Energy-Efficient Neural Network Hardware Design and Circuit Techniques to Enhance Hardware Security

A DISSERTATION
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
UNIVERSITY OF MINNESOTA
BY

Muqing Liu

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Prof. Chris H. Kim, Adviser

May 2019

Acknowledgements

The body of research detailed in this dissertation would not have been possible without the support of multiple organizations. This research has been financially supported, in part, by the National Science Foundation under award number CNS-1441639 and THE Semiconductor Research Corporation under contract number 2014-TS-2560. Additionally, I need to acknowledge the valuable contribution of the University Programs of Cadence Design Systems, Inc., Mentor Graphics, and Synopsys, Inc. for providing access to design and verification tools utilized in this research.

I would like to express my thanks and appreciation to my committee members for their guidance, instruction, and encouragement. Prof. Chris H. Kim has been a valued advisor and mentor throughout my Ph.D. study. Prof. Kim’s guidance helped me in all the time of my research. Prof. Keshab Parhi, Prof. Sachin Sapatnekar, Prof. Maria Gini and Prof. Qi Zhao have graciously agreed to take part in my prelim and final committee, and I appreciate their perspective and guidance.

My colleagues at the University of Minnesota VLSI Lab are a great team to work with on a daily basis. I thank all my fellow lab mates for the stimulating discussions and for all the fun we have had in the last four years. I would like to extend my special thanks to Qianying Tang, Chen Zhou, Po-wei Chiu and Luke Everson for their help and contribution in this research. Their timely help and valuable discussion enabled us to meet critical deadlines and keep the project on time. I thank them for their investment of time and effort.

I am thankful to Luke Everson for introducing the opportunity to me to conduct research at IMEC, Belgium. I am also thankful to Dr. Dwaipayan Biswas at IMEC for the mentoring at IMEC. The valuable support, suggestion and feedbacks he gave me during my stay at IMEC helped me a lot in finishing the LSTM architecture design.

Most importantly, I express my deepest appreciation and thanks to my parents and for their patience, prayers, and support. They are a source of constant encouragement and have provided me with the opportunity to pursue this goal.

Dedication

I dedicate this work to my family, teachers and my mentor, Prof. Chris H. Kim, who is the person who took me into the academic world in the first place and has guided me through all the way in my Ph. D. study.

Abstract

Artificial intelligence algorithms and hardware are being developed at a rapid pace for emerging applications such as self-driving cars, speech/image/video recognition, deep learning, etc. Today's AI tasks are performed at remote datacenters, while in the future, more AI workloads are expected to run on edge devices. To fulfill this goal, innovative design techniques are needed to improve energy-efficiency, form factor, and as well as the security of AI chips.

In this dissertation, two topics are focused on to address these challenges: building energy-efficient AI chips based on various neural network architectures, and designing "chip fingerprint" circuits as well as counterfeit chip sensors to improve hardware security.

First of all, in order to deploy AI tasks on edge devices, we come up with energy and area efficient computing platforms based on both multi-layer perceptron (MLP) neural network and long short-term memory (LSTM) neural network. For the MLP neural network, we built a new neural network computation paradigm based on time domain computing. Our first time-based neural network prototype shows orders of magnitude less area and lower power consumption compared to traditional digital or analog domain approaches. A parallel two-layer MLP architecture realized using the proposed core achieves a 91% accuracy in handwritten digit recognition application using MNIST database. For the LSTM neural network, we proposed a binarized LSTM architecture, which greatly simplifies the circuit complexity and reduces the memory footprint, making it suitable to be deployed on edge devices. This proposed network is demonstrated using

an application of heart rate prediction from photoplethysmorgrahpy (PPG) signals, and achieves a good prediction accuracy.

Secondly, to enhance the security of the devices and ensure secure data communication between devices, we need to make sure the authenticity of the chip. Physical Unclonable Function (PUF) is a circuit primitive that can serve as a chip “fingerprint” by generating a unique ID for each chip. The stability of this generated ID is of utmost importance. We proposed a method to select the most stable cells in a large memory array to make sure the output of a memory based PUF is always consistent. Another source of security concerns comes from the counterfeit ICs, and recycled and remarked ICs account for more than 80% of the counterfeit electronics. To effectively detect those counterfeit chips that have been physically compromised, we came up with a passive IC tamper sensor. This proposed sensor is demonstrated to be able to efficiently and reliably detect suspicious activities such as high temperature cycling, ambient humidity rise, and increased dust particles in the chip cavity.

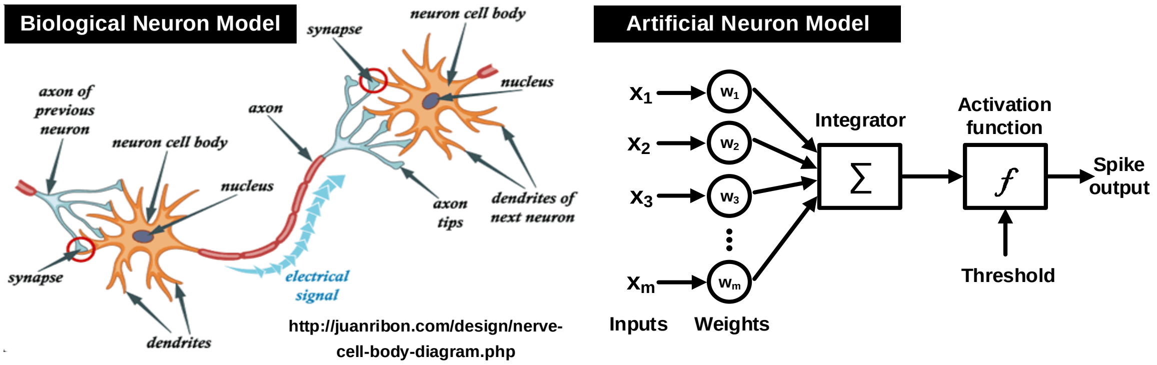
Table of Contents

List of Tables

List of Figures

Chapter 1. Introduction

Deep neural networks (DNNs) are now achieving super-human level performance in many real-life applications. For example, human Go masters turned out to be no match for Google's AlphaGo [1], and at the 2016 ImageNet challenge [2], computers could recognize images better than humans, with less than 3.5% [3] error. These breakthroughs are possible because of two primary reasons: (i) greatly improved computing power and (ii) advancements in convolution based DNN algorithms. By performing millions of convolution operations repeatedly, real-life applications such as image classification become possible. Neural networks in general have two operating phases: training (or learning) mode and inference (or prediction) mode. The weights are determined during the training mode, either supervised or unsupervised. The biological and artificial neuron models are shown in Fig. 1.1.



http://juanribon.com/design/nerve-cell-body-diagram.php

Fig. 1.1: Biological neuron model (left) and artificial neuron model (right).

DNN applications usually involve millions of weights, which may take weeks or months to obtain even on a super computer. Inference tasks are also compute intensive: for instance, the total number of multiply-and-accumulate operations required for AlexNet [2] is around 832 million with around 60 million weight parameters. Due to their high power consumption and extensive hardware requirements, most inference engines today run on the cloud (i.e. remote datacenter computers). This practice has worked so far; however moving forward, the current cloud computing model is expected to face critical challenges. For example, today's DNN algorithms are designed for cloud computing and therefore are sub-optimal for AI inference tasks performed on ultra-low power mobile devices. Thus, to deploy the AI tasks on mobile edge devices, we need to come up with efficient hardware architectures.

Meanwhile, if edge devices often handle sensitive data (i.e. medical, banking, image, voice) which can raise security concerns. As edge devices penetrate deeper into our daily lives, enhancing the security of these devices has become a critical design consideration. To address security concerns while sharing data between the edge devices and the cloud, we need to make sure that the devices involved in the communication are trustworthy.

A promising approach for ensuring trustworthiness of each device is embedding a "fingerprint" in each chip that is unique and unclonable. Physical unclonable function (PUF) [4] is a circuit that was introduced recently which harnesses inherent manufacturing variations to generate a chip fingerprint. Since manufacturing variations are random, uncontrollable, and unclonable, each device has a unique "fingerprint". One of the most important requirements for PUFs is to generate an output that is stable. A stable PUF circuit ensures that the PUF generates the same key every time, regardless of the temperature, voltage, number of year in service, etc. Another requirement for PUF is that it should be lightweight, which means it should not take a large area or require too much effort to design. Memory circuit is an attractive option for PUFs as it is readily available in practically all digital systems. The key is generated from the uninitialized power-up state of a memory cell. The power-up state is determined by the unique manufacturing variation.

While on the other hand, ensuring the authenticity of the devices might not be enough sometimes. There is another rising security concern induced by the counterfeit electronics, most of which are recycled and remarked ICs. They are usually recovered from old printed circuits boards (PCBs) and then relabeled and sold as new parts for profits; so they are authentic chips, but they can pose great concerns for customers, since they may function correctly at the beginning, but fails much earlier than expected. Detecting and preventing those counterfeit electronics from entering the market is a critical aspect of ensuring hardware security.

To fulfill the goal of running AI workloads efficiently and securely on edge devices, innovative computing architectures and circuit techniques have been proposed in this dissertation.

1.1 Time based MLP Neural Networks

The greatly improved hardware computing ability enables the deep learning algorithms to achieve great performances in many applications. To further improve the performance and energy efficiency for deep learning applications, a fully scalable light-weight integrate-and-fire neuromorphic core with brain-inspired leak and local lateral inhibition features is proposed in this dissertation. This is a fundamentally different way of implementing MLP DNNs: time domain computing, which is a much more efficient way of computation than the traditional method. The conventional approaches to implement MLP neural networks are based on bulky digital multipliers and adders to calculate the inner product, which usually has a higher precision but consumes more power and occupies more area. Therefore, the conventional approaches are not suitable for edge devices. In our proposed time based approach, we use a chain of inverters, which are one of the most basic and simplest digital circuits, with programmable delays as processing elements. The inner product function $y = \sum_i x_i \cdot w_i$ is computed purely in the time domain, where x_i is the input data, w_i is the synaptic weight and the individual stage delay is $x_i \cdot w_i$. The computation result is reflected as the accumulated delay of the inverter chain, and it can be measured by a simple readout circuit (i.e. frequency counter).

1.2 Binarized LSTM Architecture

Long short-term memory (LSTM) networks have shown to be successful in learning sequences of data. However, due to its high computation complexity and memory requirement, which leads to high power consumption, it is hard to deploy the LSTM networks on embedded devices. And Binarized Neural Networks (BNN) [5-6] become popular recently and have demonstrated to approach state-of-the-art classification accuracy. In this work, a binarized LSTM neural network is proposed and a hardware architecture is designed for accelerating this binarized LSTM network on embedded devices. This proposed network is demonstrated to be area and energy efficient, using an application of heart rate prediction from photoplethysmorgrahy (PPG) signals, and achieves a good heart rate prediction accuracy.

1.3 Chip Fingerprint based on Memory Circuits

PUF is a unique hardware that can generate a “fingerprint” for each chip based on the underlying manufacturing variation. The start-up value of an SRAM cell is unique, random, and unclonable as it is determined by the inherent process mismatch between transistors. These properties make SRAM an attractive PUF circuit for generating unique IDs/keys. The primary challenge for SRAM based key generation, however, is the poor stability when the circuit is subject to random noise, temperature and voltage changes, and device aging. Temporal majority voting (TMV) and bit masking were used in previous works to identify and store the location of unstable or marginally stable SRAM cells. However, TMV requires a long test time and significant hardware resources. In addition, the number of repetitive power-ups required to find the most stable cells is prohibitively high. To overcome the shortcomings of TMV, we propose a novel data remanence based technique to detect SRAM cells with the highest stability for reliable key generation. This approach requires only two remanence tests: writing ‘1’ (or ‘0’) to the entire array and momentarily shutting down the power until a few cells flip. We exploit the fact that the cells that are easily flipped are the most robust cells when written with the opposite data. The proposed method is more effective in finding the most stable cells in a large SRAM array than a TMV scheme with 1,000 power-up tests. Experimental studies show that the 256-bit key generated from a 512 kbit SRAM using the proposed data remanence method is 100% stable under different temperatures, power ramp up times, and device aging.

1.4 Passive Counterfeit IC Sensor

The rising security and reliability concerns induced by the counterfeit electronics necessitate the design to efficiently identify counterfeit electronics from the complex global supply chain nowadays, which is extremely challenging. To help alleviate this challenge, we present an embedded flash (eflash) memory based powerless non-volatile tamper sensor for efficiently detecting counterfeit ICs. By exposing the floating gate (FG) node of a logic-compatible eflash cell to the environment, the proposed sensor can record any subtle physical event that affects the charge stored on the exposed FG. The proposed sensor is demonstrated in both 65nm and 0.35μm standard CMOS technologies, proving that this technique is agnostic to different technology processes. Extensive test results confirm that suspicious activities such as temperature charge injection, humidity rises, and increased dust particle density in the cavity can be recorded powerlessly using the proposed sensor.

1.5 Summary of Dissertation Contribution

Several contributions have been made in this dissertation to improve the efficiency of the state-of-the-art neural network hardware architectures and enhance the security for hardware devices.

To summarize the key contributions of this research: 1) a time-based computing architecture has been proposed, which is much more area and energy efficient compared to the traditional digital or analog computing schemes. 2) To deploy the LSTM neural network, which has high computation complexity and memory requirement, on embedded devices, a binarized LSTM architecture is proposed. It greatly simplifies the circuit complexity and reduces the memory footprint of the LSTM neural network. 3) A data remanence based stable cell selection technique is presented to be able to generate 100% stable responses from a SRAM based PUF circuit. 4) A passive counterfeit electronics detection sensor is demonstrated to be effective in detecting abnormal physical attacks to the chip, thus identifying the counterfeit electronics that have been physically attacked.

The remainder of this dissertation is organized as follows. Chapter 2 presents the design details of the time based MLP neural network in 65nm CMOS technology and the measurement results of the digit recognition application. Chapter 3 demonstrates the basic idea of the proposed binarized LSTM neural network and the design overview of the hardware architecture. The simulation results of the heart rate prediction from PPG signals is also presented. Chapter 4 discusses the proposed data remanence based stable cell selection technique, and the measurement results from commercial SRAM chips. Chapter 5 illustrates the proposed counterfeit IC sensor implemented in both 65nm and 0.35µm technology. The physical attack measurement results are also demonstrated. Finally chapter 6 summarizes this dissertation.

Chapter 2. Time-based Integrate-and-Fire Neuromorphic Core Design

2.1 Introduction

Deep learning is a sub-class of machine learning algorithms, which is a brain-inspired computing algorithm. Deep learning architectures always include multiple layers of non-linear processing units, and the layers used in deep learning mostly refer to the hidden layers of an artificial neural network [7]. Artificial neural network is not a new concept, it has been around for decades [8]. The development of artificial neural network was slow in the past until the computers have gained greater computing power recently. The artificial neural network is based on a simple artificial neuron model, which is a weighted sum of the inputs, followed by an activation function. These artificial neurons are the basic processing units of the neural network and a huge amount of neurons are working in parallel to handle complex algorithms in real applications. There are two stages in deep learning tasks, training (or learning) and inference (or prediction), and the weights is determined by the training process. In this work, we focus on the inference phase. Deep learning applications usually involves millions of weights, which costs a lot of time and energy to access from memory, and large amount of parallel matrix operations, which is also a demanding task for processors. Nowadays, as the processing power of the computers are developing at an extremely fast pace, the implementation of such large scale network becomes possible. The challenge is how to perform the task efficiently so that it can be applied in mobile applications and in real-time embedded systems.

There are several common hardware implementations of artificial neural networks, including using CPU, GPU and custom ASIC. Since CPUs are optimized for latency, they are more suitable for sequential operations, so it’s not the best choice to use CPUs in massively parallel neural network applications. GPUs are optimized for memory bandwidth and the thread parallelism in GPUs can hide memory access latency, so GPUs are better suited for deep learning than CPUs. However, one big drawback of using GPUs to implement neural networks is that it always has a high power consumption, making it not suitable for mobile applications. Many researchers believe that major improvements of energy efficiency and performance should come from the specially designed hardware [9]. In [9], the Tensor Processing Unit (TPU), which is a custom ASIC developed by Google, achieved a peak throughput of 92 TeraOps/second (TOPS), which is on average about 15X-30X faster than its contemporary GPU or CPU.

Many approaches have been presented to implement the neural processing elements (PE) in custom ASIC hardware, aiming at higher power and area efficiency. Early approaches relied on analog circuits to mimic synapse and neuron functions [10]. However, using analog circuits to implement brain-inspired neural networks suffers from noise and process variation issues, so homogeneity and precision cannot be guaranteed for large scale networks. Scaling of CMOS technology also poses a challenge for analog circuits. Digital implementation of neural processing elements has been more popular recently. Compared to analog implementation, they are more robust against noise and process variation, and can benefit from technology scaling, enabling massively parallel neuromorphic ASIC systems, such as IBM's Turenorth [11]. However for digital implementation of neural networks, it usually requires a large amount of adders and multipliers to implement the MAC operations, which is area and power consuming.

There are three types of popular neural networks: Multi-Layer Perceptron (MLP), Convolutional Neural Network (CNN), and Recurrent Neural Network (RNN). According to [9], they represent 95% of the inference workload in Google's datacenters. While there are many works focusing on accelerating CNNs [12]-[18], they represent 5% of Google's datacenter workload [9]. We have dedicated to improve the efficiency in Multi-Layer Perceptron (MLP) neural network in this work.

In this work, we present an implementation of fully connected MLP neural network based on integrate-and-fire neuron model in time domain, which improves the area and power efficiency compared to digital implementation. Instead of using adders and multipliers to implement the multiply and accumulate operations, we do the calculation in time domain using digital circuits, which is more area and power efficient. And data moving can be more energy consuming than computation, as is reported in [19]-[20], so to reduce the memory access power and latency, each PE in our work has its exclusive weights, which eliminates the energy consumption of moving data from memory. This further improves the energy efficiency.

2.2 MLP Neural Network Background

In this section, we first briefly introduce the architecture of MLP neural network, then we discuss the integrate-and-fire neuron model we used in this work.

2.2.1 Multi-Layer Perceptron (MLP) Neural Network

Multi-layer perceptron (MLP) neural network is a popular feedforward neural network with one or more hidden layers between input and output layers [21], which is shown in Fig. 2.1 (a). Each node in this MLP represents an artificial neuron model that uses a non-linear activation function, shown in Fig. 2.1 (b). A supervised learning technique called backpropagation is used to train a MLP neural network. Two common activation functions for MLP are hyperbolic tangent $y(u_i) = \tanh(u_i)$ and the logistic function $y(u_i) = (1 + e^{-u_i})^{-1}$, where $y(u_i)$ is the output of the ith node (neuron), and u_i is the weighted sum of the inputs of the ith node (neuron). They are both sigmoids, so the shape of the two functions are similar, the difference is that the former function is centered around 0, ranging from -1 to 1, while the latter one ranges from 0 to 1. Most of the time, tanh can be more quickly converged than logistic function, and performs better accuracy [22]. And it is a good idea to make the input, output and hidden layers have mean values of 0 and standard deviation of 1, since in general if the average input shifts away from zero, it will bias the weight updates in a particular direction, making the learning slower [23]. However, due to the fact that we need to apply the MLP neural network in hardware and there is no negative inputs in our design, so we used the logistic function with range from 0 to 1 as the activation function in this work

.

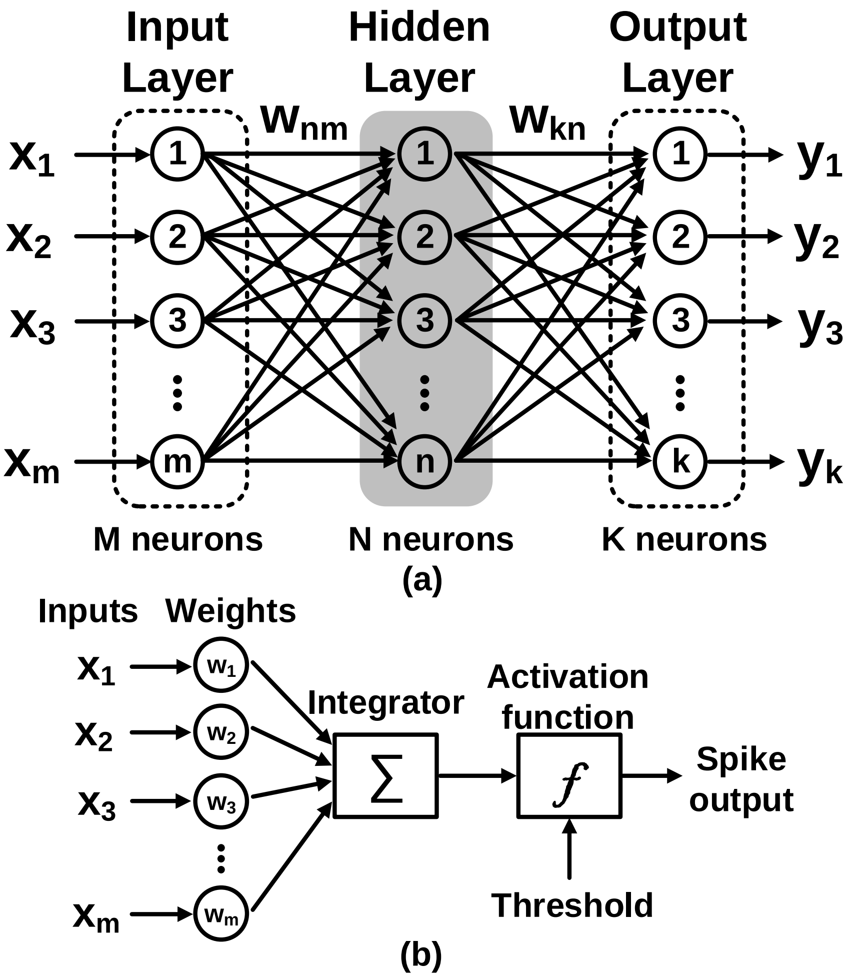


Fig. 2.1: Architecture of multi-layer perceptron (MLP) neural network with one hidden layer (a) and artificial neuron model (b).

2.2.2 Integrate-and-fire Neuron Model

Integrate-and-fire neuron model is one of the earliest models of a neuron, which was first investigated in 1907 by Louis Lapicque [24]. A neuron can be modeled by

$$I(t) = C_m \frac{dV_m(t)}{dt} \tag{2.1}$$

which is just like a capacitor. However, the neuron cell membrane is not a perfect insulator, the charges will leak away through the membrane over time. So to be more accurate, we should add a leaky term in the model [25],

$$I(t) = C\frac{dV_m(t)}{dt} + \frac{V_m(t)}{R} \tag{2.2}$$

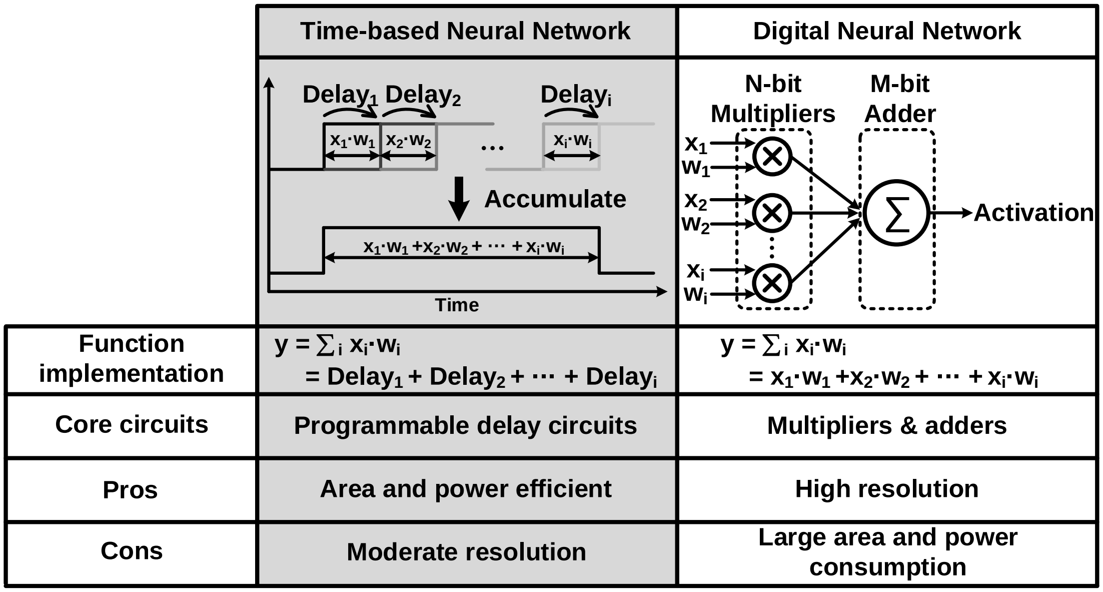
which is equivalent to a capacitor C in parallel with a resistor R. This model is also referred to as leaky integrate-and-fire (LIF) neuron model. In this work, we implemented the integrate-and-fire neuron model with leaky feature as a selective option.

2.3 Time-based Integrate-and-Fire DCO Neuromorphic

In this section, we first introduce the advantages of using time-based circuits to implement neural networks over digital implementation. Then we present the detailed implementation and characterization of our proposed time-based neuromorphic core [26].

2.3.1 Advantages of Time-based Implementation of Neural Network

Time-based circuits have several known advantages, since time is also an analog signal, so it has higher area and energy efficiency than digital implementation in low-precision computation applications [27], the time-based circuits is implemented using the same standard digital logic gates as digital circuits, so compared to voltage mode or current mode analog computing, it has excellent compatibility with advanced CMOS technologies and can tolerate low operating voltages. Besides, due to its digital nature, time-based circuits is compatible with EDA tools, thus can allow for large-scale integrated design [28].



	Time-based Neural Network	Digital Neural Network
Function implementation	$y = \sum_i x_i \cdot w_i$ $= \text{Delay}_1 + \text{Delay}_2 + \cdots + \text{Delay}_i$	$y = \sum_i x_i \cdot w_i$ $= x_1 \cdot w_1 + x_2 \cdot w_2 + \cdots + x_i \cdot w_i$
Core circuits	Programmable delay circuits	Multipliers & adders
Pros	Area and power efficient	High resolution
Cons	Moderate resolution	Large area and power consumption

Fig. 2.2: Comparison of time-based and digital implementation of neural networks.

Fig. 2.2 shows the comparison of time-based and digital implementation of neural networks. In this work, we use the programmable delay circuits as the processing elements, calculating the multiplication results, and the integration is implemented by accumulating the delay in time domain. Compared to traditional digital implementation which requires lots of adders and multipliers, time-based implementation is more area and power efficient. We can achieve 16bits/32bits fixed point or even floating point computation using digital implementation, while time-based circuits usually has a lower resolution, which is a main drawback. However, for deep learning applications, the resolution is not so critical. Recently, the binarized neural network (BNN) [5-6] has gained popularity and it has approached state-of-the-art classification accuracy. BNNs are neural networks with binary (1-bit) weights and activations. This proves that neural networks with lower resolution can also work well and it's more favorable for hardware implementation. There have been several successful hardware implementation of BNNs in time domain which show good energy and hardware efficiency [29-31].

2.3.2 Implementation of Time-based DCO Neuromorphic Core

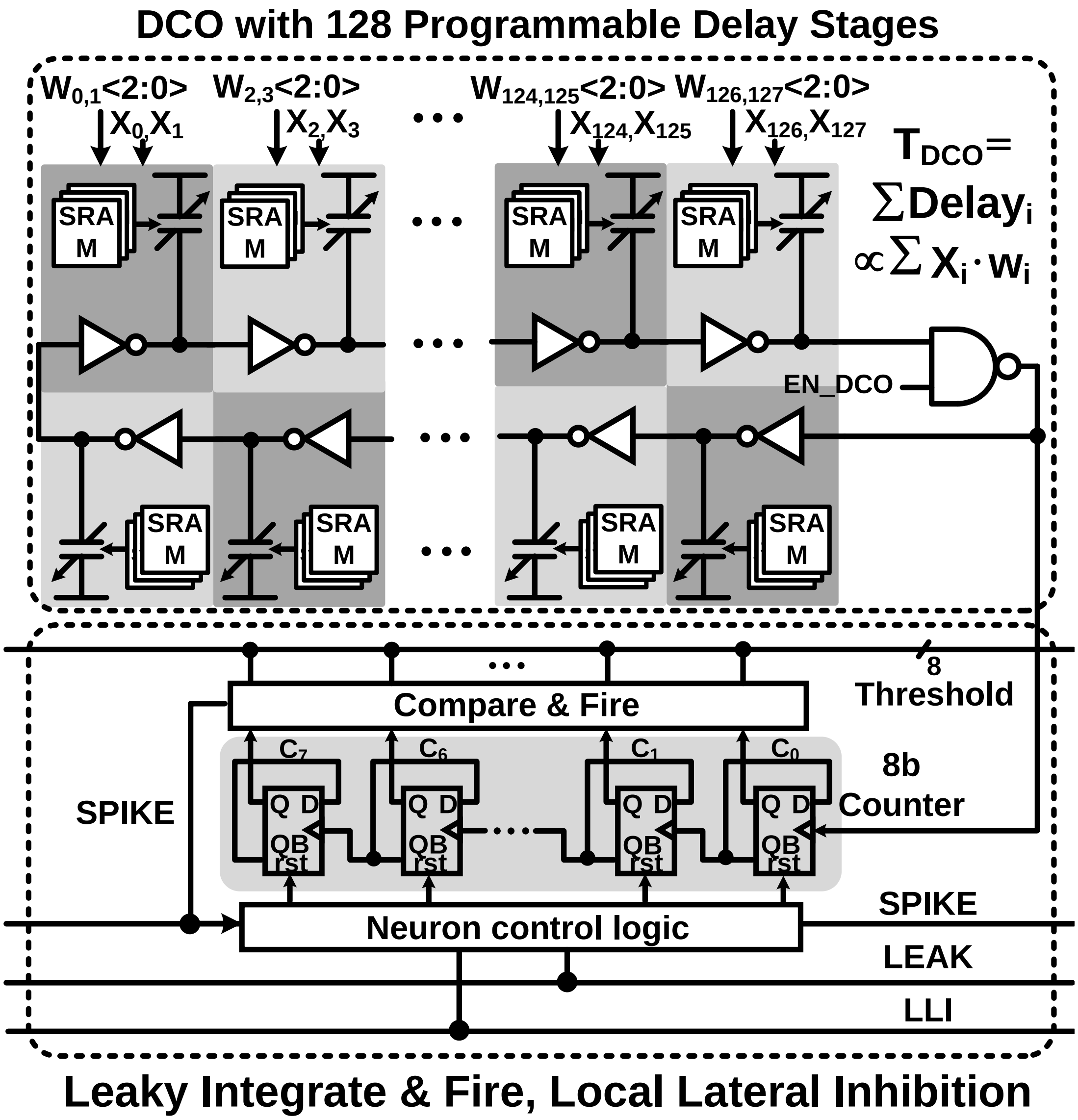


Fig. 2.3: Circuit diagram of the proposed time-based integrate & fire (I&F) DCO neuromorphic core.

Fig. 2.3 shows the proposed time-based integrate-and-fire neuromorphic core [26]. The main innovation is that it computes $y = \sum_i x_i \cdot w_i$ purely in time domain, where x_i is the input data and w_i is the synaptic weight. The upper part is a digitally controlled oscillator (DCO) with 128 programmable delay stages and one enable stage, which can compute up to 128 multiply and accumulate (MAC) operations at a time, and each programmable delay stage is the processing element (PE). The lower part is the readout circuits, implementing the leaky integrate and fire and local lateral inhibition features. Each PE computes one $x_i \cdot w_i$, and the computation result is converted to the delay of that stage. Delay of all stages are accumulated naturally in the DCO loop, which is the integration of the multiplication results. The overall delay in the DCO loop is converted to an oscillation frequency and fed to an 8-bit counter in the readout circuits. The counter increments every DCO cycle, and when the count value reaches a target count, which corresponds to the spiking threshold, a spike is generated and the counter is self-reset. The compare & fire block checks the current counter count and if the count matches the threshold, a pulse is generated as the spike. The spike is fed into the neuron control logic and the reset signal is generated to clear the counter. The output of each DCO unit is the number of spikes generated within a certain sampling period. The measurement precision of the time based DCO neuromorphic core can be easily programmed by changing the spiking threshold. For instance, with a higher spiking threshold, a smaller DCO frequency difference can be detected at the cost of longer delay, thus higher energy consumption.

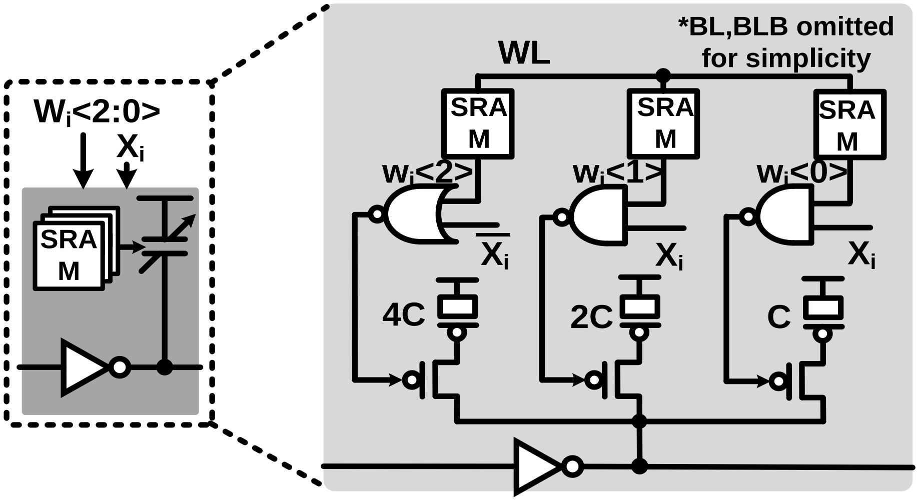


Fig. 2.4: Detailed implementation of one processing element, which is a programmable delay stage.

x_i	w_i	# Cap.	State	x_i	w_i	# Cap.	State
0	001	4C	Default	1	001	1C	Excitatory
0	010	4C		1	010	2C	
0	011	4C		1	011	3C	
0	100	4C		1	100	4C	Default
0	101	4C		1	101	5C	Inhibitory
0	110	4C		1	110	6C	
0	111	4C		1	111	7C	

Fig. 2.5: Load capacitor configuration table of all possible states.

Fig. 4 shows the detailed implementation of one processing element (PE), which is a programmable delay stage of the DCO core. Our design has a weight resolution of 3 bits, so each stage accepts 1-bit pixel x_i and 3-bit weight w_i as inputs. Each stage of the DCO core is composed of an inverter and binary-weighted MOSFET capacitors controlled by the input pixel and the 3-bit weight. Input pixels determine whether a stage is activated or not, and weights determine how many capacitors are turned on as load in that stage. Since there are both excitatory and inhibitory synapses, the delay should be able to change in both directions. So weight 100_2 is defined as weight zero, it's also the default weight when the delay stage is disabled. If weights are smaller than 100_2 (i.e. 001~011_2), fewer load capacitors are turned on, reducing the delay of that stage. These weights represent excitatory synapses. Contrarily, weights larger than 100_2 (i.e. 101~111_2) represent inhibitory synapse. Fig. 5 shows all the possible load capacitor configurations and the corresponding states. To save the power consumption of data movement during computation, each PE has its exclusive memory for use. In each programmable delay stage, there are three SRAM cells storing the 3-bit weight for each PE. This architecture is called completely parallel or fully spatially unrolled architecture [29-30].

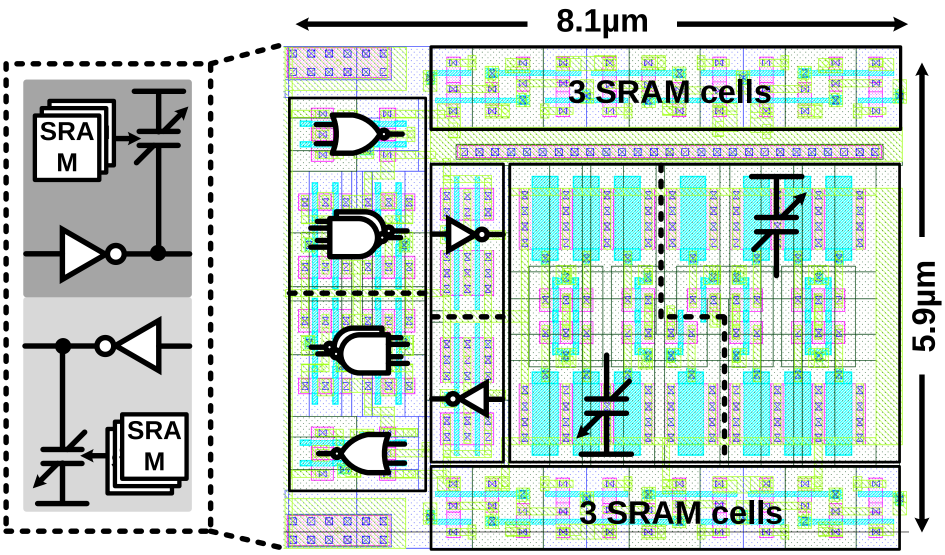


Fig. 2.6: Unit cell layout of two processing elements.

The unit cell layout that is composed of two PEs is shown in Fig. 2.6. The two PEs are made as symmetric as possible to reduce the process variation induced delay mismatch between different stages. Each unit cell has a size of 8.1μm by 5.9μm, so every PE takes an area of approximately 24μm^2 (in 65nm process), which is very compact compared to the bulky adders and multipliers in digital implementation. So this time-based implementation of neural networks can be more area and power efficient than traditional digital implementation. And because of the fully spatially unrolled architecture, after loading the weights in the SRAM cells initially, the weights don't change during the entire inference application period, so there's no power consumption for moving the data from memory to PEs. This can further increase the energy efficiency of this core.

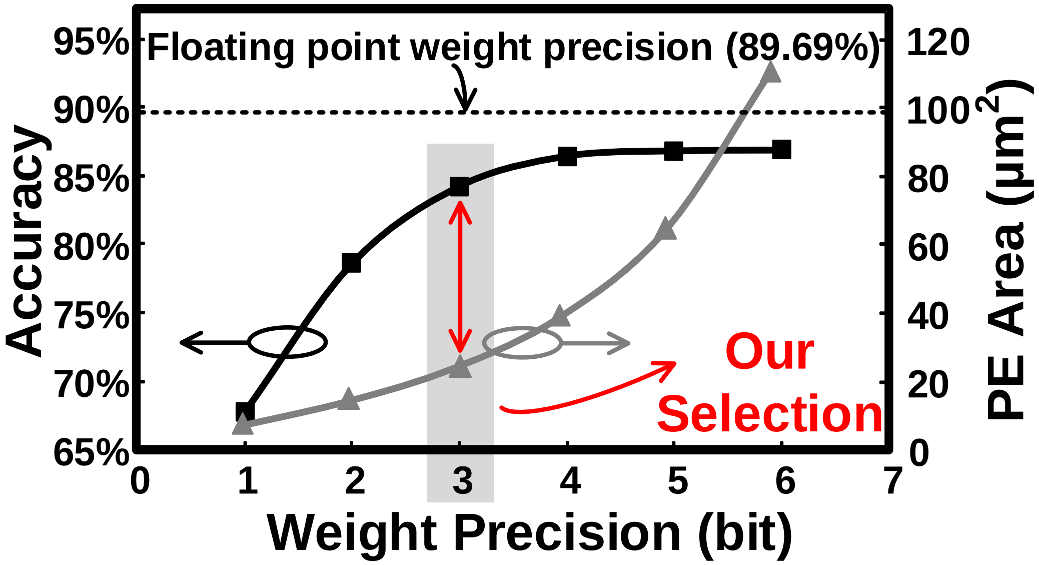


Fig. 2.7: Neural network accuracy and processing element (PE) area with different weight precision.

The reason why we use the 3-bit weight is based on the accuracy and circuit area and power tradeoff, which is shown in Fig. 2.7. We train a single layer perceptron neural network for digit recognition appication. The accuracy in the dashed line is for the floating number weight we got from training, and the black curve shows the accuracy when we round the weight to different number of bit precisions. The grey curve shows the corresponding area estimation under different weight precisions, except for the 3-bit case. The area and power consumption increases exponentially with weight precisions, however the accuracy doesn't increase that much. According to our study, 3-bit weight has the best accuracy and area tradeoff, so we decide to use 3-bit weights in this work.

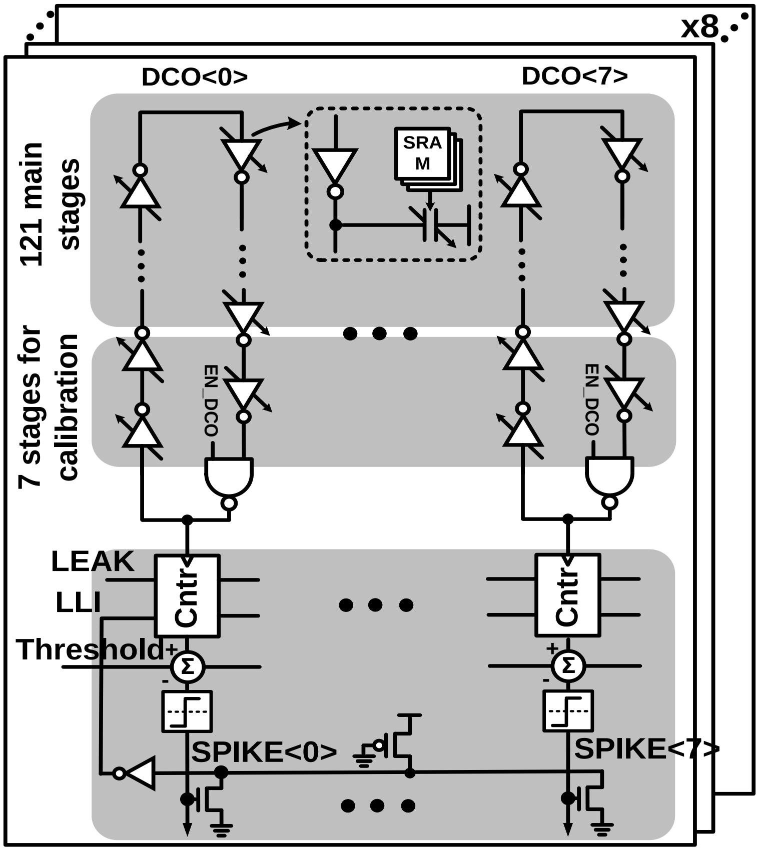


Fig. 2.8: Architecture of the proposed time-based DCO neuromorphic.

Fig 2.8 shows the overall architecture of the proposed time-based neuromorphic core with 64 DCO circuits array in parallel. The DCO array is divided into 8 groups, each consisting of 8 DCOs, to realize the local lateral inhibition feature (discussed in next section). Each DCO can be enabled or disabled independently, so any number of DCOs can be activated simultaneously according to different applications, demonstrating the scalability of this proposed neural network. The proposed neuromorphic core compares the raw spike count of each DCO to determine which neuron output is dominant. If we define the multiply and accumulate computation results as the score of each neuron, so the higher the score, the more likely it's the correct prediction. In our proposed neuromorphic core, computation results are represented by the overall delay in the DCO loop, and the DCOs with higher score has smaller delay (higher frequency), thus can generate more spikes. The dominant DCO with the most spikes is the final prediction result. However, due to process variation, different DCOs have different oscillation frequencies for identical inputs. It is critical to have a uniform DCO frequency to start with. Unlike process variation, voltage and temperature variation affect all DCOs in the same direction, so although the absolute spike counts may vary, the dominant DCO will always stay the same under V and T variations. To compensate for process variation, in our design, 7 of the 128 DCO stages are reserved for frequency calibration while the remaining 121 stages are used for the normal neural network function. For frequency calibration, all DCOs are configured to have the same pixel inputs and weights, and the spike counts are measured for a fixed period of time. By tuning the weights of the 7 frequency calibration stages, we can get a uniform baseline DCO frequency. And this is a one-time calibration, since after this calibration, further voltage and temperature variations affect all the DCO frequencies in the same way, the relative order of different DCOs won't change, so no further calibration is required. Fig. 2.9 shows the measurement results of the frequency calibration of 10 DCOs. After calibration, the frequency variation reduces from 1.17% to 0.10%. Although due to jitters and noises, the DCO frequency will vary from time to time, as long as the frequency variation is smaller than the frequency mismatch between different DCOs, the final prediction result will not be affected. And the frequency mismatches between different DCOs are determined by the weights obtained in the training process, which can be tuned and is measured to be much larger than 0.10%.

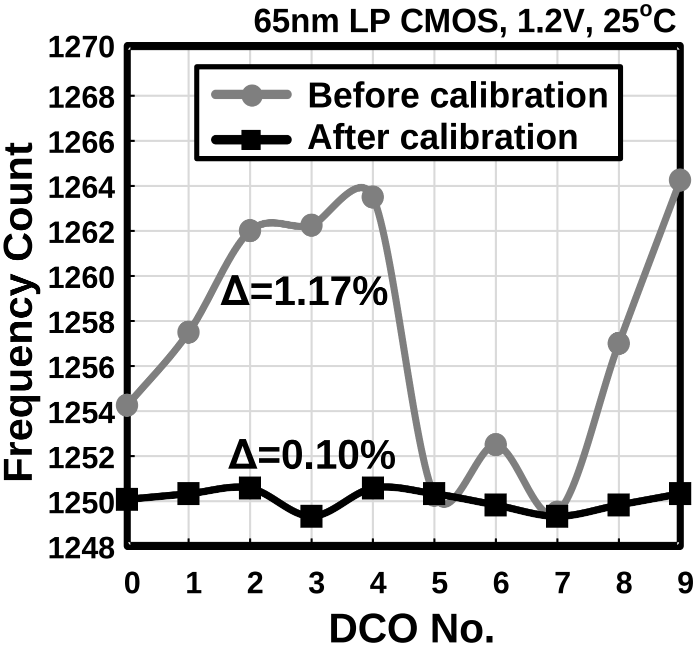


Fig. 2.9: Measurement results of DCO frequency calibration.

2.3.3 Characterization of DCO Neuromorphic Core

Before applying this time-based neuromorphic core in real applications, we need to do some further characterization to ensure the accuracy of the computation results. Linearity of the DCO frequency is one important factor that affects the computation accuracy and it determines whether the hardware results match the software simulation results. Fig. 2.10 shows the measurement results of DCO linearity characterization. It plots the mean and 3σ error bars of the frequency count when different number of stages are activated. The gray curve shows the case when only 1 out of 128 DCO delay stages is enabled, the rest stages are all turned off. In this case, when the weight is swept from 000 to 111, only 1 unit of weight is changed every time, so it corresponds to the minimum tunable frequency this DCO can achieve. The measurement results confirm a good linearity.

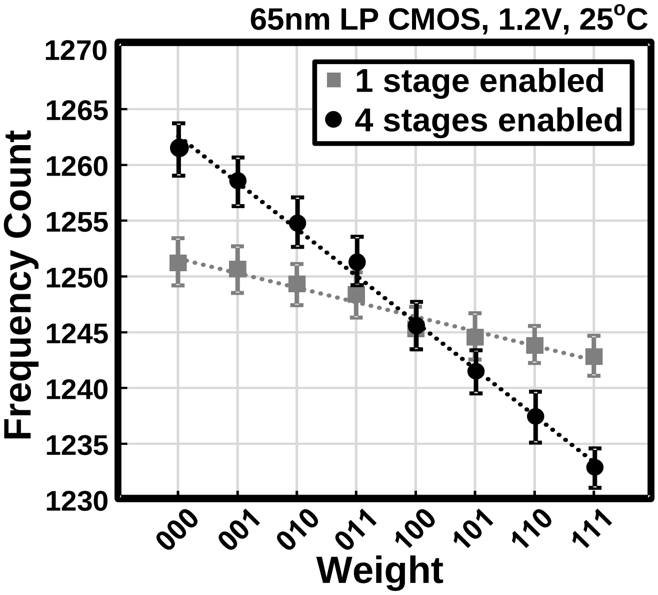


Fig. 2.10: Measurement results of DCO linearity characterization.

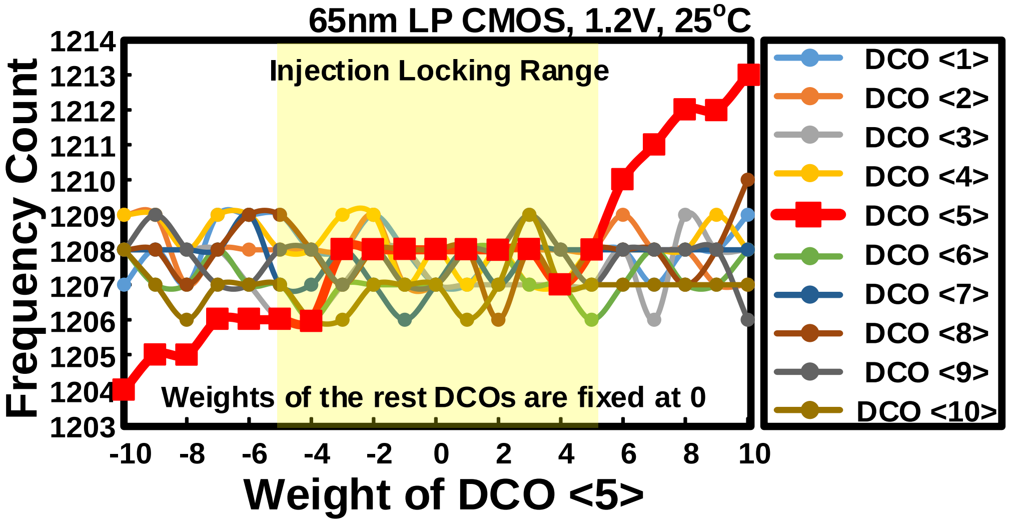


Fig. 2.11: Measurement results of injection locking phenomenon in adjacent DCOs.

Due to the fact that in this work an array of 64 oscillators can oscillate at the same time with similar oscillation frequencies, and they are in a close vicinity to each other, so injection locking might happen. To measure whether there's injection locking in this proposed architecture, we select 10 adjacent DCOs, turn on all the stages and vary the weight of the middle DCO, weights of the rest DCOs are fixed at 0. The measurement results are shown in Fig. 2.11. As the weight of the middle DCO was swept from -10 to 10, we observed that in the range of -5 to 5, when the weight of the middle DCO changed, its oscillation frequency changed little and was similar to its adjacent DCOs, so we believed that when the difference of the computation results between adjacent DCOs were smaller than 5, there was injection locking in this architecture. This injection locking phenomenon is not necessarily a bad thing in our system. Since the measurement results of classification applications show that the output difference between the dominant DCO and the rest DCOs is larger than 5, so the dominant DCO will never be affected by injection locking. And it's a favorable phenomenon that the rest DCOs are locked at a similar frequency, which makes it easier to select the dominant DCO.

2.4 Leak and Local Lateral Inhibition

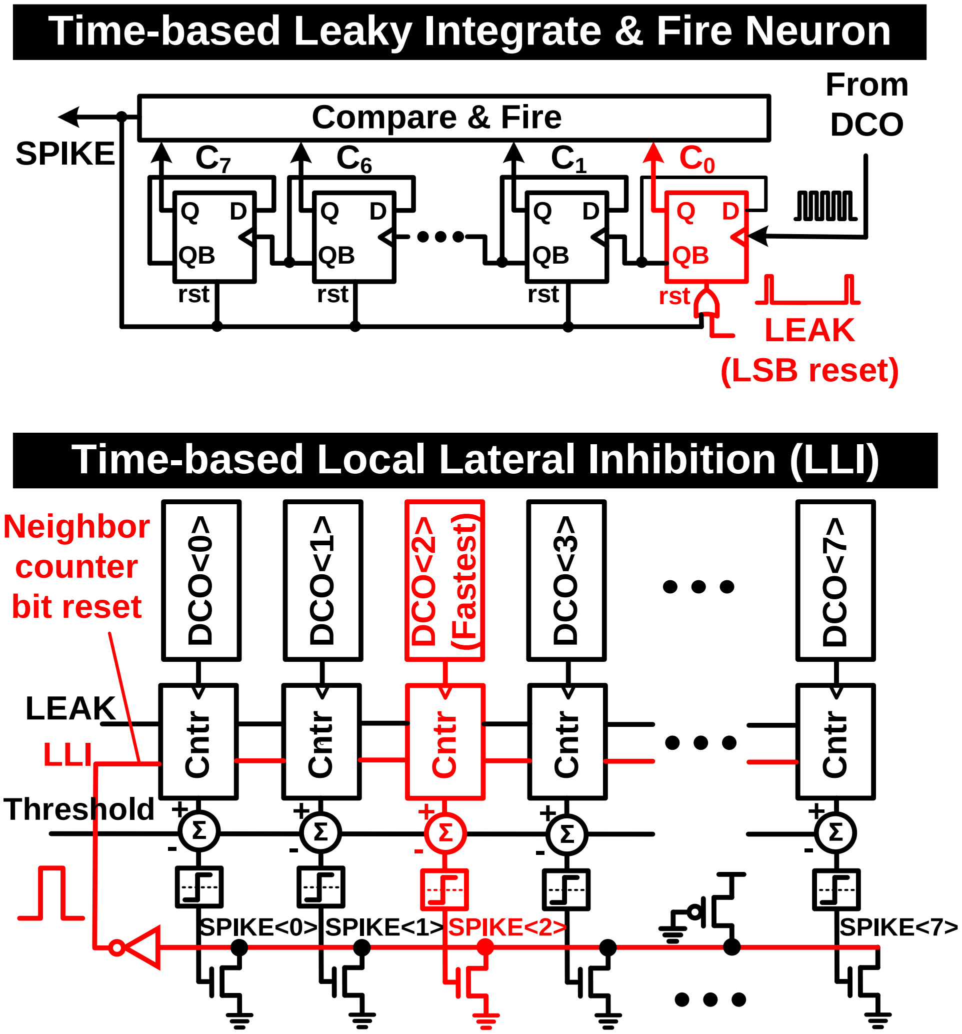


Fig. 2.12: Illustration of time-based leaky integrate-and-fire neuron and local lateral inhibition operations.

We also implemented brain-inspired leak and local lateral inhibition (LLI) features in this work to improve the performance. These two features can enhance the contrast between different neuron outputs. Fig. 2.12 illustrates the circuit implementation of leaky integrate-and-fire neuron and LLI features. We can decide separately whether to enable the leaky and LLI features or not. When the leaky feature is enabled, the LSB of the 8-bit counter in the readout circuit is not only reset by the generated spikes, but also periodically reset by a low frequency LEAK signal. This has the effect of gradually decrease the stored count value, mimicking a leaky neuron whose charge slowly leaks away through the cell membrane. The timing diagram illustrating the leaky feature is shown in Fig. 2.13 (middle), for comparison, the nominal operation without leaky or LLI features is shown in Fig. 2.13 (top). The threshold is 8 in this example. After enabling the leaky feature, the number of DCO cycles required to generate a spike gradually increases from 8 to 10. Note that the period of the LEAK signal need not be very stable and accurate, but should be several times longer than that of the DCO, otherwise the count value might increase. We also found that the pulse width of this LEAK signal can also affect the extent to which the count value is decreased. So we used an on-chip free-running VCO and pulse generator to tune the frequency and pulse width of this LEAK signal.

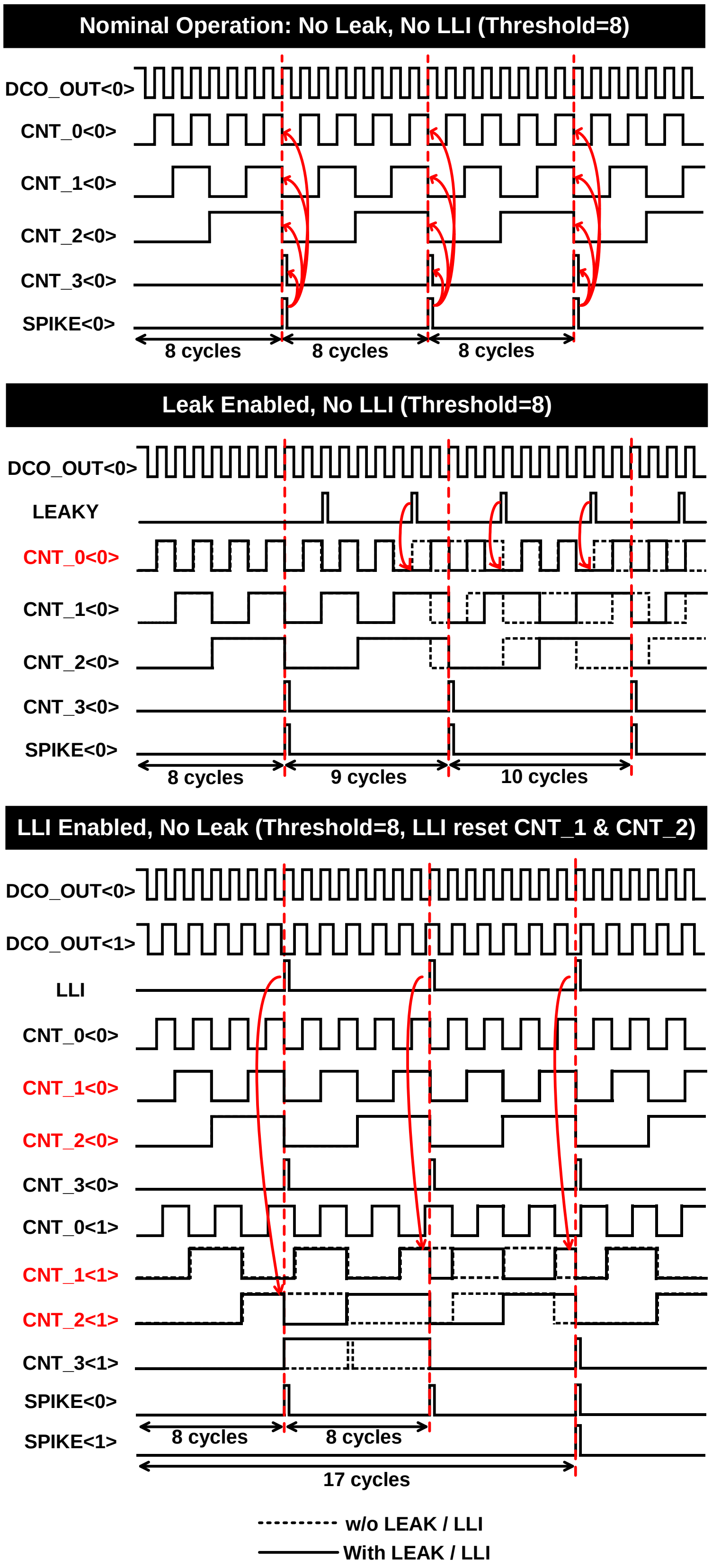
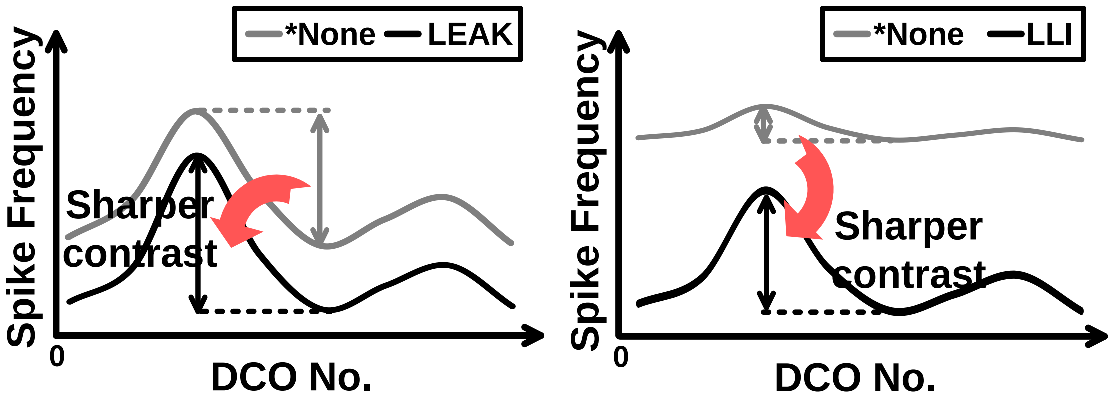

Nominal Operation: No Leak, No LLI (Threshold=8)
DCO_OUT<0>
CNT_0<0>
CNT_1<0>
CNT_2<0>
CNT_3<0>
SPIKE<0>
8 cycles
8 cycles
8 cycles
Leak Enabled, No LLI (Threshold=8)
DCO_OUT<0>
LEAKY
CNT_0<0>
CNT_1<0>
CNT_2<0>
CNT_3<0>
SPIKE<0>
8 cycles
9 cycles
10 cycles
LLI Enabled, No Leak (Threshold=8, LLI reset CNT_1 & CNT_2)
DCO_OUT<0>
DCO_OUT<1>
LLI
CNT_0<0>
CNT_1<0>
CNT_2<0>
CNT_3<0>
CNT_0<1>
CNT_1<1>
CNT_2<1>
CNT_3<1>
SPIKE<0>
SPIKE<1>
8 cycles
8 cycles
17 cycles
w/o LEAK / LLI
With LEAK / LLI

Fig. 2.13: Timing diagrams showing DCO frequency and neuron spike output with (solid waveform) and without (dashed waveforms) leaky or LLI features. Spiking threshold is 8 in this example.

Lateral inhibition is a phenomenon in which the active neuron strives to suppress the activities of its neighbors. In our design, every 8 DCOs are grouped together to realize LLI, so each DCO has 7 neighbors in its group. We can decide the inhibition amount, which is the count decrease in the counter, by setting which bits of the counter are reset. When LLI is enabled, once a DCO in the group generates a spike, there is a pulse generated as LLI signal, which resets the bits we previously defined in the neighboring counters. The fastest DCO in the group resets the other DCOs more often than it is reset by the other DCOs, enhancing the contrast between different DCO outputs. The timing diagram illustrating the LLI feature is shown in Fig. 2.13 (bottom). There are two DCOs in this example and DCO<0> is the dominant one. If there's no LLI, they need 8 cycles to generate a spike. After enabling LLI feature, the faster DCO (DCO<0>) resets the slower DCO (DCO<1> three times, so it takes the slower DCO 17 cycles to generate its first spike. So the output contrast between these two DCOs increases further after enabling the LLI feature.

The effects of leak and LLI features are illustrated in Fig. 2.14. The main benefit of the leak operation is that it can increase the relative difference between DCO spike outputs, since the absolute difference between different DCOs stays relatively the same, but the overall level of the DCO frequencies decreases, so the relative difference increases, making the prediction more accurate. And for LLI feature, the faster DCO becomes faster and slower DCO becomes slower. The contrast between different DCO outputs is sharper, so it ensures that the prediction results are more reliable. But there is usually a greater frequency drop than leaky feature, so it makes the operation slower.



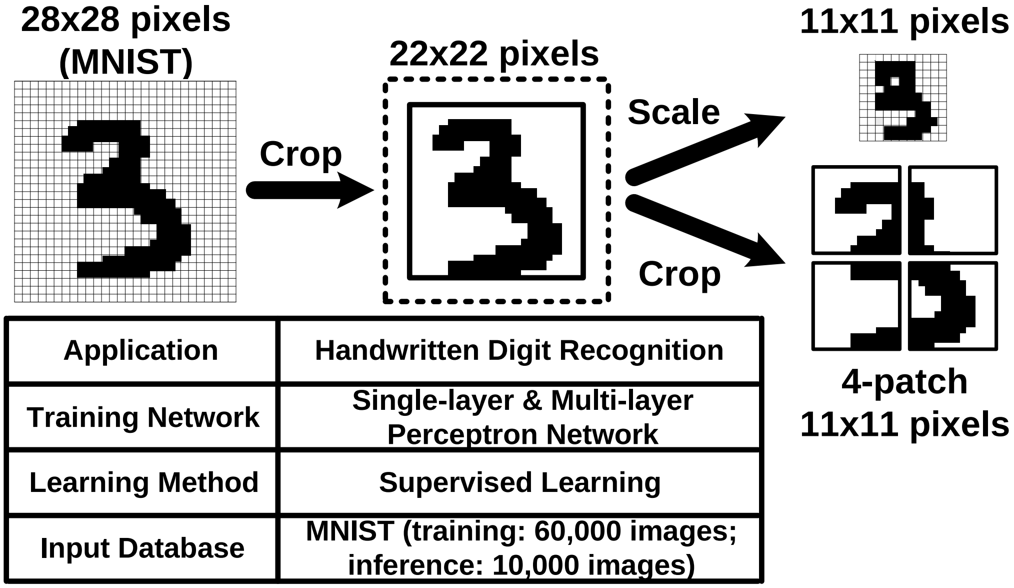
*None: No leak and no LLI, basic DCO operation.

Fig. 2.14: Effects of leak and LLI features.

2.5 Digit Recognition Application and Measurement Results

A test chip was fabricated in a 1.2V, 65nm LP CMOS to demonstrate the time-based neuromorphic core. Due to chip size limitations, we opted for a single neuromorphic core implementation with 64 DCO neurons. However, as the chip is fully scalable, we can either tile more DCOs in one neuromorphic core or tile several neuromorphic cores and operate them in parallel to handle complex deep learning algorithms.

We tested the chip with handwritten digit recognition application to showcase the versatility of the proposed core. Handwritten digit images were obtained from the MNIST database [32]. The original image size from MNIST was 28x28 pixels, while in our test chip, each DCO can accept at most 121 effective input pixels, so we did some preprocessing of the image data, which is shown in Fig. 2. 15. We first removed 3 pixels on each side without affecting the images, as they were almost all blank pixels, containing little information. Then we had two options, one is to directly scale the image from 22x22 pixels to 11x11 pixels, so that a single DCO circuit can process one complete image. Obviously, this degrades the image and deteriorates the recognition accuracy, so this is just used for proof-of-concept test architectures. In most of the applications, the pixel count is far more than 121, so to demonstrate that our core is able to handle larger images, we also crop the 22x22 images to 4-patch images with 11x11 pixels in each patch. In this case, four DCO circuits can work concurrently to process one image, which improved the throughput and recognition accuracy. The training network we used was the multi-layer perceptron neural network, and all weights were trained off-chip using supervised learning and downloaded to the chip. In the training process, we used all 60,000 images in the MNIST database and used 10,000 images for inference. The summary of the training process is shown in Fig. 2.15.



Application	Handwritten Digit Recognition
Training Network	Single-layer & Multi-layer Perceptron Network
Learning Method	Supervised Learning
Input Database	MNIST (training: 60,000 images; inference: 10,000 images)

Fig. 2.15: Data preprocessing for digit recognition application and summary for the training process.

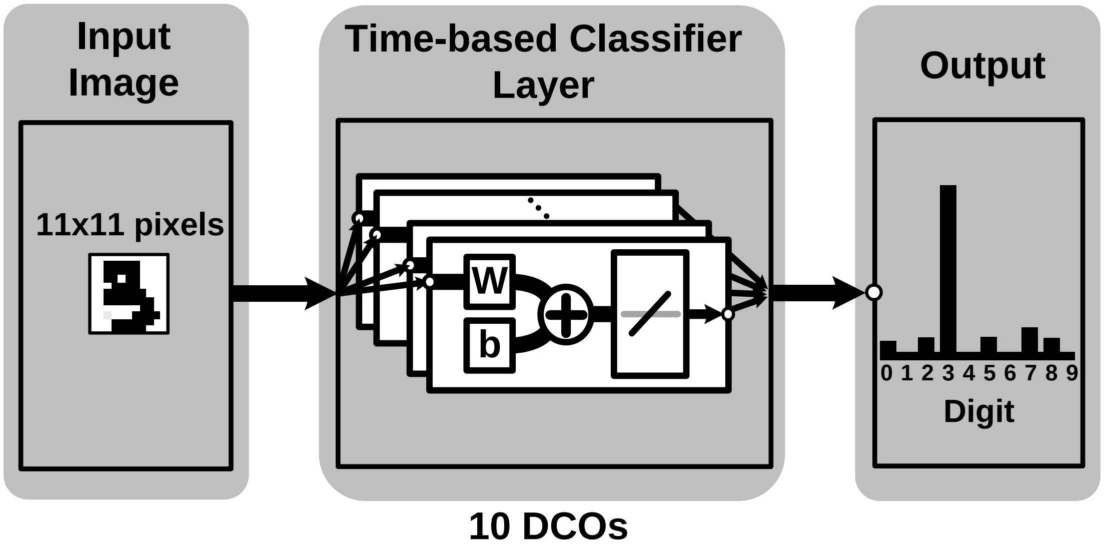


Fig. 2.16: Single-layer digit recognition application for proof-of-concept.

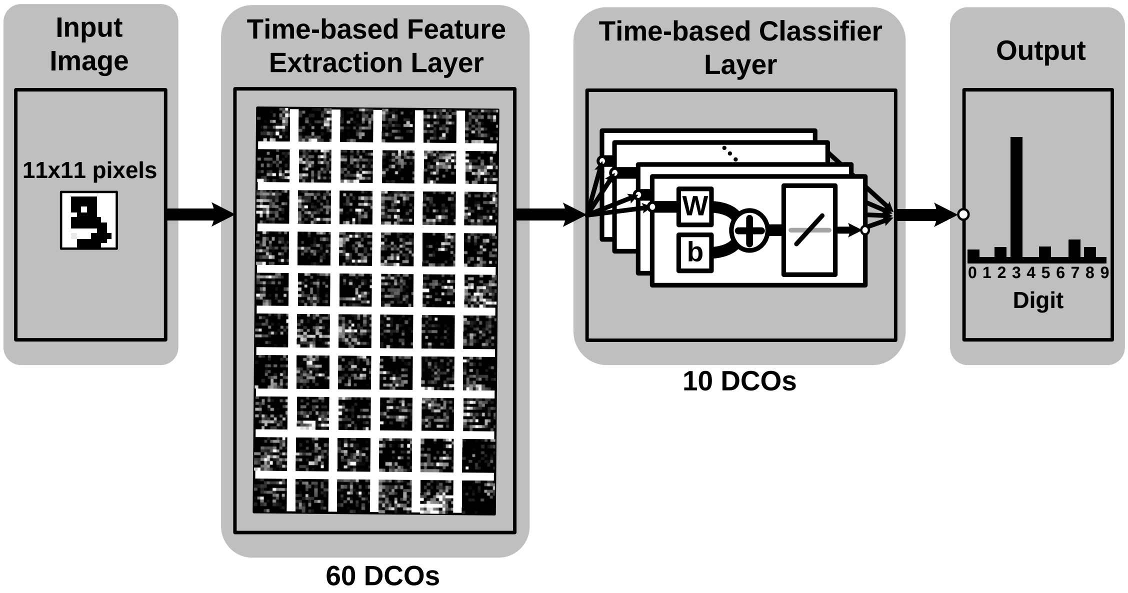


Fig. 2.17: Multi-layer digit recognition test architecture with 11x11 input images.

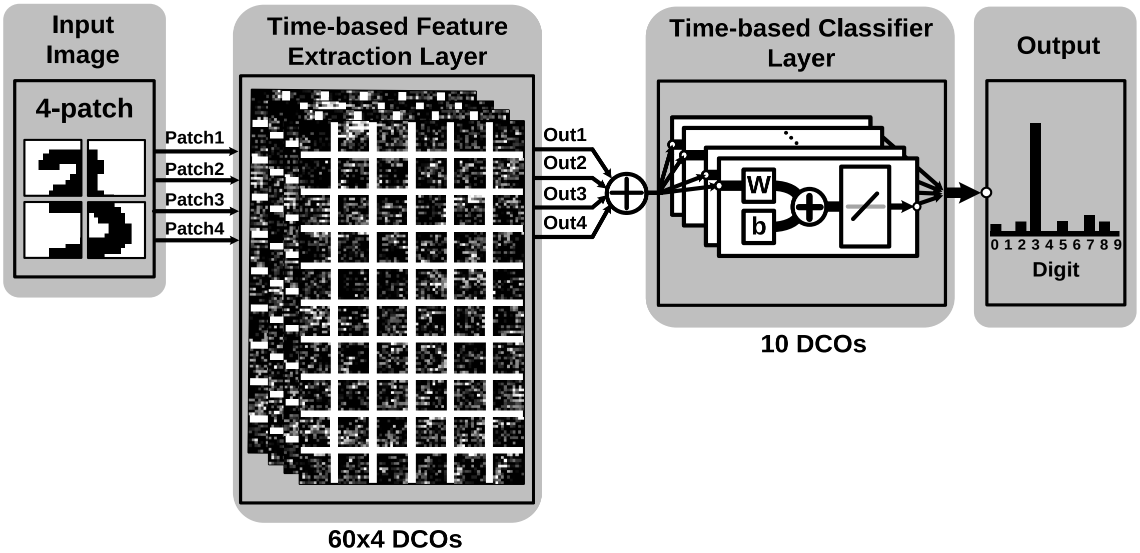


Fig. 2.18: Multi-layer digit recognition test architecture with 4-patched 22x22 input images.

First of all, for proof-of-concept, we tested the chip using a simple single-layer architecture with no hidden layers. It's a fully connected network with one input layer and one output layer, which is shown in Fig. 2.16. The input images are scaled version with 11x11 pixels. The output layer is the time-based classifier layer. There are 10 DCO neurons in the classifier layer, each neuron is trained to recognize one digit and the spike outputs from the 10 neurons are recorded. The neuron with the most spikes are the recognition output. Next, we ran the chip on a MLP architecture with one hidden layer, which is shown as time-based feature extraction layer in Fig. 2.17. The input images in this case are still 11x11 pixels. The feature extraction layer extracts 60 features from each input image. The output classifier layer is the same as before. Finally, we did our formal test of using 4-patched images with 22x22 pixels in a MLP network with one hidden layer, which is shown in Fig. 2.18. Each patch has 11x11 pixels and 4 DCOs handle the same image in parallel.

60 features are extracted from each patch by the feature extraction layer. Then we did the off-chip data processing to sum and encode the results from the feature extraction layers and feed these results to the output classifier layer to get the final recognition results. The reason why we chose a hidden layer of size 60 is based on the tradeoff between the simulated accuracy result and the circuit implementation complexity, as is shown in Fig. 19. The accuracy generally increases with the size of the hidden layer, but the increasing speed is gradually decreasing because of overfitting. And as our core has 64 DCO circuits, so if the hidden layer size is larger than 64, then it requires a multi-chip implementation, which greatly increase the complexity. In this work, we decide to use a simpler chip implementation and achieve an accuracy as high as possible. As is shown in the zoomed-in area in Fig. 2.19, the accuracy of using a hidden layer size of 60 reaches the highest and if we increase the size to 64, the accuracy doesn't improve further due to overfitting, so we decide to use the hidden layer size of 60.

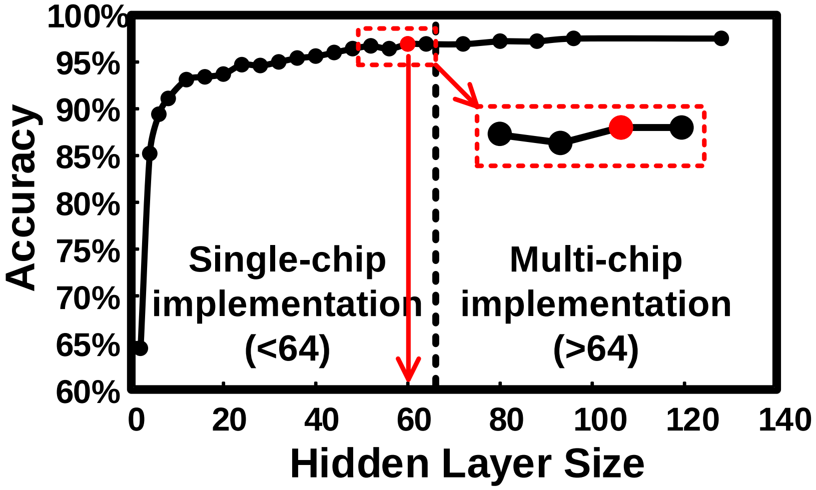


Fig. 2.19: Hidden layer size selection based on accuracy and circuit implementation complexity trade-off.

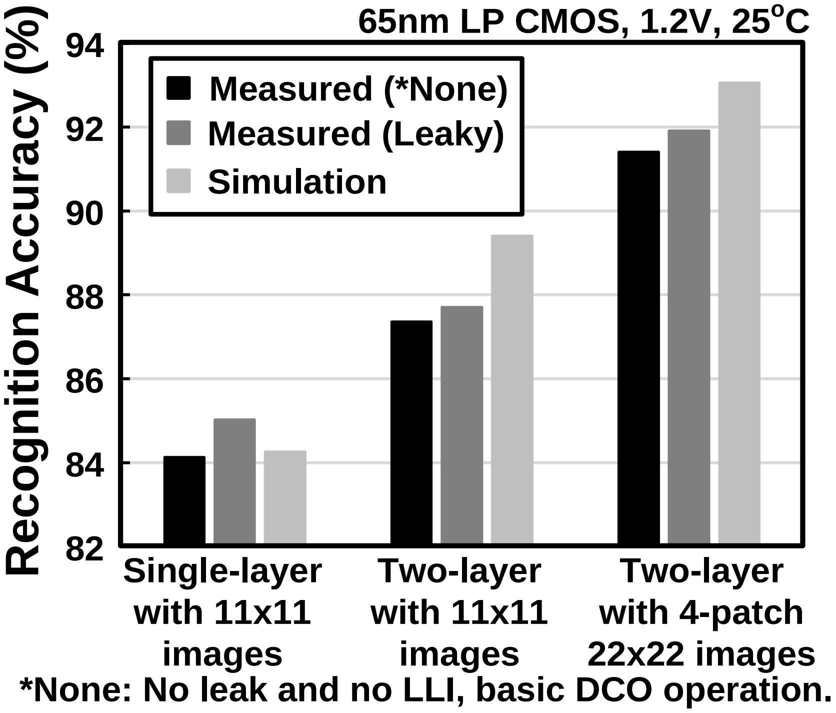


Fig. 2.20: Measured accuracy results of handwritten digit recognition application.

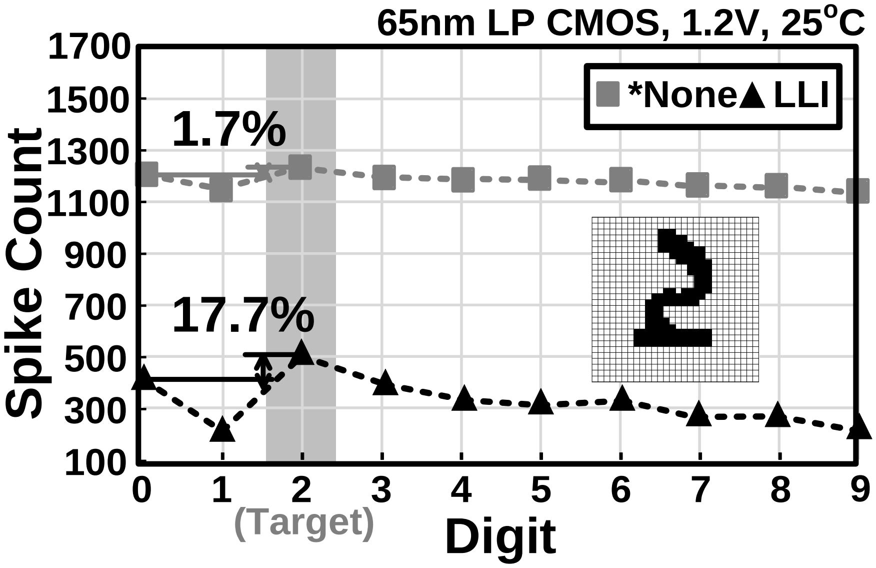


Fig. 2.21: Measured results of digit recognition application with local lateral inhibition (LLI) feature enabled.

The measured accuracy results from the above three architectures are shown in Fig. 2.20. The two-layer architecture with 4-patch inputs (22x22 pixels) achieves the highest recognition accuracy of 91.4%. With the leak feature enabled, the accuracy increases modestly to 91.9%. The measured accuracy from hardware is comparable to software simulation results, which is due to the DCO circuits have a good linearity. As seen from the measurement results in Fig. 2.20, the recognition accuracy of a single-layer architecture increases from 84.1% to 85.0% after enabling the leak feature, while the accuracy doesn’t improve as much in the two-layer architecture. This is because in the two-layer architecture, we have more weights available to improve the contrast between different neuron outputs, which means the output contrast for two layer architecture has already been pretty large. This make the leak feature less effective. Fig. 2.21 shows the measurement results of digit recognition application with LLI enabled. This figure shows the outputs from 10 DCOs in the classifier layer. For an image of digit “2”, before enabling the LLI feature, the spike count differences between different DCO neurons are very small, the minimum difference is 1.7%, which is very hard to get the correct recognition result. After enabling the LLI feature, this difference increases to 17.7%. So with a larger difference, we can make the prediction more confidently, and this prediction result is more reliable than without LLI case. The reliability of the prediction results is very important in some applications, for example, in medical field, if the doctors want to diagnose some diseases from some examination results with the help of deep learning algorithms, the reliability of the prediction result is of critical importance to the patients. We always want the doctor to be as confident as possible of the diagnosis result. Fig. 2.22 shows the measured power consumption and DCO frequency under different supply voltages. The test chip can work under a wide range of supply from 1.2V to 0.7V. The DCO circuit oscillates at 99MHz consuming 320.4μW under a nominal 1.2V supply voltage. At 0.7V supply, the DCO oscillates at 20MHz with 17.5μW power consumption.

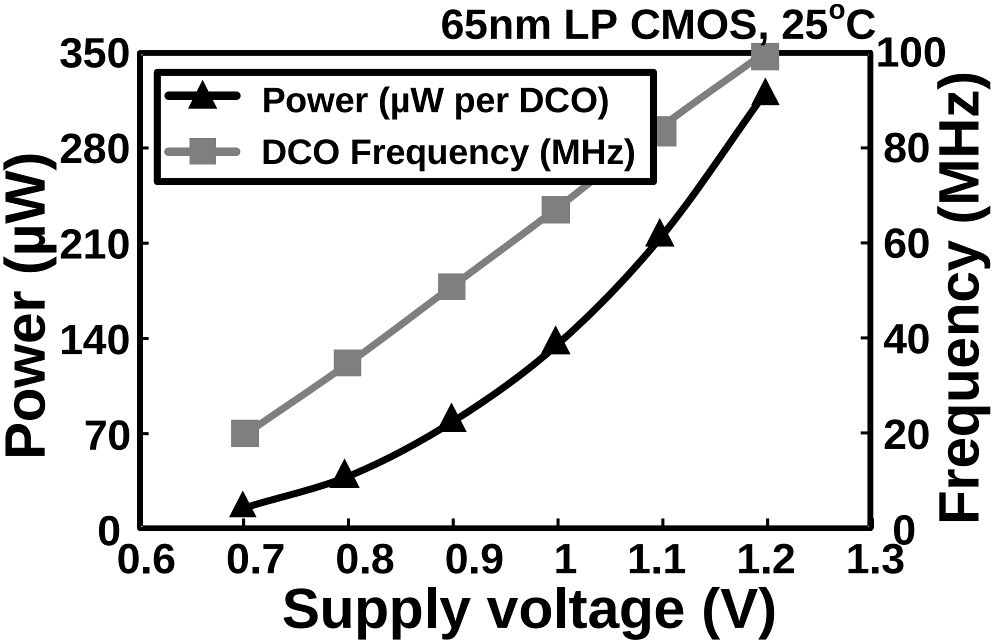


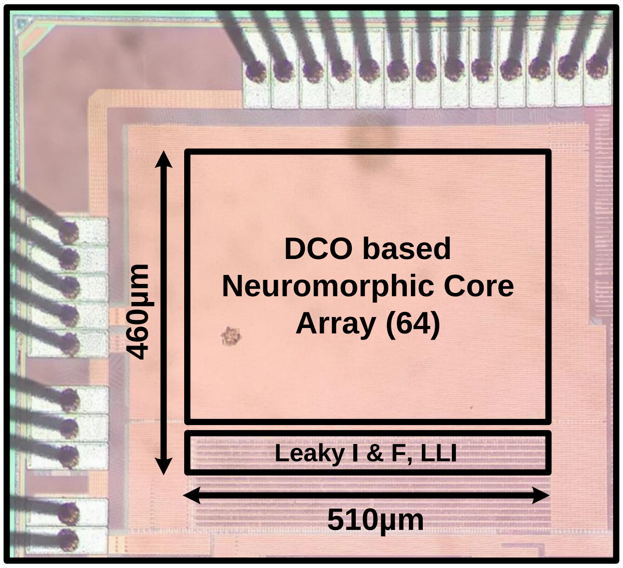
Fig. 2.22: Measured power consumption and DCO frequency of the test chip under different supply voltages.

Table 2.1 shows the performance comparison with recent neuromorphic chip designs [30, 31, 33, 34]. It’s worth noting that an apples-to-apples comparison between our time-based scheme and traditional ASIC chips can be tricky. Here, we chose to present metrics (e.g. μW/DCO, spikes/s/W) specific and relevant to our design, and we also provide the comparison of our work with prior arts in different metrics. The proposed DCO neuron can generate $3.09 \times 10^{11}/16 = 1.93 \times 10^{10}$ spikes per second per watt, for a spiking threshold value N of 16. Compared with the previous time-based neural network [29-30], our core has a slightly lower power efficiency, but the hardware efficiency is 4X better than the previous work. If we define one multiply and accumulate (MAC) as one operation (OP), our core achieves 37.4 TOPS/W efficiency, which is better than previous works [31, 33]. Note that different works have different weight resolutions, so it might not be the fairest comparison. All the performance numbers listed in the above table for this work is based on the spiking threshold value of 16. The performance can be better if smaller threshold is chosen, but it will slightly reduce the accuracy.

Table 2.1: Performance Comparison with Prior Arts

	This work	JSSC'17 [28]	VLSI'17 [29]	ISSCC'16 [31]	VLSI'15 [32]	Note
Application	Hand writing recognition	Hand writing recognition	Hand writing recognition	Object detection + intention prediction	Object Recognition	a. N=16 in our measurements. b. SOp/s/W: Synaptic operation (SOp). In DCO based time-domain neural network, one oscillation of DCO is equivalent to 121 SOp. c. 1GE: 1.44um^2(65nm). PE: processing element. d. Operation: One operation is defined as one multiplication and accumulation (MAC). In DCO based time-domain neural network, one oscillation of DCO is equivalent to 121 3-bit MAC. e. Used spiking threshold of 16, and only accounted for the power consumption of core logic circuits, memory power is not included, since weight is not updated during the inference. f. 1 MULADD = 2OPs
Neural Network Type	Multi-layer perceptron network	Binary neural network	Binary DNN	Deep neural network	SAILnet	
Circuit Type	Time-based	Time-based	Digital	Analog + Digital	Digital	
Technology	65nm	65nm	65nm	65nm	65nm	
Area	0.24mm^2 (64 DCOs)	3.61mm^2 (32K PEs)	3.9mm^2	16.0mm^2	1.8mm^2	
Voltage	1.2V	-	0.55-1.0V	1.2V	0.45V	
Frequency	99MHz (nominal DCO freq.)	-	100-400MHz	250MHz	40MHz (Inference)	
Power	320.4 μW/DCO	-	0.05-0.6W	330mW	3.65mW	
Power Efficiency	309G ÷ N spikes/s/W (N=spiking threshold[a])	48.2TSOp/s/W	6.0-2.3TOPS/W[f]	862GOPS/W	5.7pJ/pixel (memory+logic)	
Hardware Efficiency	-	76.5GE/PE	-	-	-	
Performance Comparison	37.4TSOp/s/W[b]	48.2TSOp/s/W	-	-	-	
	16.6GE/PE[c]	76.5GE/PE	-	-	-	
	37.4TOPS/W[d]	-	6.0-2.3TOPS/W	862GOPS/W	-	
	0.43pJ/pixel (logic)[e]	-	-	-	5.7pJ/pixel (memory+logic)	

Fig. 2.23 shows the die photo and the performance summary of the test chip. The neuromorphic core with 64 DCO circuits takes an area of 0.24mm^2. There are around 8K synapses and 3.1K bytes of on-chip memory.

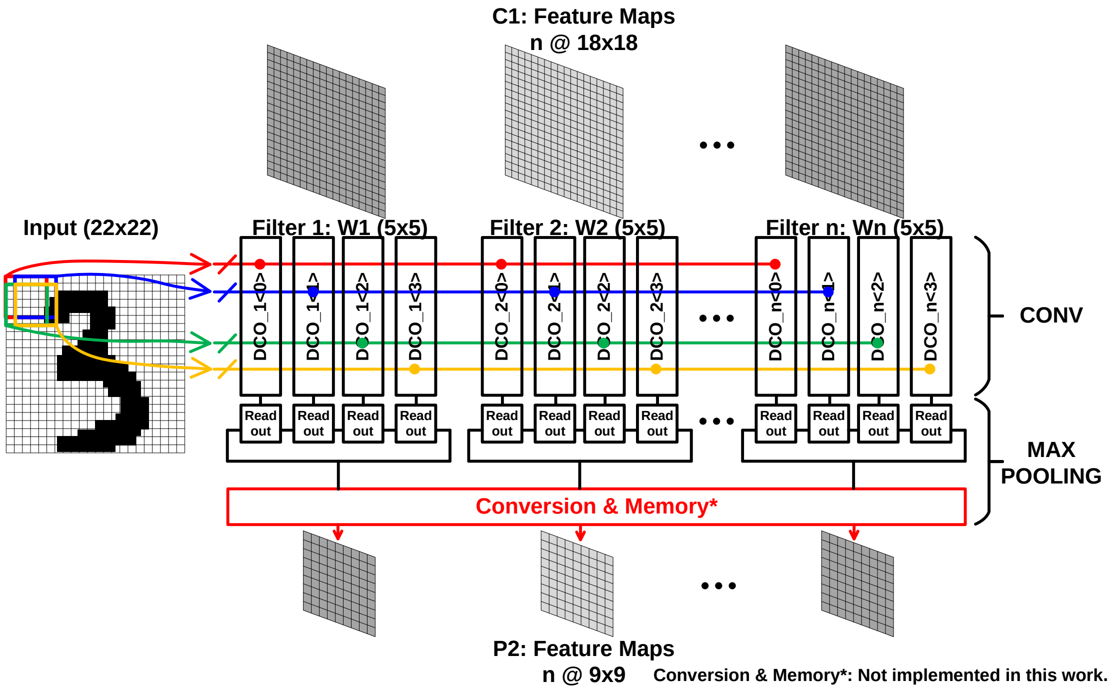


Technology	65nm LP CMOS
Core Size	510μm x 460μm = 0.24mm^2
VDD range	0.7V~1.2V
# of Neurons	64 (single core)
# of Synapses	8192 (8K)
On-chip SRAM	3.1K bytes
Throughput	746Mpixels/s ÷ N (1.2V) 148Mpixels/s ÷ N (0.7V) (*N=spiking threshold)
Power	320.4μW (per DCO, 1.2V) 17.5μW (per DCO, 0.7V)

*N = 16 in our measurements

Fig. 2.23: Die photo and performance summary of the test chip.

In this work, we have focused on the implementation of MLP using our proposed time-based neuromorphic core, however, the proposed time-based architecture is not limited to MLP only. With some modifications, we can also apply the proposed neuromorphic core in convolutional neural network (CNN), as is shown in Fig. 2.24. We take an input image of 22x22 pixels and filter size of 5x5 as an example. Each DCO can do the computation of convolution between filters and inputs. We can group 4 DCOs together for the max pooling layer to subsampling the feature maps to a smaller size by taking advantage of the LLI feature in this work. Instead of sliding the filter over input images, we fix the filters and shift the input images to feed into different DCOs, which can save the memory access cost. One limitation of the proposed work is that the outputs of the DCOs are spikes, so we need to convert this information to digital domain to feed to the next layer, and we also need some on-chip memory to store the intermediate results for implementing CNN. The main modification required is to add one conversion and memory block. And this is our possible future works for this time-based neuromorphic core.



Conversion & Memory*: Not implemented in this work.

Fig. 2.24: Implementation of convolutional neural network (CNN) using our proposed time-based neuromorphic core with some modifications.

2.6 Conclusion

In this work, we present the implementation of neuromorphic function in time domain with programmable delay stages. Brain-inspired leak and local lateral inhibition (LLI) features are also implemented on chip. The processing element of the proposed time-based neuromorphic core is based on inverter which is tiny and compact, and each processing element has its exclusive memory for use, eliminating the power for data movement, making the proposed core highly efficient in area and power. The proposed time-based neuromorphic core is tested with digit recognition application and achieves a 91.4% recognition accuracy. The energy-efficiency and versatility of the presented time-based DCO neuromorphic core makes it a promising building block for future large scale deep neural network applications.

Chapter 3. Binarized LSTM Neural Network Architecture Design

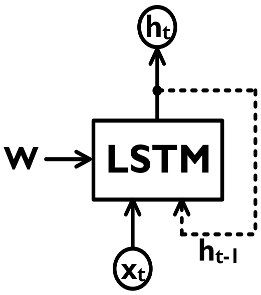
3.1 Introduction

Recurrent neural network (RNN) is very powerful in processing sequential data, and it has been proven to be successful in many applications, such as natural language processing (NLP) [35], machine translation [36], etc. Long short-term memory (LSTM) is a popular type of RNN that is good at dealing with sequential data that has long term dependencies. Photoplethysmorgraphy (PPG) signal is such a type of data, which is a popular and convenient way for heart rate monitoring. However it suffers from motion artifacts (MA) problem, deteriorating the accuracy of heart rate estimation. Recently, LSTM neural networks accompanied with convolution neural networks (CNNs) and fully connected (FC) layers are shown to work well in predicting heart rate from PPG signals [37]. This data-driven and learning-based network obviates the necessity of feature engineering, which requires domain knowledge to select hand-crafted features.

Despite the versatility of the LSTM neural networks, it is hard to implement in hardware due to its high computation complexity and memory requirement. In this work, to deploy the neural network for heart rate prediction on embedded devices, an efficient binarized LSTM hardware architecture that reduces the computation complexity and the memory footprint is proposed and the performance is evaluated with various PPG datasets. The proposed LSTM architecture is tested using the custom collected PPG signals and the simulation results demonstrates that the proposed binarized LSTM architecture achieves a good heart rate prediction accuracy.

3.2 LSTM and PPG Background

3.2.1 LSTM Neural Network



$$
\begin{aligned}
i_t &= \sigma_g(W_{xi}x_t + W_{hi}h_{t-1} + b_i)\\
f_t &= \sigma_g\left(W_{xf}x_t + W_{hf}h_{t-1} + b_f\right)\\
o_t &= \sigma_g(W_{xo}x_t + W_{ho}h_{t-1} + b_o)\\
c_t &= f_t{}^\circ c_{t-1} + i_t{}^\circ \sigma_c(W_{xc}x_t + W_{hc}h_{t-1} + b_c)\\
h_t &= o_t{}^\circ \sigma_h(c_t)
\end{aligned}
$$

Fig. 3.1: LSTM unit and equations.

The LSTM unit is composed of an input gate, a forget gate, an output gate and a cell. Each gate can be viewed as a feed-forward neural network, implementing the multiply and accumulation (MAC) computation followed by an activation function, as is shown in Fig. 3.1 [38]. The subscript *t* indexes the time step, σ_g represents the sigmoid function, and σ_c, σ_h are tanh. The operator ° denotes the element-wise product. Each gate computes the element-wise addition of two weighted sums $W_x x_t$ and $W_h h_{t-1}$, and a bias b. To simplify the neural network, we remove the bias term in this work. Therefore, each gate now computes the element-wise addition of two dot products of inputs x_t and hidden states h_{t-1} and their corresponding weights, followed by an activation function. There are two input ports for each gate, one is the actual external input signal of the current time step x_t; the other is the output (hidden state) from the previous time step h_{t-1}, which allows information to persist, and this is the loop shown in the LSTM unit in Fig. 3.1.

In this work, a binarized LSTM neural network is proposed to reduce the memory requirement and simplify the circuit implementation in hardware. During the training, the weights are binarized using loss-aware binarization scheme [39]. Instead of simply finding the closest binary approximation of weights, this loss-aware binarization also considers the loss during binarization. So in this work, the weight in each layer is binarized as

$$\boldsymbol{w}_l = \alpha_l \boldsymbol{b}_l \tag{3.1}$$

where $\alpha_l > 0$ and $\boldsymbol{b}_l$ is binary. In terms of the hardware implementation, we can still fully utilize the benefit of binary computation, and simply multiply a constant at the end of each multiply and accumulate (MAC) operation, since for each gate, α_l is a constant. And a multiplication operation with a constant number can be simplified to predefined bit shifts. During the inference mode, all the weights are available and fixed, so this multiplication is reduced to bit shifts, incurring little hardware overhead.

3.2.2 PPG Signal Basics

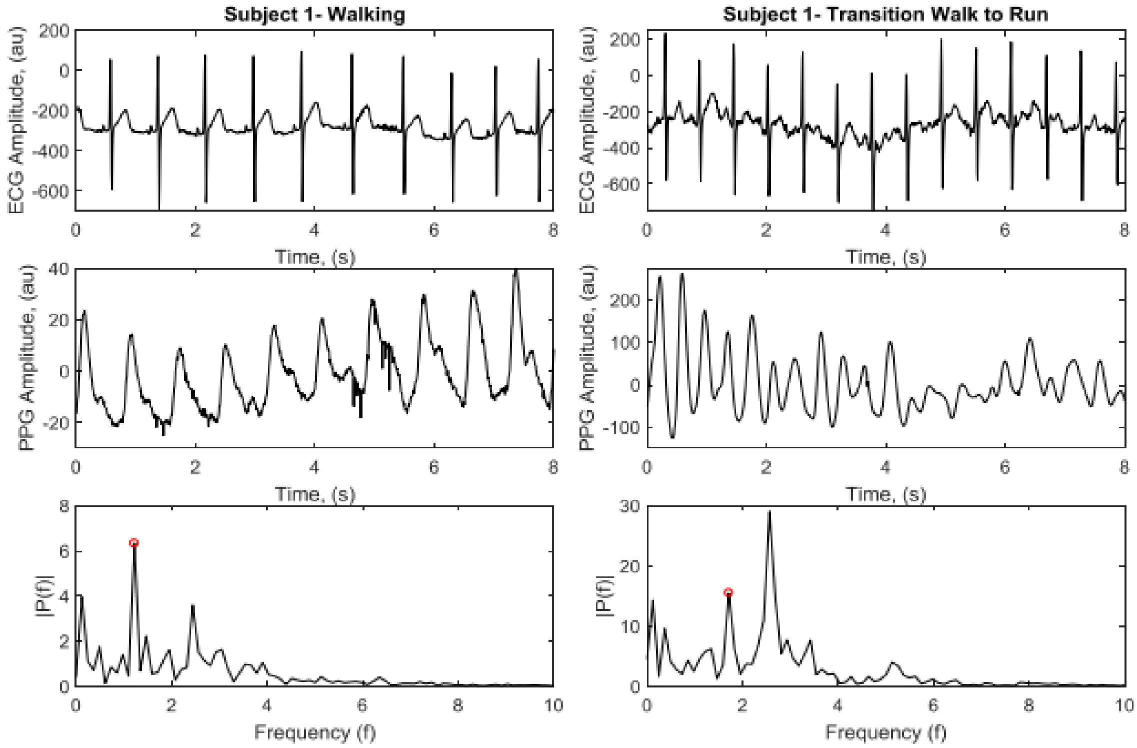


Fig. 3.2: Raw ECG (top), PPG (middle) signals and spectrums (bottom) while walking (left) and during transition from walking to running (right) respectively. The highest PPG spectral peak does not coincide with true HR (encircled) during intense motion.

PPG signals are optically obtained and are caused by the blood volume change. A PPG is usually measured by a pulse oximeter, emitting light on the skin and measuring the changes in reflected or transmitted light intensity [40]. The periodic change in the blood volume causes the light intensity to change periodically. The periodicity of the light corresponds to the cardiac rhythm, which is often used to estimate heart rate (HR). PPG signals can be acquired from peripheral locations such as fingertips, earlobes or wrist, which provides a distinct advantage of incurring low-cost and having a small form factor, making them a popular alternative for measuring HR. The conventional way of monitoring

HR is using electrocardiography (ECG), which is limited by its placement for signal fidelity. ECG measurement requires ground connection and reference sensors proximal to chest, so it is inefficient in terms of wearability for continuous HR monitoring in daily living conditions. While PPG signal can be easily embedded into wearable devices due to its small form factor and low-cost. However, PPG signals always suffer from motion artifacts (MA) problem, which distorts the signal fidelity and inhibit the robust estimation of HR. Fig. 3.2 illustrates the comparison between ECG and PPG signals and the effect of MA on the signal quality [41]. In ECG signals, the highest peak in the spectrum does not coincide with the true HR. However, in PPG signals, due to the interference of MA, the HR peak is no longer the highest, and this causes the inaccuracy in HR monitoring.

MA are caused by various factors, such as physical activities, ambient light leaking through the gap between sensor and the skin, and change in blood volume due to movements. This can cause the spectral component of MA to coincide or even overpower the heart-beat related component [42]. Conventional methods to solve this problem often rely on signal processing techniques to remove or attenuate MA using filtering [43-45], spectral subtraction [42] and feature-engineering based learning algorithms [46-47]. To avoid the necessity of feature engineering, which always requires using domain knowledge to select hand-crafted features, a learning-based framework is proposed in [41]. This proposed framework is based on the fundamentals of deep neural network (DNN), which is data-driven and has been demonstrated to be successful in predicting HR from PPG signals based on a convolution neural network (CNN) accompanied with long short-term memory (LSTM) neural network. In this work, we try to implement the LSTM neural network in the proposed framework, so that the whole framework can be embedded on the wearable devices.

3.3 Proposed LSTM Hardware Architecture

3.3.1 System Overview

Fig. 3.1 illustrates the overall neural network based heart rate prediction system [37]. The input PPG signal is divided into 8 seconds windows, and new data arrives in every 2 seconds. The minimum requirement is that the system must finish the computation in 2 seconds. The overall system composes of 2 1-dimensional CNN layers, 2 LSTM layers and a fully connected layer. LSTM layer is the most computation and memory intensive part in this system. In this work, we focus on accelerating LSTM layers and the final dense layer in hardware so that we can achieve real time heart rate prediction on embedded devices. The system parameters are illustrated in Fig. 3.1. There are 256 data points in each input PPG window, and 32 filters in both CNN layers, with size 40*1 and 40*32 respectively. There are 128 hidden units in LSTM layer. The output data with size 3*32 from the second CNN layer is the input for the first LSTM layer, so there are 3 time steps in LSTM layer and each time step has 32 data points. The second LSTM layer is basically the same as the first one, except the input size. The input of the second LSTM layer is the output from the first LSTM layer, with size 3*128, so it has 3 time steps and each time step has 128 data points.

3.3.2 Implementation Details of LSTM Layer

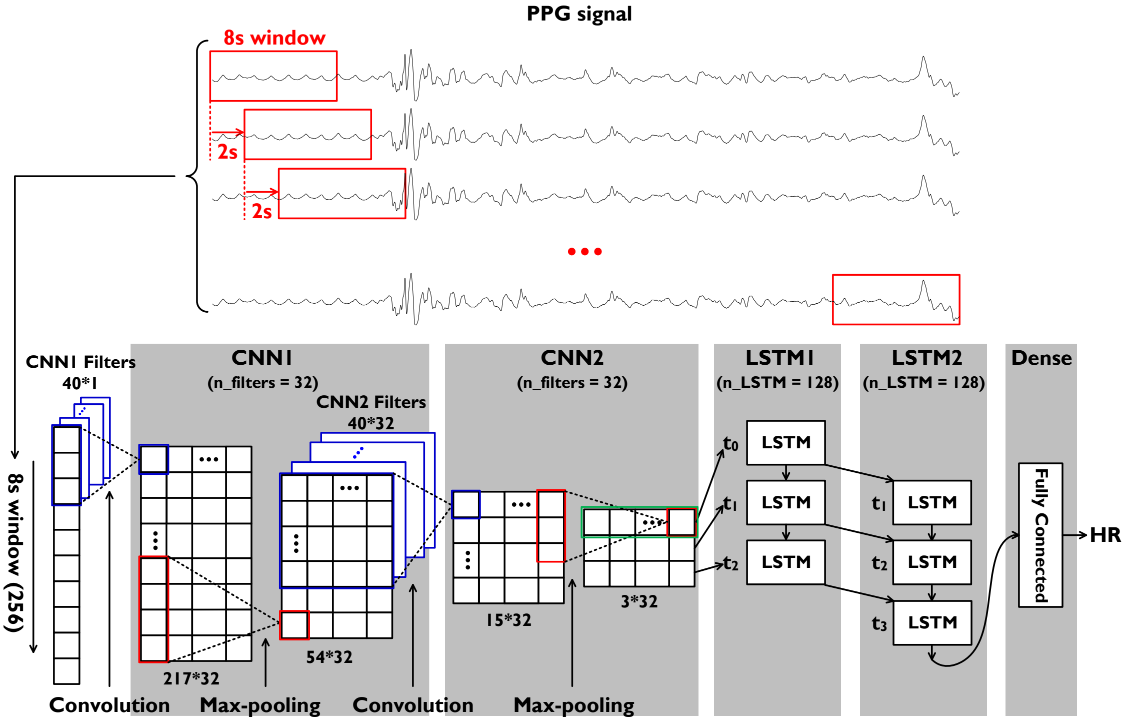


Fig. 3.3: LSTM based heart rate prediction system [37].

To accelerate the computation and alleviate the memory requirement in LSTM layers, a binarized LSTM architecture is proposed in this work. Both the input data and weight are binary numbers, either 0 or 1, this will greatly simplify the MAC computation and reduce the memory footprint. Fig. 3.3 shows the details of the proposed hardware implementation of the first LSTM layer. The LSTM layer implementation is divided into 2 stages: MAC computation in parallel (stage 0) and C_i and h_i computation in serial (stage 1). In this proposed binarized LSTM network, both the inputs and weights are binary, so the MAC computation is reduced to bit-wise XNOR followed by a population count, thus allowing the parallel computation, which is shown in Fig. 3.4 (a). For LSTM1, the input vector size is 32, the weight matrix is 32 by 128, so the MAC unit for each gate is computing the dot product of the above two matrixes, and the output vector has size 128. The MAC unit contains 128 XNOR units in parallel, and each unit computes 32 bit-wise XNOR. The input vector is shared by 128 units. This is combinational logic, which can be finished within one clock cycle, however, this requires all 32 1-bit inputs and 32*128 1-bit weights are available at the same time, which consumes a large memory bandwidth. While in this application, the computation latency requirement is not that strict, to reduce the memory bandwidth requirement, an optimized MAC computation is proposed, which is shown in Fig. 3.4 (b). The 128 parallel MAC computation units are broken into 32 groups and within each clock cycle only 1 group is computed, so only 4 XNOR units are left in each MAC unit and they are reused 32 times. We tradeoff the computation time for memory requirement. This increases the computation time, but reduces the memory footprint by 32 times. We choose 32 to have a balanced computation time and memory footprint. To achieve the best possible computation accuracy, we use 24-b data to represent intermediate calculation results.

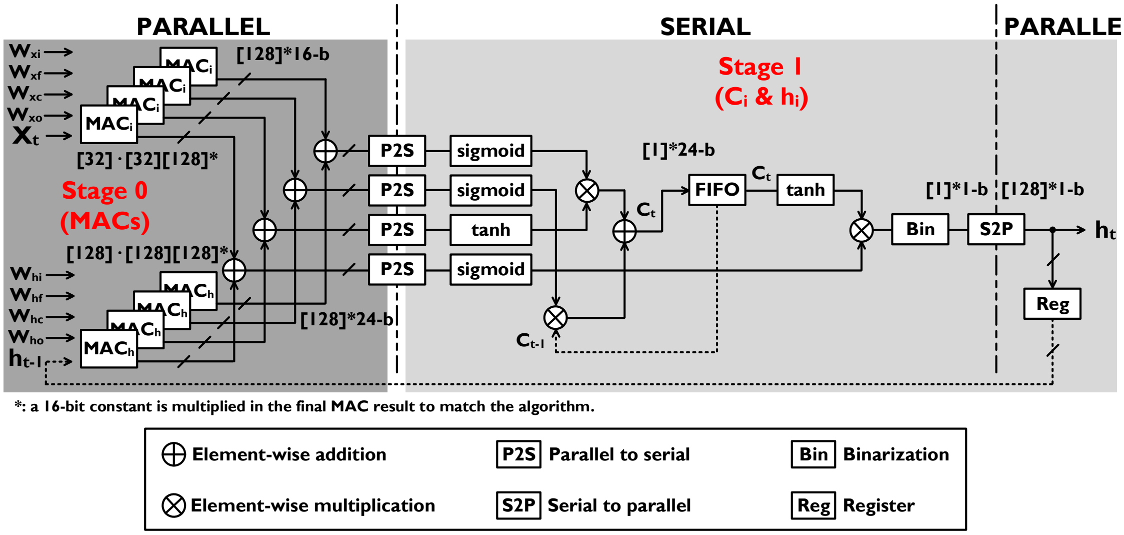
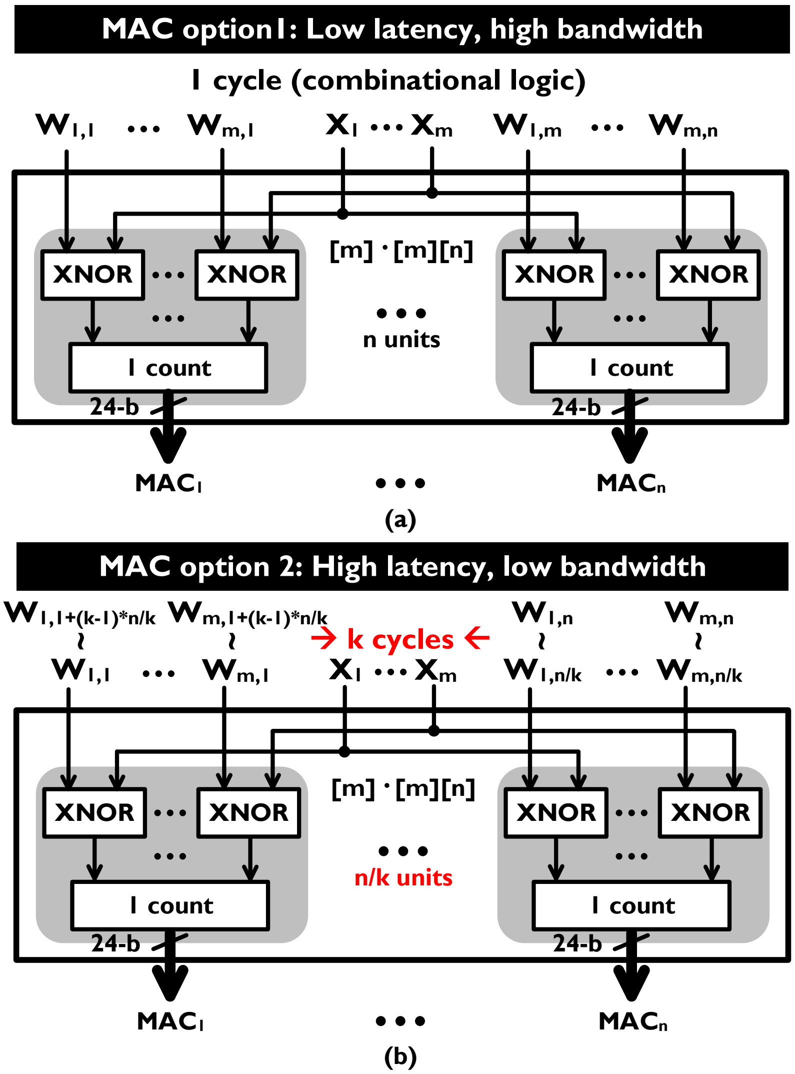


Fig. 3.4: Proposed binarized first LSTM layer architecture.

The non-linearity blocks, including sigmoid and tanh, are implemented with CORDIC module [48]. It is a relatively large block, so parallel implementation is not allowed. The parallel data is serialized before being fed into the non-linearity block. We reuse the CORDIC block for the element-wise multiplication to save area. A first-in-first-out (FIFO) buffer is inserted to save the current cell state for the next time step computation, which is shown in Fig. 3.5. In each clock cycle, a 1-bit output h_t is computed, and the serial to parallel (S2P) block collects the 1-bit h_t and outputs the final 128-bit h_t after 128 clock cycles and then feeds it to the next time step computation. The LSTM2 architecture is almost the same with the LSTM1 architecture, except that the input X is the output h_t from LSTM1, which is 128 in size. So the input MAC computes the dot product of two matrixes with size 128 and 128 by 128.



(a)

(b)

In this application: m = 32, n = 128, k = 32

Fig. 3.5: Detailed MAC implementation, original (a) and optimized (b).

The final fully connected dense layer computes the dot product of input vector and weights, which are both 128 in size. The input is binary, which is the 128-bit output from the LSTM2 layer, and the weight is non-binary, since the output of the dense layer is the predicted heart rate. Due to the binary input, dense layer MAC computation can be implemented with an accumulator, and the input h_t decides whether to add or subtract the current weight to the accumulated result, which is shown in Fig. 3.6. This dense layer computation is implemented in parallel with the LSTM2 layer during the final time step computation.

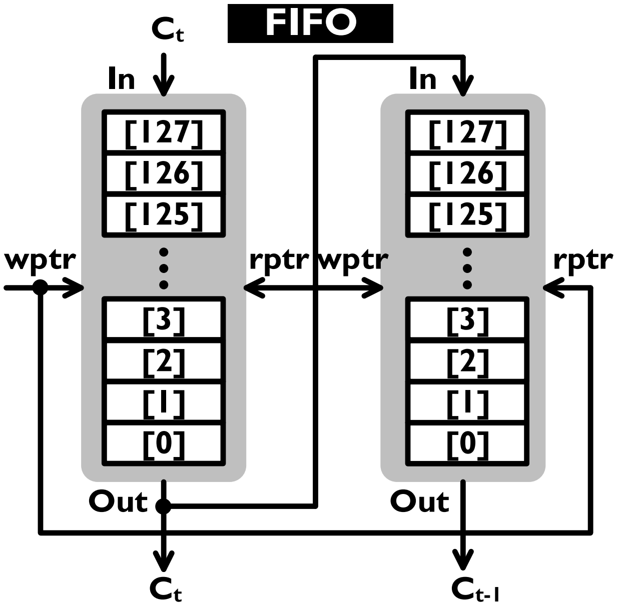


Fig. 3.6: FIFO implementation for storing the current time step and the previous time step cell states.

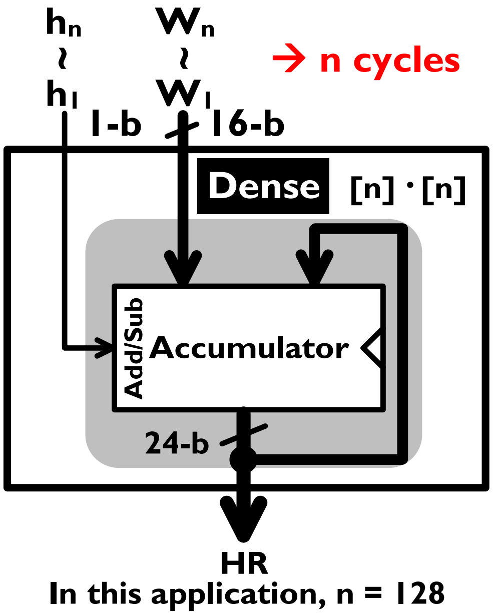


Fig. 3.7: Dense layer implementation using an accumulator.

The overall timing diagram of the system, including two LSTM layers and the final dense layer, is shown in Fig. 3.7. The entire system can be pipelined, so that the HR prediction can be finished in (n+1) time steps, where n is the time steps in each LSTM layer. There are 3 time steps in the dataset shown in this application. When the first time step of the first LSTM layer is computing, the other blocks should wait. After the output from the first time step of LSTM1 is ready, the first time step computation of LSTM2 can start. So the output h_1 is fed to both LSTM1 and LSTM2. It's similar for the rest of the time step computations. During the final time step of LSTM2 computation, the output h_3 comes out 1-bit by 1-bit, without being collected by the S2P module as that in the previous time step computations. This 1-bit h_3 output directly goes to the input port of the final dense layer, which accumulates the input data bit by bit. Thus the dense layer requires no extra computation time. As soon as the computation of the final LSTM layer is finished, the dense layer is also finished. So in this case, only 4 time steps of computation time is required to finish the computation of the entire system. The block diagram of the proposed system is shown in Fig. 3.8.

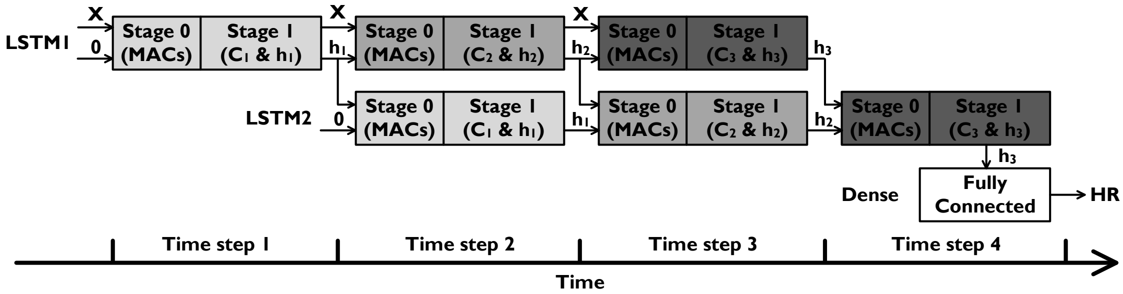


Fig. 3.8: Timing diagram of the proposed LSTM based heart rate prediction system.

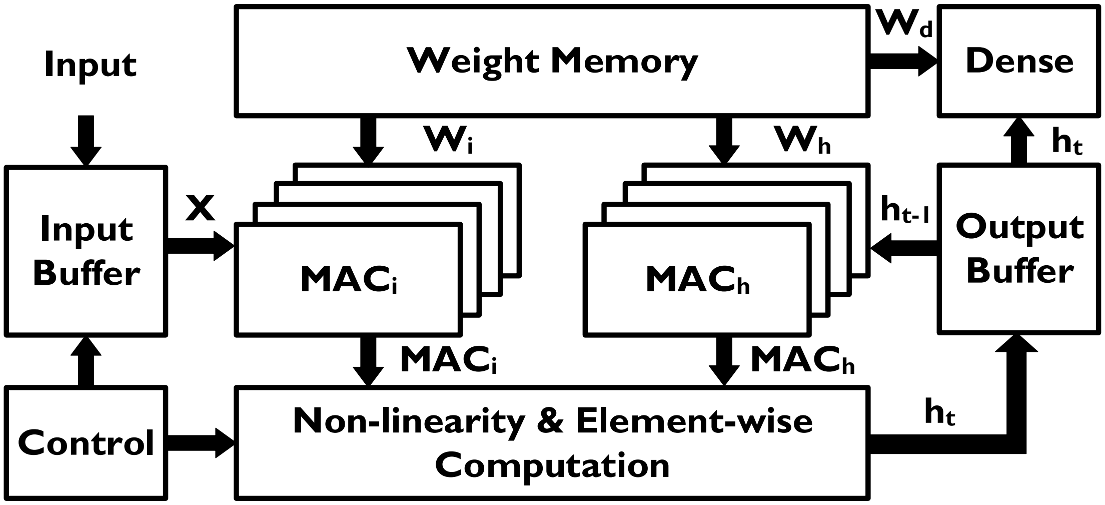


Fig. 3.9: Block diagram of the proposed LSTM based heart rate prediction system (with only LSTM layers and the final dense layer).

3.4 Heart Rate Estimation using the Proposed LSTM Architecture

To demonstrate the performance of this proposed binarized LSTM neural network, we simulated the neural network using custom collected PPG and ECG signals. The ECG signals are used as the true HR label during the training process. Fig. 3.10 shows both the ECG and PPG sensing platform [41]. The raw PPG data is first preprocessed before feeding into the proposed neural network. The ECG and PPG signals were originally sampled at 256 Hz, so there are 2048 data points in each 8s window. To save training time and reduce the memory requirement, we used wavelet transform to down-sample the PPG data into 256 data points in 8s window. There is a limited accuracy degradation after the data down-sampling according to the software simulation results.

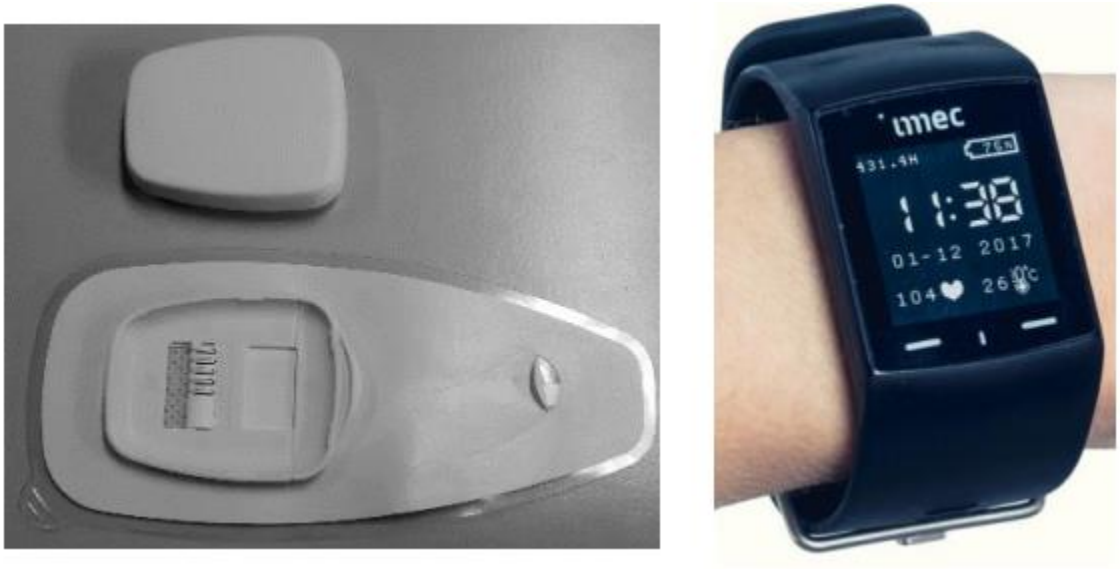


Fig. 3.10: Customized chest patch supporting two-lead ECG acquisition (left) and wristband with green LED PPG component.

For proof-of-concept, we feed the output data from the second CNN layer to the proposed system, including 2 LSTM layers and the final dense layer, and we also compare the HR estimation result with the software simulation result. The comparison of HR estimation results is shown in Table 3.1. The estimation results shown in the table is the average HR of one 8s window. We can see from the table that the proposed hardware system achieves a relatively good accuracy.

Table 3.1: Comparison of HR Estimation Results

	Estimated HR (BPM)	**Error (BPM)**
Hardware	65.81	2.99
Software	61.92	0.9
HR Label	62.82	-

3.5 Conclusion

In this work, we implement a binarized LSTM neural network architecture in hardware. This proposed binarized LSTM architecture reduces the circuit complexity and memory footprint compared to a normal LSTM network, making it possible to be deployed in embedded devices. The proposed architecture is simulated using the custom collected PPG data and achieves a good HR estimation accuracy.

Chapter 4. A Data Remanence based Approach to Generate 100% Stable Keys from an SRAM Physical Unclonable Function

4.1 Introduction

Physical Unclonable Function (PUF) is a circuit that harnesses inherent manufacturing variation to generate a random and unique key used for secure hardware authentication. The input to a PUF is referred to as “challenge”, and is provided by the server. The output of a PUF is called “response” which is sent back to the server for authentication purposes. If the response from the PUF matches the correct response stored on the server, then the user is granted to access to the system.

Two categories of PUFs exist: “strong” PUF and “weak” PUF. Strong PUFs like Arbiter PUF [49] and ring oscillator PUF [50] can generate an exponential number of unique challenge response pairs (CRPs), making them suitable for authentication applications without the use of encryption algorithms. Weak PUFs on the other hand, can only generate a linear number of CRPs and hence are used for key generation. Keys generated by weak PUFs can be used in conjunction with encryption algorithms for authentication applications [51]. The main requirement for keys generated by weak PUFs is that their value should not change with temperature and voltage changes, or with device aging.

SRAM is an attractive option for weak PUFs [52] since it is readily available in digital processors. Compared to dedicate PUFs such as arbiter PUF or ring oscillator PUF, the amount of effort needed to implement an SRAM PUF is negligible. The “challenge” to an SRAM PUF is the memory cell address while the “response” is the uninitialized power-up value of the cell. The layout of a 6T SRAM cell is perfectly symmetric and hence no systematic offset exists. Hence, the power-up state is determined by process variation induced mismatch between the two cross-coupled inverters. The manufacturing variability is random, unclonable and uncontrollable, which gives each chip a unique key. The main design consideration for SRAM PUFs is making sure the key is 100% stable. Given the same challenge, we expect the PUF to generate the same key regardless of the operating condition. This is difficult to achieve since the static mismatch of a SRAM cell may not always be large enough to overpower the random thermal noise under all operating conditions.

Temporal majority voting (TMV) is a popular technique for improving the stability of PUF responses [53-54]. The basic principle is to repetitively test the PUF using the same challenge and take the majority value of the responses as the final output. Increasing the number of repetitive tests allows the tester to find keys that are more stable. The main drawback of TMV is that it usually involves a large number of tests (e.g. 100’s or 1000’s of power-ups for SRAM PUF), which is prohibitive in terms of test time and test hardware.

Furthermore, even with such a large number of trials, the stability criterion cannot be made very stringent, so there’s a high possibility that the stable cells found using TMV will become unstable in future evaluations. In [53], a combination of TMV, burn-in hardening and ECC circuits were used to meet the stability requirement. However, these techniques introduce significant hardware overhead. To make matters worse, TMV may have to be performed under extreme voltage and temperature conditions to ensure the responses are truly stable. This is very time consuming and difficult to implement in a high-volume production flow. A bit selection algorithm proposed in [55] utilizes just two test conditions; high-temperature/low-voltage and low-temperature/low-voltage. This is more efficient and less costly for selecting stable bits, however, it involves changing the test temperature which is undesirable. Error Correcting Codes (ECC) can be used to correct the unstable outputs using a software algorithm. However, ECC may leak secret information and introduce extra design complexity and communication overhead.

The instability of TMV selected cells stems from the marginally stable cells, i.e., cells that appear to be stable during TMV tests but become unstable under extreme environmental conditions. These cells are more stable than an average cell, but less stable than the strongest cells that consistently produce the same response. Finding the strongest cells in a large SRAM array requires a prohibitively large number of repetitive tests and may involve changing the voltage and/or temperature. To overcome the limitations of TMV, we propose a method for selecting the most stable cells in an SRAM array based on just two power-up tests. Compared to TMV, our approach reduces the test time and obtains more accurate information pertaining to the stability of cells. Experiment results from off-the-shelf SRAM chips show that the cells selected by our proposed strategy are 100% stable under extreme test conditions.

4.2 Data Remanence Based Stable Key Selection

4.2.1 Data Remanence Based Approach

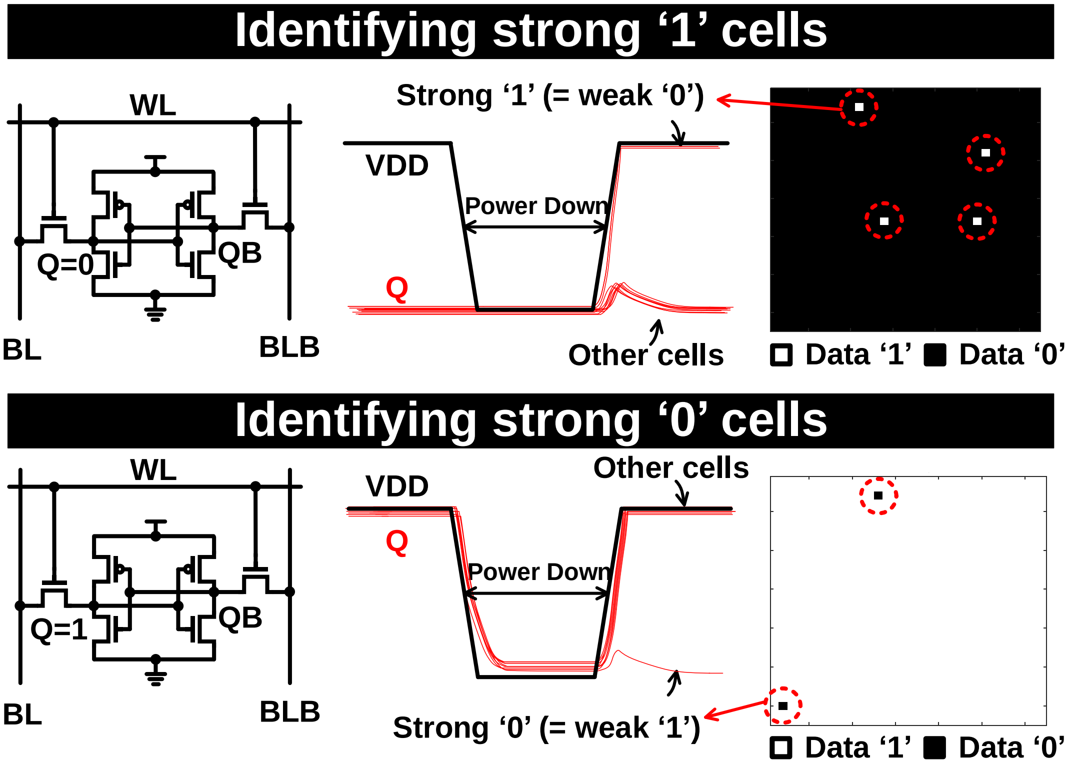


Fig. 4.1: Proposed data remanence based technique to rapidly select the most stable cells in a large SRAM array.

Fig. 4.1 illustrates the basic principle of the proposed data remanence based stable cell selection method. According to Wikipedia, remanence is defined as "the magnetization left behind in a ferromagnetic material after an external magnetic field is removed" [56].

Similar to this concept, we remove the supply voltage for a short period after initializing the array to all 0's or all 1's.

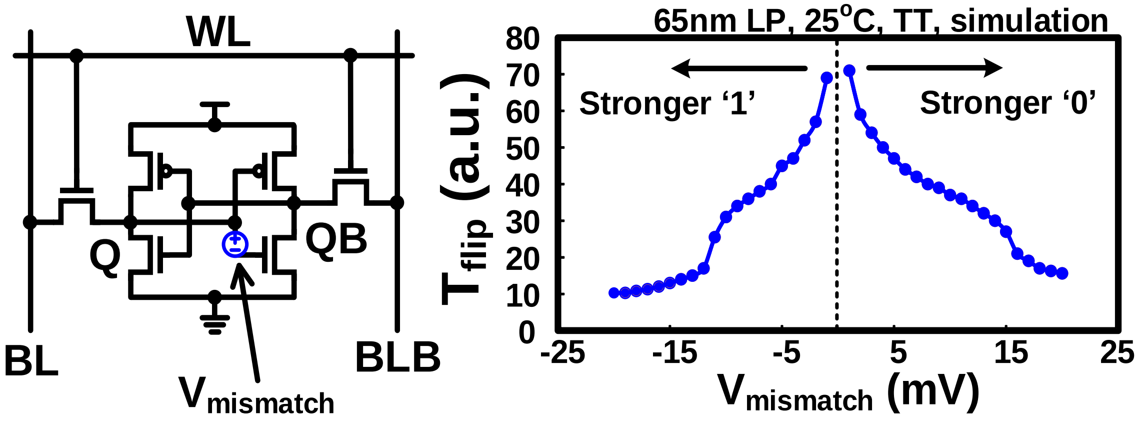


Fig. 4.2: Required power down period to flip SRAM data with different skew.

As shown in Fig. 4.2, the first few bits to flip after the brief power down period are ones that are strongly biased to the opposite value. For instance, if the entire array is initialized to 0's, the first bits to flip to 1's after the short power down period are the strongest '1' bits in the array. In traditional SRAM PUF power up operation, the response is only related to the inherent transistor mismatch of each SRAM cell. Data written to the cell doesn't affect the power up state because all storage nodes have fully discharged to an unbiased state due to leakage current. In other words, the data remanence is fully decayed. However, if the cell is powered back immediately after a power down, then the storage node data will revert to the previous data because the data remanence is very strong. If the power down time is long enough to make the data remanence comparable to the transistor mismatch, then some cells will revert to the previous data, while other cells will flip to the opposite value. As shown in Fig. 4.1 (top), if '0' is written to all the cells, node Q will be 0V and node QB will be VDD before the power down. After a short power down period, the majority of the QB nodes will revert back to VDD upon a power up due to the remanence charge on the Q and QB nodes. However, the cells with the strongest bias towards the opposite value will flip to ‘1’ as illustrated in Fig. 4.1 (top). The transistor mismatch in these cells produces a strong bias which cannot be overpowered by the small data remanence. We utilize this behavior to find the most stable ‘1’s in a large SRAM array. Similarly, by writing ‘1’ to all the cells in the SRAM array and asserting a short power down period, we can find the most stable ‘0’s, which are the first cells to flip when the power is turned back on, as highlighted in Fig 4.1 (bottom).

Note that a “remanence decay” based side-channel attack method was proposed in [57] where a pulsed power supply was used to recover the secret keys generated by an SRAM based PUF. Our approach employs the same method but for a totally different application: i.e., finding the most stable bits in an SRAM PUF with minimal test time and test hardware overhead.

4.2.2 Characterization of Data Remanence Effect

To verify the proposed technique in real hardware, we performed data remanence tests on off-the-shelf SRAM chips from Microchip Technology. Each chip contains 512 k memory cells. The first step is to determine the appropriate power down time. If the power down time is too long, then the data stored in the array is completely collapsed and the SRAM will power up to its uninitialized state. On the other hand, if the power down period is too short, then the data will deterministically revert to the previous written state. Therefore, the power down period should be carefully characterized. Fig. 4.3 shows the percentage of flipped cells when the power down time is swept from 100ms to 1000ms.

The SRAM chips we tested were fabricated in an ultra-low leakage technology, requiring a relatively long power down time to observe data remanence effects. We expect a much shorter data remanence time (e.g. microseconds) for SRAMs built in advanced CMOS technologies. The overall data remanence trends will be agnostic to the technology node.

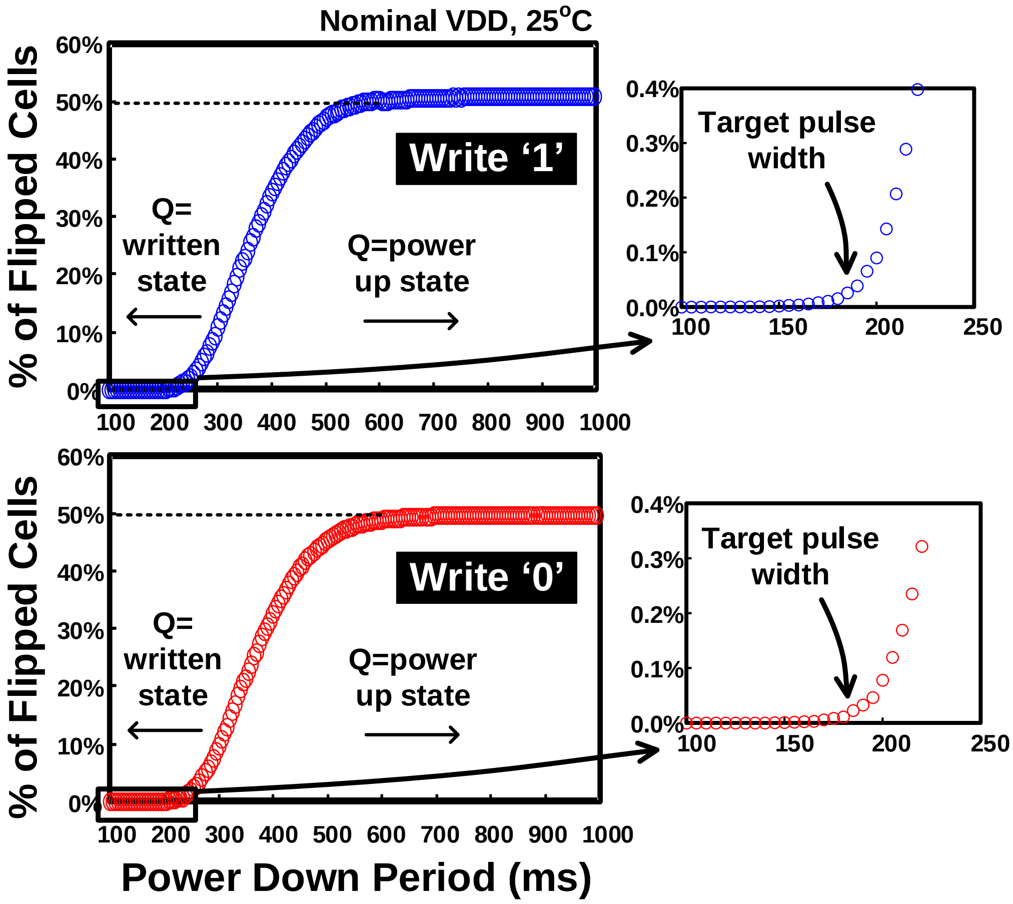


Fig. 4.3: Percentage of flipped cell versus when writing all '1's (upper) and writing all '0's (lower) to the SRAM cells.

Data was collected from all 512 k cells of each SRAM chip. In both write '1' and write '0' cases, the cells start to flip after a power down period of about 130ms. When the power down period increases to about 600ms, the flip ratio reaches 50% which corresponds to the SRAM power up state. For authentication applications, we are only interested in finding the most stable ‘1’s and ‘0’s in the entire array, and therefore we need to select a power down period that is short enough so that only the most oppositely biased cells flip. This time is usually less than 200ms, which is about 3 times shorter than the power down time required for a standard SRAM PUF evaluation (approximately 600ms in our case). Although the proposed data remanence method requires all SRAM cells to be written to ‘1’ or ‘0’ before the power down, the time needed to write data into the array is negligible compared to the power down time required to clear the data remanence in the SRAM. Moreover, our approach only requires two tests to select the most stable cells in the SRAM array; one test for selecting stable ‘1’ cells and the other for selecting stable ‘0’ cells. TMV may require hundreds or more power ups to find the robustly stable cells, and we must wait at least 600ms between two consecutive power-up tests. In short, compared to TMV, the proposed technique requires not only fewer power-ups (hundreds or thousands → 2) but also shorter power down periods (600ms → 200ms) which significantly reduces the overall test time.

For a better understanding of the proposed technique, Fig. 4.4 shows data remanence of a small 1kbit sub-array for different power down periods. Fig. 4.4 (upper) shows the bit map for selecting stable ‘1’s. Data ‘0’, denoted in black, is first written to the whole array, and then the power supply is turned off, letting the data stored in the SRAM to decay. When the power supply is turned on after 130ms, the first cell in the 1kbit array flips. This cell corresponds to the most stable ‘1’ cell in this array. When the power down period increases further, more cells flip, which are the next most stable ‘1’ cells. Depending on the number of stable cells we want to select, the amount of data remanence needs to be tuned accordingly by changing the power off period. Stable ‘0’s can be selected in a similar way, as shown in Fig. 4.4 (lower).

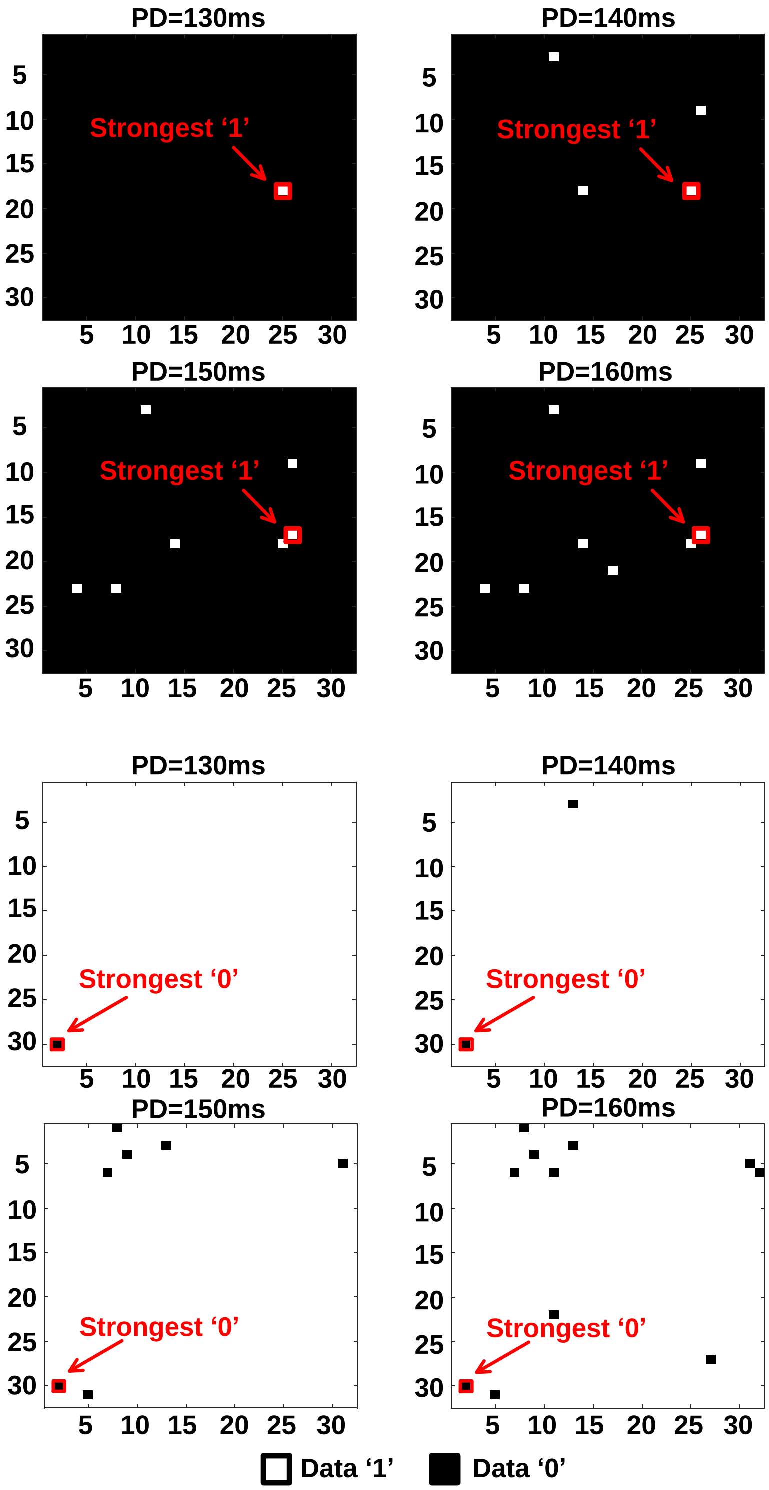


Fig. 4.4: Cell flip maps of randomly selected 1K SRAM cells under different power down periods (PD).

As seen in Fig. 4.4, the data remanence based technique allows us to measure the extent to which a cell is stable, by looking at the order of the cell flips. By sweeping the power down time and recording the order of the cell flips, we can sort and list the strength levels of each cells from the strongest '0' cell to the strongest '1' cell. As such, we can obtain complete knowledge of the cell's strength of the whole SRAM array by sweeping the power down time. For example, if we want to sort the cells from strongest '0' to balanced '0', we first write data '1' to the whole array and sweep the power down period from 100ms to 600ms for the SRAM chips used in our experiment. The responses of all cells are recorded and sorted by retention time, as shown in Fig. 4.5 (upper). The sorting order is shown for 50% of the cells (i.e., 256 k), since the other half will always generate a '1' irrespective of the power down time. When applying the data remanence method to generate stable keys, we only select the strongest cells. The most biased cells can be seen more clearly in the zoomed-in plot. Depending on how many stable bits we want to select, we can vary the power off period. For example, for a 256-bit key, we select roughly 128 stable '0's and 128 stable '1's from 512 kbit cells, which is 0.05% of the total cells available. The power off period should be around 185ms. If we want to select 512 bits, we can increase the power off period to around 195ms to allow more flips. In a realistic scenario, we can select more bits than we need and then pick the number of stable bits requested by our target application. Similar plots of the data '1' case are shown in Fig. 4.5 (lower). The bit index shows the order from the strongest '0's to balanced cells to strongest '1's, from top to bottom. We can observe from Fig. 4.5 that by using the proposed data remanence technique and sweeping the power down period, we can sort the cell strength levels in very fine steps. For comparison, we used the conventional TMV method to find stable bits in the same SRAM array. 1,000 power-up tests were performed and the probability of each cell being '1' or '0' were calculated. We found that 40% of the cells are stable '1' through all 1,000 tests and 41% of the cells are stable '0' all the time. However, many of the allegedly stable cells will show unstable behavior at different voltage and temperature conditions, or when the SRAM is subject to aging. To determine the 256 most stable bits from a 512 kbit SRAM array, which is only 256/512k = 0.05%, we may need millions of repetitive power up tests for TMV, which is impractical.

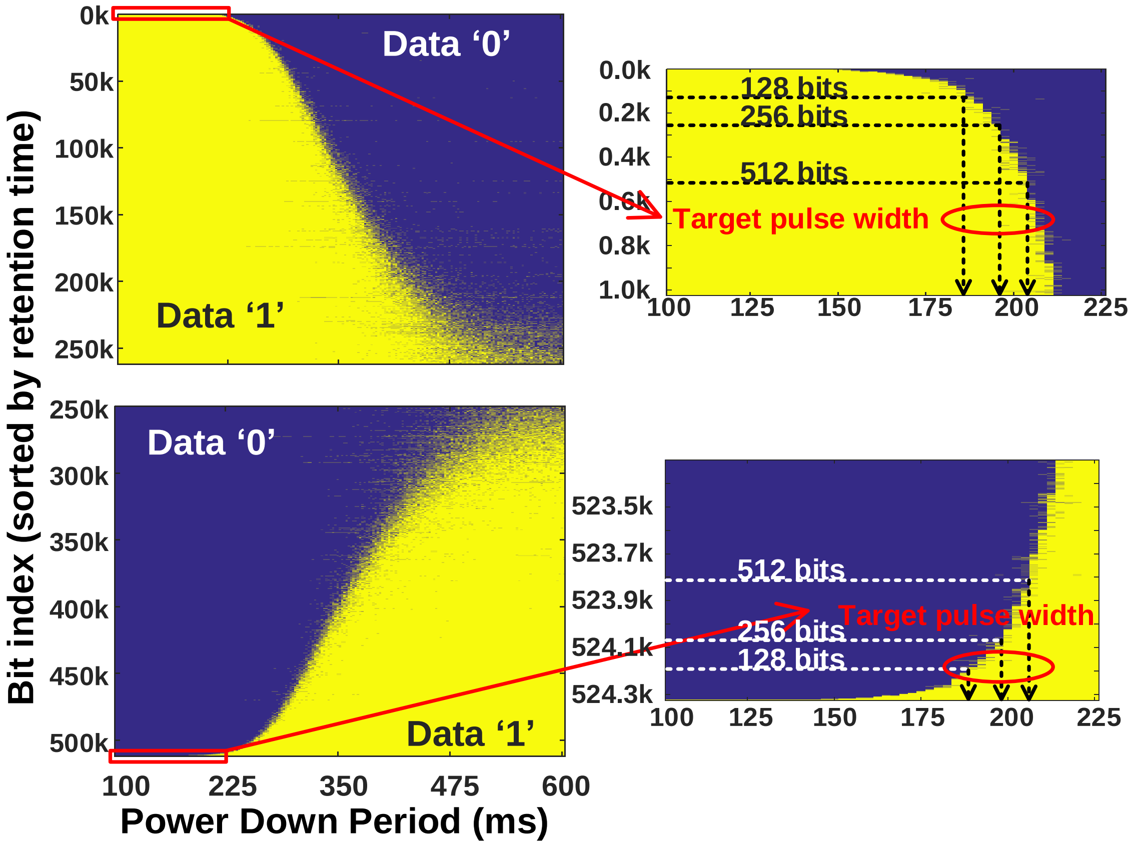


Fig. 4.5: Measured SRAM data remanence for data 1 (upper) and data 0 (lower). Cells with the shortest retention times are highlighted in the zoom-in plots.

Fig. 4.6 (left) summarizes the test flow for characterizing data remanence while sweeping the power down time. Note that we perform this extensive test on one of the chips to determine the appropriate power down period of all chips. An attractive feature of the data remanence test is that it can be performed at any temperature. The top 0.05% stable cells found from the power down sweep test will remain stable at different temperatures and voltage conditions. For actual SRAM PUF applications, we use the power down period found from the extensive data remanence test and run the test only two times; one for selecting strong '1's and the other for selecting strong '0's. The enrollment test shown in Fig. 4.6 (middle) stores the location of the most stable bits on-chip. In the key generation phase, we simply power up the SRAM, and key values are retrieved from the stable bit locations.

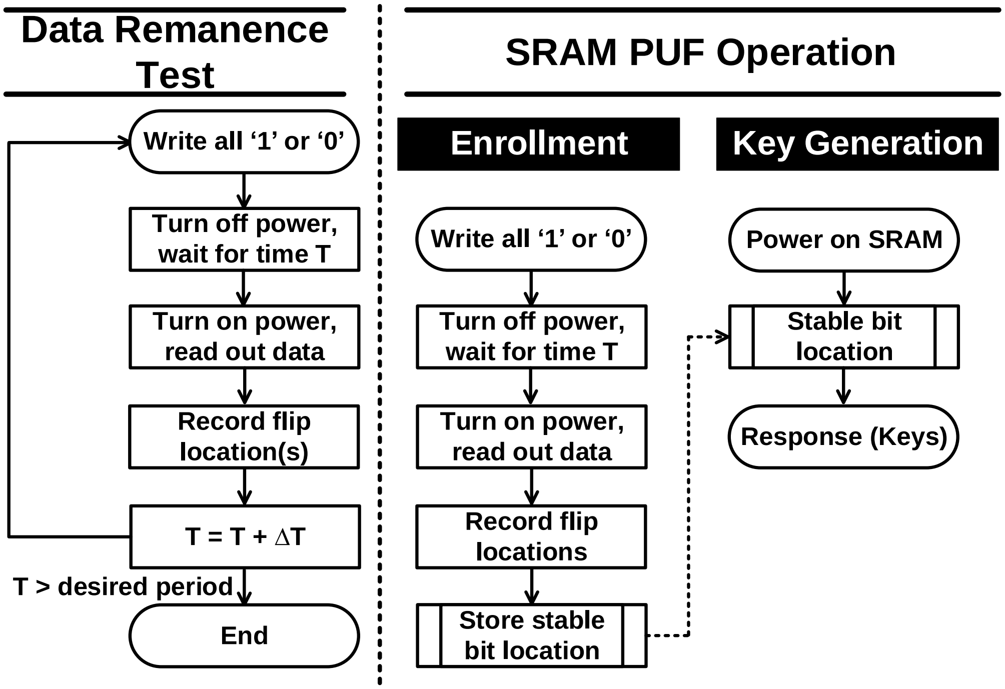


Fig. 4.6: Flow chart for data remanence test, enrolment test, and key generation test. The power down time T for enrolment phase can be determined based on a one-time data remanence test performed at any temperature.

4.3 SRAM PUF Measurement Results

This section shows detailed measurement results verifying that the stable cells selected using our proposed technique are indeed stable across different environmental and aging conditions. Fig. 4.7 shows the measurement set up. The pulsed power supply and other digital signals are provided by a PXI based data acquisition system. GPIB controlled power supplies were used to stress the chip. Chips were measured inside a temperature controlled chamber.

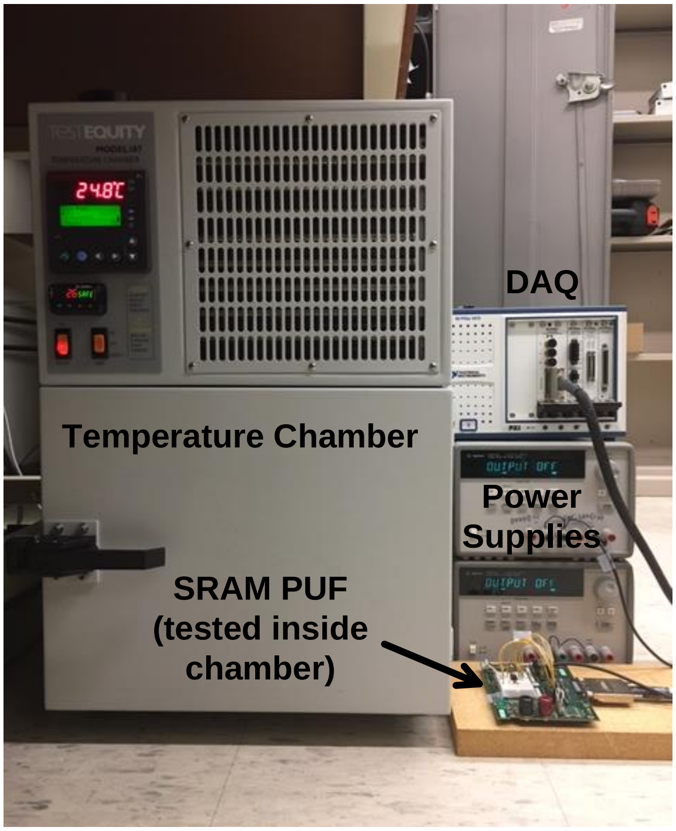


Fig. 4.7: SRAM PUF measurement set up including a temperature control chamber.

4.3.1 Uniqueness of Key

The maximum number of bits for encryption algorithms like AES, is usually 256 bits [51]. So, the target number of bits for our SRAM PUF based key generation is 256 bits. However, we also present results for generating 512 bit and 1024 bit keys for better understanding. Each SRAM chip we tested has 512 kbits in total, so the goal is to select the 128 most stable '1's and the 128 most stable '0's from 512 kbits, which correspond to the most stable 0.05% of the bits. The power supply off period for selecting 256 stable bits in this test was about 190ms. Alternatively, we can select more than 256 bits and randomly pick 256 cells and record their locations to generate the bit location address. Fig. 4.8 shows 256 bit keys generated from 4 different SRAM chips showing an average inter-chip Hamming distance of 0.4935, confirming the uniqueness of the keys. Note that the precise location of the stable bits is different in each SRAM chip.

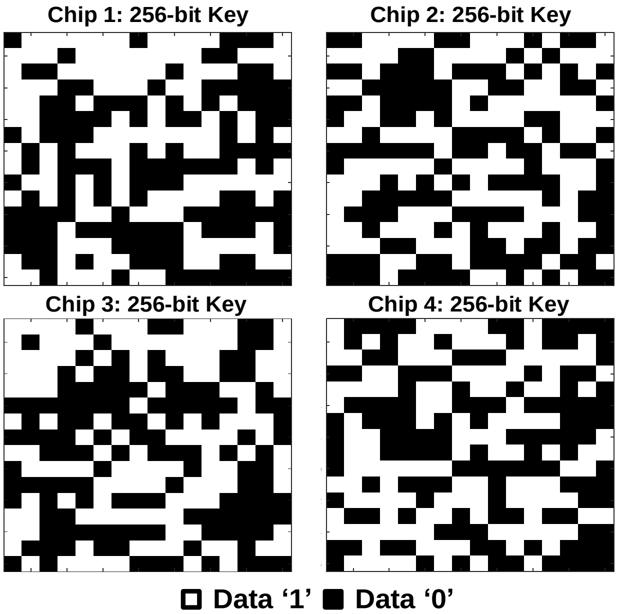
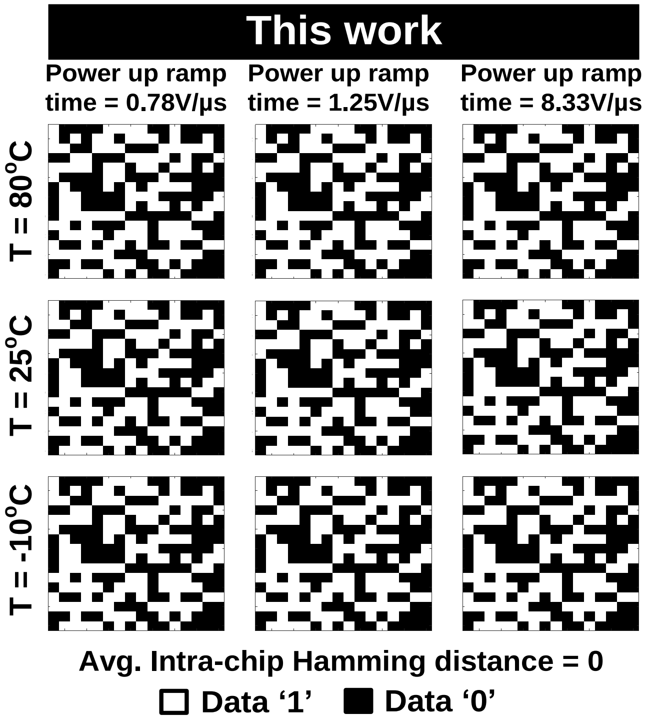
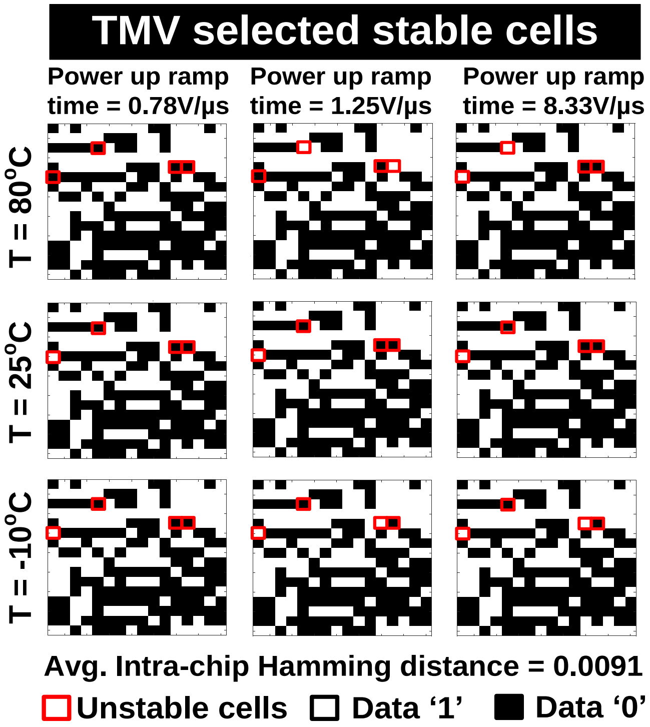


Fig. 4.8: Keys generated from 4 different SRAM chips. The inter-chip Hamming distance between the 4 different keys is 0.4935.

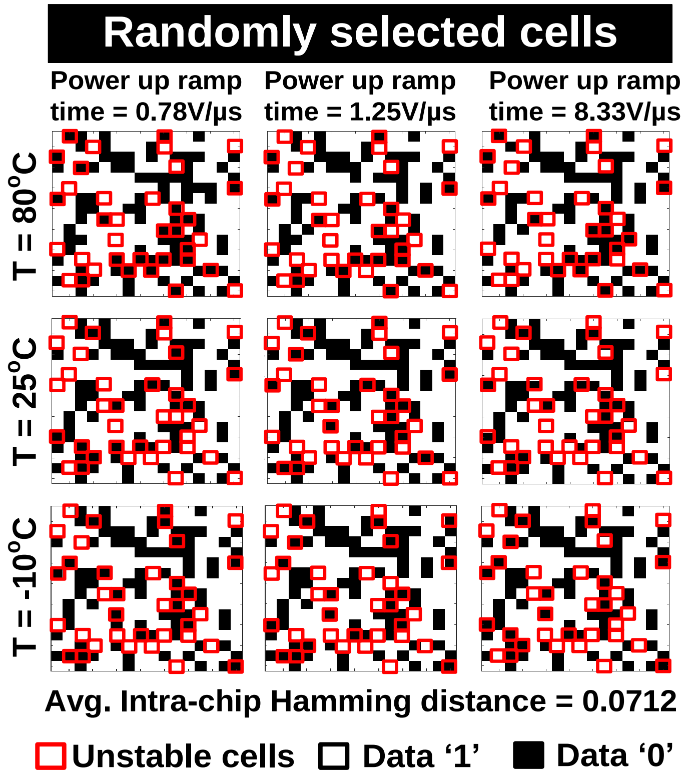
4.3.2 Effect of Power Ramp-up Time and Temperature



(a)



(b)



(c)

Fig. 4.9: Measured SER cross-section for inverter, NAND and NOR gates at multiple supply voltages.

To verify that the key selected using the proposed technique is stable under different environmental conditions, the voltage ramp up rate and temperature were varied. Note that during the SRAM power up, the state is resolved during the very beginning of the power supply ramp up, so the final power supply level will not affect the stability of the SRAM PUF. Instead, the ramp up rate of the power supply may have an effect on the stability of the responses. To evaluate this effect, the ramp up rate of the supply voltage was changed from 0.78µV/s to 8.33µV/s. Testing was performed at three temperatures; 80°C, 25 °C and -10 °C. Fig. 4.9 (a) shows the measured SRAM PUF responses and the average intra-chip

Hamming distances under different test conditions using the proposed technique. Power up tests were repeated 10 times under each condition to ensure that the responses are absolutely stable. Since the responses are always stable, there is no need for further processing of the responses using ECC. This reduces the circuit complexity and communication overhead. For comparison, we also select 256 stable bits using the TMV method based on 1,000 repetitive power ups. That is, we only chose the cells that are consistently '0' or consistently '1' throughout the entire 1,000 trials. As mentioned earlier, even with 1,000 repetitive power up tests, we are only able to discriminate the top 81% stable cells which includes marginally stable cells. As a reminder, the proposed data remanence technique can select the top 0.05% stable cells with just two power-up tests. The responses using the 1,000 trial TMV method are shown in Fig. 4.9 (b). The unstable bits are highlighted in red. It can be seen from the cell maps that 4 cells are unstable when the temperature or power supply ramp up rate changes, which is not acceptable for ECC-less key generation. Finally, the power up responses from 256 randomly selected SRAM cells are shown in Fig. 4.9 (c). As expected, many bits are unstable when tested under different conditions. These measurement results confirm that the data remanence technique proposed in this chapter can reliably identify the most stable bits in an SRAM array with only two power-up tests. The stable keys can be selected under the nominal voltage and room temperature condition, so it can greatly reduce the test cost and test time. We also selected the 512 most stable bits and 1024 most stables bits and their responses were proven to be 100% stable under various voltage and temperature conditions, confirming the effectiveness of this technique.

4.3.3 Effect of Device Aging

Device aging may cause the PUF response to change over the lifetime of a product, which is undesirable [58]. In particular, bias temperature instability (BTI) is known to be the dominant aging mechanism in SRAM cells due to the low activity factor and DC stress nature [59-60]. BTI manifests as an increase in threshold voltage, and occurs when PMOS or NMOS transistors are biased with a negative or positive gate voltage, respectively [61]. Depending on the data stored in the SRAM cell during stress, BTI can either emphasize or de-emphasize the process variation induced mismatch. Emphasizing the mismatch will harden the responses and make them more stable, while de-emphasizing the mismatch will have the opposite effect [58]. Since our goal is to verify the stability under the worst case condition, we stress the SRAM array with the power-up state which will decrease the mismatch between the two cross-coupled inverters. This de-emphasizes the mismatch and makes the bits more unstable during the actual power up test. The SRAM chips were stressed under a static DC condition (i.e., no switching or toggling) for 72 hours using a 1.5xVDD supply voltage. Before applying the stress voltage, the fresh PUF response is read out for reference. The SRAM PUF responses of the selected 256/512/1024 most stable cells are read out every hour and the intra-chip Hamming distances are calculated against the fresh response (Fig. 4.10).

For comparison, the intra-chip Hamming distances of stable cells selected using the TMV method and the random selection method are also shown in Fig. 4.10. We can see that the stable cells selected using the proposed technique are 100% stable throughout the entire stress experiment. TMV leads to 8% bit flips at the end of the 72 hour stress period, while the number of bit flips for randomly selected cells is 15%. Experimental results confirm that the cells selected using the proposed technique remain 100% stable for the stress condition used in this work.

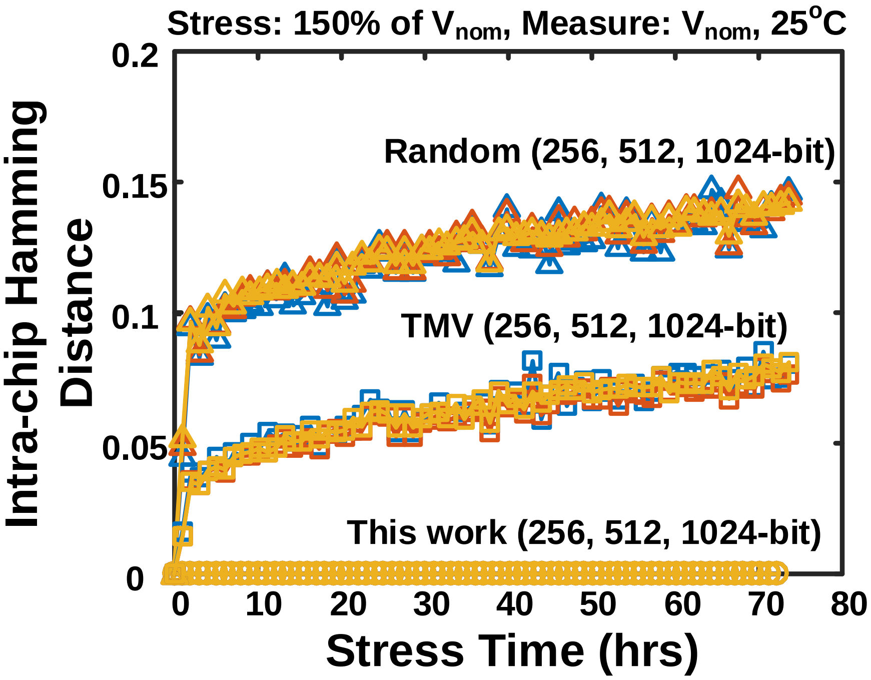


Fig. 4.10: Average intra-chip Hamming distances of different techniques (i.e., proposed, TMV, and random selection) versus stress time for 256, 512, 1024 selected bits.

4.4 Conclusion

In this work, we proposed a data remanence based technique, to efficiently generate 100% stable keys from an SRAM PUF. By writing ‘1’s or ‘0’s to the entire SRAM array and recording the bit flip locations after a brief power down period, we can identify the strongest ‘1’ and strongest ‘0’ cells in a large SRAM array with just two power-up tests. We have confirmed that the responses selected based on the proposed technique are 100%

stable under different voltage and temperature variations, as well as under BTI aging. The proposed technique doesn’t require repetitive power-up tests as in conventional TMV methods. Since the responses are 100% stable, there’s no need for ECC, which simplifies the authentication hardware.

Chapter 5. A Passive IC Tamper Sensor Design based on an Exposed Floating Gate Device in Standard Logic Processes

5.1 Introduction

Counterfeit ICs entering the supply chain have been causing significant financial damage to the electronics industry. According to recent estimates, the electronics industry is losing $100 billion in terms of worldwide revenue every year because of counterfeit parts [62-65]. Different types of counterfeit methods have been reported, including recycling, remarking, overproducing, cloning, forged documentation, defective and tampered chips [66]. Among them, recycled and remarked counterfeit electronics account for more than 80% of all reported counterfeiting cases [67]. Recycled ICs are usually recovered from old printed circuits boards (PCBs) and then relabeled and disguised as new parts. Recycled ICs can pose great concern for customers, since they may function correctly initially, but cause early failures down the road. Detecting these counterfeit electronics efficiently is a critical aspect of preventing counterfeit ICs. Several techniques have been proposed to detect counterfeit electronics. They can be broadly classified into two categories: physical inspection and electrical test [68]. Physical tests are usually destructive and must utilize test instruments, making it very costly and time-consuming. Furthermore, it usually requires a human expert to interpret the test results [69]. Electrical inspection on the other hand uses on-chip sensors [70-71] to automatically flag compromised chips. Despite their promise, on-chip sensors cannot stop untrusted foundries from overproducing chips [68], and they cannot fully detect physical attacks such as die removal and desoldering. These physical attacks are known to be very common in today's global supply chain.

In this work, we present a logic-compatible embedded flash (eflash) based tamper sensor which utilizes an exposed floating gate (FG) structure to detect whether or not an IC has been physically compromised. The proposed eflash based sensor doesn't require any power source for the sensing operation which is a critical requirement for counterfeit IC detection. Additionally, the sensor can be implemented using IO devices readily available in any standard CMOS technology. Part of this work was presented in [72].

5.2 5T Eflash Basics

The sensor utilizes an eflash cell whose FG node is exposed to the environment such as the chip cavity. Fleeting changes in the electron charge stored in the eflash cell can be captured by the proposed sensor. The eflash circuit is implemented using discrete IO devices available in a standard CMOS process so no modification is required to the process technology [73]. The proposed eflash based sensing has the advantage of offering a secure non-volatile storage solution as well as the ability to retain stored data without a power source. Fig. 5.1 shows the 5T eflash cell schematic and layout along with the bird's eye view of the cell [74]. The eflash cell consists of five transistors: coupling device M_1, erase device M_2, program/read device M_3 and two selection devices S_1 and S_2. The FG node is formed by connecting the transistors M_1-M_3 in a back-to-back fashion. The width of M_1 is made larger than those of M_2 and M_3 to achieve a high coupling ratio (CR). A large CR ensures that the FG voltage closely follows the PWL voltage applied to the coupling device, maximizing the electric field for efficient Fowler-Nordheim (FN) tunneling through the dielectric of M_2 and M_3.

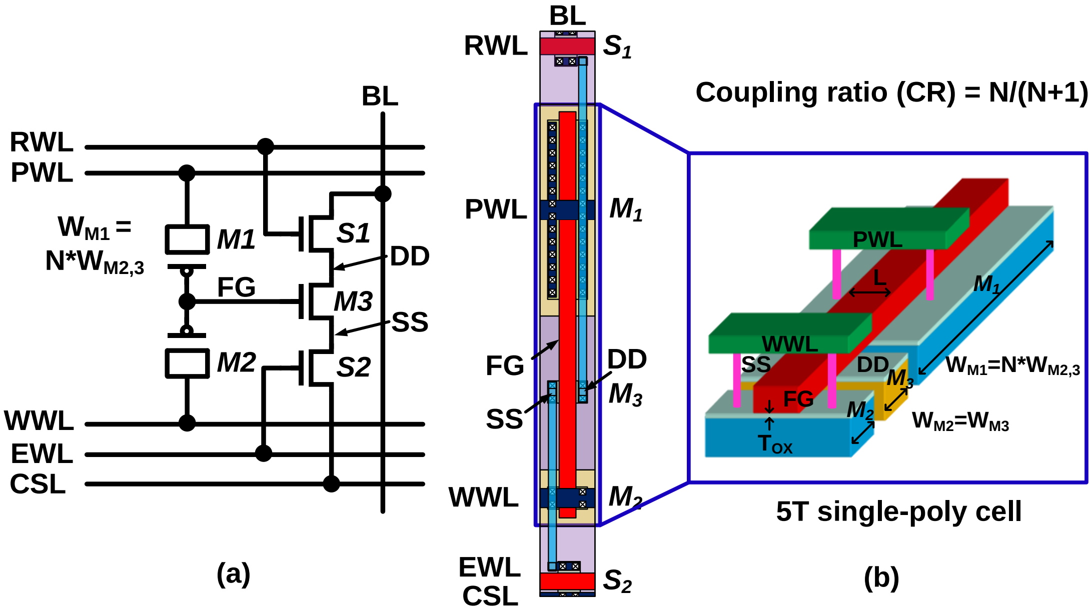


Fig. 5.1: (Left) 5T eflash cell structure; (middle) layout view; (right) bird's eye view of the cell.

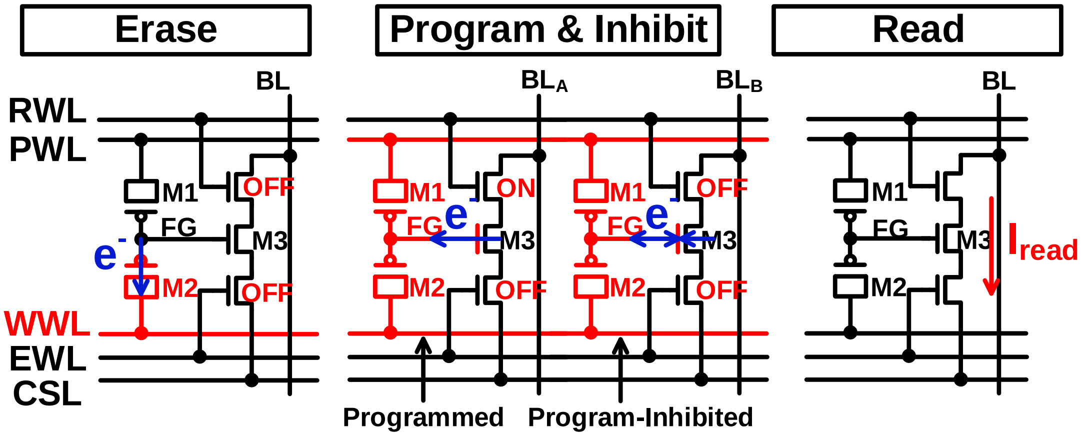


Fig. 5.2: Three operating modes of the 5T eflash cell.

Fig. 5.2 shows the three operating modes of the proposed eflash cell. In erase mode, a high erase voltage is applied to WWL which removes the electrons from FG through M_2. In program mode, the upper selection device is turned on while a high program voltage is applied to both PWL and WWL, causing electrons to tunnel from M_3 into FG. The 5T eflash cell also has program inhibit capability, which is achieved by turning off the upper selection device during the program operation. During program inhibition, the source voltage of the read device is boosted, preventing the electrons from being injected to the FG node via M_3, thus the program operation in this cell is inhibited. This allows us to selectively program a specific eflash cell in a row.

Electron charge stored on the FG node affect the threshold voltage of M_3, and in read mode, the threshold is read out by measuring the bitline (BL) current. In our proposed eflash based sensor, the FG node is exposed to the chip cavity through pad openings, so any physical sources that affect the number of electrons in the FG node can be detected. To activate the sensor, the eflash cells must be programmed once to populate the FG with electrons. After that, the sensor can record tamper events without a power source.

5.3 Sensor Test Structure in 65nm Technology

5.3.1 Basic Concept

To validate the proposed eflash based sensor concept, we first implemented a single test structure shown in Fig. 5.3 in a 65nm CMOS technology. All the transistors in the eflash cell are standard 2.5V I/O devices with a tunnel oxide thickness (T_{OX}) of approximately 5nm [73]. The FG node of the eflash cell is exposed to the environment by connecting a stack of metals from the bottom M1 layer to the top M7 layer as shown in Fig. 5.3. The dimension of the opening window is 7µmx7µm. Dummy floating metals are placed around the FG opening window to serve as charge collection metals. The opening windows of the surrounding dummy metals can be seed in Fig. 5.3 (left). Since the dummy metals are very close to the FG, they can help attract and collect electrons from the FG, which enhances the sensitivity. Threshold voltage of the read transistor is a function of the FG charge; i.e. fewer electrons on the FG node translates into a lower threshold voltage and hence a higher BL current.

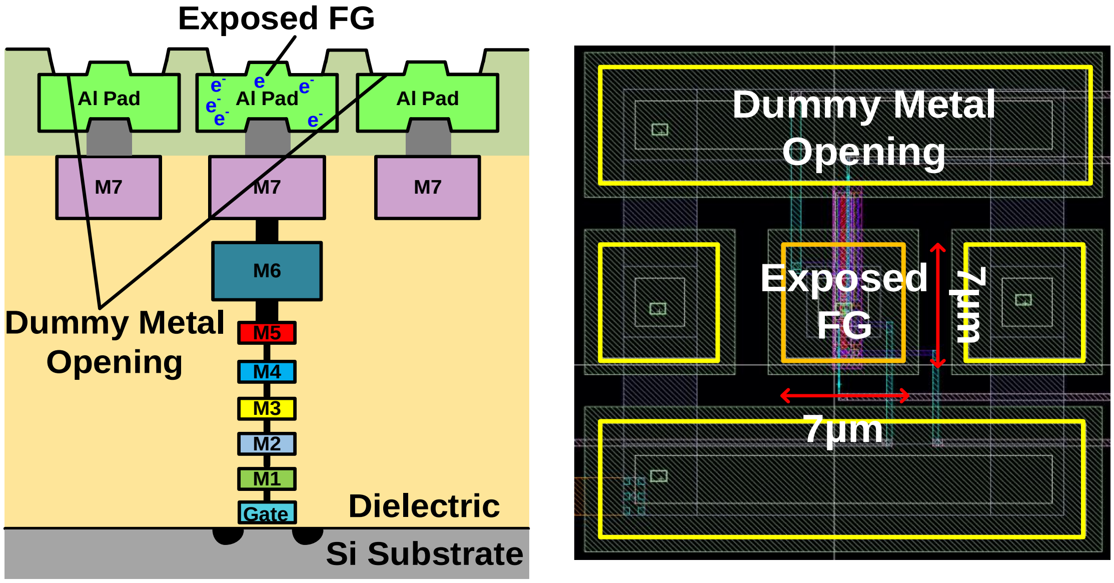


Fig. 5.3: (Left) Cross section view of eflash with exposed FG; (right) top layout view of exposed FG and surrounding dummy metals.

Since recycled ICs are usually recovered from a discarded circuit board, the chips are likely to be exposed to high temperatures during the removal process. High temperature facilitates FN tunneling of electrons, so this type of physical attack can be detected by our proposed eflash sensor. Another form of physical attack is opening the chip package to gain access to the silicon die. When the chip package is opened, even temporarily, the humidity or the dust particle density may change. Increased moisture in the air increases the surface conductivity of the FG node, thereby causing electrons on the FG to escape more rapidly. Dust particles with a positive net charge are attracted to the exposed FG node containing electrons with a negative charge. It’s also possible that the parasitic capacitance surrounding the FG node changes due to the presence of extra particles, which affects the erase and program operations. These can all cause variations in the FG node charge, and thus can be detected by the proposed sensor.

5.3.2 Measurement Results

We performed temperature and humidity tests mimicking physical attacks that a chip may encounter. Fig. 5.4 (left column) shows a permanent BL current jump after the temperature spike. This indicates that the electrons on the FG node are permanently lost, allowing the sensor to successfully record the event. Note that between the readout intervals, the power supply of the eflash sensor was shut down. In other words, the sensor is able to record physical attacks without any power.

Humidity test results are shown in Fig. 5.4 (right). We opened the lid of the chip and increased the moisture content in the air. The experiment was performed in an enclosed room, and the relative humidity near the chip was monitored. When the relative humidity was increased, a permanent jump in the cell current was observed. The measurements were highly repeatable which indicates that the proposed sensor can reliably detect humidity changes.

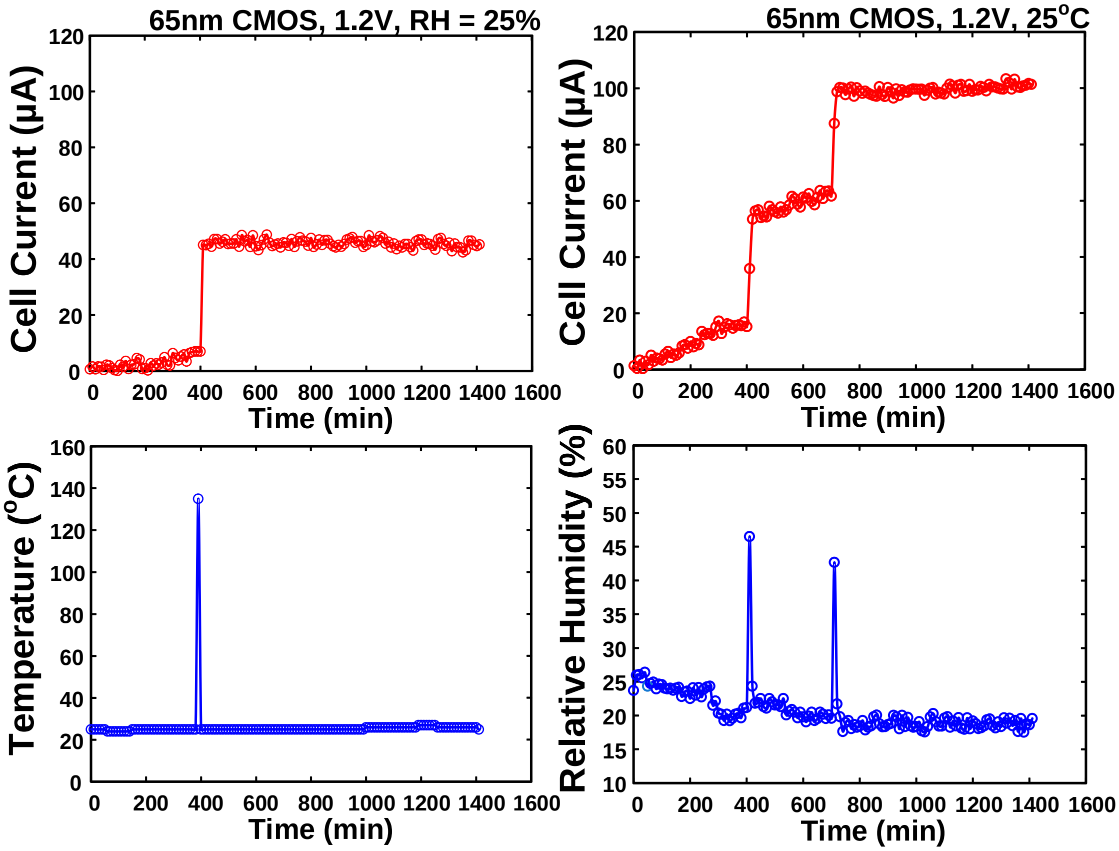


Fig. 5.4: Temperature (left) and humidity (right) attack test results of 65nm eflash sensor test structure.

After several humidity tests, we were no longer able to erase or program the sensor. The same behavior was found in multiple chips. Interestingly, we found that all chips with this specific behavior had particles landed on the FG node as shown in Fig. 5.5. The erase and program characteristics of the cells before and after the particle landing are shown in Fig. 5.6. This suggests that the proposed sensor can even detect particles entering the chip cavity which is another indication that a chip has been compromised.

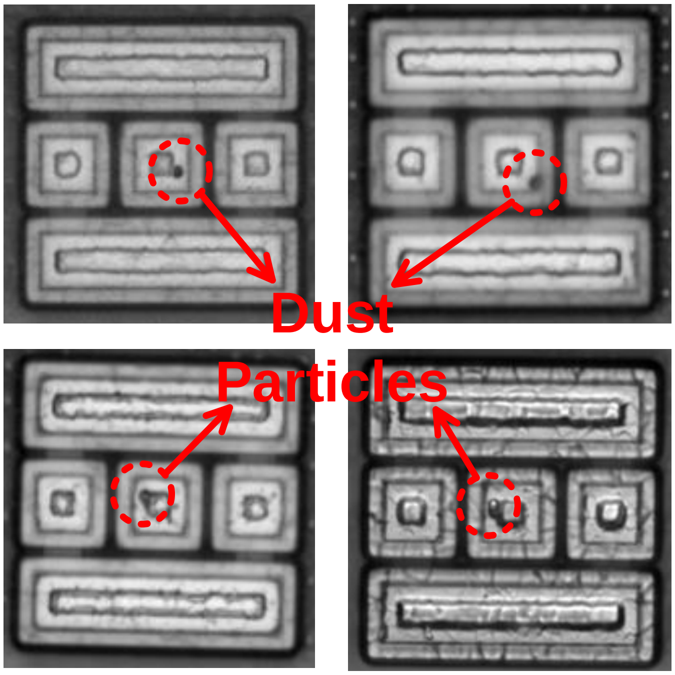


Fig. 5.5: Microscope images of 4 different chips with particles landed on the FG node of 65nm eflash sensor test structure.

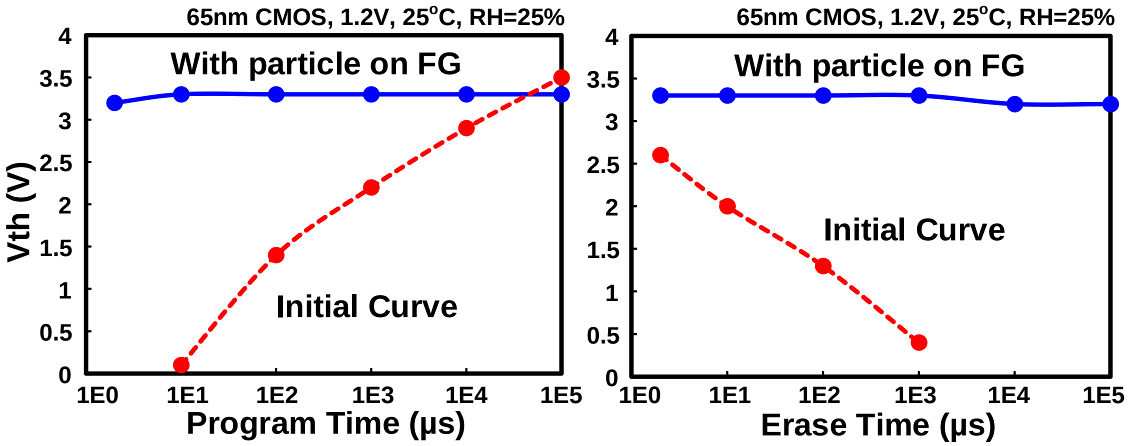


Fig. 5.6: Program (left) and erase (right) characteristics for eflash sensor with particles landed on the FG node.

5.4 16x16 Sensor Array in 0.35µm Technology

Encouraged by the 65nm test structure results, we implemented an array based test chip with complete peripheral circuitry. The new chip was fabricated in a 0.35µm technology for cost reasons. It's worth reiterating that the proposed sensor is agnostic to process technology so long as transistors with an oxide thickness greater than 5nm are available.

5.4.1 Implementation

A 16x16 sensor array was fabricated in a 0.35µm logic process. The differential cell structure with exposed and buried floating gates is shown in Fig. 5.7 [72]. The exposed cell is the working sensor and the buried cell is the reference cell. A differential read out enables higher sensitivity. Standard core devices in the 0.35µm technology with a nominal supply voltage of 3.3V and a typical 7.6nm tunnel oxide thickness (T_{OX}) was used for the cell implementation. No special devices were needed for the sensor implementation. Each sensing unit consists of one working sensor with an exposed FG and a reference sensor with a buried FG.

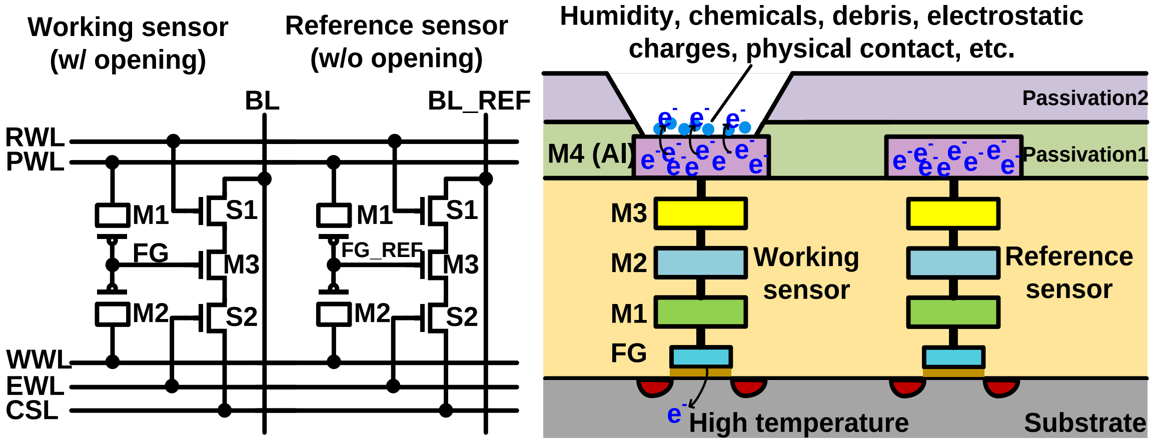


Fig. 5.7: Differential eflash sensor cell with exposed and buried floating gates implemented in 0.35µm technology.

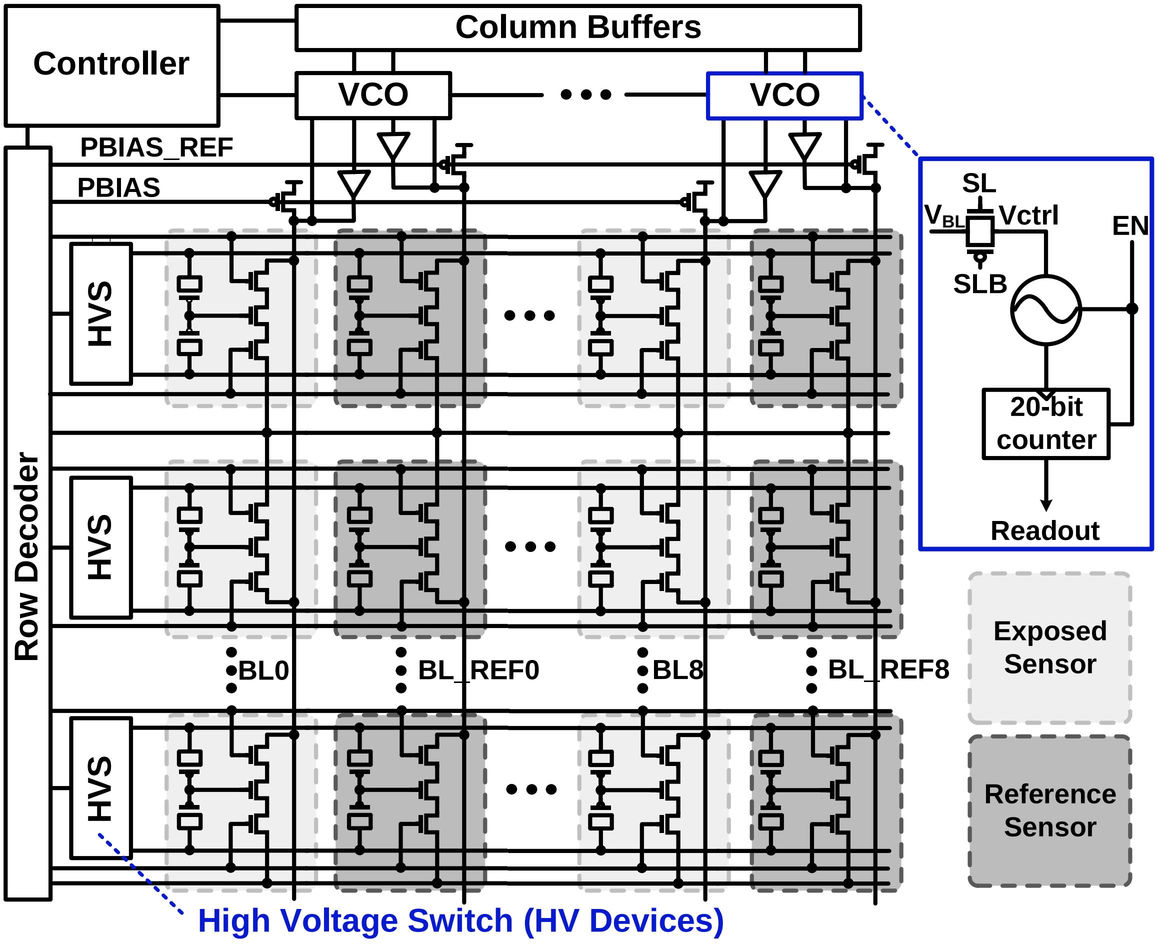


Fig. 5.8: 16x16 sensor array architecture.

The complete 16x16 array architecture is shown in Fig. 5.8. A high voltage switch (HVS) circuit is used to generate the high voltage signals for erase and program operations with correct timing, as well as the read voltages. Cascoding (or device stacking) was used extensively to prevent overstress issues in the high voltage switch circuit. The working sensors and reference sensors share the same readout path which cancels out common process, voltage, and temperature variation effects. The readout circuit consists of a voltage controlled oscillator (VCO) and a counter. The detailed readout operation is shown in Fig. 5.9. The BL voltage which reflects the charge stored on the FG node is converted to the corresponding frequency by the VCO circuit. Simulation results show that a 10mV difference caused by the FG node charge difference translates to a BL voltage of 0.3V. As mentioned earlier, more electrons on the FG means a higher threshold voltage and a lower BL current. This means that the final BL voltage is higher for the same pull up bias voltage PBIAS, resulting in a higher output frequency count.

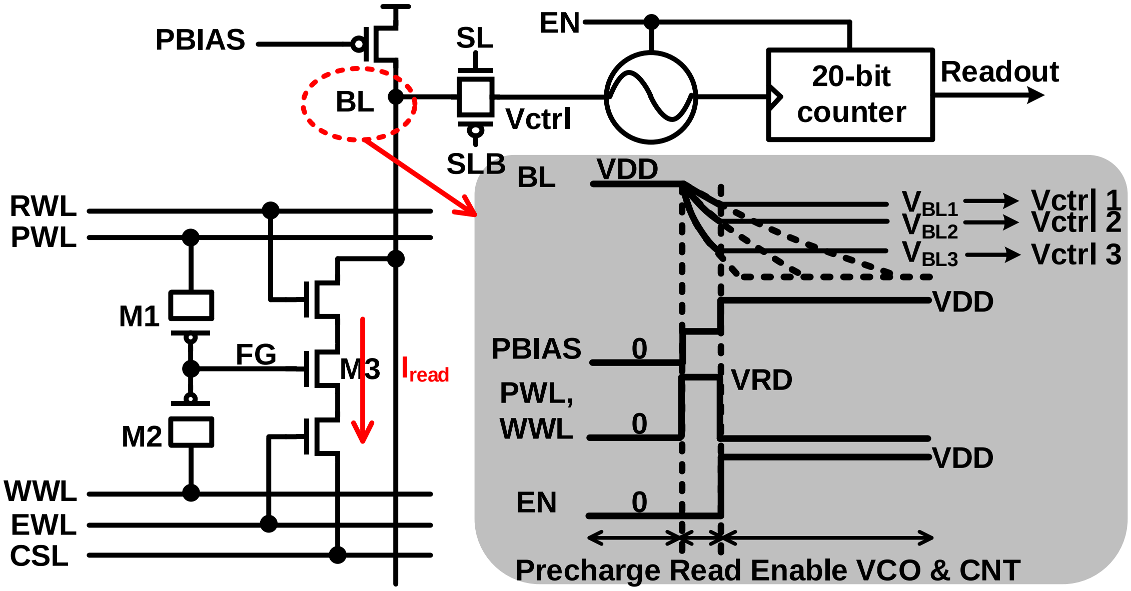


Fig. 5.9: VCO based sensor readout circuit and timing diagram.

To test a wide range of design choices, the sensor array consists of eflash cells with different transistor sizes and different opening sizes. The detailed sensor size and opening size configuration are shown in Fig. 5.10 (left). There are 16 different transistor size and opening size combinations. The die photo is shown in Fig. 5.10 (right).

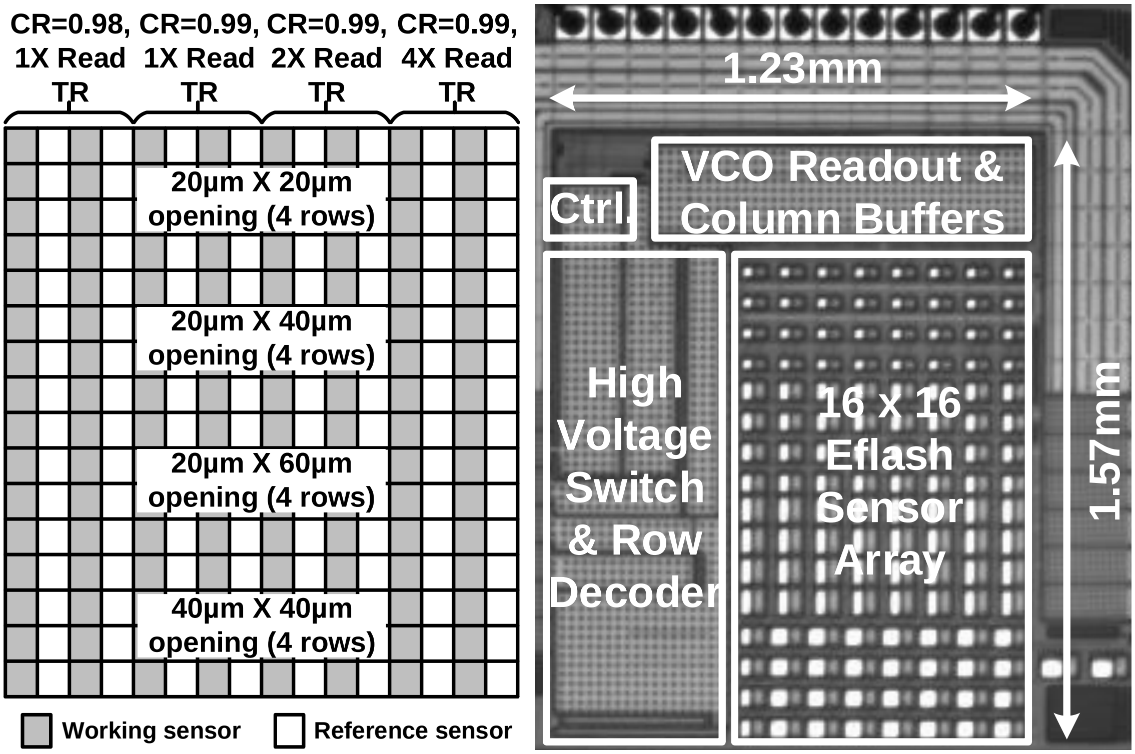

CR=0.98, 1X Read TR
CR=0.99, 1X Read TR
CR=0.99, 2X Read TR
CR=0.99, 4X Read TR
20μm X 20μm opening (4 rows)
20μm X 40μm opening (4 rows)
20μm X 60μm opening (4 rows)
40μm X 40μm opening (4 rows)
Working sensor
Reference sensor
1.23mm
1.57mm
Ctrl.
VCO Readout & Column Buffers
High Voltage Switch & Row Decoder
16 X 16 Eflash Sensor Array

Fig. 5.10: Sensor circuit and opening size configurations (left). Die photo (right).

5.4.2 Basic Functionality Tests

Fig. 5.11 shows the VCO frequency characterization results before any program or erase operations. The frequency variation of the same VCO for different readout trials was 0.27% while the frequency variation between different VCOs was 1.51%. Process, voltage, and temperature (PVT) effects can be cancelled out by measuring the frequency difference between the working sensor and the reference sensor. Also note that the frequency shift caused by physical attacks, which will be shown later, is usually 20% or more. This shift is significantly higher than the frequency variation induced by PVT effects.

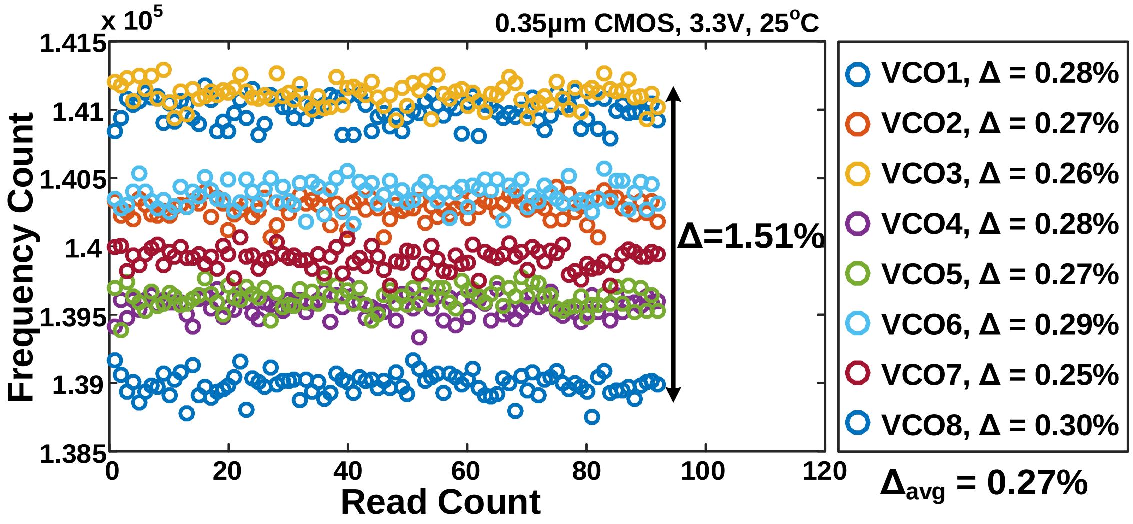


Fig. 5.11: Characterization of VCO frequency variation of the readout circuits.

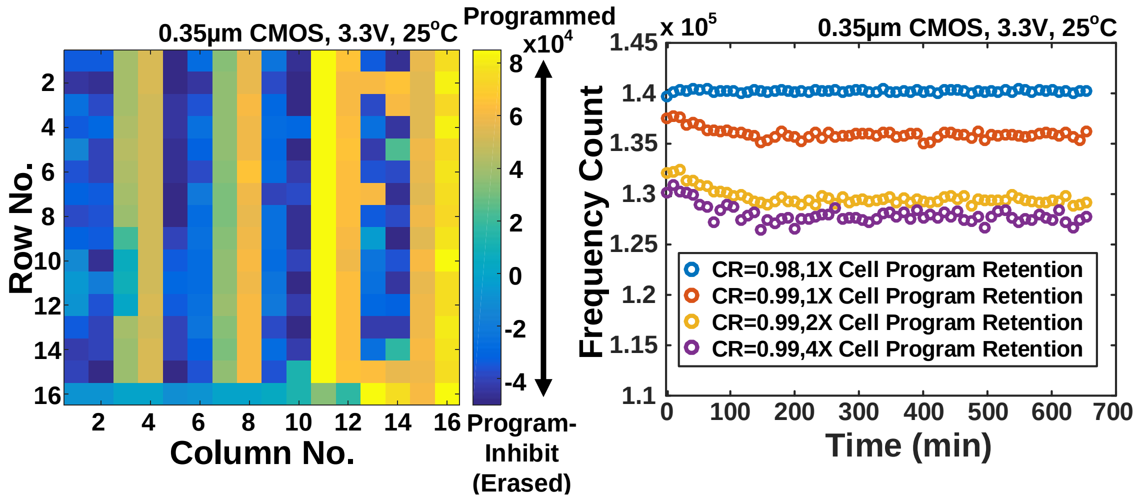


Fig. 5.12: (Left) Erase, program and program-inhibit characteristics and (right) cell retention results.

Fig. 5.12 (left) shows the erase, program and program inhibit characteristics of the 16x16 sensor array. The entire array was initially erased, and alternating columns were programmed while the other columns were program inhibited. The program inhibited cells remain in the erased state. Since the electron charge stored on the FG is directly proportional to the readout frequency count, the programmed cells have a higher count and the erased cells have a lower count. Fig. 5.12 (left) displays the frequency change of the sensor array after the erase and program operations. Yellow blocks represent frequency increase (electron increase), corresponding to the programmed cells, and blue blocks correspond to the erased cells. The cell color becomes darker on the right side of the color map, meaning that with higher CR, the eflash cell can be more efficiently programmed/erased and with higher read transistor size, a similar threshold change can be amplified further. The retention characteristics of the program operation is shown in Fig. 5.12 (right). This figure displays the frequency count over time for 4 cells on the same row with different CR and read transistor size. Note that in this figure, the raw frequency count is plotted without canceling the intrinsic frequency variation between different VCOs, so only the frequency change of different curves over time matters in this figure. Comparing the top two curves, we can see that eflash cells with a higher CR of 0.99 has a larger retention loss. This might due to the eflash cell with a higher CR has more abundance of charges on the FG, so there is also a higher amount of charge loss. Comparing the bottom three curves with the same CR but different device sizing, a larger device size results in a higher retention loss. For a given amount of the charge loss on the FG nodes, which results in a similar threshold change on the read transistors, the larger device will have a larger current change, reflected as a larger frequency count drop. Due to the limited number of chips and the lack of an accurate temperature control setup, we only performed retention tests at room temperature. Considering that the gate oxide thickness of this 0.35μm process is 50% thicker than that in a previous work [73], we expect the cell retention to be adequate at different PVT conditions. We have verified that under normal operating conditions, the sensor readout frequency is relatively constant. Therefore, we can say with high confidence that any significant changes in the output frequency is attributed to a physical attack.

5.4.3 Physical Attack Tests

We performed temperature, humidity and particle/debris attack tests to evaluate the array based sensor. Before the tests, the rows of the sensor array were alternately erased and programmed. The programmed rows are the sensing nodes, and the erased rows serve as the charge collection metals. Fig. 5.13 (top) shows the temperature test results. We used a simple heat gun to raise the temperature of the chip, mimicking an attempt to desolder the package from the printed circuit board. This was repeated three times. The red curve in Fig. 5.13 (top) plots the temperature profile during the test. The sensor frequency was readout between the temperature spikes, and the power supply of the test chip was shut off between the readouts. The test chip was packaged in a ceramic DIP48 package with a removable lid. During the first attack, the lid of the chip was kept closed. This did not result in any appreciable frequency change so for the second and third attacks, we opened the lid for better heat exposure. Note that this was necessary only because of the large DIP package and small test dies used in our experiments. Smaller packages are expected to be significantly more susceptible to temperature attacks due to their low thermal mass. After the heat is removed for the second temperature attack, the readout frequency of the exposed and unexposed cells went down, indicating a charge loss in the FG nodes. After the third temperature attack, the readout frequency remained low. This indicates that high temperature facilitates the FN tunneling and once the electrons have gained enough energy to pass the barrier, later attacks with similar temperature won't cause further frequency change. FN tunneling induced by high temperature affects both exposed and unexposed cells, so as expected, permanent charge loss occurs in both type of cells.

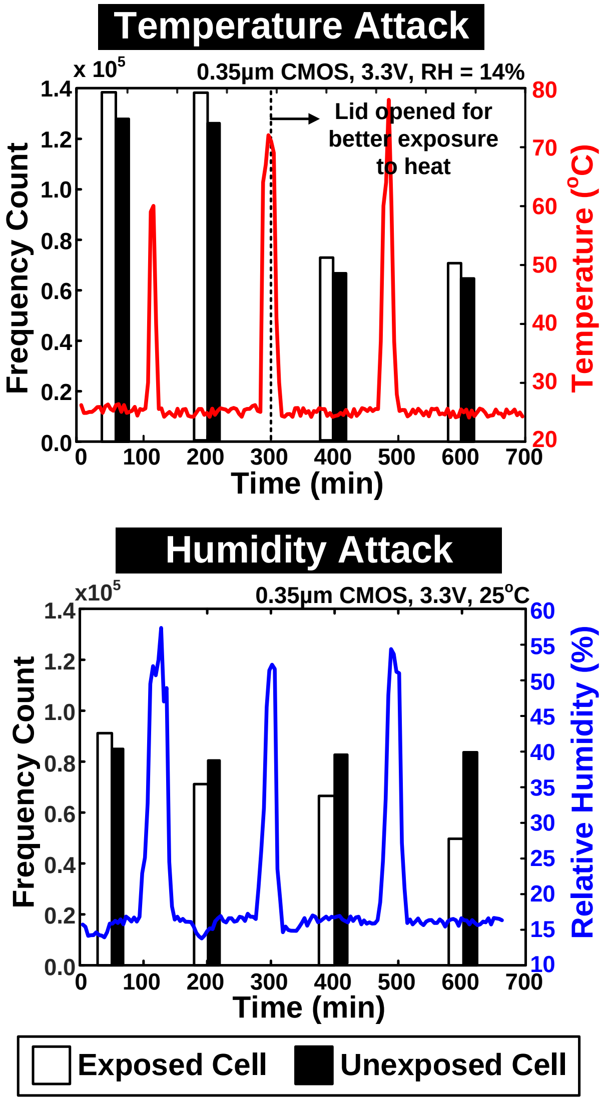


Fig. 5.13: Temperature (top) and humidity (bottom) attack test results.

Fig. 5.13 (bottom) shows humidity attack test results. High humidity was applied three times, which is shown in the blue curve. The black/white bars in Fig. 5.13 (bottom) plot the frequency readout of the exposed and unexposed cells before and after humidity attacks. After each humidity attack, the frequency of only the exposed sensor decreased further, which matches the previous results from the 65nm test structure. The frequency of the unexposed reference sensor remained at the same level after each humidity attack. These measurement results indicate that humidity causes a permanent charge loss only in the exposed sensor. Humidity attack only changes the surface conductivity of the exposed sensors, so we can only observe frequency shift in the exposed sensors. By comparing the frequency change of the exposed sensor and unexposed sensor, we can determine the source of the attack.

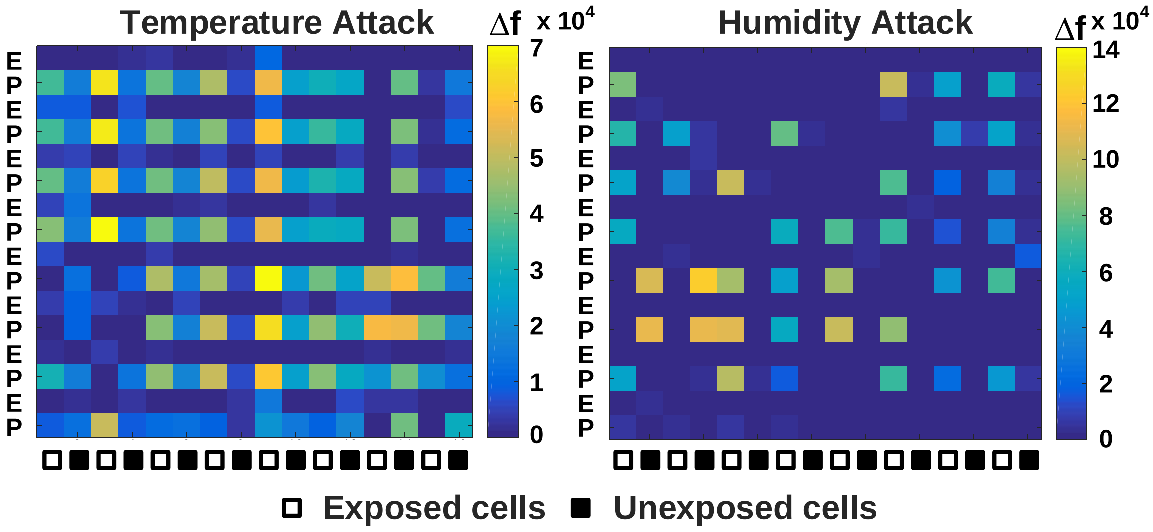


Fig. 5.14: Frequency change maps of temperature attack (left) and humidity attack (right).

By analyzing the entire frequency map data for the sensor array, which is shown in Fig. 5.14, we can extract further information on the type of attack. For instance, temperature attack results in frequency change in both the exposed and unexposed sensors, while the exposed cells have a larger drop due to better exposure to the heat source. This trend can be seen in Fig. 5.14 (left), where the cells on the programmed rows generally have a brighter color, with the exposed cells having the brightest color. Abnormal behavior can be seen in the upper right or lower left regions which underscores the importance of having an array structure rather than a single node sensor. Unlike temperature attack, humidity attack causes a large frequency drop only in the exposed cells, which is illustrated in Fig. 5.14 (right). Most of the non-blue color blocks appear on the odd columns, which represent the frequency change in exposed cells. Some abnormal cells can be found in the bottom left corner. Note that in this frequency change map, the programmed working sensors are only on the even rows, so the frequency change happens only in the even rows.

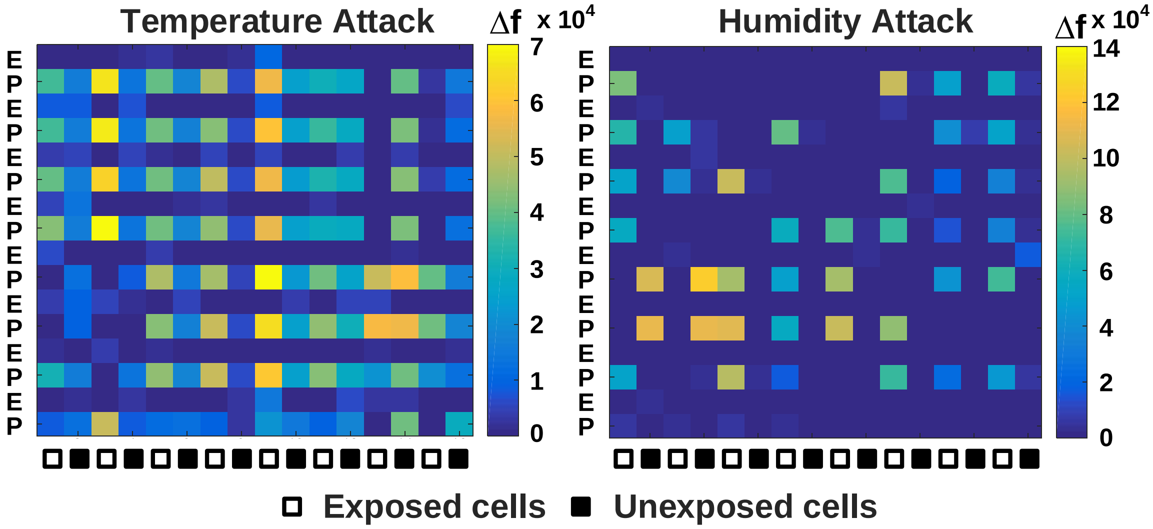


Fig. 5.15: Particle/debris test results. Frequency change map (top) and microscope images of the region that was affected by the particles (bottom).

Finally, particle/debris attacks were performed, which is shown in Fig. 5.15. We introduced some fine particles to the chip cavity intentionally to speed up the test. Rows in the sensor array were programmed and erased in an alternating fashion for better charge collection. During the measurement, the frequency of the sensor cells was continuously recorded. The chip lid was opened and some talc particles were introduced right after 100 minutes. The frequency of some cells dropped almost instantaneously). The locations of the 4 cells showing large frequency drops are highlighted in the red box in Fig. 5.15 (top). The microscope images before and after the particle/debris attack shows the sensor array area inside the red box in the frequency change map. By cross-checking the microscope image with the location of the sensor cells showing a large frequency drop, we found that the particles/debris tend to be attracted to the erased sensor cells, while frequency drops occur in the adjacent programmed cells. This suggests that either the electrons on those programmed cells have been collected by the nearby particles/debris or the parasitic capacitance of the FG changes because of the particles/debris landed on the die. We could also observe that in the frequency change map of particles attack, there are only several nearby exposed cells showing frequency change, illustrated by the yellow blocks, which indicates the possible locations of the particles. This pattern is different from the temperature and humidity tests, so the proposed sensor array can also discriminate the particle/debris effects from other forms of attacks.

5.5 Conclusion

In this work, we presented an eflash based counterfeit IC detection sensor with an exposed FG node. The proposed 5T eflash cell is built using I/O transistors readily available in any logic process, and hence incurs no process overhead. Measurement results from a 65nm sensor test structure and a 0.35µm sensor array test chip validates that the proposed eflash sensor can sense and distinguish between different physical attacks. Any physical source that changes the charge stored on the exposed FG can be detected by this sensor. This includes humidity, high temperature, dust particles, chemicals, and electrostatic charges. Test chip results shows that the proposed eflash based sensor can efficiently and reliably detect many types of counterfeit attempts.

Chapter 6. Summary

In this dissertation, in order to efficiently and securely deploy the AI workloads on edge devices, several hardware architectures and circuit techniques are proposed to improve the energy and area efficiency of the MLP and LSTM neural networks, and enhance the hardware security as well. The performance of the proposed designs are verified with the simulation results, as well as the measurement results from the working test chips fabricated in advanced CMOS technologies.

Chapter 2 introduces a novel time-based computing scheme to implement MLP neural network, which is a much more efficient way of computing multiply and accumulate (MAC) than the traditional methods. The proposed neural network is based on integrate-and-fire neuron model and is featured with brain-inspired leak and local lateral inhibition capabilities, which can be enabled to further improve the classification accuracy. The processing element of the proposed time-based neuromorphic core is based on inverters, which is tiny and compact, and each processing element has its exclusive memory, avoiding the extra power for data movement, thus the proposed neuromorphic core is highly efficient in area and power. The performance of the proposed core is tested with digit recognition application and achieves a 91.4% accuracy.

Another neural network architecture: LSTM neural network is explored in Chapter 3. LSTM neural network is very powerful in processing sequential data and has been proven to be successful in various applications. The high computation complexity and memory requirement of the LSTM neural network make it difficult to be implemented in hardware.

In order to deploy the LSTM based neural networks on embedded devices, a binarized LSTM architecture is proposed and implemented in our work. The binarized LSTM architecture greatly simplifies the MAC computation complexity and reduces the memory requirement for storing the weights. A pipelined architecture is proposed to reduce the total computation time of the multi-layer neural network system. The performance of the proposed LSTM neural network is verified with the application of heart rate estimation from PPG signals.

Chapter 4 discusses a data remanence technique to generate 100% stable keys from an SRAM PUF to enhance the hardware security. The power-up state of a SRAM array is a very attractive option for weak PUFs, but it is not always stable. The proposed data remanence technique can efficiently select the stable SRAM cells in a large SRAM array, and the PUF response is generated with only the stable cells, thus the stability of the PUF response is guaranteed. The efficacy of the proposed technique is demonstrated with the measurement results from the off-the-shelf SRAM chips. The PUF response generated using the proposed technique is 100% stable under various temperature, voltage ramp up rate and aging conditions.

Finally chapter 5 illustrates a passive IC tamper sensor design based on an exposed floating gate device in standard logic processes. This proposed tamper IC is based on a non-volatile eflash memory cell, and can record the event history powerlessly, which can be used to detect and record suspicious tampering attacks to the chip. The performance of the proposed sensor is tested with physical tamper attacks, such as high temperature cycling, humidity rises, and increased dust particle density in the chip cavity. The measurement results prove that the proposed sensor is able to detect many type of counterfeit attempts efficiently and reliably.

Bibliography

[1] D. Silver, *et al*., “Mastering the game of Go with deep neural networks and tree search,” *Nature*, vol. 529, no. 7587, pp. 484–489, Jan. 2016.

[2] A. Krizhevsky, I. Sutskever, and G. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Neural Information Processing Systems*, 2012.

[3] K. He, *et al*. (2016), “Identity mappings in Deep Residual Networks,” [Online]. Available: https://arxiv.org/abs/1603.05027.

[4] B. Gassend, D. Clarke, M. van Dijk and S. Devadas, “Silicon Physical Random Functions,” in *Proceedings of the Computer and Communications Security Conference*, Nov. 2002.

[5] M. Courbariaux, I. Hubara, D. Soudry, R. El-Yaniv and Y. Bengio, “Binarized Neural Networks: Training Neural Networks with Weights and Activations Constrained to +1 or -1,” [Online]. Available: https://arxiv.org/abs/1602.02830.

[6] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv and Y. Bengio, “Binarized neural networks,” *Proceedings of Advances Neural Information Processing Systems*, vol. 29, pp. 4107–4115, 2016.

[7] Y. Bengio, “Learning deep architectures for AI”, *Foundations and Trends in Machine Learning*, vol. 2, no. 1, pp. 1-127, 2009.

[8] W. S. McCulloch and W. Pitts, “A logical calculus of the ideas immanent in nervous activity,” *The bulletin of mathematical biophysics*, vol. 5, no. 4, pp. 115-133, Dec. 1943.

[9] N. P. Jouppi, *et al*. (2017), “In-datacenter performance analysis of a Tensor Processing Unit^{TM},” [Online]. Available: https://arxiv.org/abs/1704.04760.

[10] C. Mead, “Neuromorphic Electronic Systems,” *Proc. IEEE*, vol. 78, no. 10, pp. 1629-1636, Oct. 1990.

[11] P. Merolla, *et al*., “A Million Spiking-Neuron Integrated Circuit with a Scalable Communication Network and Interface,” *Science*, vol. 345, no. 6197, pp. 668-673, Aug. 2014.

[12] S. Park, *et al*., “A 1.93TOPS/W scalable deep learning/inference processor with tetraparallel MIMD architecture for big-data applications,” in *Proc. IEEE Int. Solid-State Circuits Conf.* (ISSCC), pp. 1–3, Feb. 2015.

[13] L. Cavigelli, *et al*., “Origami: A convolutional network accelerator,” in *Proc. 25th Ed. Great Lakes Symp. VLSI*, pp. 199–204, 2015.

[14] J. Sim, *et al*., “A 1.42TOPS/W deep convolutional neural network recognition processor for intelligent IoE systems,” in *Proc. IEEE Int. Solid-State Circuits Conf.* (ISSCC), pp. 264–265, Feb. 2016.

[15] Y.-H. Chen, T. Krishna, J. Emer, and V. Sze, “Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks,” in *Proc. IEEE Int. Solid-State Circuits Conf.* (ISSCC), pp. 262–263, Feb. 2016.

[16] Y.-H. Chen, T. Krishna, J. S. Emer, and V. Sze, “Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks,” *IEEE J. Solid-State Circuits*, vol. 49, no. 1, pp. 73–83, Jan. 2017.

[17] G. Desoli, *et al*., "A 2.9TOPS/W deep convolutional neural network SoC in FD-SOI 28nm for intelligent embedded systems," in *Proc. IEEE Int. Solid-State Circuits Conf.* (ISSCC), pp. 238–239, Feb. 2017.

[18] K. Bong, *et al*., "A 0.62mW ultra-low-power convolutional-neural-network face-recognition processor and a CIS integrated with always-on Haar-like face detector," in *Proc. IEEE Int. Solid-State Circuits Conf.* (ISSCC), pp. 248–249, Feb. 2017.

[19] R. Hameed *et al*., "Understanding sources of inefficiency in general purpose chips," in *Proc. 37th Annu. Int. Symp. Comput. Archit.*, pp. 37–47, 2010.

[20] M. Horowitz, "Computing's energy problem (and what we can do about it)," in *IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers* (ISSCC), pp. 10–14, Feb. 2014.

[21] A. K. Jain and J. Mao, "Artificial neural networks: A tutorial," in *IEEE Comput. Mag*., pp. 31–44, Mar. 1996.

[22] B. Karlik, and A. V. Olgac, "Performance analysis of various activation functions in generalized MLP architectures of neural networks," *Int. J. Artif. Intell. Expert Syst.* (IJAE), vol. 1, no. 4, pp. 111-122, 2011.

[23] Y.A LeCun, L. Bottou, G.B. Orr, and K.-R. Muller. "Efficient backprop," in *Neural networks: Tricks of the trade*. Springer Berlin Heidelberg, pp. 9-48, 2012.

[24] L. Abbott *et al*., "Lapicque's introduction of the integrate-and-fire model neuron (1907)," *Brain Research Bulletin*, vol. 50, no. 5, pp. 303-304, 1999.

[25] Gerstner and Kistler, "Formal spiking neuron models," in *Spiking Neuron Models, Single Neurons, Populations, Plasticity*, Cambridge University Press, 2002, ch. 4, sec. 1.1.

[26] M. Liu, L. Everson, and C. H. Kim, “A scalable time-based integrated-and-fire neuromorphic core with brain-inspired leak and local lateral inhibition capabilities,” in *Proc. IEEE Custom Integr. Circuits Conf.*, pp. 1-4, May 2017.

[27] R. Sarpeshkar, “Analog versus digital: Extrapolating from electronics to neurobiology,” *Neural Comput.*, vol. 10, no. 7, pp. 1601–1638, 1998.

[28] D. Miyashita *et al.*, “An LDPC decoder with time-domain analog and digital mixed-signal processing,” *IEEE J. Solid-State Circuits*, vol. 49, no. 1, pp. 73–83, Jan. 2014.

[29] D. Miyashita, S. Kousai, T. Suzuki, and J. Deguchi, “Time-domain neural network: A 48.5 TSOp/s/W neuromorphic chip optimized for deep learning and CMOS technology,” in *IEEE Asian J. Solid-State Circuits*, Nov. 2016, pp. 25-28.

[30] D. Miyashita, S. Kousai, T. Suzuki, and J. Deguchi, “Neuromorphic chip optimized for deep learning and CMOS technology with time-domain analog and digital mixed-signal processing,” *IEEE J. Solid-State Circuits*, vol. 52, Issue 10, Oct. 2017.

[31] K. Ando, *et al.*, “BRein memory: A 13-layer 4.2 K neuron/0.8M synapse binary/ternary reconfigurable in-memory deep neural network accelerator in 65nm CMOS,” in *Proc. IEEE Symp. VLSI Circuits*, pp. 24–25, Jun. 2017.

[32] Y. A. LeCun, *et al.*, “The MNIST Database of Handwritten Digits.”(1998).

[33] K. J. Lee, *et al.*, “A 502GOPS and 0.984mW dual-mode ADAS SoC with RNN-FIS engine for intention prediction in automotive black-box system,” in *IEEE Int. Solid-State Circuits Conf.* (*ISSCC*), pp. 256-257, Feb. 2016.

[34] J. K. Kim, P. Knag, T. Chen, and Z. Zhang, “A 640M pixel/s 3.65mW sparse event-driven neuromorphic object recognition processor with on-chip learning,” in *Proc. IEEE Symp. VLSI Circuits*, pp. 50–51, Jun. 2015.

[35] T. Young, D. Hazarika, S. Poria, and E. Cambria, “Recent Trends in Deep Learning Based Natural Language Processing,” [Online]. Available: https://arxiv.org/abs/1708.02709, Nov. 2018.

[36] D. Bahdanau, K. Cho, and Y. Bengio, “Neural Machine Translation by Jointly Learning to Align and Translate,” [Online]. Available: https://arxiv.org/abs/1409.0473, May, 2016.

[37] L. Everson, *et al*., “BiometricNet: Deep Learning based Biometric Identification using Wrist-Worn PPG,” in *IEEE Int. Symp. On Circuits and Systems (ISCAS)*, May, 2018.

[38] F. A. Gers, J. Schmidhuber, and F. Cummins “Learning to Forget: Continual Prediction with LSTM,” *Neural Computation*, vol. 12, no. 10, pp. 2451-2471.

[39] L. Hou, Q. Yao and J. T. Kwok, “Loss-aware Binarization of Deep Networks,” [Online]. Available: https://arxiv.org/abs/1611.01600, Nov. 2016.

[40] K. Shelley and S. Shelley, “Pulse Oximeter Waveform: Photoelectric Plethysmorgraphy,” in *Clinical Monitoring*, Carol Lake, R. Hines, and C. Blitt, Eds.: W. B. Saunders Company, pp.420-428, 2001.

[41] D. Biswas, *et al*., “CorNET: Deep Learning Framework for PPG based Heart Rate Estimation and Biometric Identification in Ambulant Environment,” in *IEEE.Trans. on Biomedical Circuits and Systems*, vol. 13, no. 2, pp. 282-291, Apr. 2019.

[42] Z. Zhang, *et al*., " TROIKA: A General Framework for heart rate monitoring using wrist-type photoplethysmorgraphic signals during intensive physical exercise," *IEEE Trans. Biomed. Eng*., vol. 62, Issue 2, pp. 522-531, 2015.

[43] M. Mashhadi, *et al*., "Heart Rate Tracking using Wrist-type Photoplethysmorgraphic (PPG) Signals during Physical Exercise with Simultaneous Accelerometry," in *IEEE Signal Processing Letters*, v. 23, 2016.

[44] B. Lee, *et al*., "Improved Elimination of Motion Artifacts from a Photoplethysmorgraphic Signal using a Kalman Smoother with Simultaneous Accelerometry," *Physiol Meas*., vol. 31, no. 12, pp. 1585-1603, 2010.

[45] A. Temko, "Accurate Heart Rate Monitoring during Physical Excercises using PPG," *IEEE Trans. Biomed. Eng*., vol. 64, no. 9, pp. 2016-2024, 2017.

[46] E. Grisan, *et al*., "A Supervised Learning Approach for the Robust Detection of Heart Beat in Plethysmographic Data," in *Engineering in Medicine and Biology Society (EMBC), 2015 37th Annual International Conference of the IEEE*, pp. 5825-5828, 2015.

[47] M. Essalat, *et al*., "Supervised Heart Rate Tracking using Wrist-type Photoplethysmographic (PPG) Signals during Physical Exercise without Simultaneous Acceleration Signals," in *Signal and Information Processing (GlobalSIP), 2016 Global Conference on*, pp. 1166-1170, 2016.

[48] J. E. Volder, "The CORDIC Trigonometric Computing Technique," *IRE Trans. On Electronic Computers*, vol. EC-8, pp. 330-334, Sept. 1959.

[49] S. Devadas, E. Suh, S. Paral, *et al.*, "Design and Implementation of PUF-Based 'Unclonable' RFID ICs for Anti-counterfeiting and Security Applications," in *Proceedings of IEEE International Conference on RFID*, May 2008, pp. 58-64.

[50] A. Maiti, J. Casarona, L. McHale, *et al.*, "A Large Scale Characterization of RO-PUF", *IEEE International Symposium on Hardware-Oriented Security and Trust*, 2010, pp. 94-99.

[51] C. Herder, M. Yu, F. Koushanfar, and S. Devadas, "Physical Unclonable Functions and Applications: A Tutorial," *Proceedings of the IEEE*, vol. 102, no. 8, pp. 1126-1141, 2014.

[52] D. Holcomb, W. Burleson, and K. Fu, "Power-up SRAM State as an Identifying Fingerprint and Source of True Random Numbers." *IEEE Transactions on Computers*, vol. 58, no. 9, pp. 1198-1210, Sep. 2009.

[53] S. K. Mathew, S. K. Satpathy, M. A. Anders, *et al.*, "A 0.19pJ/b PVT-variant-tolerant Hybrid Physically Unclonable Function Circuit for 100% Stable Secure Key Generation in 22nm CMOS," in *IEEE International Solid State Circuits Conference* (ISSCC), Feb. 2014, pp. 278-279.

[54] C. Zhou, S. Satapathy, Y. Lao, *et al*, "Soft Response Generation and Thresholding Strategies for Linear and Feed-forward MUX PUFs," in *International Symposium on Low Power Electronics and Design*, Aug., 2016.

[55] K. Xiao, M. Rahman, D. Forte, *et al.*, "Bit Selection Algorithm Suitable for High-Volume Production of SRAM-PUF," in *IEEE International Symposium on Hardware-Oriented Security and Trust*, 2014, pp. 101-106.

[56] Wikipedia, https://en.wikipedia.org/wiki/Remanence.

[57] Y. Oren, A. Sadeghi, and C. Wachsmann, "On the Effectiveness of the Remanence Decay Side-channel to Clone Memory-based PUFs," in *Cryptographic Hardware and Embedded Systems*, vol. 8086, pp. 107-125, 2013.

[58] R. Maes and V. van der Leest, “Countering the Effects of Silicon Aging on SRAM PUFs,” *IEEE International Symposium on Hardware-Oriented Security and Trust*, 2014, pp. 148-153.

[59] A. Jain, A. Paul, and C. H. Kim, “A 32nm SRAM Reliability Macro for Recovery Free Evaluation of NBTI and PBTI,” *International Electron Devices Meeting*, Dec. 2012.

[60] T. Kim, W. Zhang, and C. H. Kim, “An SRAM Reliability Test Macro for Fully-automated Statistical Measurements for Vmin degradation,” *Custom Integrated Circuits Conference*, Sep. 2009.

[61] D. K. Schroder and J. A. Babcock, “Negative Bias Temperature Instability: Road to Cross in Deep Submicron Silicon Semiconductor Manufacturing,” in *Journal of Applied Physics*, vol. 94, pp. 1-18, Jul. 2003.

[62] Internet of Things Global Standards Initiative [Online]. Available: https://www.itu.int/en/ITU-T/gsi/iot/Pages/default.aspx

[63] A. Nordrum (2016, Augst 18). Popular Internet of Things Forecast of 50 Billion Devices by 2020 is Outdated [Online]. Available: https://spectrum.ieee.org/tech-talk/telecom/internet/popular-internet-of-things-forecast-of-50-billion-devices-by-2020-is-outdated

[64] G. Fink, D. V. Zarzhistsky, T. E. Carroll and E. D. Farquhar, “Security and Privacy Grand Challenges for the Internet of Things,” in *Int. Conf. Collaboration Technologies and Systems* (CTS), pp. 27-34, Jun. 2015.

[65] M. Pecht and S. Tiku, “Bogus: Electronic Manufacturing and Consumers Confront a Rising Tide of Counterfeit Electronics,” *IEEE Spectrum*, vol. 43, no. 5, pp. 37-46, May 2006.

[66] U. Guin and M Tehranipoor, “On Selection of Counterfeit IC Detection Methods,” in *Proc. IEEE North Atlantic Test Workshop*, pp. 1-5, May 2013.

[67] L. W. Kessler and T. Sharpe (2010), Faked Parts Detection [Online]. Available: http://publish-it-online.com/article/Faked+Parts+Detection/411055/39826/article.html

[68] U. Guin, K. Huang, D. DiMase, J. M Carulli, M. Tehranipoor and Y. Makris, “Counterfeit Integrated Circuits: A Rising Threat in the Global Semiconductor Supply Chain,” *Proc. IEEE*, vol. 102, no. 8, pp. 1207-1228, Aug. 2014.

[69] S. Shahbazmohamadi, D. Forte and M. Tehranipoor, “Advanced Physical Inspection Methods for Counterfeit IC Detection,” in *40th Int. Symp. For Testing and Failure Analysis*, pp. 55-64, Nov. 2014.

[70] T, Kim, R. Persaud and C. H. Kim, “Silicon Odometer: An On-chip Reliability Monitor for Measuring Frequency Degradation of Digital Circuits,” *IEEE J. Solid-State Circuits*, vol. 43, no. 4, pp. 874-880, Apr. 2008.

[71] X. Zhang, N. Tuzzio and M. Tehranipoor, “Identification of Recovered ICs using Fingerprints from a Light-weight On-chip Sensor,” in *Proc. IEEE Design Autom. Conf.*, pp. 703-708, Jun. 2012.

[72] M. Liu and C. H. Kim, “A Powerless and Non-volatile Counterfeit IC Detection Sensor in a Standard Logic Process based on an Exposed Floating-gate Array,” in *IEEE Symp. On VLSI Technology*, pp. 102-103, Jun. 2017.

[73] S. Song, K. Chun and C. H. Kim, “A Logic-compatible Embedded Flash Memory for Zero-standby Power System-on-chips Featuring a Multi-story High Voltage Switch and a Selective Refresh Scheme,” *IEEE J. Solid-State Circuits*, vol. 48, no. 5, pp. 1302-1314, May 2013.

[74] S. Song, J. Kim and C. H. Kim, “A Comparative Study of Single-poly Embedded Flash Memory Disturbance, Program/Erase Speed, Endurance, and Retention Characteristic,” *IEEE Trans. Electron Devices*, vol. 61, no. 11, pp. 3737-3743, Nov. 2014.