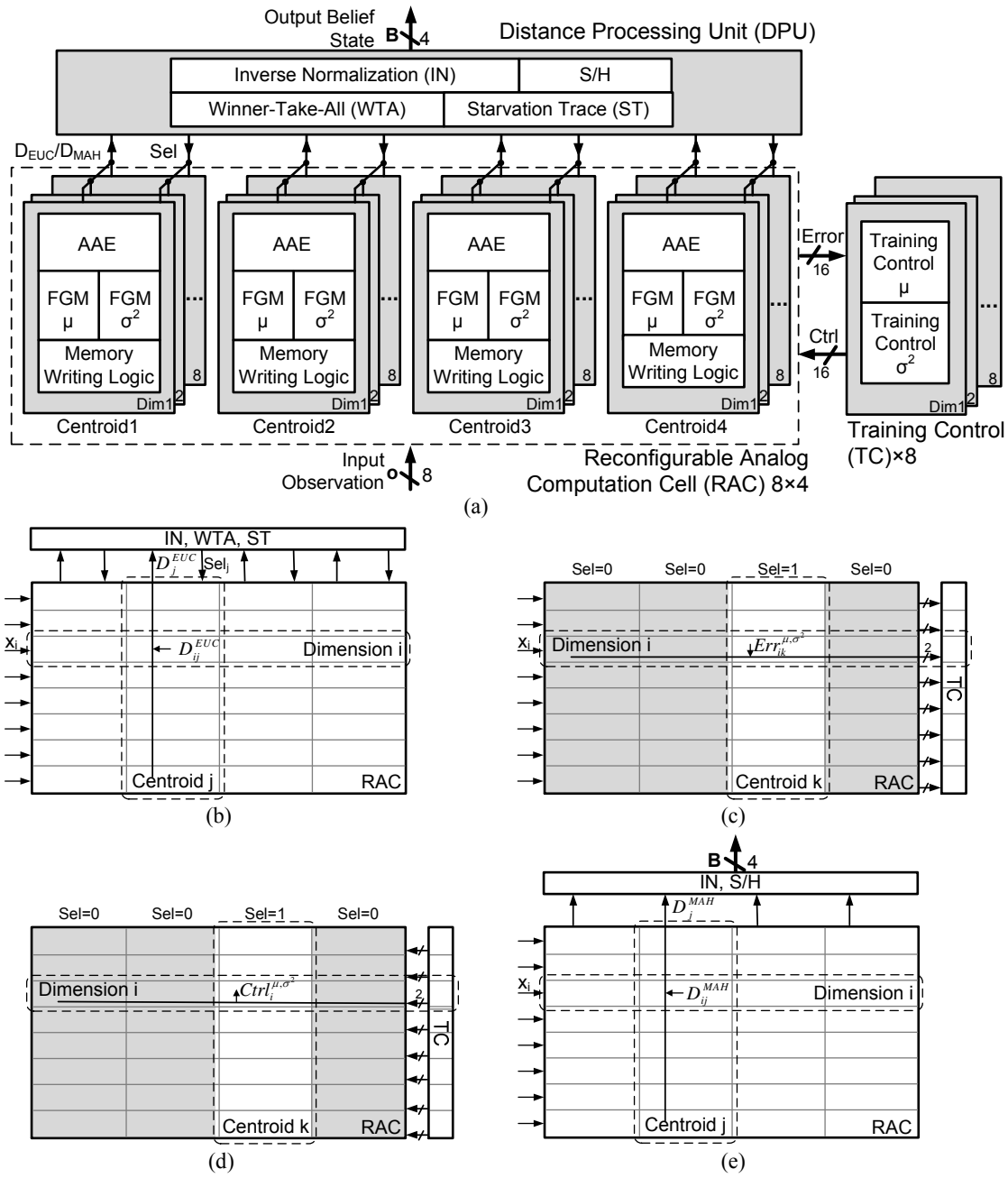


Figure 4-2(a): The node architecture. The clustering algorithm implemented by the node is illustrated in (b)-(e).

and selects it by asserting Sel_k. For robust learning against unfavorable initial conditions, a starvation trace (ST) [62] circuit in the DPU monitors and corrects situations wherein some centroids are initialized too far away from populated regions of the inputs and never get selected,

or "starved". In the next phase (Figure 4-2(c)), the selected centroid propagates its mean and variance error vectors to the training control (TC) block. The *i*-th elements of the 8-D error vectors are given by

$$\begin{aligned} Err_{ik}^{\mu} &= o_i - \hat{\mu}_{ik} \\ Err_{ik}^{\sigma^2} &= \left(o_i - \hat{\mu}_{ik}\right)^2 - \hat{\sigma}_{ik}^2 \end{aligned} . \tag{4.3}$$

The TC is shared across all centroids because only one centroid is selected for training each cycle. After the TC loads the errors, it generates memory writing control signals ***Ctrl*** for both mean and variance memories in the selected centroid, respectively. ***Ctrl*** is broadcasted along the row, the memory writing logic ensures that only the memories in the selected centroid get updated (Figure 4-2 (d)). The magnitudes of update are proportional to the errors in (4.3):

$$\begin{aligned} \hat{\mu}_{ik}[n+1] &= \hat{\mu}_{ik}[n] + \alpha \, \mathrm{Err}_{ik}^{\mu} \\ \hat{\sigma}_{ik}^2[n+1] &= \hat{\sigma}_{ik}^2[n] + \beta \, \mathrm{Err}_{ik}^{\sigma^2} \end{aligned} , \tag{4.4}$$

where α and β are the learning rates. The proportional updates cause the centroid means and variances to follow exponential moving averages and converge to the true means and variances of the data clusters. All the memories are written simultaneously. Finally, the 4-D belief state ***B*** is constructed, which represents the probability that the input vector belongs to each of the 4 centroids (Figure 4-2(e)). Simplified 8-D Mahanalobis distances (assuming diagonal covariance matrix) from each centroid to the input are calculated in a way similar to (4.1):

$$\mathrm{D}_j^{MAH} = \sum_i D_{ij}^{MAH} = \sum_i \frac{(o_i - \hat{\mu}_{ij})^2}{\hat{\sigma}_{ij}^2} . \tag{4.5}$$

Compared to the Euclidean distance, the Mahalanobis distance is a better metric of statistical similarity in that it takes both the mean distance and spread of data into account. Then the inverse-normalization (IN) block in the DPU converts $\boldsymbol{D}^{\boldsymbol{MAH}}$ to valid probability distribution ***B*** satisfying:

$$B_j = \frac{\lambda}{D_j^{MAH}}, \quad \sum_j B_j = 1 \tag{4.6}$$

where λ is the normalization constant. A sample and hold (S/H) holds $\boldsymbol{B}$ for the rest of the cycle to allow parallel operation across the hierarchy. After the training converges, the ADE can operate in recognition mode. In this mode, the memory adaptation is disabled to save power and the ADE continuously extracts rich features from the input based on its previously learned model parameters.

Careful considerations at architecture and algorithm level facilitate scaling, and improve area and energy efficiency. First, each node is identical, making is easy to scale up the system for deeper hierarchy and larger input dimensionality to solve more complex problem. Second, the DPU and TC are shared along the columns and rows, respectively, and kept peripheral to the computation array, so as that their area and power scales up slower. Third, the similarity metrics used in the algorithm (D^{EUC}/D^{MAH}) allow easier scaling up of input dimension. The distances are summed in current to form multivariate distribution: the increased current level reduces the time constant at the summing node, and all the 1-D elements can be computed in parallel.

The ADE goes through four distinct operation phases in each cycle, and in each phase only a part of the system is active. Based on this observation, the circuits are partitioned into several power domains based on the functions they perform, and power gating is applied whenever possible to save biasing power. The resulting timing diagram of the flexible intra-cycle power gating is shown in Figure 4-3. Measurement results show a reduction of power consumption by 22% in training mode and 37% in recognition mode.

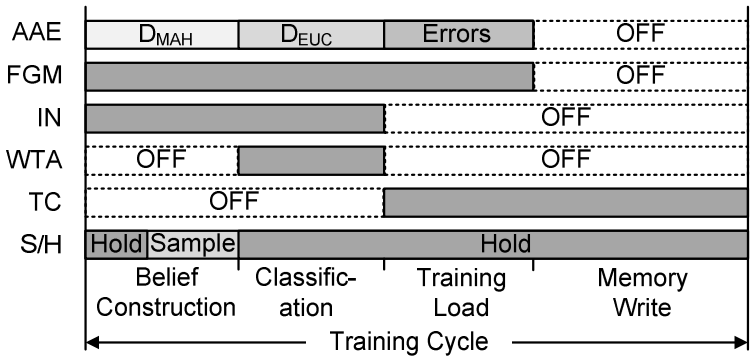


Figure 4-3: Timing diagram of the intra-cycle power gating.

4.3 Circuit Implementation

4.3.1 Floating-Gate Analog Memory (FGM)

The FGM provides non-volatile storage for the centroid parameters. It can be accessed by on-chip circuits, as well as from off-chip through scanning registers for initialization. Its schematic is shown in Figure 4-4. The design is based on the FG memory in Chapter 2. Significant improvement on power and is achieved by replacing the differential pair V-to-I converter with a single-ended structure which realizes similar transfer function. The "single-

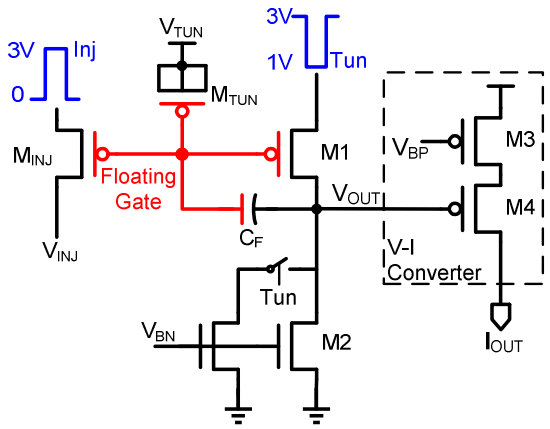


Figure 4-4: The schematic of the improved floating gate analog memory.

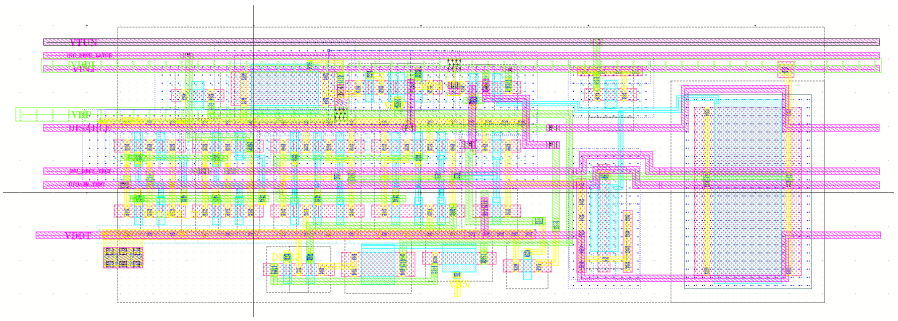


Figure 4-5: The layout of the new FGM.

TABLE III. PERFORMANCES SUMMARY AND COMPARISON OF THE IMPROVED FG MEMORY

Parameter	***Proposed FGM***	*Digital Register (8bit)*	*Digital Register2 (20bit)*
Active Area	**140 μm^2**	332 μm^2	830 μm^2
Power supply	**3 V**	1.5 V	1.5 V
Power consumption	**15 nW (avg.)**	21.3 nW	53.4 nW
Output range	**0 - 10 nA**	-	-
Dynamic range	**46 dB**	48 dB	120 dB

ended differential pair" design is modified from that in [63]. This structure achieves more than 50% area reduction, while still providing a sigmoid transfer function and reduced input swing required by the FGM. Another attractive feature is that this V-I converter does not consume any current other than its own output current. On average, this halves the power consumption compared to the constant biasing in normal differential pair. In order to further reduce the area and power, the cascode output current mirror is removed. The virtual ground provided by the input of the absolute value circuit in the AAE helps to reduce the error caused by the finite output resistance.

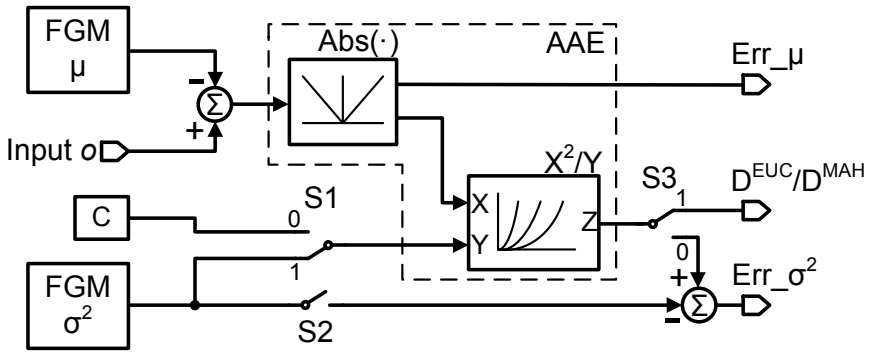
The biasing of the floating gate inverting amplifier is as low as 0.5 nA. In normal operation, the bandwidth is not strongly affected because the capacitance at the node V_{OUT} is relative small. However, during tunneling, the voltage at V_{OUT} needs to slew down by about 2 V. The small biasing current makes the slew rate excessively low, prohibiting fast memory writing. To solve

this, an additional biasing current source is added to boost the slew rate for the memories being tunneled. The power penalty is small because on average, only 1/8 of the total memories are tunneled in each cycle.

The layout of the proposed FGM is shown in Figure 4-5. The performance is listed in Table III. Compared to the original FGM in Chapter 2 [27], the proposed FGM occupies only 28.5% of the area and consume 66% less current. The FGM has an SNR equivalent to an 8-bit digital register, while the minimum fractional update equivalence is estimated to be 20 bits. Table III compares the proposed analog FGM to 8-bit and 20-bit digital registers in the same process operating at same speed of 8.3 kHz and shows advantages in both area and power.

4.3.2 Reconfigurable Analog Computation (RAC)

The RAC is the most computation-intensive block in the system and utilizes subthreshold



Phase	Switch Positions			Output
	S1	S2	S3	
Classification	0	Open	1	D^{EUC}
Training Load	0	Close	0	Err_μ,σ^2
Belief Construction	1	Open	1	D^{MAH}

Figure 4-6: The schematic of the reconfigurable analog computation cell and the switch positions for three operation modes.

current-mode computation to implement efficient arithmetic functions. It performs three different operations through reconfigurable current routing. The schematic and the current switch configurations for the three modes are shown in Figure 4-6. The input current o and the centroid mean $\hat{\mu}$ stored in the FGM_μ are added in opposite direction, the difference current $o - \hat{\mu}$ is rectified by the absolute value circuit Abs. The unidirectional current is then fed into the X^2/Y operator circuit, the Y component can be either the centroid variance $\hat{\sigma}^2$, or a constant C, depending on whether D^{MAH} or D^{EUC} is required. In training load phase, the Abs circuit duplicates its X input to get Err_μ (the error for mean memory training), and the difference current between D^{EUC} and $\hat{\sigma}^2$ is used as Err_σ^2 (the error for variance memory training) because the Euclidean distance has the same form as the square error in (4.3). The reconfigurability of the RAC allows the computational circuits to be reused for different operations, therefore saving area. It reduces the number of error sources in the circuit. In addition, it reduces the system's sensitivity to mismatch errors by having correlated errors for memory training and feature extraction. For example, if there is an offset at the input, it has no system level impact because it will simply shift the location of the origin of the input space. And since both training and feature extraction are concerned with the relative distances, it has no effect on the ultimate belief output. Similarly, if there's offset or gain errors in the memory, this error will be included in the feedback loop and the memory output will still adapt to the input statistics.

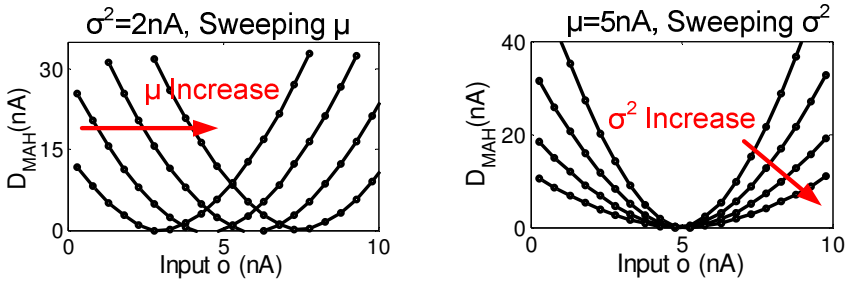


Figure 4-7: The measured transfer functions with the RAC configured to belief construction mode.

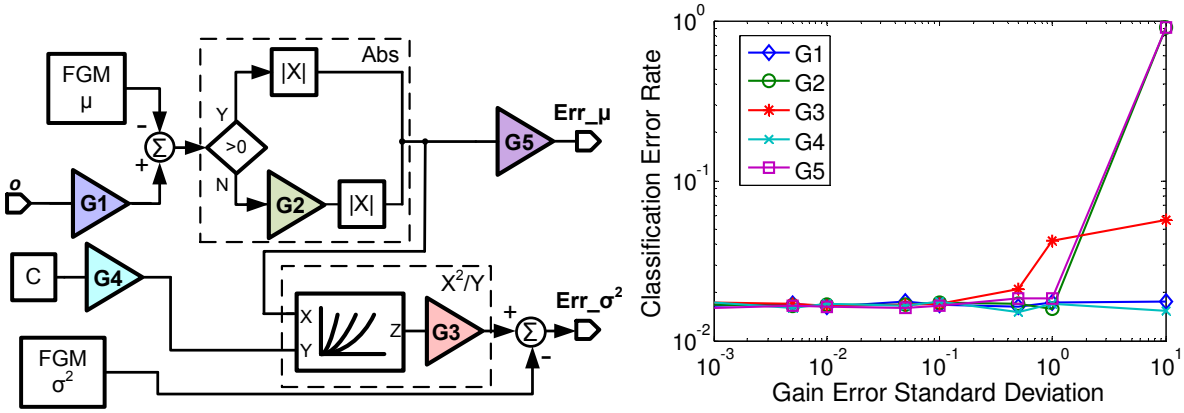
The design of the analog arithmetic element (AAE) is similar to the distance computation block in Chapter 3.3.2. The measurement results of the AAE are plotted in Figure 4-7, showing variable center and width of the quadratic transfer function by varying $\hat{\mu}$ and $\hat{\sigma}^2$.

ASP suffers from circuit imperfections such as noise and mismatch due to its lack of restoring mechanisms found in digital logic. Any ASP-based system needs to address these non-idealities, without excessively affecting the other performances metrics. The current noise power of transistors biased in subthreshold is given by $2qI_D\Delta f$ [64], where Δf is the noise bandwidth, proportional to g_m of the transistors (the relative contribution of flicker noise is negligible at very low current level). As g_m/I_D ratio is fairly flat in subthreshold region, the computational throughput of a current-mode circuit biased in sub-threshold grows roughly linearly with the signal current level (or power consumption) while the system SNR remains nearly constant. Mismatch and efficiency place two contradictory requirements to the circuit design: device matching can be improved by increasing the areas of the devices [65], however, sizing-up devices comes with the cost of both area and energy efficiency. Fortunately, the learning algorithm provides robustness to mismatch by desensitizing the system to static errors using algorithm-level feedback [15]. To take full advantage of this robustness, the behavioral model of the RAC is built to include the mismatch errors found in the circuit. In sub-threshold circuits, the

threshold voltage mismatch is the dominant source of mismatch. Even though the notion of threshold voltage is precisely applicable to subthreshold operation, the same variation causes the same effect as a shift in the gate voltage. Therefore the drain current of a transistor with a shift in threshold voltage can be expressed as

$$I_D = I_0 e^{\frac{V_{GS}-\Delta V}{U_T}} = I_0 e^{\frac{V_{GS}}{U_T}} e^{-\frac{\Delta V}{U_T}}, \tag{4.7}$$

where I_0 is a device-specific constant and U_T is the thermal voltage. It can be seen that the $e^{-\Delta V/UT}$ term results in gain errors in current-mode circuits. In the model in Figure 4-8(a) (training load mode is shown), each gain block Gx corresponds to the gain error introduced by each sub-circuit. System simulations were performed with progressively increasing gain errors to evaluate the effect of each error on the ADE system performance. The results are plotted in Figure 4-8(b). It can be seen that the system performance does not degrade until the errors are quite large, showing the robustness of the algorithm. The knowledge of the system sensitivities and tolerances allows aggressive reduction of the device sizes to place each gain error around its knee point of the performance curve in Figure 4-8(b), improving efficiency with negligible performance penalty.



(a) (b)

Figure 4-8: Behavioral model of the RAC with gain errors. (b) System's classification error rate as a function of each error.

In current mode sub-threshold circuit as this work, SNR can be improved by increasing the signal current level while keeping the bandwidth constant, or keeping the current level constant while reducing the bandwidth, both hurt efficiency. With increased resolution, the advantage of analog signal processing will diminish because the cost for analog accuracy grows faster than digital. This work was designed for 7 bit resolution, because system simulation indicated adequate performance with this resolution. And the measurement results agree well with the design target.

4.3.3 Distance Processing Unit (DPU)

The distance processing unit (DPU) performs various operations on the 8-D distances from the four centroids. It has a modular design with four identical channels interconnected, one for each centroid. And it performs collective operations such as IN and WTA with a single communication wire along all the channels. Both facilitate scaling of the number of centroids. The simplified schematic of one channel is shown in Figure 4-9. In belief construction phase, the IN blocks converts Mahalanobis distance $\boldsymbol{D}^{\boldsymbol{MAH}}$ to belief state $\boldsymbol{B}$. The algorithm requires these

two values to follow (4.6), as ***B*** represents collectively exhaustive probability measures of the input's similarity to each centroid. The translinear loop formed by M1 and M2 (denoted by the arrow) causes the product of the two drain currents to be a function of the difference between the bias voltage V_B and the voltage on the communication wire V_C, $I_{IN} \cdot I_{OUT} = f(V_C - V_B)$. Since all the channels see the same V_B and V_C, they all have: $I_{IN} \cdot I_{OUT} = \lambda$, where λ is constant across the four channels. In addition, the sum of the four output currents is dictated by the normalization current I_{NORM}, common to all the channels. Thus the inverse normalization function is implemented with only 3 transistors per channel without any additional biasing. The output belief states are sampled then held for the rest of the cycle to enable parallel operation of all the layers. The sampling of ***B*** starts from the top layer and propagates to the bottom, opposite to the data path; this pipelined processing eliminates the need to wait for the data to settle before

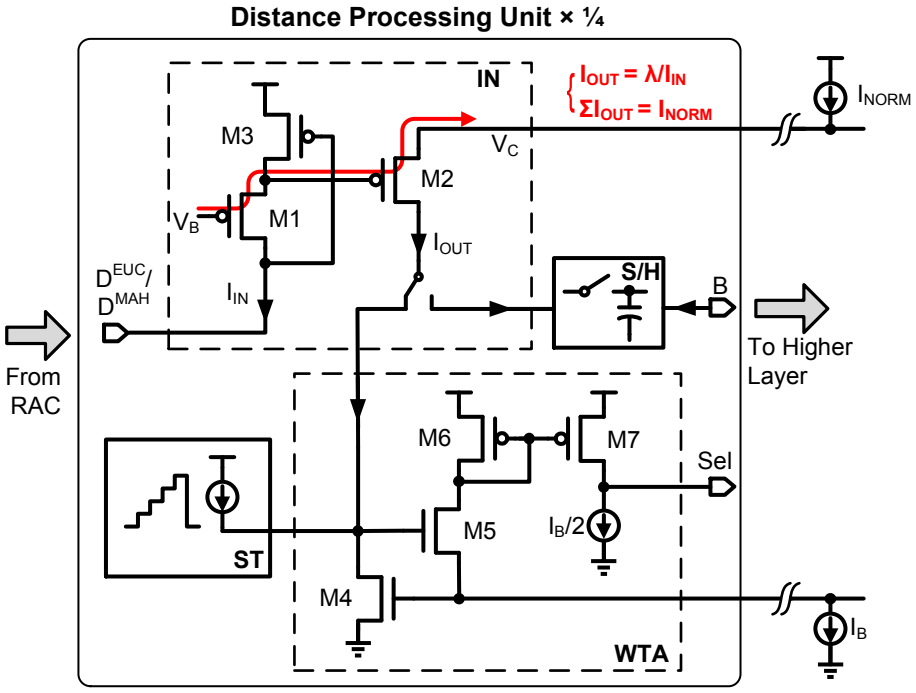


Figure 4-9: The schematic of one channel of the distance processing unit.

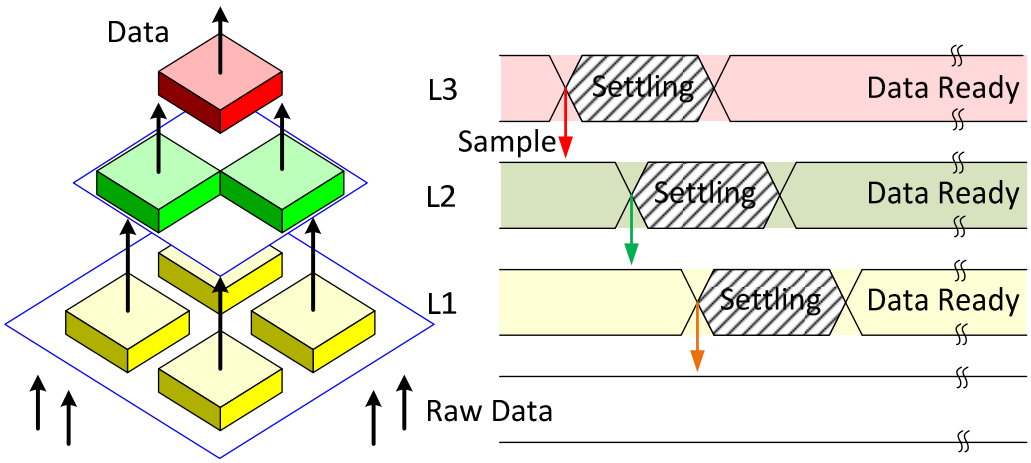
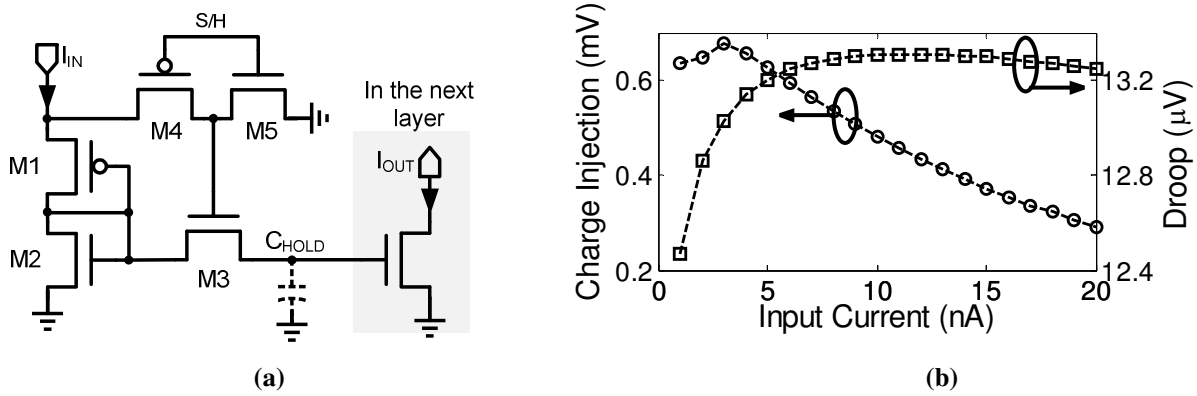


Figure 4-10: Timing diagram of data sampling across the hierarchy to enable pipelined operation.

sampling, improving the throughput, the timing diagram is shown in Figure 4-10. In classification phase, reconfigurable current routing allows the IN circuits to be reused together with the WTA to yield a loser-take-all operation to find the minimum Euclidean distance. The WTA (M4-M7) is based on the design in [49]. The voltage on the common wire is determined by the cell with the largest input current (winner). And the entire biasing current I_B will flow through M5/6 in the winner cell, making its output high. A starvation trace (ST) circuit is implemented to inject current into the WTA when the centroid is starved.

The schematic of the current mode sample and hold (S/H) is shown in Figure 4-11(a). To maximize the power efficiency, the holding capacitor C_{HOLD} is realized entirely with the wiring parasitic capacitances between nodes. These wires are carefully laid-out to be shielded from noisy signals, and a low-charge-injection switch is designed to mitigate the charge injection errors exacerbated by low valued C_{HOLD} and current-mode sub-threshold operation. During sample mode, *S/H* is low and the switch M3 is turned-on with near-minimum necessary V_{GS} to minimize its channel charge. This V_{GS} is generated by the diode-connected PMOS M1: body effect causes it to have slightly higher V_{TH} than M3, ensuring reliable turn-on in worst case mismatch situation. The post-layout simulation results are shown in Figure 4-11 (b). The S/H



(a) **(b)**

Figure 4-11: (a) The schematic of the sample and hold and (b) simulated charge injection and droop errors.

achieves less than 0.7 mV of charge injection error, and less than 17 μV of droop across a cycle with about 80 fF C_{HOLD}.

The schematic of the starvation trace circuit is shown in Figure 4-12, together with the typical timing diagram. C1, D1 and M1 form a charge pump, which removes a certain amount of charge from the storage capacitor C2 at every negative edge of the system clock when it is not selected, increasing the output current I_{OUT}. Since the ST output adds to the inversed Euclidean distance, this current progressively decreases the apparent distance from the starved centroid to the input, forcing it to eventually get selected for update and pulling it toward more populated areas of the input space. The ST output current is reset once the centroid gets selected (*Sel =1*).

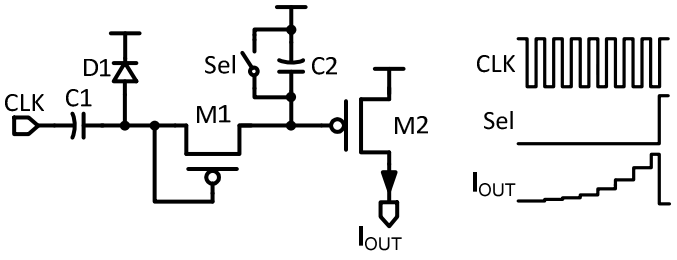


Figure 4-12: The schematic and timing diagram of the starvation trace circuit.

4.3.4 Training Control (TC)

The training control circuit converts the memory error current to pulse width to control the memory adaptation. The design is similar to the memory adaptation circuit in Chapter 3.3.4.

4.3.5 Biasing and Layout Design

Like other ASP systems, ADE requires biasing in many blocks, for example, V_{BP} in the FGM sets the full scale output; and the amplifier in the Abs circuit requires tail current. Distribution of biasing efficiently and accurately is important to the system's performance. A

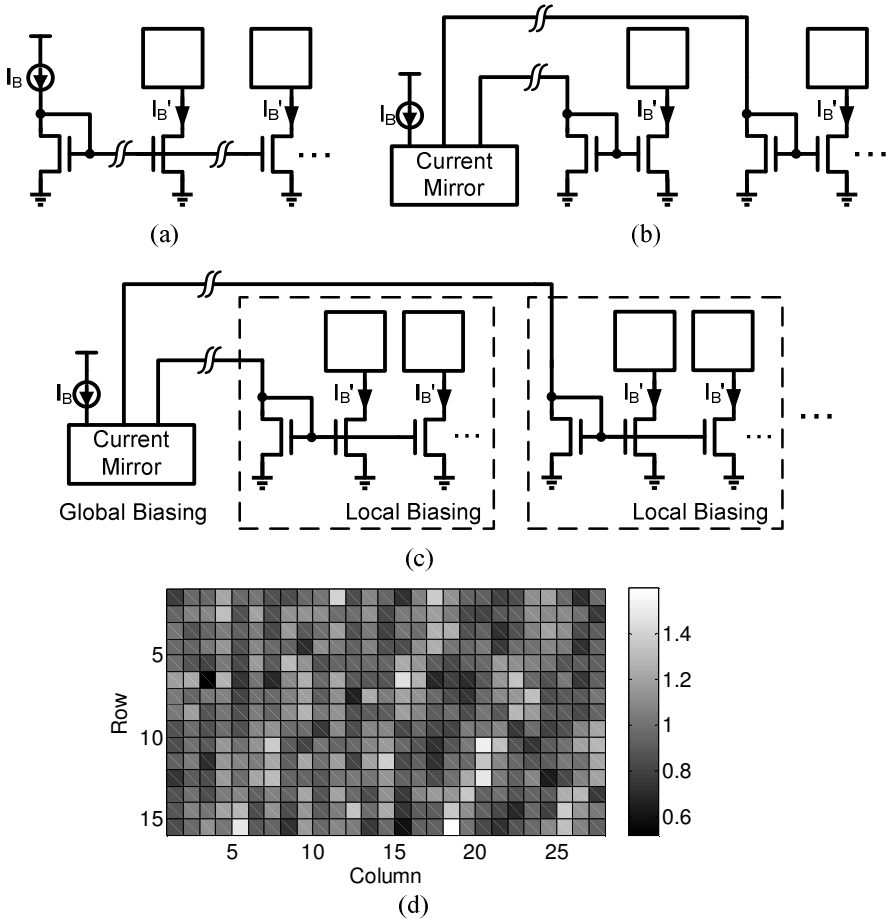


Figure 4-13: Biasing schemes (a) Voltage distribution. (b) Current distribution. (c) Proposed hybrid biasing. (d) Measured mismatch of biasing.

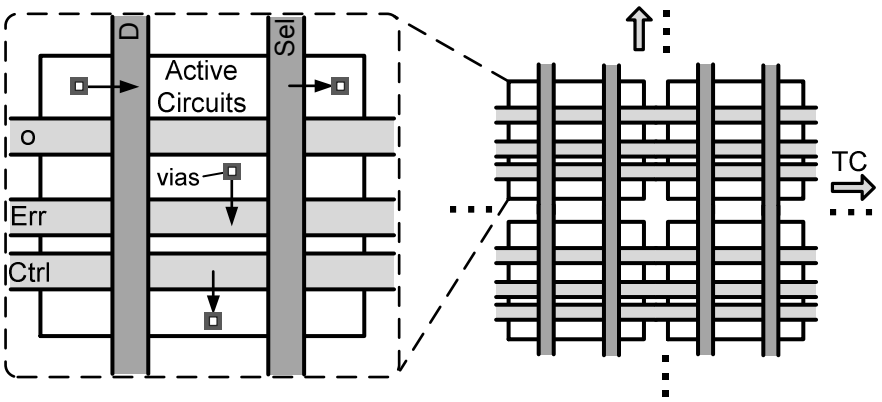
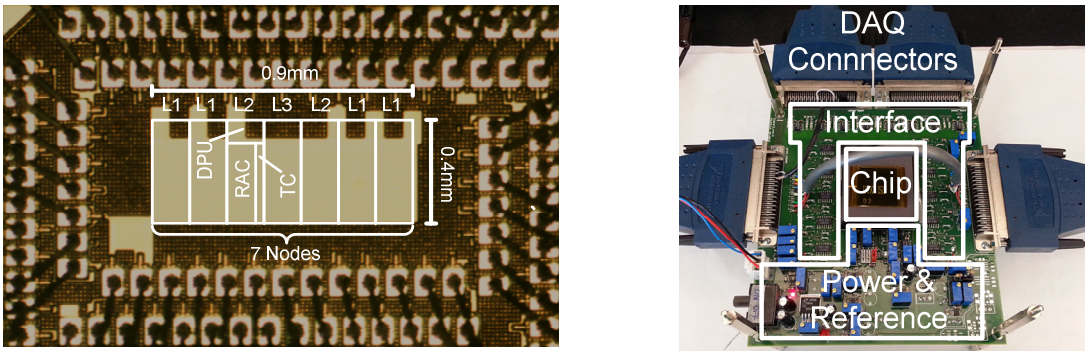


Figure 4-14: Conceptual diagram showing how the RAC array is assembled from the RAC cells.

tight tolerance in biasing current allows less safety margin in the system design and operation, because the block with lowest biasing current is usually the performance bottleneck. Biasing can be distributed across the chip using voltage as in Figure 4-13(a). However this scheme results in poor matching performance in large-scale systems due to process, stress and thermal gradients [65]. A current distribution scheme as in Figure 4-13(b) achieve better immunity to gradients by keeping both sides of current mirror close, but consumes large biasing current and wiring overhead. The biasing scheme adopted in this design is a trade-off between the above two: current distribution is used for global biasing, and voltage distribution is used for local biasing, as shown in Figure 4-13(c). The resulting biasing current accounts for only about 5% of the total current consumption, without observable gradient effects, shown in Figure 4-13(d).

The layout design of the ADE is non-trivial. A high density is required not only due to chip area constraint, but also for maximizing computational throughput and reducing mismatch across the system. Dense layout is achieved by floor-planning, minimizing the areas of repetitive components and running interconnections above the active circuits. Care was exercised to avoid noise coupling and component mismatch caused by density. The layout labor can be greatly



(a) (b)

Figure 4-15: (a) Chip micrograph and (b) custom test board.

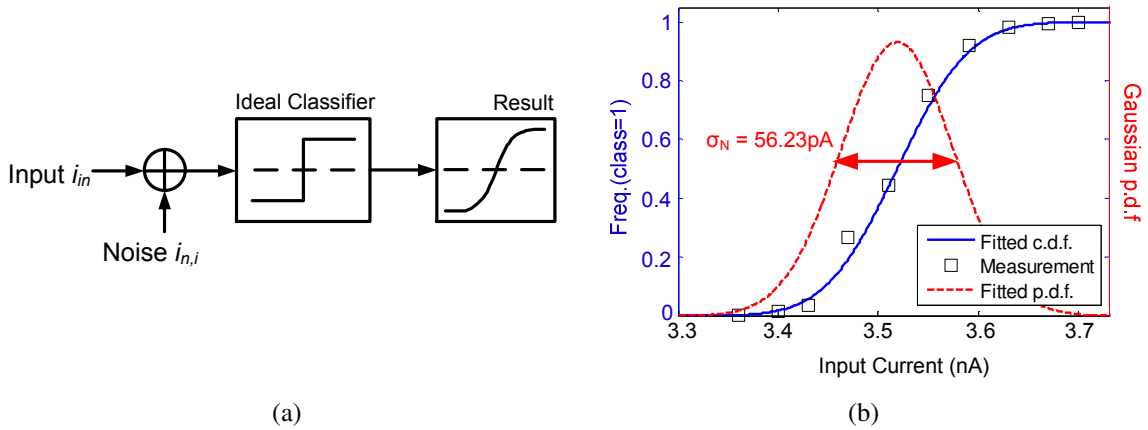
reduced by exploiting the regularity of the structure. Figure 4-14 shows conceptually how the RAC array is assembled from the RAC cells.

4.4 Measurement Results

The ADE was fabricated in a 0.13 μm standard CMOS process, and has an active area of 0.36 mm^2, including the biasing circuits and program ming registers, shown in Figure 4-15(a). Each RAC cell occupies 792 μm^2. Thick-oxide IO FETs are used to reduce charge leakage in the FGMs. With 3 V power supply, it consumes 27 μW in training mode, and 11.4 μW in recognition mode. To characterize the chip, a custom test board is developed with circuits to interface with the current mode IO of the chip, shown in Figure 4-15(b). For practical use, the design is intended for system-on-chip applications where the inputs and outputs are generated and processed on-chip. The data is streamed between the chip and PC through data acquisition hardware. And the acquired data is post-processed in MATLAB.

4.4.1 Input Referred Noise

We use a statistical approach to measure the input referred noise of the non-linear ADE system. In the measurement, memory adaptation is disabled and the system is configured into a classifier, modeled as an ideal classifier with an input referred current noise (Figure 4-16(a)). With two centroids competing, the circuit classifies the inputs to one centroid (class=1) or the other (class=0). When the inputs are close to the decision boundary and the classification is repeated for multiple times, the noise causes uncertainty in the outcome. Assuming additive Gaussian noise, it can be shown that the relative frequency of the event class=1 approaches the cumulative density function (c.d.f.) of a normal distribution. The standard deviation σ_N of this distribution is extracted using curve fitting, shown in Figure 4-16(b), and can be interpreted as the input-referred rms noise. The measured input referred current noise is 56.23 pA_{rms} and with an input full scale of 10 nA, we get an SNR of 45dB, or 7.5 bit resolution.



(a) (b)

Figure 4-16: (a) The system model for noise measurement. (b) Measured classification results and extracted Gaussian distribution.

4.4.2 Clustering Test

The performance of the node is demonstrated with clustering tests. 40000 8-D input vectors are generated, consisting 4 underlying clusters, each drawn from a Gaussian distribution with different mean and variance. The centroids are first initialized to separated means and equal variance (the initial condition is not critical since the circuit will adaptively adjust to the inputs). During the test, the centroid means are read out every 0.5 sec., plotted on top of the data scatter in Figure 4-17, and shown together is the learned variance values at the end of test. For easier visual interpretation, 2-D results are shown. The extracted cluster means and variances from several tests are compared to the true values and show good matching in Figure 4-18. The gain error in μ extraction is due to component mismatch; and the deviation of exponent from 2 in σ^2 extraction is due to body effect in the X^2/Y circuit; both can be tolerated by the algorithm. The performance of the starvation trace is verified by presenting the node with an ill-posed clustering problem. It can be seen that one of the centroids is initialized too far away from the input data,

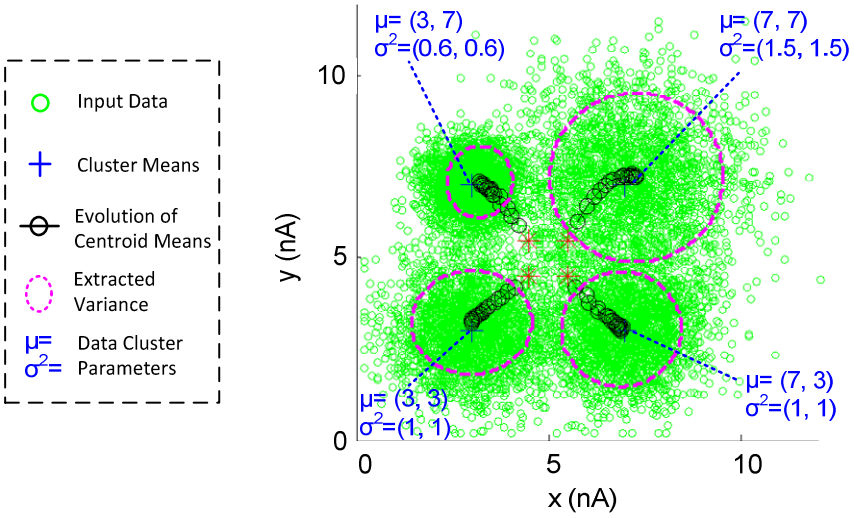


Figure 4-17: The clustering test results.

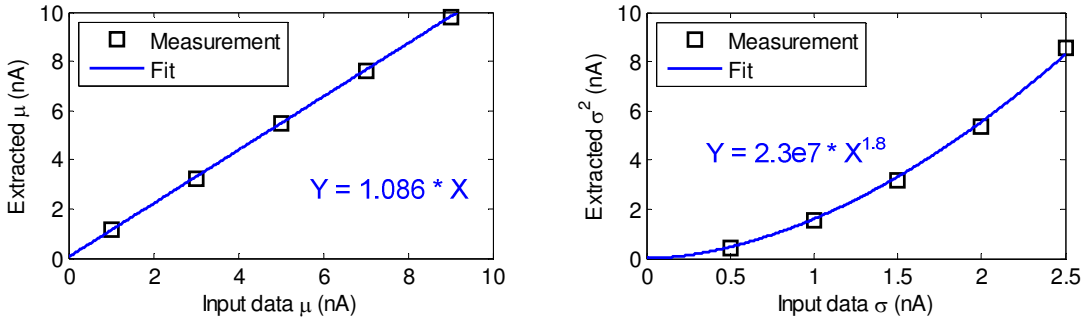


Figure 4-18: The extracted parameters plotted versus their true values.

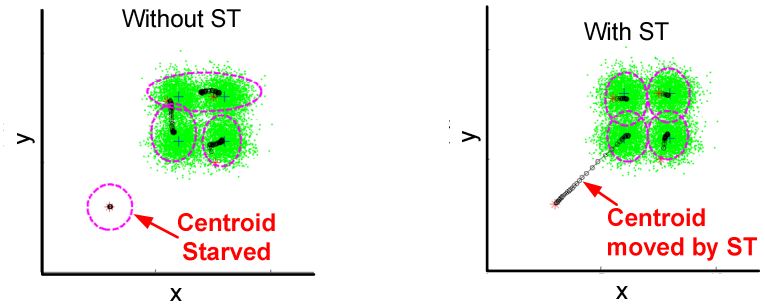


Figure 4-19: Clustering results with bad initial condition without and with the starvation trace enabled.

therefore never gets updated without the ST enabled. However, with the starvation trace enabled, the starved centroid is slowly pulled toward the area populated by the data, achieving a correct clustering result, shown in Figure 4-19.

4.4.3 Feature Extraction Test

We demonstrate the full functionality of the chip by doing feature extraction for pattern recognition with the setup shown in Figure 4-20. The input patterns are 16×16 symbol bitmaps corrupted by random pixel errors. An 8×4 moving window defines the pixels applied to the ADE's 32-D input. First the ADE is trained unsupervised with examples of patterns at 4.5 kHz. The training converges after about 30 k samples (7 sec.), as shown in Figure 4-21(a). After the training converges, adaptation can be disabled and the circuit operates in recognition mode at 8.3 kHz. The 4 belief states (Figure 4-21(a)) from the top layer are used as rich features, achieving a dimension reduction from 32 to 4. A software neural network then classifies the reduced-dimension patterns. Three chips were tested and average recognition accuracies of 100% with corruption lower than 10%, and 94% with 20% corruption are obtained, which is

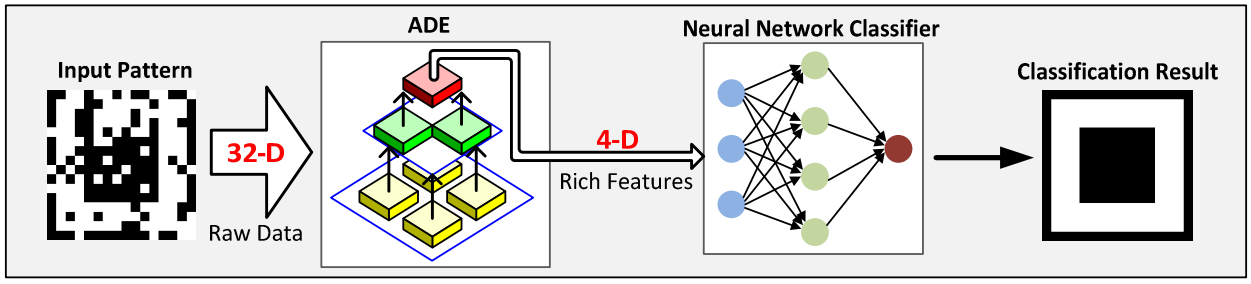
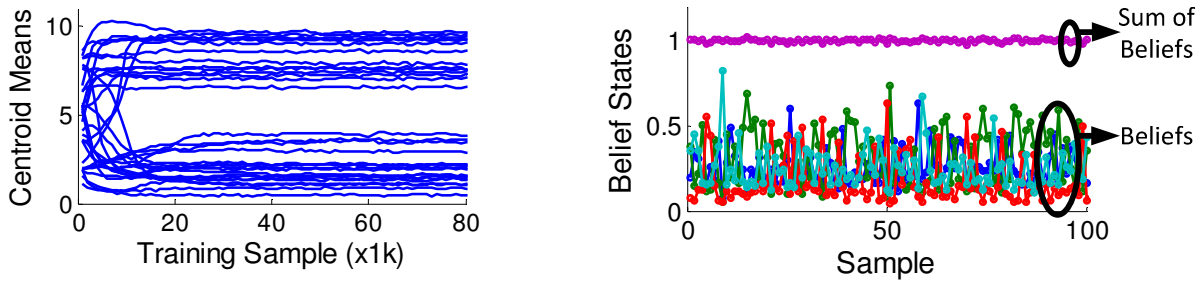


Figure 4-20: The feature extraction test setup.



(a) (b)

Figure 4-21: (a) The convergence of centroid during training. (b) Output rich feature from the top layer.

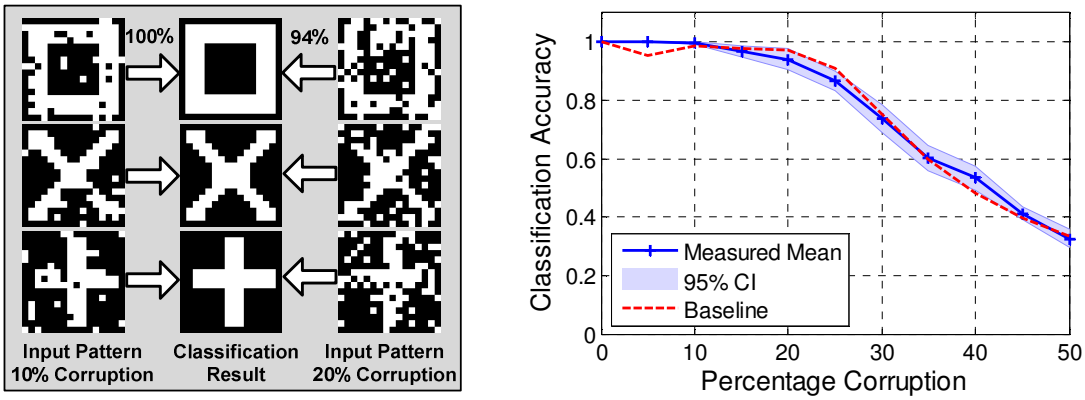


Figure 4-22: Measured classification accuracy using the feature extracted by the chip.

comparable to the floating-point software baseline, demonstrating robustness to the non-idealities of analog computation, as shown in Figure 4-22. The plot on the right shows the mean accuracy and 95% confidence interval (2σ) from the three chips tested, compared to the software baseline.

4.4.4 Performance Summary and Comparison

The measured performance of the ADE is summarized in Table IV. It achieves an energy efficiency of 480 GOPS/W in training mode and 1.04 TOPS/W in recognition mode. The performance and energy breakdown in the training mode are shown in Figure 4-23, the performance pie chart shows the mega operations per second each phase performs, and the energy chart shows how much energy each phase consumes per cycle. Table V compares this work with state-of-art bio-inspired parallel processors utilizing analog computation. It can be seen that this work has very high energy efficiency in both modes. Although it operates relatively slow, the ultra-low power consumption, together with the advantages of nonvolatile memory and unsupervised online trainability makes it ideal for autonomous sensory applications.

TABLE IV. PERFORMANCES SUMMARY OF THE ANALOG DEEP LEARNING ENGINE

Techonology	1P8M 0.13µm CMOS	
Power Supply	3V	
Active Area	0.9mm×0.4mm	
Memory	Non-Volatile Floating Gate	
Memory SNR	46dB	
Training Algorithm	Unsupervised Online Clustering	
Input Referred Noise	56.23pA$_{rms}$	
System SNR	45dB	
I/O Type	Analog Current	
Operating Frequency	Training Mode	4.5kHz
	Recognition Mode	8.3kHz
Power Consumption	Training Mode	27µW
	Recognition Mode	11.4µW
Energy Efficiency	Training Mode	480GOPS/W
	Recognition Mode	1.04TOPS/W

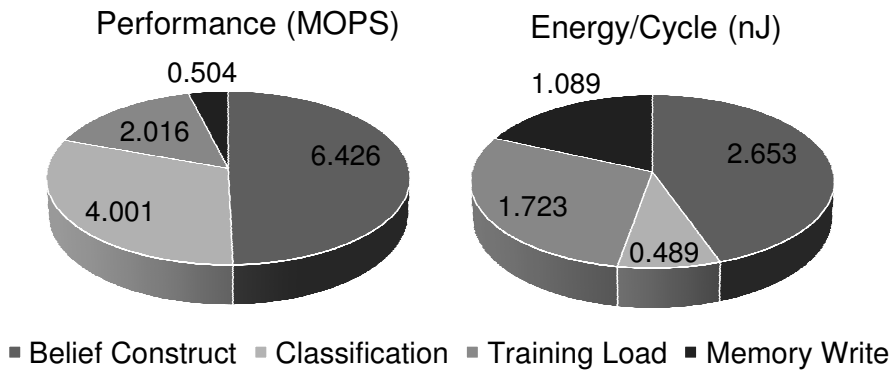


Figure 4-23: The performance and energy breakdown in the training mode.

TABLE V. COMPARISON TO PREVIOUS WORKS

	This work	JSSC'13 [54]	ISSCC'13 [55]	JSSC'10 [56]
Process	**0.13µm**	0.13µm	0.13µm	0.13µm
Purpose	**DML Feature Extraction**	Neural-Fuzzy Processor	Object Recognition	Object Recognition
Non-volatile Memory	**Floating Gate**	NA	NA	NA
Power (W)	**11.4µW**	57mW	260mW	496mW
Peak Energy Efficiency	**1.04TOPS/W**	655GOPS/W	646GOPS/W	290GOPS/W

Chapter 5 A nano-power tunable bump circuit

An ultra-low-power tunable bump circuit is presented. It incorporates a novel wide-input-range tunable pseudo-differential transconductor linearized using drain resistances of saturate transistors. The circuit is fabricated in a 0.13 μm CMOS process. Measurement results show that the proposed transconductor has a 5 V differential input range with less than 20% of linearity error. The circuit demonstrates tunability of bump center, width and height with a power consumption of 18.9 nW from 3 V supply, occupying 988 μm^2.

5.1 Introduction and Literature Review

Circuitries with bell-shaped transfer functions are widely used to provide similarity measures in analog signal processing systems such as pattern classifier [35], [58], support vector machine [46], and deep learning engine [60]. This non-linear radial basis function can be realized with the classic bump circuit [66]. However, the original implementation lacks the ability to change the width of its transfer function. Variable width can be obtained by pre-scaling the input voltage before connecting to the bump generator. The pre-scaling circuit using multi-input floating gate transistors [35] or digital to analog converter [46] consumes area and power overhead. In [58], [67], the widths of bump-like circuits are varied by switching binary sized transistors, but the number of possible widths is limited. A Gaussian function can be directly synthesized by exponentiating the Euclidean distance [68], however, this approach can lead to complex circuit and large area.

In this work, we propose to implement a bump circuit by preceding the current correlator [66] with a tunable transconductor to achieve variable width and height. The design of linear transconductors in subthreshold CMOS is challenging as the linear range of a conventional differential pair diminishes with the gate overdrive, and reaches its minimum in subthreshold region [69]. Common linearization techniques such as source degeneration [69], bias offset [70], source coupling [71] and triode transconductor [72] become either less effective or practical due to the nano-amp biasing current and exponential transfer function of the transistors. The novel transconductor proposed in this work exploits the drain resistance of saturate transistors to obtain wide input range and tunable trans-conductance. And the pseudo-differential structure allows operation with low supply voltage.

5.2 Circuit Design

The schematic of the proposed bump circuit with wide-input-range pseudo-differential transconductor is shown in Figure 5-1. In subthreshold, the current correlator M5-10 [66] computes a measure of the correlation of its two inputs (with a current scaling factor of 4):

$$I_{out} = 4\frac{I_1 I_2}{I_1 + I_2}. \tag{5.1}$$

The tunable transconductor (M1-M4 and I_W) converts the differential input V_{in1}, V_{in2} to current output I_1, I_2. The input transistors M1, M2 act as source follower. In subthreshold and assuming saturation, their source voltages are given by:

$$V_{s1,2} = \kappa V_{in1,2} - U_T \ln\left(\frac{I_{1,2}}{I_0}\right), \tag{5.2}$$

where $\kappa \approx 0.7$ is the gate coupling factor, $U_T \approx 26$ mV is the thermal voltage and I_0 is the pre-exponential current factor dependent on process and device dimension. In (5.2), the first term

indicates a linear relationship between $V_{in1,2}$ and $V_{s1,2}$, while the second term causes non-linearity. This non-linearity is mild as it is in a logarithm term. M3 (M4) serves as the current source for follower M1 (M2), its gate length is intentionally made smaller to exploit its channel length modulation. With first order approximation, the drain current in M3 is

$$I_D = I_{D0}(1 + \lambda \mathrm{V_{s1}}), \tag{5.3}$$

where $I_D = I_W + I_1$, λ is its channel length modulation coefficient, and I_{D0} is the drain current without channel length modulation, same for both M3 and M4. We utilize this dependence of I_D to V_{s1} to implement a large-value resistor tunable by current I_W. A common mode feedback (CMFB) circuit M11-M14 controls the gates of M3 and M4 to provide common mode rejection for the pseudo-differential structure and ensures that $I_1 + I_2 = I_H$. Combining this with (5.3), the output currents are:

$$\begin{aligned} I_1 &= \lambda \frac{2I_W + I_H}{2 + \lambda(V_{s1} + V_{s2})}(1 + \lambda V_{s1}) - I_W \\ I_2 &= \lambda \frac{2I_W + I_H}{2 + \lambda(V_{s1} + V_{s2})}(1 + \lambda V_{s2}) - I_W . \end{aligned} \tag{5.4}$$

Assuming a balanced input of $V_{in1} + V_{in2} = 2V_{cm}$, and that the second term in (5.2) can be

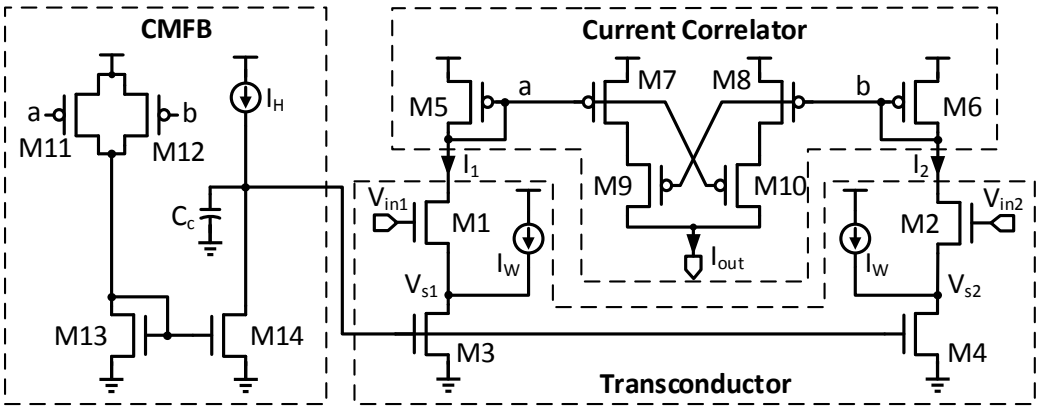


Figure 5-1: Schematic of the proposed tunable bump circuit.

neglected, the transconductance is given by:

$$g_m = \frac{d(I_1 - I_2)}{d(V_{in1} - V_{in2})} = \lambda\kappa\delta(2I_W + I_H), \text{where } \delta = \frac{1}{2 + 2\lambda\kappa V_{cm}}. \quad (5.5)$$

It can be seen that the transconductor is controlled by both I_W and I_H. The V_{cm} term in δ causes slight asymmetry in the bump transfer function, which is tolerable in the application. The pseudo-differential structure allows a wide differential input range, and the circuit can operate at very low supply voltage of about $V_{GS5} + 6U_T$.

When $V_{in1} = V_{in2}$, $I_1 = I_2 = 0.5I_H$, and the maximum bump current output (bump height) is given by $I_{out,max} = I_H$. With I_H fixed, changing I_W varies the transconductance of the transconductor, therefore changes the width of the bump. As I_1 and I_2 are linear related to the input voltages, the shape of the bump output is quadratic:

$$I_{out} = \frac{4}{I_H}\left[\lambda^2\gamma^2(2I_W + I_H)^2(1 + \lambda\kappa V_{in1})(1 + \lambda\kappa V_{in2}) - I_W(I_W + I_H)\right]. \quad (5.6)$$

5.3 *Measurement Result*

The proposed bump circuit is fabricated in a 0.13 μm CMOS process, thick oxide IO FETs are used to extend the V_{DD}, therefore the input dynamic range. The active area is 26×38 μm^2, shown in Figure 5-2, shown together is the test setup with data acquisition hardware and the

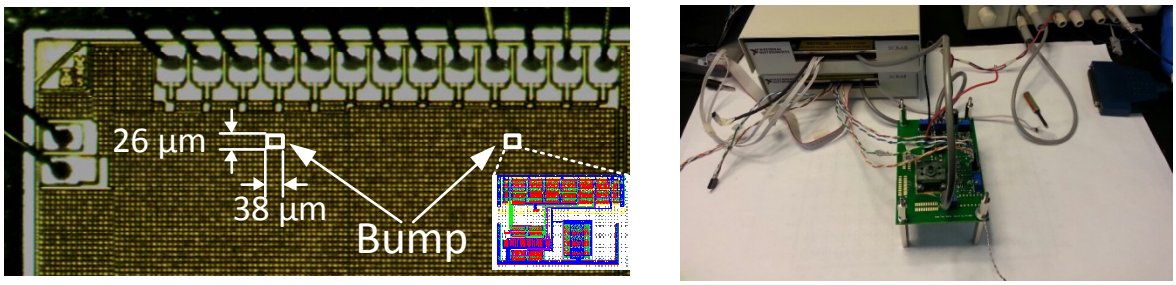
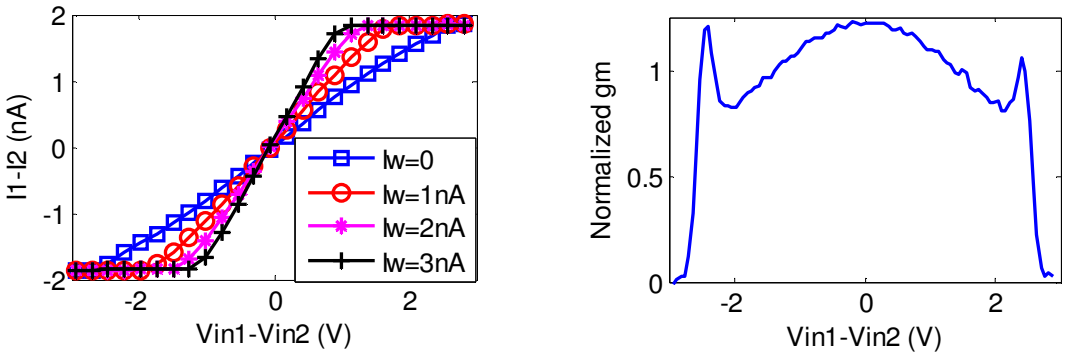


Figure 5-2: Bump circuit micrograph, layout, and the test setup.



(a) (b)

Figure 5-3: (a) Transconductor output, (b) normalized gm (I_W=0).

custom-designed test board. Biased at $I_W = I_H = 1$ nA and $V_{in1} = V_{in2}$, it consumes 6.3 nA current from the 3 V supply. The circuit is functional with V_{DD} down to 0.5 V, however the input range is limited at such low supply.

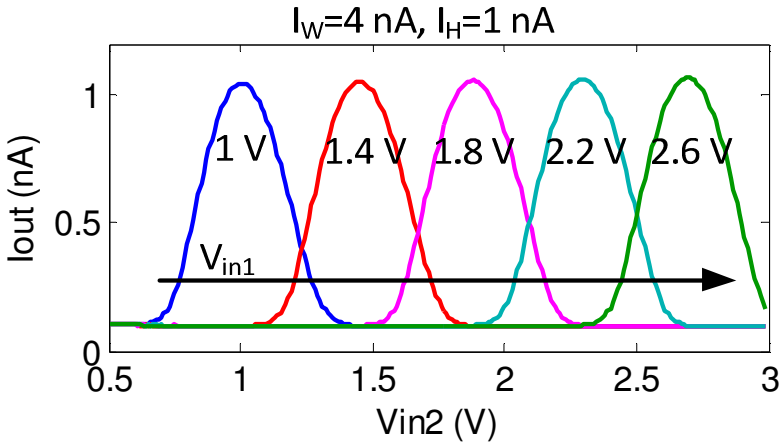
The transconductor outputs I_1 and I_2 are copied offchip by two additional PMOSs at node a and b, omitted in Figure 5-1. The differential output current with different I_W is plotted in Figure 5-3(a) with $I_H = 2$ nA, and balanced input voltage with $V_{cm} = 1.5$ V. The normalized g_m when I_W = 0 is plotted in Figure 5-3(b), showing an input range of 5 V with g_m error below 20%, covering almost the entire input common mode range. The nonlinearity can be attributed to the second term in (5.2), as well as the second order effects such as the dependence of λ on V_{DS}. It is tolerable in bump generator application as the bump output itself is an approximation of a highly nonlinear function. The offset of about 100 mV is due to device mismatch and can be calibrated out by utilizing floating gate techniques such as that described in Chapter 2.

The transfer functions of the bump circuit with regard to one input V_{in2} are plotted in Figure 5-4, showing variable center, width and height by varying V_{in1}, I_W, and I_H, respectively. Figure 5-4 also demonstrates that the circuit works properly with unbalanced input.

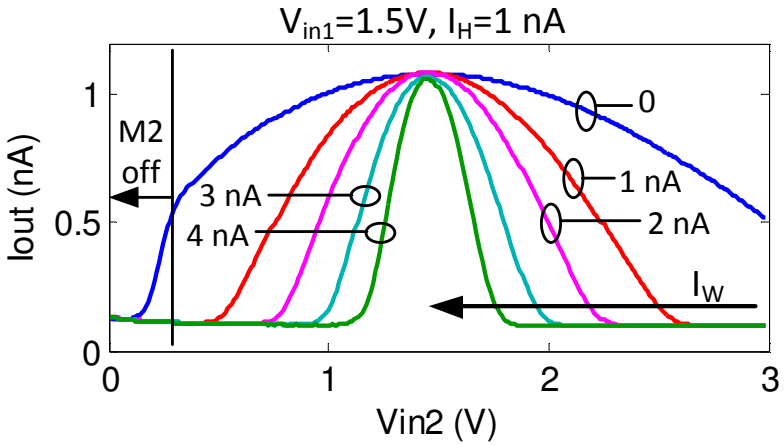
The 1-D bump output can be extended to high dimension to represent multivariate probability distribution by cascading multiple bump circuits, i.e., connecting I_{out} of one circuit to the I_H input of the next circuit. The measured 2-D bump output is plotted in Figure 5-5. Same as the 1-D case, each dimension's parameters are individually tunable.

To evaluate the computational throughput of the proposed bump circuit, the step response time is measured. With $I_W = I_H = 1$ nA, the response time for the output current to settle to 95% of its final value is 45 μs when the differential input steps from 0 V to 1 V.

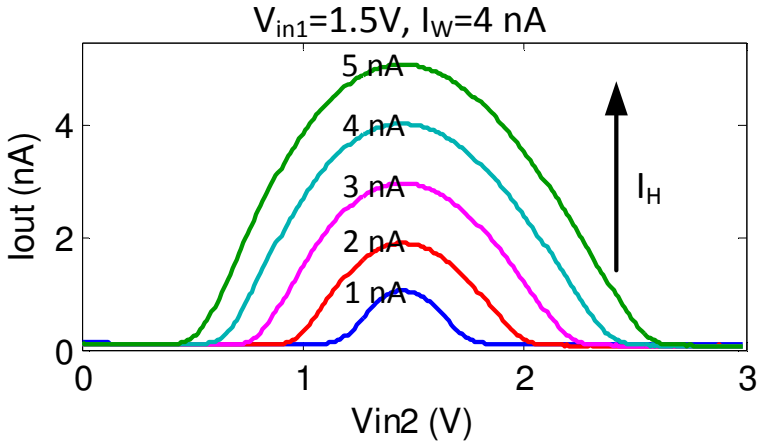
Table VI summarizes the measured performances of the proposed bump circuit. Compared to other recently reported works, the proposed circuit occupies smaller area and consumes significantly lower power.



(a)



(b)



(c)

Figure 5-4: **The measured bump transfer functions showing (a) variable center, (b) variable width, (c) variable height.**

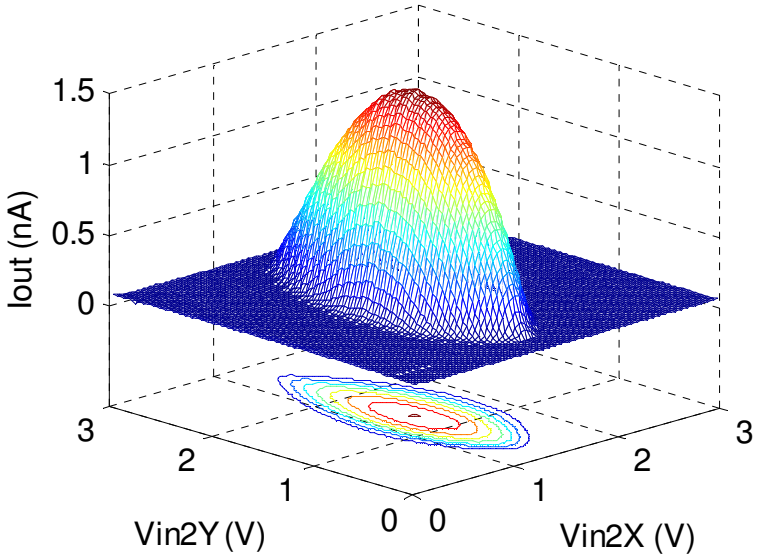


Figure 5-5: **The measured 2-D bump output with different width on x and y dimensions.**

Table VI. Performance Summary and Comparison of the Bump Circuit

	This work	**[35]**	**[67]**	**[68]***
Technology	**0.13 µm**	0.5 µm	0.13 µm	0.18 µm
Supply voltage	**3 V**	3.3 V	1.2 V	0.7 V
Power	**18.9 nW**	90 µW	10.5 µW	485 nW
Area	**988 µm^2**	3444 µm^2	1050 µm^2	-
Response time	**45 µs**	10 µs	-	9.6 µs

*: measurement results

Chapter 6 Conclusions and Future Work

This chapter summarizes this dissertation and proposes future work in this research area.

6.1 Conclusions

This dissertation investigates the implementation of machine learning systems with analog signal processing systems. The main conclusions are summarized below.

First, I presented a floating-gate current-output analog memory in a 0.13 μm standard digital CMOS process. The novel update scheme allows random-accessible control of both tunneling and injection without the needs for high-voltage switches, charge pumps or complex routing. The update dynamics is sigmoid, suitable for many adaptive and neuromorphic applications. FG model parameters have been extracted to facilitate predictive programming. Measurement and simulation shows that with 45 nW power consumption, the proposed memory achieves 7-bit programming resolution, 53.8 dB dynamic range and 86.5 dB writing isolation.

Second, I proposed an analog online clustering circuit. It uses the floating-gate memory I designed to achieve non-volatile storage. An analog computation block utilizes translinear principles to obtain 3 different distance metrics with significantly lower energy consumption than an equivalent digital implementation. A TD-LTA is proposed to improve energy efficiency, and an MA circuit implements a robust learning algorithm. The prototype circuit fabricated in a 0.13 μm digital CMOS process demonstrates unsupervised real-time classification, statistical parameter extraction and clustering of the input vectors with a power consumption of 15 μW.

Third, I developed an analog deep machine learning system, first reported in the literature to the best of my knowledge. It overcomes the limitations of conventional digital implementations by taking the efficiency advantage of analog signal processing. Reconfigurable current-mode

arithmetic realizes parallel computation. A floating-gate analog memory compatible with digital CMOS technology provides non-volatile storage. Algorithm-level feedback mitigates the effect of device mismatch. And system level power management applies power gating to inactive circuits. I demonstrated online cluster analysis with accurate parameter learning, and feature extraction in pattern recognition with dimension reduction by a factor of 8. In these tests, the ADE achieves a peak energy efficiency of 1 TOPS/W and an accuracy in line with the floating-point software simulation. The system features unsupervised online trainability, nonvolatile memory and good efficiency and scalability, making it a general-purpose feature extraction engine ideal for autonomous sensory applications or as a building block for large-scale learning systems.

Finally, I designed an ultra-low-power tunable bump circuit to provide similarity measures in analog signal processing. It incorporates a novel transconductor linearized using drain resistances of saturated transistor. I showed in analysis that the proposed transconductor can achieve tunable g_m with wide input range. Measurement results demonstrated 5 V differential input range of the transconductor with less than 20% of linearity error, and bump transfer functions with tunable center position, width and height. I also demonstrated 2-D bump outputs by cascading two bump circuits on the same chip.

6.2 Future Work

Based on this dissertation, the following can be considered for future research.

First, the energy efficiency can be further improved. One possible direction is to use lower power supply for the circuit. In this work, 3 V supply voltage is used mainly to achieve good tunneling isolation for the floating gate memory. For other computation circuits, a lower supply voltage can be domain used to save power. To accommodate for low supply, some of the circuits

need to be redesign to remove stacked transistors, and thin oxide transistors with lower threshold voltage can be used.

Second, a reconfigurable machine learning chip can be developed. The reconfigurability will allow the circuit to implement different machine learning algorithms based on the application requirement, making the system more flexible.

Third, a scaled-up version of the ADE can be implemented. This will help us to understand the effect of scaling of the system. And the larger-scale system will be able to solve more complex problems.

Finally, analog signal processing can be applied to other applications. One possible application is the analog classifier. It can be used to classify the rich features generated by the ADE and achieve a complete analog patter recognition engine.

References

[1] "Learning," Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Learning.

[2] K. P. Murphy, Machine learning: a probabilistic perspective, Cambridge, MA: The MIT Press, 2012.

[3] E. Alpaydın, Introduction to machine learning, 2nd ed., Cambridge, MA: The MIT Press, 2010.

[4] R. Bellman, Adaptive control processes: a guided tour, Princeton, NJ: Princeton University Press, 1961.

[5] I. Arel, D. Rose and T. Karnowski, "Deep machine learning - a new frontier in artificial intelligence research," *Computational Intelligence Magazine, IEEE,* vol. 5, no. 4, pp. 13-18, 2010.

[6] I. Arel, D. Rose and R. Coop, "Destin: A scalable deep learning architecture with application to high-dimensional robust pattern recognition," in *Proc. of the AAAI 2009 Fall Symposium on Biologically Inspired Cognitive Architectures (BICA)*, Nov. 2009.

[7] A. Coates, et al., "Scalable learning for object detection with gpu hardware," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on.*, 2009, pp. 4287-4293.

[8] R. C. Merkle, "Brain, Energy Limits to the Computational Power of the Human," [Online]. Available: http://www.merkle.com/brainLimits.html.

[9] M. Fischetti, "Computers versus Brains," Scientific American, 25 Oct. 2011. [Online]. Available: http://www.scientificamerican.com/article.cfm?id=computers-vs-brains.

[10] C. Mead, "Neuromorphic electronic systems," *Proc. IEEE,* vol. 78, no. 10, pp. 1629-1636,

Oct. 1990.

[11] M. Bohr, "A 30 year retrospective on Dennard's MOSFET scaling paper," *Solid-State Circuits Society Newsletter, IEEE,* vol. 12, no. 1, pp. 11-13, 2007.

[12] Y. Taur, "CMOS design near the limit of scaling," *IBM Journal of Research and Development,* vol. 46, no. 2.3, pp. 213-222, 2002.

[13] V. Subramanian, B. Parvais and J. Borremans, "Planar bulk MOSFETs versus FinFETs: an analog/RF perspective," *Eectron Devices, IEEE Transactions on,* vol. 53, no. 12, pp. 3071-3079, 2006.

[14] R. Sarpeshkar, "Analog versus digital: extrapolating from electronics to neurobiology," *Neural Comput.,* vol. 10, pp. 1601-1638, Oct. 1998.

[15] S. Young, J. Lu, J. Holleman and I. Arel, "On the impact of approximate computation in an analog DeSTIN architecture," *IEEE Trans. Neural Netw. Learn. Syst.,* vol. PP, no. 99, p. 1, Oct. 2013.

[16] W. Bialek, et al., "Reading a neural code," *Science,* vol. 252, pp. 1854-1857, June 1991.

[17] R. R. de Ruyter van Steveninck, et al., "Reproducibility and variability in neural spike trains," *Sicience,* vol. 275, pp. 2406-2419, 1997.

[18] M. van Rossum, "Effects of noise on the spike timing precision of retinal ganglion cells," *J. Neurophysiology,* vol. 89, pp. 2406-2419, 2003.

[19] M. Konijnenburg, et al., "Reliable and energy-efficient 1MHz 0.4V dynamically reconfigurable SoC for ExG applications in 40nm LP CMOS," *ISSCC Dig. Tech. Papers,* pp. 430-431, Feb. 2013.

[20] J.-S. Chen, C. Yeh and J.-S. Wang, "Self-super-cutoff power fating with state retention on a

0.3V 0.29fJ/cycle/gate 32b RISC core in 0.13μm CMOS," *ISSCC Dig. Tech. Papers,* pp. 426-427, Feb. 2013.

[21] J. Chang, et al., "A 20nm 112Mb SRAM in high-κ metal-gate with assist circuitry for low-leakage and low-VMin applications," *ISSCC Dig. Tech. Papers,* pp. 316-317, Feb. 2013.

[22] P. Pavan, et al., "Flash memory cells—an overview," *Proc. of IEEE,* vol. 85, no. 8, pp. 1248-1271, Aug. 1997.

[23] H. P. McAdams, et al., "A 64-Mb embedded FRAM utilizing a 130-nm 5LM Cu/FSG logic process," *IEEE J. Solid-State Circuits,* vol. 39, no. 4, pp. 667-677, Apr. 2004.

[24] "Overview for FRAM Series MCU," TI, [Online]. Available: http://www.ti.com/lsds/ti/microcontroller/16-bit_msp430/fram/overview.page. [Accessed Sep. 2013].

[25] T-Y. Liu, et al., "A 130.7mm^2 2-layer 32Gb ReRAM memory device in 24nm technology," *ISSCC Dig. Tech. Papers,* pp. 210-211, Feb. 2013.

[26] M. Jefremow, et al., "Time-differential sense amplifier for sub-80mV bitline voltage embedded STT-MRAM in 40nm CMOS," *ISSCC Dig. Tech. Papers,* pp. 216-217, Feb. 2013.

[27] J. Lu and J. Holleman, "A floating-gate analog memory with bidirectional sigmoid updates in a standard digital process," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2013, vol. 2, pp. 1600-1603.

[28] C. Diorio, "Neurally inspired silicon learning: from synapse transistors to learning arrays," Ph.D. dissertation, Caltech, Pasadena, CA, 1997.

[29] D. Kahng and S. M. Sze, "A floating-gate and its applications to memory devices," *The Bell*

System Technical Journal, vol. 40, pp. 1288-1295, July-Aug. 1967.

[30] L. R. Carley, "Trimming analog circuits using floating-gate analog MOS memory," *IEEE J. Solid-State Circuits,* vol. 24, no. 6, pp. 1569-1575, Dec. 1986.

[31] *IEEE Standard Definitions and Characterization of Floating Gate Semiconductor Arrays,* IEEE Std 1005-1991, 1991.

[32] M. Lenzlinger et al., "Fowler-Nordhiem tunneling in the thermally grown SiO2," *J. Appl. Physics,* vol. 40, p. 278, 1969.

[33] R. R. Harrison, J. A. Bragg, P. Hasler, B. A. Minch and S. P. Deweerth, "A CMOS programmable analog memory-cell array using floating-gate circuits," *IEEE Trans. Circuits Syst.II, Analog Digit. Signal Process.,* vol. 48, no. 1, pp. 4-11, Jan. 2001.

[34] B. K. Ahuja, et al., "A very high precision 500-nA CMOS floating-gate analog voltage reference," *IEEE J. Solid-State Circuits,* vol. 40, no. 12, pp. 2364-2372, Dec. 2005.

[35] S. Peng, P. Hasler and D. V. Anderson, "An Analog programmable multidimensional radial basis function based classifier," *IEEE Trans Circuits and Syst. I, Reg Papers,* vol. 54, no. 10, pp. 2148-2158, Oct. 2007.

[36] P. Hasler and J. Dugger, "An analog floating-gate node for Supervised learning," *IEEE Trans Circuits and Syst. I, Reg Papers,* vol. 52, no. 5, pp. 834-845, May 2005.

[37] M. Figueroa, S. Bridges, D. Hsu and C. Diorio, "A 19.2 GOPS mixed-signal filter with floating-gate adaptation," *IEEE J. Solid-State Circuits,* vol. 39, no. 7, pp. 1196-1201, July 2004.

[38] C. Diorio, "A p-channel MOS synapse transistor with self-convergent memory writes," *IEEE Trans. Electron Dev.,* vol. 47, no. 2, pp. 464-472, Feb. 2000.

[39] K. Rahimi, C. Diorio, C. Hernandez and M. Brockhausen, "A simulation model for floating-gate MOS synapse transistors," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2002, vol. 2, pp.532-535.

[40] J. Sanchez and T. DeMassa, "Review of carrier injection in the silicon/silicon-dioxide system," *IEE Proc. G–Circuits, Devices Systems,* vol. 138, no. 3, pp. 377-389, Jun. 1991.

[41] J. Lu, et al., "An analog online clustering circuit in 130nm CMOS," in *IEEE Asian Solid-State Circuits Conference*, 2013.

[42] D. J. C. MacKay, Information Theory, Inference and Learning Algorithms, New York, NY, USA: Cambridge University Press, 2003.

[43] S. Chakrabartty and G. Cauwenberghs, "Sub-microwatt analog VLSI trainable pattern classifier," *IEEE J. Solid-State Circuits,* vol. 42, no. 5, pp. 1169-1179, May 2007.

[44] R. Chawla, A. Bandyopadhyay, V. Srinivasan and P. Hasler, "A 531nW/MHz, 128x32 current-mode programmable analog vector-matrix multiplier with over two decades of linearity," in *Proc. IEEE Custom Integr. Circuits Conf. (CICC)*, Oct. 2004.

[45] J. Lubkin and G. Cauwenberghs, "A micropower learning vector quantizer for parallel analog-to-digital data compression," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 1998, pp. 58-61.

[46] K. Kang and T. Shibata, “An on-chip-trainable Gaussian-kernel analog support vector machine,” *IEEE Trans. Circuits Syst. I, Reg. Papers,* vol. 57, no. 7, pp. 1513-1524, Jul. 2010.

[47] Z. Wang, "Novel pseudo RMS current converter for sinusoidal signals using a CMOS precision current rectifier," *IEEE Trans. Instrum. Meas.,* vol. 39, no. 4, pp. 670-671, Aug.

1990.

[48] B. Gilbert, "Translinear circuits: a proposed classification," *Electron. Lett.,* vol. 11, no. 1, pp. 14-16, 1975.

[49] J. Lazzaro, S. Ryckebusch, M. A. Mahowald and C. Mead, "Winner-take-all networks of O(n) complexity," *Advances in Neural Information Processing Systems 1,* pp. 703-711, Morgan Kaufmann Publishers, San Francisco, CA, 1989.

[50] "Machine Learning Surveys," [Online]. Available: http://www.mlsurveys.com/.

[51] J. Bergstra, F. Bastien, O. Breuleux, P. Lamblin, R. Pascanu, O. Delalleau, G. Desjardins, D. Warde-Farley, I. Goodfellow, A. Bergeron and Y. Bengio, "Theano: deep learning on GPUs with Python," in *Big Learning Workshop, NIPS'11*, 2011.

[52] N. Cottini, M. Gottardi, N. Massari, R. Passerone and Z. Smilansky, "A 33 uW 64 x 64 pixel vision sensor embedding robust dynamic background subtraction for event detection and scene interpretation," *IEEE J. Solid-State Circuits,* vol. 48, no. 3, pp. 850-863, Mar. 2013.

[53] J. Holleman, A. Mishra, C. Diorio and B. Otis, "A micro-power neural spike detector and feature extractor in .13um CMOS," in *Proc. IEEE Custom Integrated Circuits Conf. (CICC)*, Sept. 2008, pp. 333-336.

[54] J. Oh, G. Kim, B.-G. Nam and H.-J. Yoo, "A 57 mW 12.5 μJ/Epoch embedded mixed-mode neuro-fuzzy processor for mobile real-time object recognition," *IEEE J. Solid-State Circuits,* vol. 48, no. 11, pp. 2894-2907, Nov. 2013.

[55] J. Park, I. Hong, G. Kim, Y. Kim, K. Lee, S. Park, K. Bong and H.-J. Yoo, "A 646GOPS/W multi-classifier many-core processor with cortex-like architecture for super-resolution

recognition," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2013, pp. 17-21.

[56] J.-Y. Kim, M. Kim, S. Lee, J. Oh, K. Kim and H.-J. Yoo, "A 201.4 GOPS 496 mW real-time multi-object recognition processor with bio-inspired neural perception engine," *IEEE J. Solid-State Circuits,* vol. 45, no. 1, pp. 32-45, Jan. 2010.

[57] R. Robucci, J. Gray, L. K. Chiu, J. Romberg and P. Hasler, "Compressive sensing on a CMOS separable-transform image sensor," *Proc. IEEE,* vol. 98, no. 6, pp. 1089-1101, June 2010.

[58] T. Yamasaki and T. Shibata, "Analog soft-pattern-matching classifier using floating-gate MOS technology," *Neural Networks, IEEE Transactions on,* vol. 14, no. 5, pp. 1257-1265, Sept. 2003.

[59] Y. Zhang, F. Zhang, Y. Shakhsheer, J. Silver, A. Klinefelter, M. Nagaraju, J. Boley, J. Pandey, A. Shrivastava, E. Carlson, A. Wood, B. Calhoun and B. Otis, "A batteryless 19 uW MICS/ISM-band energy harvesting body sensor dode SoC for ExG applications," *IEEE J. Solid-State Circuits,* vol. 48, no. 1, pp. 199-213, Jan. 2013.

[60] J. Lu, S. Young, I. Arel and J. Holleman, "A 1TOPS/W analog deep machine learning engine with floating-gate storage in 0.13μm CMOS," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2014, pp.504-505.

[61] S. Young, A. Davis, A. Mishtal and I. Arel, "Hierarchical spatiotemporal feature extraction using recurrent online clustering," *Pattern Recognition Letters,* vol. 37, pp. 115-123, Feg. 2014.

[62] S. Young, I. Arel, T. Karnowski and D. Rose, "A fast and stable incremental clustering

algorithm," in *proc. 7th International Conference on Information Technology*, Apr. 2010.

[63] J. Mulder, M. van de Gevel and A. van Roermund, "A reduced-area low-power low-voltage single-ended differential pair," *IEEE J. Solid-State Circuits,* vol. 32, no. 2, pp. 254-257, Feb. 1997.

[64] G. Reimbold and P. Gentil, "White noise of MOS transistors operating in weak inversion," *Electron Devices, IEEE Transactions on ,* vol. 29, no. 11, pp. 1722-1725, Nov. 1982.

[65] M. Pelgrom, A. C. J. Duinmaijer and A. Welbers, "Matching properties of MOS transistors," *IEEE J. Solid-State Circuits,* vol. 24, no. 5, pp. 1433-1439, Oct. 1989.

[66] T. Delbruck, "`Bump' circuits for computing similarity and dissimilarity of analog voltages," in *Proc. Int. Joint Conf. on Neural Networks*, Jul. 1991, pp. 475-479.

[67] K. Lee, J. Park, G. Kim, I. Hong and H.-J. Yoo, "A multi-modal and tunable Radial-Basis-Funtion circuit with supply and temperature compensation," in *Proc. IEEE Int. Symp. Circuits Syst.*, May 2013, pp.1608-0611.

[68] F. Li, C.-H. Chang and L. Siek, "A very low power 0.7 V subthreshold fully programmable Gaussian function generator," in *Proc. Asia Pacific Conf. on Postgraduate Research in Microelectronics and Electron.*, Sept. 2010, pp. 198-201.

[69] P. Furth and A. Andreou, "Linearised differential transconductors in subthreshold CMOS," *Electron. Lett. ,* vol. 31, no. 7, pp. 545-547, Mar. 1995.

[70] Z. Wang and W. Guggenbuhl, "A voltage-controllable linear MOS transconductor using bias offset technique," *IEEE J. Solid-State Circuits,* vol. 25, no. 1, pp. 315-317, Feb. 1990.

[71] A. Nedungadi and T. R. Viswanathan, "Design of linear CMOS transconductance elements," *IEEE Trans. Circuits Syst.,* vol. 31, no. 10, pp. 891-894, Oct. 1984.

[72] J. Pennock, "CMOS triode transconductor for continuous-time active integrated filters," *Electron. Lett.,* vol. 21, no. 18, pp. 817-818, Aug. 1985.

Vita

Junjie Lu was born in Shanghai, China on May 13, 1986. He received his B.S. degree in electrical engineering from Shanghai Jiao Tong University, China in 2007. From 2007 to 2010, he worked as a R&D engineer at Philips. He started his Ph.D. study in electrical engineering at the University of Tennessee, Knoxville in 2010. His research interests include low-power, high-performance analog and mixed-signal circuit design.