8-2023

Reclaiming Fault Resilience and Energy Efficiency With Enhanced Performance in Low Power Architectures

Noel Daniel Gundi
Utah State University

RECLAIMING FAULT RESILIENCE AND ENERGY EFFICIENCY WITH ENHANCED PERFORMANCE IN LOW POWER ARCHITECTURES

by

Noel Daniel Gundi

A dissertation submitted in partial fulfillment
of the requirements for the degree

of

DOCTOR OF PHILOSOPHY

in

Electrical Engineering

Approved:

Sanghamitra Roy, Ph.D. Major Professor	Koushik Chakraborty, Ph.D. Committee Member
Greg Droge, Ph.D. Committee Member	Zhen Zhang, Ph.D. Committee Member
Vicki H Allan, Ph.D. Committee Member	D. Richard Cutler, Ph.D. Vice Provost of Graduate Studies

UTAH STATE UNIVERSITY
Logan, Utah

2023

ABSTRACT

Reclaiming Fault Resilience and Energy Efficiency with Enhanced Performance in Low Power Architectures

by

Noel Daniel Gundi, Doctor of Philosophy

Utah State University, 2023

Major Professor: Sanghamitra Roy, Ph.D.
Department: Electrical and Computer Engineering

Shrinking technology node and the massive increase in data workloads has witnessed a swift migration of the system towards the Low-Power Computing (LPC) paradigm. Additionally, to accelerate the redundant yet mammoth AI instructions, novel ASIC design architectures have been explored. Google's Tensor Processing Unit (TPU) is one such architectural innovation deployed in the commercial space to speedup the processing of AI workloads.

In an effort to achieve a superior energy efficiency, Near-Threshold Computing (NTC) has been marginalized to be an efficient LPC paradigm. Due to an underscaling of voltage, NTC offers quadratic savings is power consumption in comparison to operating the system at its nominal counterpart i.e., Super-Threshold Computing (STC). However, NTC exhibits an extreme sensitivity to Process Variation (PV). Moreover, the reduced speed of transistors at NTC exacerbates the overall performance of the system. Hence, the integration of NTC into the conventional semiconductor workspace has been restricted. In this work, distinct methodologies are explored to provide improved performance at NTC. Furthermore, effects of PV, which are unnoticed at STC but posing a severe threat to the reliability of the low-power AI computing is addressed. This dissertation exploits the disparate

computational delays of arithmetic units to provide up to 2.5× improved performance and 1.35× better energy efficiency at NTC. Additionally, the distinct dataflow patterns of the TPU are statistically analyzed to employ selective voltage levels and further enhance the performance of the TPU. Also, the homogeneous architecture of the TPU systolic array is thoroughly investigated to design a low-overhead faulty Processing Element (PE) detection scheme. The locality of the faulty PE is later utilized to tackle the impending faults.

(105 pages)

PUBLIC ABSTRACT

Reclaiming Fault Resilience and Energy Efficiency with Enhanced Performance in Low Power Architectures

Noel Daniel Gundi

Rapid developments of the AI domain has revolutionized the computing industry by the introduction of state-of-art AI architectures. This growth is also accompanied by a massive increase in the power consumption. Near-Theshold Computing (NTC) has emerged as a viable solution by offering significant savings in power consumption paving the way for an energy efficient design paradigm. However, these benefits are accompanied by a deterioration in performance due to the severe process variation and slower transistor switching at Near-Threshold operation. These problems severely restrict the usage of Near-Threshold operation in commercial applications. In this work, a novel AI architecture, Tensor Processing Unit, operating at NTC is thoroughly investigated to tackle the issues hindering system performance. Research problems are demonstrated in a scientific manner and unique opportunities are explored to propose novel design methodologies.

To the Almighty God, Mummy, Daddy, friends and all my loved ones whom I have lost in the past years.

ACKNOWLEDGMENTS

First and foremost, I want to thank GOD — The LORD Almighty, for giving me a wonderful life and caring for me all the while.

I would like to offer my sincere heartfelt gratitude to everyone who have helped and guided me in this Ph.D. journey. I would like to thank my major advisor Dr. Sanghamitra Roy and co-advisor Dr. Koushik Chakraborty for giving me the opportunity to join the USU Bridge Lab. I thank them for consistently encouraging and guiding me with their valuable insight, constant encouragement, and helping me to give my full effort and strive towards a meaningful research. Additionally, thank you to my advisors for all their constructive criticism and enabling me to envision the future possibilities in a research. Also, I want express my gratitude to them for always hosting me with hospitality and ensuring I am not alone in Logan. I want thank my Ph.D. committee members Dr. Greg Droge, Dr. Zhen Zhang and Dr. Vicki Allan for all their valuable feedback on my research and motivating presence. I would like to thank Tricia Brandenburg, Diane Buist and Kathy Phippen for assisting me with all documentation work and always willing to address my doubts. I also appreciate and thank Brady for always helping me out with the lab and computer work. A special thank you to the ECE Department for all the support.

I am very thankful and extremely delighted to work with all the members of the BRIDGE lab. I want to thank Tahmoures for all the intellectual insight bestowed on me during the initial days. Pramesh and Prabal, for being wonderful research partners, correcting and encouraging me every time. Sourav, for instilling in me the confidence when I was down. Chidham, for the calm and intellectual demeanor. Aatreyi, for being the strict class teacher/sister. Rajesh for introducing me to the world of schematic drawing. I also want to thank all my labmates with whom all I have associated with, Heidi, Mitchell, Jyotirmoy, Bishal, Zinnia, Andrew C., Bijoy, Andrew G., Mason and Tim for making my journey memorable.

Thanks to all my colleagues at Tata Elxsi, Nagesh, Aditya, Raghu, Prakash, Ganesh, Abhijit, Teena and Rohit. Thanks again to Rohit, Hitesh, YK, Nevin, Mamta and Anshuman from AMD for guiding me during my internship. Special thanks to Dr. Acken, Dr. Johnson, Dr. Samarakoon and Dr. Stine from my days at Oklahoma State University.

I want to thank all my friends Surya, Spandan, Pradeep, Rohit, Rakshit, Theja, Libin, Naveen, Santosh J., Avinash, Robin, Swithin, Praveen, Prashant, Pavan, Makutum, Shiva, Utpal, Santosh M., Varun, Ricky, Vivian, Eldho, Ryan, Chris, Isaac, Krishnaprasad, Anirudh, Rupesh, Dilip, Rajashekar, Sujit and Steve for being there for me. Additionally, thanks to all the friends I gained during my Ph.D. journey, Avik, Vedang, Jatin, Akshat, Ashu, Vaibhav, Supratik, Saju, Raushan, Rejoy, Supriyo and Colby. Special thanks to my little friend Deep for being a great geography partner. Big thanks to my church family, Matthew, Joseph, Derek, Jonathan S., Thomas, Jonathan B., Pastor Dane, Jason, Drew, Dan, Jordan, Benny, Jim and Ann, Ibukun, Bolaji, Olusola, Yessica, Alban, Preeti, Susan, entire EU, GCCF, ACF, The Shofar Call and Navigators. Special thanks to Dr. McGee, Dr. Bond, Hy, Don, Joel and the Oklahoma City bible study group for always remembering me in prayers. Thanks to brothers Varghese, Oberon, uncles Anthony, Ernst, Prakash, Naveen, Joaquim, Robert, K.G. Samuel, aunty Lalitha, Gay, sister Autumn and their respective families. Thanks again to Pastor Zac. Thank you again to sister Lakshmi and brother Peter for always hosting me. Thank you to brothers Prem, Shekhar, Daniel K., sister Ranjini and Pastors Saji, Stephen, Molly and Joy.

I thank my parents Kamala and Madhukar for their love and unconditional support. Thank you to my brothers and sisters, Jenet, Chandrakala, Pradeep, Steven, Sabina, Mamta, Solomon, Vishal, Simon, Archana, Shirlin, Sherlock, Praveen, Ullas, Anet, Alvita and Adrian. Thanks again to my uncles and aunts Sulochana, Pamela, Sara, Vanita, Vijaykumar H., Janet, David, Lata and Shaila. Remembering all my loved ones, grandparents Simon, Rosa, John, Emily, aunts Hilda, Margaret, Sundra and Advin, uncles Sadashiv, Suresh, Vijaykumar and Stanley, sister Seema and brothers Daniel, Naveen and Sam.

Thank you everyone for making my journey memorable.

Noel Daniel Gundi

CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACRONYMS

VLSI	Very Large Scale Integration
LPC	Low-Power Computing
TPU	Tensor Processing Unit
CPU	Control Processing Unit
GPU	Graphics Processing Unit
AI	Artificial Intelligence
ASIC	Application Specific Integrated Circuit
DNN	Deep Neural Network
PE	Processing Element
NTC	Near-Threshold Computing
NTV	Near-Threshold Voltage
STC	Super-Threshold Computing
FPGA	Field Programmable Gate Array
RTL	Register Transfer Level
PV	Process Variation
MAC	Multiplier-and-Accumulate
CDF	Cumulative Distribution Function
CostCo	Costless Correction
MUX	Multiplexer
EXOR	Exclusive-OR
SIPO	Serial Input Parallel Output
SPICE	Simulation Program with Integrated Circuit Emphasis
FinFET	Fin Field-Effect Transistor
MSB	Most Significant Bit
LSB	Least Significant Bit
STA	Static Timing Analysis

TOPS	Tera Operations Per Second
DVS	Dynamic Voltage Scaling
DVFS	Dynamic Voltage and Frequency Scaling
TB	Time-Borrow
EMU	Error Management Unit
ECU	Error Collection Unit
VCU	Voltage Control Unit
MRFF	Modified Razor Flip Flop
RFF	Razor Flip Flop
TEC	Total Error Count
CU	Control Unit
BU	Boost Unit
CF	Curve Factor
BR	Boost Register
PTM	Predictive Technology Model
BE	Boost Energy
FO4	Fan-Out-of-Four
LP-fault	Low-Power Fault
FCU	Fault Control Unit
FHop	Fault Hop
FHop-TB	Fault Hop Time-Borrow
FDR	Fault Detection Register
SRAM	Static Random Access Memory
DRAM	Dynamic Random Access Memory

CHAPTER 1

INTRODUCTION

Deployment of Artificial Intelligence (AI) into various spheres of the daily life and the massive developments in the domain-specific architectures has witnessed a remarkable growth in the usage of Deep Neural Networks (DNNs) in the computing ecosystem. Additionally, the utilization of AI models has seen a significant rise in multiple domains ranging from infotainment, academia and biomedical applications. However, the proliferation of AI workloads has also been accompanied by a larger carbon footprint [1]. Furthermore, the advent of Edge Computing has further pushed the energy efficiency limits by narrowing the thermal budget. Hence, the adoption of Low-Power Computing (LPC) into the computing paradigm has been inevitable.

For semiconductor devices, dynamic power in the is linearly dependent on the signal switching and frequency, and quadratically dependent on the operating supply voltage. Additionally, static power is dependent on the leakage current. Therefore, lowering of voltage closer the device's threshold region can aid in lowering power consumption. Hence, to facilitate LPC, Near-Threshold Computing (NTC) can be a singled out as viable solution to cater the system power restrictions. In NTC, the device is operated at a voltage slightly above the device threshold voltage. While NTC offers quadratic benefits in energy efficiency, it is accompanied by noticeable degradation in performance. Computational delays of electronic circuits are inversely proportional to the operating voltage. Hence, a lower operating voltage significantly increases the switching delays due to which the clock frequency is appropriately scaled to meet the timing requirements of computation. Furthermore, NTC is extremely sensitive to Process Variation (PV) due to which faults which completely concealed during nominal operation become massive bottlenecks at NTC. To overcome these performance and reliability concerns, enhancements in the circuit-architectural level are of utmost importance. Moreover, these innovations will aid

in the migration of LPC into the computing paradigm.

In this dissertation, two bodies of work propose enhancements in timing error resilience and energy efficiency by uncovering distinct component delay profiles and workflow. Chapter 3 provides an in-depth investigation on the unequal delay characteristics of the arithmetic units, the unique systolic dataflow pattern and provides orders of improvements in performance. Chapter 4 further exploits the unique dataflow and uses a mathematical approach to tackle the errors at higher frequency points. The third body of work presented in Chapter 5, addresses the hidden threat due to PV which manifests itself at NTC and demonstrates a innovative solution to marginalize the fault locally and tackle it using minor additions in the processing unit.

Additionally, Chapter 2 provides the literature survey of the various research efforts relevant to this dissertation. Chapter 6 summarizes all the works demonstrated through Chapter 3 to Chapter 5 and concludes this dissertation. Section 1.1 presents all the conference and journal publications stemming from this dissertation and serve as formal contributions to the academic field.

1.1 Contributions of This Dissertation

The works presented in this dissertation have been published in several conference proceedings and journal articles, including 2020 IEEE Asia and South Pacific Design Automation Conference (ASPDAC), 2023 IEEE Design Automation Conference (DAC), 2020 and 2022 Journal of Low Power Electronics and Applications (JLPEA) and 2021 IEEE Transactions on Very Large Scale Integration (TVLSI).

1.1.1 Conference Papers

- STRIVE: Enabling Choke Point Detection and Timing Error Resilience in a Low-Power Tensor Processing Unit, Noel Daniel Gundi, Zinnia Muntaha Mowri, Andrew Chamberlin, Sanghamitra Roy and Koushik Chakraborty, Accepted for publication in *IEEE/ACM Design Automation Conference (DAC)*, 2023.

- EFFORT: Enhancing energy efficiency and error resilience of a near-threshold tensor processing unit, Noel Daniel Gundi, Tahmoures Shabanian, Prabal Basu, Pramesh Pandey, Sanghamitra Roy, Koushik Chakraborty and Zhen Zhang, *IEEE Asia and South Pacific Design Automation Conference (ASPDAC)*, 2020.

1.1.2 Journal Articles

- Implementing a Timing Error-Resilient and Energy-Efficient Near-Threshold Hardware Accelerator for Deep Neural Network Inference, Noel Daniel Gundi, Pramesh Pandey, Sanghamitra Roy and Koushik Chakraborty, *Journal of Low Power Electronics and Applications (JLPEA)*, vol. 12, no. 2, p. 32, 2022.

- EFFORT: A comprehensive technique to tackle timing violations and improve energy efficiency of near-threshold tensor processing units, Noel Daniel Gundi, Tahmoures Shabanian, Prabal Basu, Pramesh Pandey, Sanghamitra Roy and Koushik Chakraborty, *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 29, no. 10, pp. 1790–1799, 2021.

- Challenges and Opportunities in Near-Threshold DNN Accelerators around Timing Errors, Pramesh Pandey, Noel Daniel Gundi, Prabal Basu, Tahmoures Shabanian, Mitchell Patrick, Koushik Chakraborty and Sanghamitra Roy, *Journal of Low Power Electronics and Applications*, 10(4), 33, 2020.

CHAPTER 2

LITERATURE REVIEW

This chapter presents the comprehensive literature survey on the research related to the works in this dissertation. The include the contemporary research on NTC, low-power computing, schemes to increase energy efficiency and error resilience in DNN accelerators, and works to reduce the impact of faulty elements in low-power computing. Consequently, the works listed in the chapter demonstrates the research efforts in embracing the low-power realm and the corresponding methodologies facilitating the adoption of this migration. Section 2.1 present works adopting the benefits and challenges in NTC. Section 2.2 addresses the research pertaining to enhancement techniques centered on improving the performance and energy efficiency of for AI architectures. Section 2.3 discusses the works on the impact of faults on various PEs and the improvement schemes to reduce the effect of such faults.

2.1 Exploring the works at NTC :

Although the benefits of utilizing NTC as a prominent LPC paradigm are massive, the vulnerabilities accompanying NTC are quite noticeable. Section 2.1.1 present the works highlighting the evident benefits provided by the NTC paradigm. Section 2.1.2 addresses the research efforts in facilitating the mitigation of the problems plaguing the NTC realm.

2.1.1 Opportunities at NTC

- **Dreslinski et al. [2]:** This work introduces and proposes low-voltage operation as solution to the problem of power consumption and the deployment of NTC across all computing domains. They demonstrate 10× or higher energy efficiency due to the employment of NTC. The authors also highlight a 5× increase in performance variation and an increase in memory failure rates. However, adoption of enhancements

at the circuit architecture levels and other areas can aid in overcoming the barrier of utilization of NTC for all the computing platforms. This work provides a in-depth analysis of the impact of NTC on the various device parameters.

- **Markovic et al. [3]:** This work develops a energy-delay modeling framework to study the various inversion regions and analyze the effect of operating voltage and transistor sizing on the overall performance of the system, while operating in a subthreshold region. The authors propose the utilization of a pass-transistor based logic for operating in subthreshold region. The authors indicate the usage of time-multiplexing in leakage dominated designs to facilitate lower energy consumption.

- **Friedberg et al. [4]:** This work tackles the negative effect on circuit performance due the variations introduced during manufacturing process. The authors propose to lower the variations in device attributes by applying a newer approach on the process control. They also propose to adopt simulation based approaches to reduce the PV sensitivity for circuits.

- **Stine et al. [5]:** This work presents modeling techniques to analyze the contributions at distinct levels of manufacturing towards spatial variation. They propose filtering, spine and regression schemes for wafer-level estimations, Fourier transformations for die-level estimations and spline and frequency oriented methods for wafer-die estimations.

- **Karpuzcu et al. [6]:** This work elaborate the impacts of PV at NTC using an architecture model. The authors present the drawbacks of employing the enhancement techniques for STC in tackling the PV at NTC. They also present a roadmap to be followed to design efficient techniques to overcome the impacts of PV at NTC.

2.1.2 Challenges at NTC

- **Borkar et al. [7]:** This work proposes to tackle the problem PV using design and CAD based approach. The authors emphasize on the usage of design flows incorporating

probabilistic and statistical based tools. They also elaborate a change from continuous to discreet due to the adoption of stricter regulation for device parameters such as interconnect pitches, spacings, lengths etc.

- **Karpuzcu et al. [8]:** This work introduces a microarchitectural model of PV at NTC. The model elaborates the impact of variations on the operating frequency, power usage by the processing cores, memory types in NTC and various faults in the memories while operating at NTC. They present a gate-delay model, specific memory for NTC, memory failure modes and effect on the various models by the leakage current.

- **Pinckney et al. [9]:** This work analysis the energy minimum point for parallelized NTC systems. The authors investigate the energy benefits, operating voltage and process of tasks being parallelized. They study the Leakage overhead, Amdahl overhead and Architectural overhead, and conclude that these overheads are interrelated and restrict the possible energy efficiency gains due to voltage underscaling.

- **Kaul et al. [10]:** The authors in this paper strongly uplift the usage of NTC for energy efficiency purposes. The design challenges associated with NTC (i.e., circuit performance variations, subthreshold leakage etc.,) are thoroughly investigated. Additionally, improvements in memories, gates and level converters are proposed. The paper also places a unique emphasis on Design Automation tools to aid in the efficient design of devices operating at Near-Threshold Voltage (NTV).

- **Kaul et al. [11]:** This work presents the challenges associated with the NTV operation of complex System-on-Chip (SOC) designs. The authors design an IA processor capable of working at NTC and investigate the power performance, operation of the processor at different process skews,improvements in register files and logic, and the power and performance of NTV Single-Instruction-Multiple-Data (SIMD) engine.

2.2 Improving the performance and energy efficiency with enhancements in architecture and memory

- **Reagen et al. [12]:** This work presents a co-design technique across the algorithm, architecture and circuit level to improve the energy efficiency of DNN by applying selective pruning through lowering SRAM voltages without compromising the accuracy. The authors use cross layer optimization to the baseline design and obtain up to 8 $\times$ reduction in power.

- **Chen et al. [13]:** This work proposes a run-time pruning technique, called row stationary, that enhances the efficiency of a convolutional neural network by re-configuring the spatial architecture, in order to map its computations. They optimize the energy efficiency by ensuring a maximum reusage of local data to lower the data movement.

- **Wang et al. [14]:** This work presents an elastic DNN accelerator architecture to detect the adversary sample attacks by organizing the execution of DNN and the detect algorithm concurrently.

- **Zhang et al. [15]:** This work introduces an aggressive voltage underscaling method to improve the energy efficiency of DNN accelerators while keeping accuracy drop less than 1%. Their architecture drops the subsequent MAC operation utilizing the erroneous data and uses the extra cycle to correct the error-prone operation.

- **Chandramoorthy et al. [16]:** This paper proposes a technique to enable neural network acceleration at low voltages. The technique provides low voltage operation for almost the entire application run, thereby increasing the energy efficiency and also ensures the mitigation of failures at low voltages. The authors propose to dynamically boost supply voltage during the SRAM read/write accesses.

- **Yu et al. [17]:** This work presents a hardware pruning technique which applies SIMD-aware weight and node pruning synergistically at the design time to improve the energy efficiency of the DNN by reducing the size of the underlying hardware. The authors propose to remove the redundant nodes in each layer, in order to compress the DNN model and thereby avoid the overhead of sparsity.

- **Ozen et al. [18]:** This work presents a highly resilient DNN using modified algorithms by comparing the reductions in the vulnerability surface through appropriate quantization techniques using efficient training methods.

- **Salami et al. [19]:** This work investigates the vulnerability due to permanent and transient faults on the neural network accelerator components and proposes a technique to reduce the faults by retrieving the corrupted bits. They analyze the resilience of the Neural Network model at Register-Transfer level (RTL) and segregate the vulnerability of RTL components. The methodology corrects the bit flips by 47 %, in comparison to the other state-of-art techniques.

- **Kim et al. [20]:** This work demonstrates a memory adaptive training with in-situ canaries, which improves the energy efficiency by enabling an aggressive voltage scaling of DNN accelerator weight memories.

- **Libano et al. [21]:** This work analyzes the impact of radiation-induced errors for a neural network in FPGAs and employ a selective triple modular redundancy on more vulnerable neural network layers to mask the faults efficiently. The authors classify the errors due to radiation as critical errors and triplicate only the most vulnerable layers of the neural network using the proposed technique.

- **Ghodrati et al. [22]:** This work addresses the problems in mixed-signal circuitry caused by the limited range of information encoding, noise susceptibility and Analog-to-Digital conversion overhead by bit-partitioning the vector dot-product into groups of lower bandwidth operations operating in parallel and distributed across various vector elements.

- **Mackin et al. [23]:** This work demonstrates the execution of MAC operations in the data location using the NVM crossbar arrays. The weights are simultaneously programmed at optimal hardware conditions and its efficacy is inspected under notable NVM variability.

- **Shafiee et al. [24]:** This work proposes a pipelined architecture, with each layer of the neural network being assigned specific crossbars and eDRAM buffers used to accumulate the data between pipe stages. The Analog-to-Digital conversion overhead is reduced using a novel data encoding technique and the balance between memristor storage/compute, buffers and ADCs on the chip is handled by performing a design space inspection.

- **Eshraghian et al. [25]:** This work reduces the power/area using a frequency oriented approach to generate analog weights, by utilizing the switching factor of digitized conductance. The authors allocate the kernel information to the device conductance and time-varying input frequency by exploiting the dependency of *v*-*i* plane hysteresis on frequency.

- **Whatmough et al. [26]:** This work proposes a bypass operation to prevent an erroneous value from entering an adder chain by implementing a logic consisting of Razor flip-flop, multiplexer and delay register in between a pipeline stage of two multiply-accumulate operations.

- **Yang et al. [27]:** This work exploits the error resilience of Convolutional Neural Networks to achieve energy efficiency gains by operating the system at reduced power.

- **Mauro et al. [28]:** This work proposes a hybrid memory scheme to improve the energy efficiency of the Binay Neural Networks by replacing the error prone SRAMs with reliable standard-cell memories. They characterize the SRAM memories at ultra-low voltages using a self-test strategy to measure the Bit Error Rate on larger SRAMs.

- **Whatmough et al. [29, 30]:** This work achieves improved throughput by reusing the data and exploiting the sparsity in DNN data structures. Additionally, the timing error tolerance is realized by utilizing DNN algorithmic resilience in DNNs and

demonstrating circuit-level time borrowing in the datapath. They have also proposed a time borrow tracking technique using Razor and approximate error correction methodology to tackle timing errors.

The work (i.e., *EFFORT*) in this dissertation is the first one to explore the disparate combinational delays of arithmetic units, exploit the systolic dataflow pattern and employ a minimal circuit-architectural manipulation to detect and correct the impeding timing errors, along with lowering the dynamic power usage. Additionally, the work (i.e., *PREDITOR*) in the first work to reduce the impact of *undetectable timing errors* using the statistical based approach.

2.3 Improving the reliability at LPC for extreme PV

- **D.Xu et al. [31]:** This work proposes a hybrid computing architecture to recompute the operations mapped to the faulty PEs in arbitrary locations using dot-production processing units (DPPUs). They show the toleration of faulty PEs at random locations by the DPPU and at various fault distributions. The authors demonstrate the utilization of parallel processing in each operation the sequential processing of the network operations.

- **Zhang et al. [32]:** This work prunes the weights mapped to the faulty MAC units to mitigate the impact of permanent faults on a TPU. They also retrain the weight based on the positional knowledge of the MAC units to restore the classification accuracy to the baseline, but with requires an extra time for each TPU chip.

- **Spyrou et al. [33]:** This work demonstrates a fault-tolerant Spiking Neural Network architecture with simplified error detection and recovery scheme. The authors propose to nullify the effect of specific faults using passive fault tolerance modeled on dropout.

- **Salami et al. [34]:** This work evaluates an undervolting technique for Neural Network acceleration in Field Programmable Gate Arrays (FPGAs) to improve the power-efficiency. They investigate the multiple components of real FPGAs at reduced voltage operation and identify reliability behavior of the CNN accelerators. The authors propose to integrate optimization techniques with undervolting to lower the shortcomings of reduced-voltage operation.

- **Givaki et al. [35]:** This work experimentally evaluates the effect of aggressive voltage underscaling of block RAMs in an FPGA by emulating the real fault maps of SRAM memories. The authors proposed to increase the training iteration by 10 % to fit the gap in accuracy, arising due to the undervolting the memories.

- **Tang et al. [36]:** This work investigates the impact of GPU dynamic voltage and frequency scaling on the energy consumption and performance of DNNs. They observed that in comparison to the core frequency operation, operating at optimal frequency can conserve 8.7% $\sim$ 23.1% energy for DNN training and 19.6% $\sim$ 26.4% energy for inference.

- **Lee et al. [37]:** This work explores the optimization methods for hardware architectures for energy-efficient DNN processing on edge devices. The authors also explore hardware architecture optimization and data-path structure, in addition to hardware co-designed ASICs and DNN algorithm.

- **Nguyen et al. [38]:** This work presents a stretchable DRAM refresh control technique to replace non-critical bits with parity bits of error correction schemes resulting in improved energy efficiency without performance degradation of DNNs. The DRAM chip consists of an error-free zone and an error zone, with the error-free zone having the same refresh time as the normal ones and the error zone having its refresh time stretched adaptively based on the bit-error-rate of the users.

- **Koppula et al. [39]:** This work proposes a general framework that reduces DNN energy consumption by using approximate DRAM devices while meeting the user-specified DNN accuracy requirements. They propose increasing the DNN's error tolerance by retraining the DNN for the approximate DRAMs and mapping the error tolerance of DNN data types with the appropriate DRAM partitions.

- **Jiang et al. [40]:** This work develops a Dynamic Voltage and Frequency Scaling (DVFS) framework on FPGAs to analyze the impact of DVFS on CNNs in terms of performance, power, energy efficiency and accuracy.

- **Elbtity et al. [41]:** This work introduces approximate PEs to replace the direct quantization of inputs and weights, utilized per-approximate units and shared them with approximate PEs to reduce the overhead arising from the element-wise operation. The authors demonstrate a reduction in the critical path delay in a PE required for the different types of multiplication and lower the processing time in a large scale multi-element arrays for a forward pass of data.

- **Ruospo et al. [42]:** This work reduces the fault simulation times for DNN inference execution using a fault injector framework, which replicates the pipeline flow. They elaborate the Neural Network (NN) layers as a pipeline in a processor core and utilize a process to ensure the synchronization of individual computations involved in an inference phase.

- **Hosseini et al. [43]:** This work proposes algorithms to approximate the value of fault-free bits in the defective DNN weight memories and reduce the drop in DNN accuracy due to faulty Non-volatile memories. The authors utilized the algorithms to lower the deviations in the DNN weights and perform the deviation minimization process at weight deployment stage to reduce the overheads due to hardware and runtime.

- **Hoang et al. [44]:** This work demonstrates a technique to develop a unified fault map from smaller fault maps lower the retraining rounds and reduce the retraining overhead. The authors investigate the different quantization methodologies and provide a detailed analysis on the DNNs error resilience for permanent faults present in the on-chip weight memory.

To the best of knowledge, the work (i.e., *STRIVE*) in this dissertation is the first one to emphasize the impact of the PV in transforming a zero computation to a non-zero value and design a low-overhead fault detection and mitigation scheme to mitigate the effect of PV in a TPU systolic array containing faulty gates.

CHAPTER 3

RECLAIMING THE ENERGY EFFICIENCY OF A NEAR-THRESHOLD TENSOR PROCESSING UNIT WITH TIMING ERROR RESILIENCE AND CLOCK GATING

3.1 Background and Contributions of This Work

Advancements in artificial intelligence have entered a new realm owing to the development of domain specific architectures dedicated to neural networks (NN) processing. Tensor processing unit (TPU), a custom application specific integrated circuit (ASIC) built by Google, is one such accelerator, which is exclusively built to handle most of the deep neural networks (DNN) inference workloads in their servers.

The rapidly increasing workloads calls for an increase in the processing speed and deployment volume [45]. It, however, comes at a cost of a heavy power usage, thus affecting the energy efficiency of the system. In order to preserve the energy efficiency, the TPU is operated at the near-threshold computing (NTC) region, where the transistor supply voltage is scaled down to just above its threshold voltage [2].

Accelerators like TPUs are designed to offer a very high throughput for DNN inference workloads. Although NTC operating conditions can ensure a low energy consumption, the throughput is heavily declined due to the slower transistors and longer computational delays. As the technology nodes shrink in size, inefficiency in strictly controlling the fabrication process introduces Process Variation (PV) into the system. PVs are caused due to a combination of systematic and random effects, which induces undesired delays in the computation paths [4, 5, 7, 46]. Shabanian et al and Bal et al, demonstrated that both NTC-GPU and NTC-CPU are highly susceptible to PV, which cause timing violation induced performance bottleneck [47, 48]. NTC-TPU is not an exception, this sensitivity can impact the DNN inference accuracy significantly [2, 6]. This work underlines the significance of the computational delays and order of execution of the arithmetic units, to handle

timing violations in NTC TPUs. Additionally, the energy efficiency of the TPU is enhanced by exploring the predictable data flow pattern in its systolic array, thereby promoting an error-resilient and energy-efficient TPU design paradigm.

Several timing error resilient schemes have been explored for CPUs and custom ASICs [15,49–65]. However, these schemes are inefficient for combating timing violation in TPUs. Razor is one such popular timing violation detection method which uses a double sampling flip-flop to detect the errors [49]. Using instruction replay, the erroneous data is recomputed and the correct value is propagated to the next stage of the pipeline. TPU has a massive systolic array of 256×256 multiplier-and-accumulate (MAC) units. So, using an instruction replay in one MAC unit, results in stalling the operation of the entire systolic array, leading to a massive drop of throughput and increase in the energy consumption. TE-Drop is a recently proposed technique to handle timing violations in TPU-like systolic arrays. In this technique, the MAC unit encountering a timing error, steals an execution cycle from its downstream MAC, and recomputes the correct value [15]. In the process, the downstream MAC's computation is bypassed. However, there will be multiple levels of bypassing, in case of timing errors in consecutive rows of the same column of MACs, in the same clock cycle. Bypassing multiple computations can cause a severe drop in the inference accuracy. Additionally, the timing errors encountered in the last row of MACs will not be tackled by TE-Drop, also resulting in an accuracy drop. A naive approach to tackle timing violations is to allow the erroneous data to flow through the successive stages of operations [50, 66]. This technique undermines the effects of the erroneous data in DNN computations, as a large number of timing errors causes a significant drop in the inference accuracy [67].

In order to overcome the drawbacks of these error handling schemes, this work proposes a unique timing error correction technique which handles timing errors in the same cycle of the execution while enhancing the energy efficiency of the TPU. It is observed that in a MAC, multiplier takes relatively higher execution time than accumulator (Section 3.2.3). Additionally, it is observed that a predictable data flow pattern in the TPU systolic

array (Section 3.3.3). Analyzing these computational delays, data flow patterns, and utilizing the computational order of the arithmetic units, this work proposes EFFORT—an error resilient, low-power, novel TPU design paradigm. Following are the specific contributions of this work:

- It is experimentally demonstrated that a 8-bit multiplier takes higher computation time than a 24-bit accumulator (Section 3.2.3). The computational delays and operational order of these arithmetic units are exploited to tackle the timing errors.
- A predictable data flow pattern is observed in the TPU and this data flow pattern is utilized to reduce the energy consumption in the systolic array (Section 3.3.3).
- EFFORT—an energy efficient dynamic timing error detection/correction technique is proposed, that detects the timing errors, obtains the corrected data and propagates it to preserve the output accuracy(Section 3.3.2), while simultaneously employing a low-overhead clock gating technique to improve the energy efficiency (Section 3.3.3) of the TPU.
- In comparison to TE-Drop [15] and the Baseline-TPU, EFFORT delivers 2.5× better performance for 6 out of 8 DNN benchmarks, while incurring only 4% loss in inference accuracy (Section 3.5.2).
- It is demonstrated that EFFORT consumes up to 6% and 27% less power and gives up to 1.06× and 1.35× better performance per unit power, than Baseline-TPU and TE-Drop (Section 3.5.3).

3.2 Motivation

In this section, the unseen opportunities that can be availed to tackle timing errors in a TPU systolic array are illustrated. Section 3.2.1 sheds light on the background of the TPU systolic array and inherent opportunities which can be exploited for an improved performance. Using the cross-layer methodology in Section 3.2.2, the MAC units' delay profiles

are investigated. Section 3.2.3 elaborates the significance of the results and establishes the ground work for the timing error correction and dynamic power management scheme.

3.2.1 Background

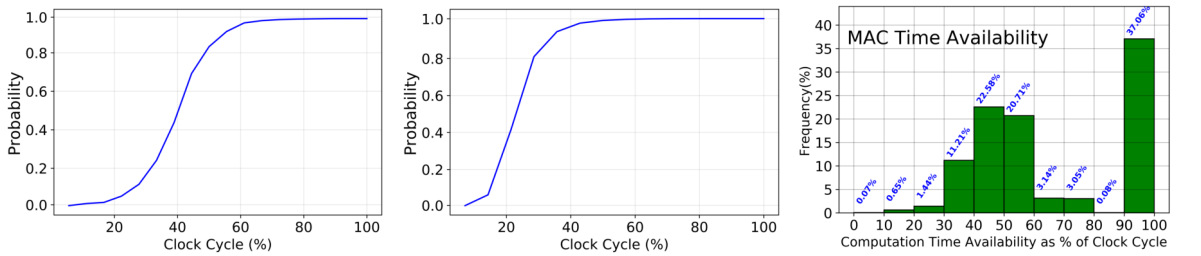
DNN Accelerators

DNN obtains inference using multiple layers of computation. Outputs of neurons from each layers are referred to as activation streams. An activation matrix is multiplied with the weight matrix in each layer. To accelerate the matrix multiplication, a systolic array of MAC units are employed in DNN accelerators [68]. TPU–a DNN accelerator–uses a 256×256 systolic array of MACs. The weight matrices are pre-loaded into the MACs. The activation streams flow from left to right in consecutive clock cycles. The activation and weight matrices maintain an 8-bit integer precision, while the accumulator maintains a 24-bit integer precision.

Opportunities in a Systolic Array

The asymmetric delay distributions of the multiplier and accumulate blocks in a MAC unit, open up a unique opportunity to tackle timing errors in a systolic array. The accumulate operation in a MAC, adds the output of the upstream MAC to the output of its own multiplier block. Due to a relatively large computation time of the multiplication operation, the output from the upstream MAC has ample time to reach the current accumulate unit, presuming the synchronization takes place at the primary output of the MAC. *Exploiting this available timing window, correcting an erroneous operation at the upstream MAC can be overlapped with the multiplication operation of the current MAC, without paying any additional performance penalty.*

The wavefront propagation of data in a systolic array leads to a static pattern of busy and idle phases. Such predictable pattern creates an avenue to conserve power of the idle MAC units. The experimental methodology used to demonstrate these opportunities in the systolic array of an NTC TPU is briefly discussed in the next section.



(a) CDF of multiplier delay distribution.

(b) CDF of accumulator delay distribution.

(c) Histogram of time availability in MAC computations.

Fig. 3.1: *CDFs of the delay distributions for multiplier (Figure 3.1(a)) and accumulator (Figure 3.1(b)) show that the multiplier has higher computational delay compared to the accumulator. Accumulator takes less than a half clock cycle for its part of computation. Figure 3.1(c) shows the time available for recomputation in downstream MACs when timing errors occur in the respective upstream MACs.*

3.2.2 Methodology

A multiplier and an accumulator unit at NTC are synthesized, using 15-nm FinFET library from NanGate [69]. To model the PV at NTC for FinFET, VARIUS-NTV models [70] are used. PV-induced delays in randomly chosen 2% of the gates in the circuit is considered for a conservative estimate [71,72]. The in-house statistical timing analysis tool is used to investigate the delay distribution of the sensitized path for different inputs to the multiplier and accumulator unit. The computational time availability in the MAC units in real time is analyzed by simulating the in-house TPU systolic array simulator in a real time environment. Section 3.4 provides further elaboration on the cross-layer methodology.

3.2.3 Results and Significance

Figures 3.1(a) and 3.1(b) show the delay distributions of the multiplier unit and accumulate unit, respectively. The multiplier unit is tested for a set of all possible 8-bit activation streams created against all possible 8-bit weight streams, which results in a total of 65536, 16-bit output combinations. Each one of these 65536 outputs serve as one of the inputs for the accumulator while the other input of the accumulator is fed with its own output from the previous cycle.

The cumulative distribution functions (CDFs) for the multiplier and accumulator in

Figures 3.1(a) and 3.1(b) indicate that the multiplier utilizes between 20-60% of the clock cycle and the accumulator uses less than 40% of the clock cycle. From Figure 3.1(a) and 3.1(b), it can be inferred that, even if the output of multiplier unit is sensitized only to the activation stream due to the preloaded weight, the multiplier unit still induces a higher combinational delay to the MAC operation in comparison to the accumulator, during a clock period of operation. Figure 3.1(b) shows that the accumulation requires less than half cycle of the clock period. *This disparate timing characteristics of the multiplier and accumulate operations create an opportunistic timing window to correct any timing violation in an upstream MAC, thereby preventing any erroneous value to be propagated down the column of the systolic array.*

Figure 3.1(c) demonstrates the computational time available in the downstream MACs, when the respective upstream MACs encounter a timing error. The TPU simulator is tested using the appropriate activation inputs and trained weights extracted from the MNIST [73] dataset. The computational time availability pattern of a MAC unit in a timing error scenario is depicted as a histogram in Figure 3.1(c). The X-axis is split into ten bars, with each bar indicating the percentage of MAC computations having time availability in the particular range of the clock cycles. From Figure 3.1(c), it is evident that more than 60% of the clock cycle period is available for re-computation process in a downstream MAC in the event of a timing error. The time availability is majorly attributed to the minuscule accumulator operation and also to the large number of zero-skip operations [12, 13, 74].

Figure 3.2 depicts the decrease in dynamic power and domination of static power in a MAC unit as the region of operation is changed from super-threshold computing (STC) to NTC. The X-axis is normalized to 1GHz. Operating voltage is set at 0.85V and scaled linearly to depict the shift in operating conditions. Static energy consumption can be reduced by operating the systolic array at frequencies, above the nominal NTC frequency. However, increasing the operating frequency linearly increases the dynamic energy consumption of the MAC units. In order to curb the increase in dynamic energy, an in-situ clock gating technique can be employed in the systolic array. With this opportunistic win-

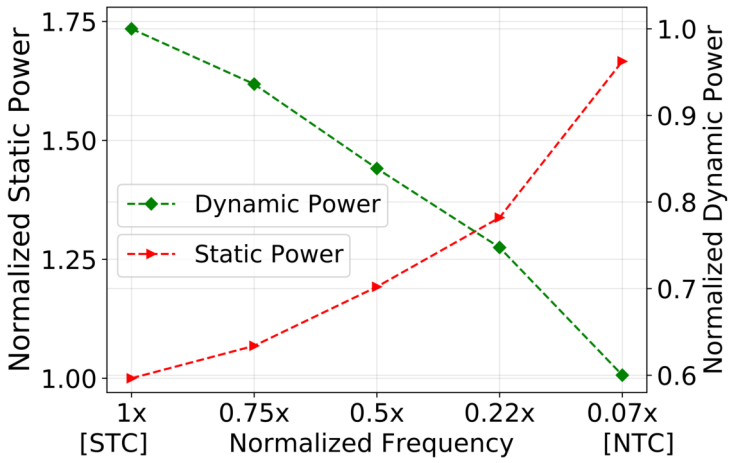


Fig. 3.2: *Figure 3.2 portrays the increase in static power and decrease in dynamic power for decreasing frequencies. Voltages are scaled accordingly to depict the shift from STC to NTC.*

dow in-sight, the performance enhancing TPU systolic array design–EFFORT, is analyzed next.

3.3 EFFORT Design

Energy eFFicient and errOr Resilient TPU (EFFORT), is a novel design paradigm to improve the performance of an NTC TPU by enhancing the timing error resilience of its MAC units and managing the dynamic energy consumption of the systolic array. The overview of EFFORT is described in Section 3.3.1. The detailed components of EFFORT are explained from Section 3.3.2 to Section 3.3.5.

3.3.1 Design Overview

Figure 3.3 demonstrates the high-level design of EFFORT. Two key modifications are added to the baseline NTC TPU. First, each MAC unit is augmented with a novel penalty-free error detection and correction logic, thus preserving a high performance. Second, a low-overhead clock gating technique is implemented to conserve the dynamic power of the systolic array. These two components are discussed next.

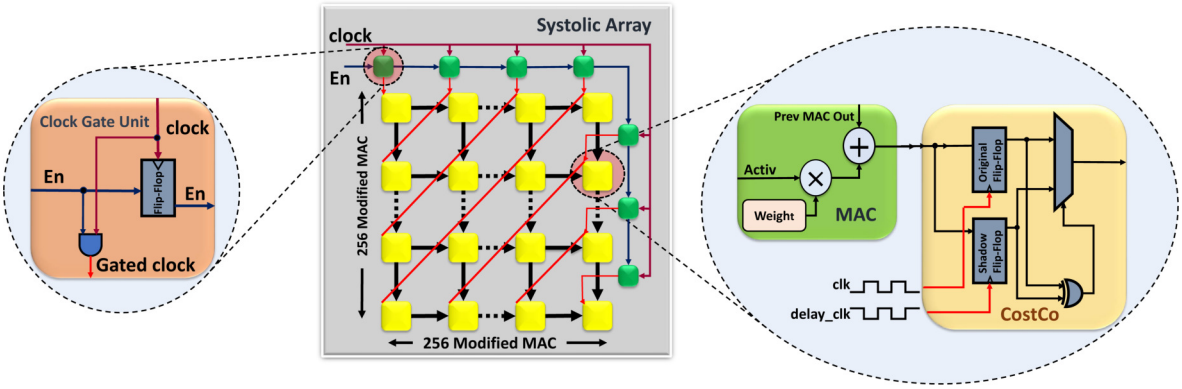


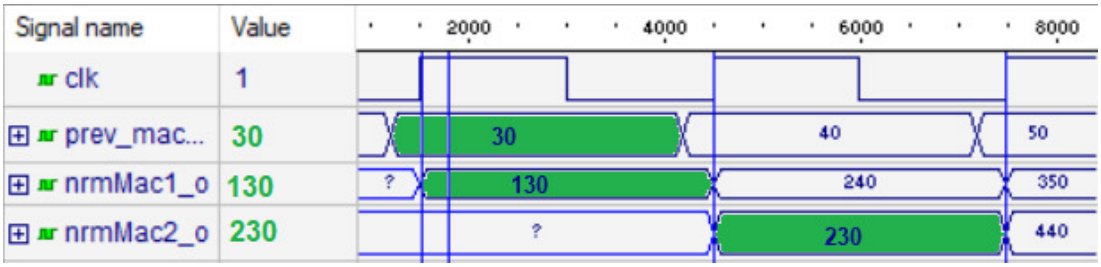
Fig. 3.3: *CostCo implemented inside the MAC unit to detect and correct timing violations.*

3.3.2 Costless Correction (CostCo)

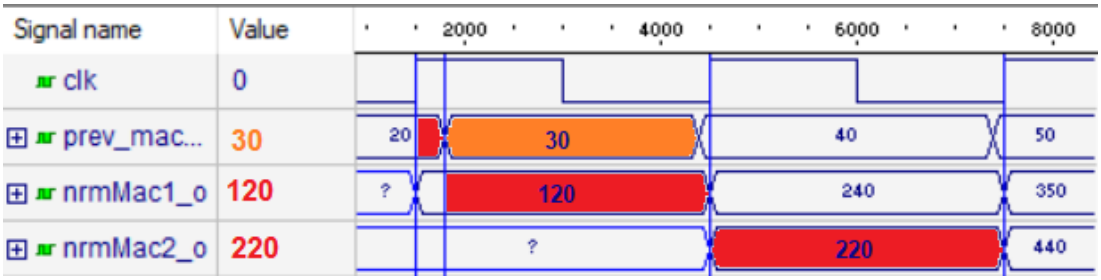
In this section, costless correction (CostCo) is introduced. The conventional Razor [49] is augmented with a multiplexer (MUX) and an Exclusive-OR (XOR) gate, as demonstrated in Figure 3.3. Since CostCo controls its output with the comparison of the shadow latch and the main latch, it is capable of propagating the correct value to the downstream logic within the same clock cycle that timing error detection happens. Figure 3.4 demonstrates the RTL simulation waveforms for two consecutive MAC units within a column, in a systolic array, both in absence and presence of a timing violation. Note that, *only* the primary output of the MAC units is synchronized with the system clock, to enable the proposed CostCo design.

Figure 3.4(a) demonstrates the normal output waveforms of two consecutive column MACs in absence of any timing violation. Figure 3.4(b) shows how a small additional delay in the input of the first MAC, engenders timing violation in its immediate downstream MAC, leading to an erroneous result. Figure 3.4(c) exhibits how CostCo can detect the timing violation and propagate the correct value to its succeeding downstream MAC, within the same clock cycle. CostCo can be employed as a competent method to tackle timing violation if the downstream combinational logic has sufficient time-window before the next rising edge of the clock, to replay its logical operations on the corrected data.

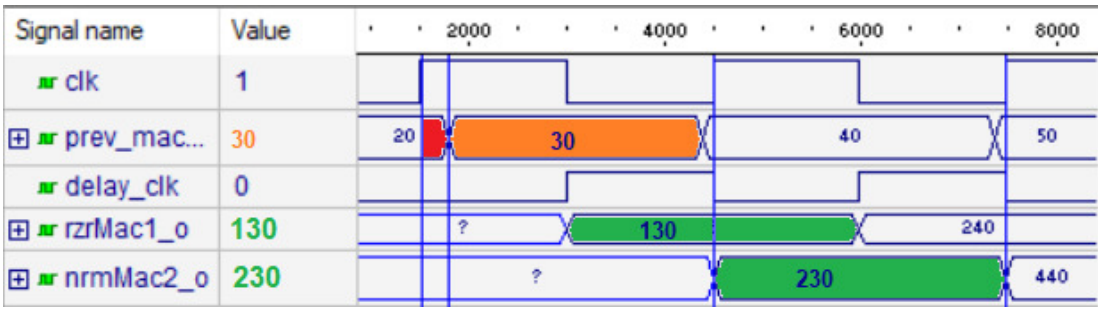
In case of a timing error, 24 CostCo flip-flops are considered at the output of each MAC to provide the correct values to the downstream MACs. The output of each MAC



(a) No timing violation.



(b) Timing violation with no correction.

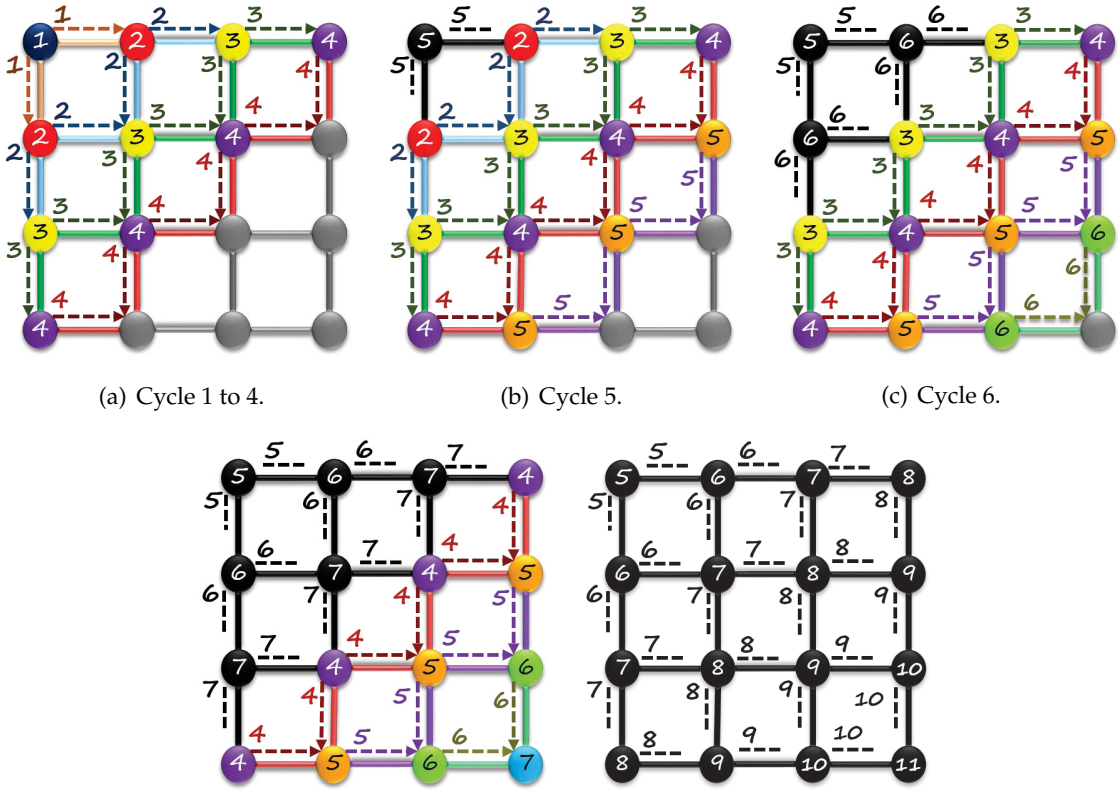


(c) Timing violation with correction.

Fig. 3.4: *CostCo can diagnose timing violation and propagate the corrected value within one clock cycle.*

is utilized in the accumulation operation in its succeeding MAC. As accumulation in each MAC requires less than 50% of the clock cycle (Section 3.2.2), a 50% shift in the system clock is considered to provision the CostCo flip-flop clock. This shift of clock provides an opportunity to detect timing errors up to 50% beyond the system clock, while it guarantees the succeeding MACs to have adequate time to accomplish their accumulation operations in the remaining time window. The hardware overhead and performance gain of this design is discussed in Section 3.5.

3.3.3 Systolic Clock Gating



(a) Cycle 1 to 4. (b) Cycle 5. (c) Cycle 6.

(d) Cycle 7. (e) Cycle 8 to 11.

Fig. 3.5: *Data flow pattern in a* 4×4 *systolic array for 11 consecutive cycles. Gray MACs are yet to receive their inputs, black MACs have completed their operations, while the rest of the MACs are presently computing their respective outputs.*

In EFFORT, the operating frequency is increased, while keeping the supply voltage at the nominal NTC value, in order to provide a better performance compared to a baseline NTC TPU. To reduce the power consumption due to a high-frequency operation, the application independent data-flow pattern within the TPU systolic array is exploited, and employ a low-overhead clock gating technique.

Application Independent Data Flow

Figure 3.5 demonstrates the pattern of data flow inside a 4×4 systolic array for 11 consecutive clock cycles. In this figure, the gray nodes represent the MACs that have not received their data yet, the nodes in black denote the MACs that completed their opera-

tions, and other colored nodes demonstrate the MACs which are doing their operations. Numbers on black nodes show the cycle when they completed their operations, numbers on other colored nodes represent the cycle in which they received their first data, and numbers on each edge display the cycle in which the preceding node attempts to activate or deactivate its subsequent nodes either on the right-hand side or down a row. As Figure 3.5(a) exhibits, considering the upper left node as the start point from cycle 1 to 4, all the MACs from start point down to the main diagonal, receive their data respectively in a sequential fashion, while the rest of them are yet to receive their data. After cycle 4 (Figure 3.5(b) through Figure 3.5(d)), as a new set of MACs receive their data in each cycle, another set of MACs accomplish their tasks. Figure 3.5(e) displays the systolic array after 11 clock cycles, when the entire systolic array operation is completed. It is observed that all the MACs on the same diagonal of the systolic array, are active or idle in the same cycles.

Clock Gating Components

Based on the activity pattern of a systolic array, a low overhead clock gating technique is proposed, to shutdown the clock of idle MACs, improving the dynamic energy consumption of the TPU. Since all of the MACs on the same diagonal of the systolic array are active or idle in the same clock cycle, instead of endowing a separate clock gating unit for each MAC, only one clock gating unit for each set of MACs on each diagonal. Since an $n \times n$ matrix has $(2n-1)$ diagonals, the total number of required clock gating units is reduced from n^2 to $(2n-1)$. Figure 3.3 shows that each clock gating unit consists of one flip-flop to register the *enable signal* for the downstream clock gating unit, and an AND gate to control the clock for its corresponding group of MACs.

MAC Activity Analysis

Generalizing from Figure 3.5, an $n \times n$ systolic array needs $(3n-2)$ cycles to complete its operation. However, not all MACs are active during each clock cycle. For an $n \times n$ systolic array, at each clock cycle in the range [Cycle 1, Cycle n] and [Cycle ($2n$ - 1), Cycle ($3n$ - 2)], the total number of active MACs is $\frac{n \times (n+1)}{2}$. Furthermore, in the interval [Cycle

$(n+1)$, Cycle $(2n)$], at each cycle i, the number of active MACs change by $(n-(2\times i))$, where a positive (negative) value indicates an increase (decrease) in the number of MACs. Applying the proposed clock gating technique by summing the number of active MACs during the $(3n-2)$ clock cycles, the order of active sequential logic is reduced from $(3n^3)$ to (n^3).

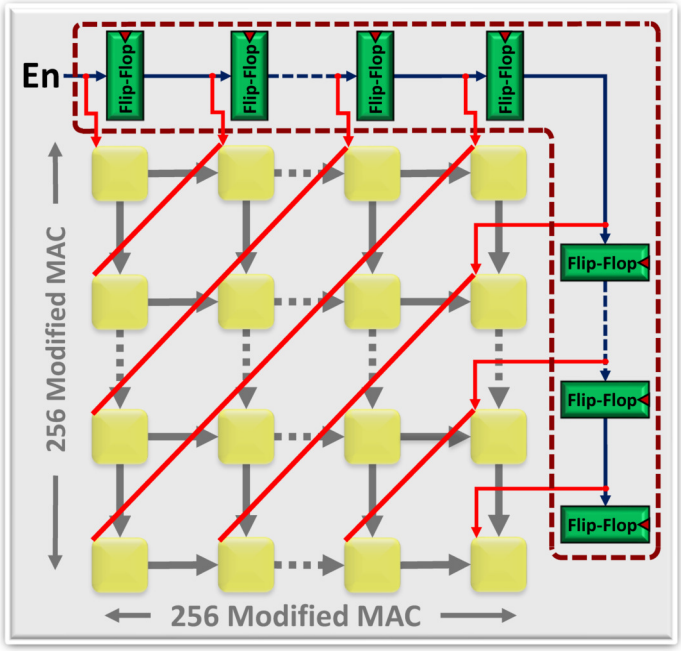


Fig. 3.6: *Serially connected flip-flops in the Clock Gate units effectively exhibit the characteristics of Right Shift Register, thereby suitably enabling/disabling the MAC units along the diagonal of the systolic array.*

Clock Gate Unit

The inherent dataflow along the diagonal of systolic array facilitates the generation of Enable (En) bit for consecutive clock gate units. Based on the MAC activity analysis (Section 3.3.3), a simple heuristic as illustrated in Algorithm 1 is developed, to enable/disable MACs during the systolic array operation. The consecutive clock gate units, connected to each other effectively work as a Serial Input Parallel Output (SIPO) Right Shift Register as depicted in Figure 3.6. For each clock cycle in the range [Cycle 1, Cycle n], En bit is set

Algorithm 1 Gating Enable Algorithm

1: $Systolic\ Array\ Dimension\ (n) = 256$
2: $Tot.\ Sim.\ Cycles\ (tot_{cycles}) = 766 \leftarrow ((3 * n) - 2)$
3: $activ \leftarrow 1,\ inactive \leftarrow 0$
4: **while** $Sim_{cycle} \leq tot_{cycles}$ **do**
5: **if** $Sim_{cycle} \leq n$ **then**
6: $En \leftarrow high$
7: **else**
8: $En \leftarrow low$
9: **end if**
10: **end while**

to active high state (lines 5-6 in Algorithm 1), thereby appropriately enabling the horizontally aligned clock gate units in a bitwise order. Further, from clock interval [Cycle $(n+1)$, Cycle $(3n-2)$], En bit is set to active low state (lines 7-8 in Algorithm 1), disabling the horizontally aligned clock gate units and appropriately enabling/disabling the vertically aligned clock gate units in the consecutive cycles. The experimental results and hardware overhead of this technique is discussed in Section 3.5.

3.3.4 EFFORT Variants

Two different variants of the EFFORT scheme are explored with regard to the timing error resilience provided by CostCo by underlining the significance of Most Significant Bits (MSBs) of the MAC unit's accumulator.

EFFORT-24

EFFORT-24 is a variant encompassing the comprehensive architecture, elaborated so far. It includes the 24 CostCo flip-flops (Section 3.3.2) at the output of each MAC unit to ensure a safe passage of corrected values to the downstream MACs in the event of a timing error.

EFFORT-14

EFFORT-14 is variant with only 14 CostCo flip-flops [15] augmented to the output of each MAC unit. The 14 CostCo flip-flops are assigned to the 14 Most Significant Bits

(MSBs) of the accumulator output. EFFORT-14 will detect any modifications in the 14 MSBs due to timing errors and the updated value will be provided to downstream MAC unit. However, any modifications to the remaining 10 bits (i.e., 10 Least Significant Bits (LSBs)) due to timing errors will be ignored. Since almost half of the accumulator bits are protected, the differential impact of the timing errors can be observed by comparing the outputs of EFFORT-24 and EFFORT-14.

EFFORT-8

EFFORT-8 is a trimmed version of EFFORT-24 with only 8 CostCo flip-flops supplemented to the MAC output. EFFORT-8 will have only 8 MSBs of the MAC unit output protected by CostCo flip-flops. EFFORT-8 aids in further analyzing the impact, the varying degree of timing errors has on the system output. EFFORT-8 also helps in understanding the performance/energy trade-off, due to the lower degree of protection it offers in comparison to EFFORT-24.

3.3.5 Design Summary

In this section a brief summary of the EFFORT design for deeper clarity is provided.

- ***CostCo*** consists of the a modified Razor flip-flop (2 flip-flops, XOR gate and Multiplexer). CostCo exploits the time borrowing window (Section 3.2.3) to detect the late transitioning output to improve the overall inference accuracy (Section 3.5.2).
- ***Systolic Clock Gating*** utilizes the diagonal wavefront dataflow pattern in the systolic array to reduce the dynamic energy consumption, and enhances the energy-efficiency gains (Section 3.5.3).
- Hence, ***EFFORT*** (EFFORT-24) constitutes of CostCo and Systolic Clock Gating to provide a significant increase in the overall performance of an NTC TPU (Section 3.5.3).

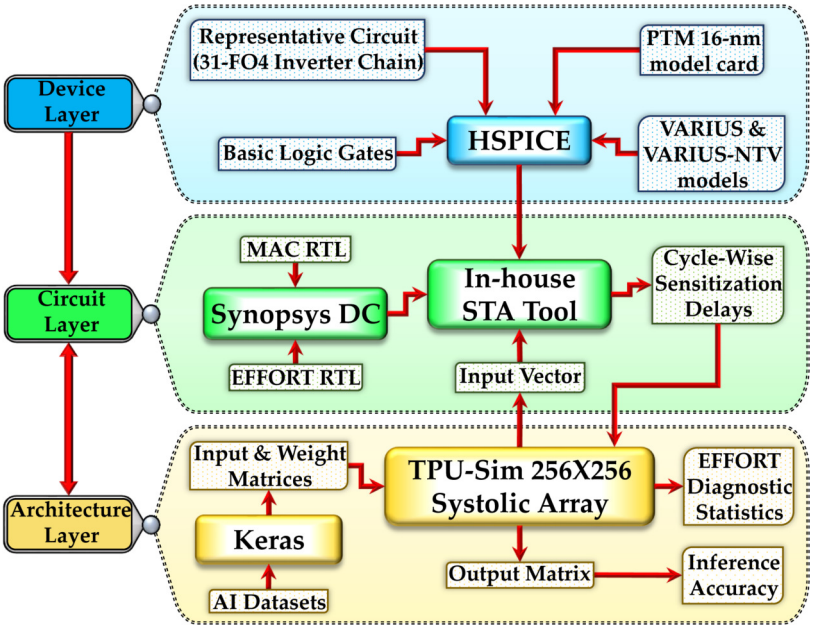


Fig. 3.7: *Cross Layer Methodology*

- ***EFFORT-14*** and ***EFFORT-8*** are lightweight variants of EFFORT (EFFORT-24), which provides bit protection to only 14 and 8 MSBs respectively.

3.4 Methodology

In this section, the comprehensive cross-layer methodology, used to implement the proposed design and evaluate its capabilities across DNN applications is expounded. Figure 3.7 depicts the cross-layer methodology, which will be outlined in the following sections.

3.4.1 Device Layer

The delay distributions of the basic logic gates are measured (e.g., NOR, NAND and Inverter) by performing HSPICE simulations with 16-nm Predictive Technology Model [75]. The impact of with-in die process variation at NTC is considered by using VARIUS-NTV model [8]. In addition, VARIUS-TC model is employed to integrate the FinFET attributes [76]. The delay values are used in the circuit layer (Section 3.4.2) to analyze the delays in a MAC unit.

3.4.2 Circuit Layer

A TPU systolic array is implemented, as well as, the components of the proposed design in Verilog RTL. The developed RTLs are synthesized using Synopsys Design Compiler. The synthesized netlists are used in the in-house statistical timing analysis (STA) tool. The STA tool employs libraries of delay distributions for basic logic gates from HSPICE simulations (Section 3.4.1), to provide the delays of the sensitized paths in the MAC circuit. The sensitized delays are utilized for further evaluations of the proposed technique.

Benchmarks		**Error free Accuracy**
Name	**Basic Layers Architecture**	
SVHN [77]	CONV: (32, 32, 3)x(32, 32, 32)x(32, 32, 32)x(14, 14, 64)x(14, 14, 64)x(5, 5, 128)x (5, 5, 128), FC: 512x512x10	0.94
GTSRB [78]	CONV: (3, 48, 48)x(32, 48, 48)x(32, 46, 46)x(64, 23, 23)x(64, 21, 21)x(128, 10, 10)x (128, 8, 8),FC: 2048x512x43	0.97
Reuters [79]	FC: 2048x256x256x46	0.80
IMDB [80]	CONV: 400x(400x50)x(398, 256), FC: 256x1	0.89
MNIST [73]	FC: 784x256x256x10	0.98
CIFAR-10 [81]	CONV: (32, 32, 3)x(32, 32, 32)x(32, 32, 32)x(16, 16, 64)x(16, 16, 64)x(8, 8, 128)x (8, 8, 128),FC: 2048x512x10	0.77
FMNIST [82]	FC: 784x256x512x10	0.89
AMNIST [83]	CONV: (20, 25, 1)x(20,25,128)x(20,25,64), FC: 32000x256x128x40	0.92

Table 3.1: *List of DNN benchmarks used and the error free accuracy.*

3.4.3 Architecture Layer

The in-house TPU systolic array simulator developed using C++ is used. The TPU systolic array is based on the detailed architecture of a TPU [68]. The delays from the STA tool (Section 3.4.2) are incorporated into the TPU Simulator to simulate timing errors in the MAC units. Keras [84] is interfaced with the TPU simulator to replicate a real-life inference engine. The DNN applications listed in Table 3.1 are trained using Keras. Activation inputs

and trained weights from each layers are extracted and separated into 256×256 matrices. Inference accuracy is obtained by combining the output matrices from the simulator.

3.5 Experimental Results

In this section, the efficacy of different schemes for NTC TPU operation is compared. The baseline frequency of operation for the schemes is (0.45V, 67.5MHz), which offers an error-free execution of the systolic array. Section 3.5.1 introduces the comparative schemes. Section 3.5.2 presents the inference accuracies. Section 3.5.3 discusses the power savings and Section 3.5.4 presents the overheads of EFFORT.

3.5.1 Comparative Schemes

- **Baseline-TPU :** This scheme operates an NTC TPU without any error detection and correction. It allows the erroneous data to propagate through all the computation stages in the systolic array [50].
- **TE-Drop :** This technique handles the timing errors by dropping the subsequent downstream MAC operation [15]. The erroneous MAC recomputes the output by stealing the clock cycle from its downstream MAC.
- **EFFORT-24 :** This is the proposed technique which uses the opportunistic timing window in the MAC operation to detect and correct timing errors (Section 3.3). However, if a computational delay falls beyond that opportunistic timing window, an erroneous value will be propagated.
- **EFFORT-14 :** This is a variant of EFFORT, with only 14 MSBs of the MAC unit output protected by CostCo flip-flops (Section 3.3.4).
- **EFFORT-8 :** This is a lighter variant of EFFORT providing security to only 8 MSBs of the MAC unit output (Section 3.3.4). Timing errors causing modifications to the remaining 16 LSBs will not be corrected.

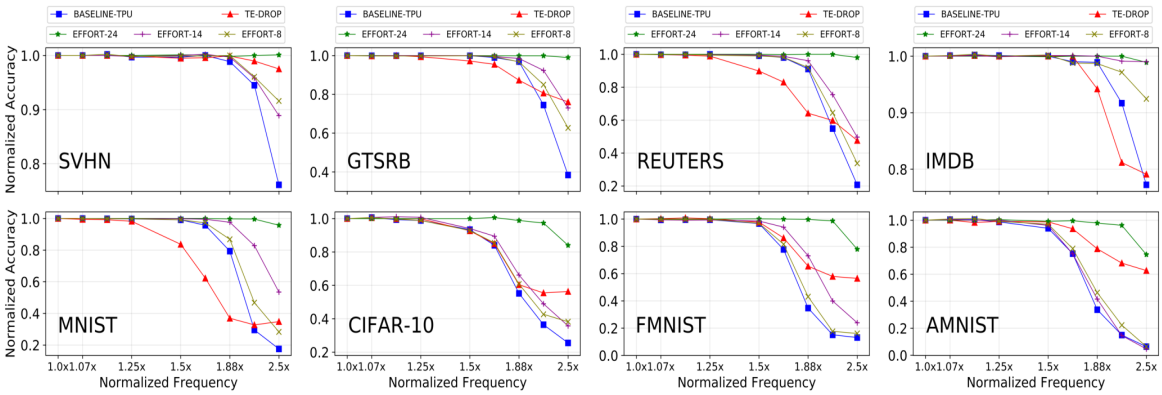


Fig. 3.8: *Normalized inference accuracies of the 8 DNN benchmarks for different comparative schemes.*

3.5.2 Inference Accuracy

Figure 3.8 shows the normalized accuracies for different comparative schemes at various operating frequencies. The operating voltage is set to 0.45V for all frequencies. Y-axis is normalized to the error free accuracy for the baseline operation and X-axis is normalized to the baseline frequency. Error free accuracy for the DNN benchmarks are SVHN: 0.94, GTSRB: 0.97, REUTERS: 0.80, IMDB: 0.89, MNIST: 0.98, CIFAR-10: 0.77, FMNIST: 0.89 and AMNIST: 0.92 respectively. The normalized accuracies for the 8 DNN benchmarks drip at different rates from the maximum for the various resilient schemes, emphasizing the impact of timing errors as the performance points are increased.

A modest timing error resilience can be observed in all the schemes up to 1.25× the baseline frequency of operation. Accuracy begins to decline as the number of errors drastically increases at higher frequencies. EFFORT-24 and EFFORT-14 outperforms other schemes by detecting and correcting most of the timing errors. However, for CIFAR-10, FMNIST and AMNIST, the computational delay at the highest frequency is relatively higher than other benchmarks, which increases the number of undetected errors and consequently, causes more reduction in the inference accuracy. The complete protection provided to all the accumulator bits aids EFFORT-24 in providing a better error resilience compared to EFFORT-14. At higher frequencies, significant rise in timing errors leads to a huge number of timing errors getting unnoticed by EFFORT-14, thereby restricting its

error handling capability. Since EFFORT-8 offers protection to only a third of the accumulator bits, it significantly falls short of EFFORT-24 and EFFORT-14. However, the varying degree of protection offered to the MSBs of the accumulator bits (8-bit $\rightarrow$ 14-bit $\rightarrow$ 24-bit), enables us to assess the orders of improvement in performance achieved from EFFORT-8 (8-bit) to EFFORT-24 (24-bit). Baseline-TPU has a relatively sudden fall in inference accuracy as propagating errors in successive stages massively deteriorates the quality of the output matrices [67]. Inference accuracy for TE-Drop, however, falls at a slower pace, compared to the Baseline-TPU. At higher frequencies, due to a large number of timing errors, TE-Drop bypasses a higher number of MAC computations, resulting in inferior accuracies compared to EFFORT-24. Hence, an NTC TPU, enhanced with EFFORT, results in only 4% average accuracy loss, when operated up to 2.5$\times$ the baseline frequency, for 6 out of 8 DNN benchmarks.

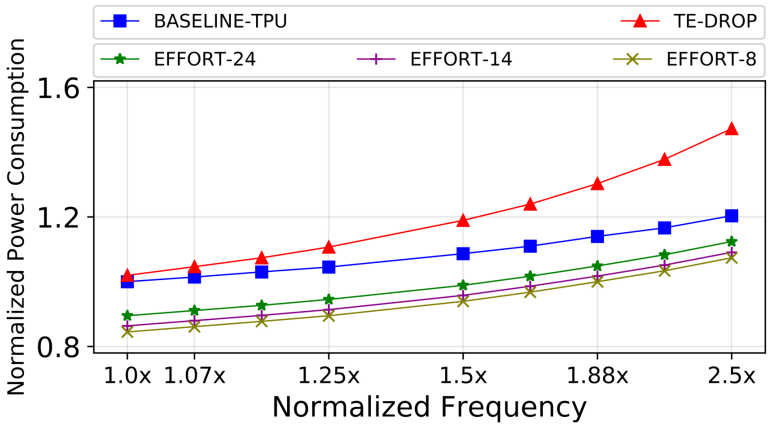


Fig. 3.9: *Power Consumption (Lower is better).*

3.5.3 Energy Efficiency

Figure 3.9 shows the average power consumption for the 8 DNN benchmarks for different comparative schemes. Power consumption for the comparative schemes are normalized to the power consumption of the Baseline-TPU at the baseline frequency (Power

consumed by Baseline-TPU at 1.0x frequency is 1.3 W). With the increasing operational frequency, power consumption steadily increases for all the schemes. However, EFFORT-24 has lower power consumption compared to TE-Drop and Baseline-TPU. Despite the addition of CostCo into MAC units, the clock gating mechanism (Section 3.3.3) implemented in EFFORT-24 yields lower dynamic power in MAC units which are idle. Hence, the overall power consumption for the systolic operation is reduced. Thus, EFFORT-24 consumes up to 6% and 27% less power when compared to Baseline-TPU and TE-Drop. TE-Drop, due to its Razor flip-flops, has the highest power consumption. Limited protection provided to the accumulator bits in the EFFORT variants yields a lower area footprint compared to EFFORT-24, thereby incurring lower power consumption.

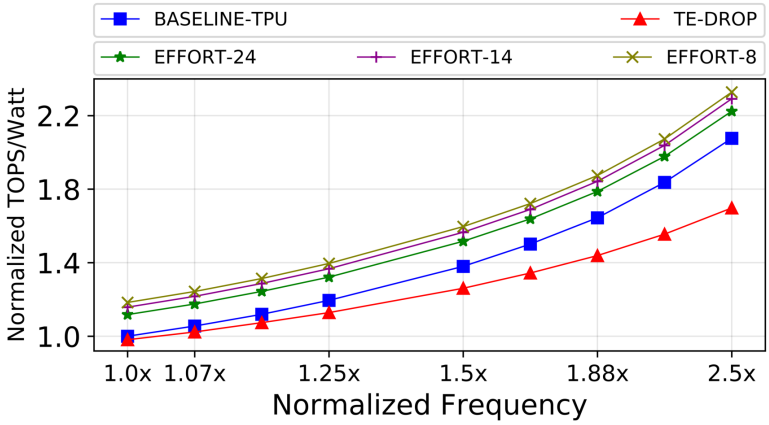


Fig. 3.10: *TOPS/Watt (Higher is better).*

Figure 3.10 depicts the average of the energy-efficiency, measured in Tera Operations Per Second (TOPS)/Watt, for 8 DNN benchmarks with the normalized frequencies. TOPS/Watt for all the scheme are normalized to that of the Baseline-TPU at the baseline frequency (Baseline-TPU at 1.0x frequency delivers an energy efficiency of 53 TOPS/Watt). All the schemes have the same TOPS measure. However, TE-Drop has the lowest energy-efficiency due to its relatively high power footprint compared to other schemes. Owing to the clocking gating, EFFORT-24 boasts the highest energy-efficiency. EFFORT-24 de-

livers up to 1.06× and 1.35× better performance per unit power consumption, relative to Baseline-TPU and TE-Drop. Owing to lesser power consumption in EFFORT-8 and EFFORT-14 compared to EFFORT-24, a better TOPS/Watt performance is observed in these EFFORT variants. EFFORT-24 delivers up to 85% of the energy efficiency of an STC TPU. *Hence, EFFORT is a superior NTC TPU design paradigm, offering a high energy-efficiency while providing a high timing error resilience.*

3.5.4 Implementation Overhead

EFFORT incurs hardware overheads due to the clock gating circuit, and the CostCo logic added to each MAC. As the systolic array takes almost 24% of the TPU die area [68], EFFORT incurs an area overhead of only 5%.

In terms of gate count, a MAC unit without any enhancements uses 212 gates, while a MAC enhanced with CostCo uses 240 gates. The Clock Gate Unit is composed of 4 gates.

CHAPTER 4

RECLAIMING THE PERFORMANCE OF A NEAR-THRESHOLD TENSOR PROCESSING UNIT WITH SELECTIVE VOLTAGE BOOSTING

4.1 Background and Contributions of This Work

Rapid progress in the Artificial Intelligence (AI) realm has evidenced an emergence in the domain-specific AI architectures to sustain the colossal boom in the development of Deep Neural Network (DNN) applications. The rising efficiency of DNN accelerators further bears witness to the ongoing empirical advancements within the sphere of *Deep Learning* [12,85]. Google's Tensor Processing Unit (TPU), a systolic array of multiplier-and-accumulate (MAC) units, has been spearheading this race, by demonstrating a massive performance boost over the contemporary CPUs and GPUs [45].

However, the growth in AI processing is accompanied by a monumental increase in the power consumption as affirmed by carbon footprint for a single AI training [86]. To sustain the performance and trim the energy consumption, operating the TPU in the Near-Threshold Computing (NTC) realm is conceived for this research. Operating a TPU at NTC, benefits in a massive energy savings as the operating voltage is scaled close to the threshold voltage. However, extreme Process Variation (PV) sensitivity [87] introduced at NTC induces a high rate of timing errors which degrades the overall system performance [67], thereby reducing the energy-efficiency gains at low-voltage operation [6]. In this chapter, the significance of *undetectable timing errors* (Section 4.2.1) on the performance of an NTC TPU, is emphasized and the architectural homogeneity is exploited to detect and tackle timing errors, thereby presenting a reliable and energy efficient TPU design paradigm.

Razor is a very popular timing speculation methodology using a double sampling flip-flop to detect and correct timing errors [49]. Razor employs an instruction replay to correct

the erroneous computation. The tightly pipelined architecture of the TPU will result in inflating the execution time, thereby restricting the use of an instruction replay in a TPU. TE-Drop, a recently proposed technique prompts the MAC unit encountering timing error to steal a clock cycle from the downstream MAC to correct the timing error [15]. The erring MAC overrides the downstream MAC's output with its corrected value. However, TE-Drop does not address the timing errors which cannot be detected by Razor Flip Flop [49] which leads to the erroneous value to propagate down the output. Additionally, various timing error resilient schemes have been proposed for the processing architectures [88–104].

To overcome the limitations in existing schemes, a novel timing error prediction strategy based on the foreseeable timing error occurrence pattern in the TPU systolic array is proposed. The impacts of *undetectable timing errors* on the inference accuracy (Section 4.2.1) is observed. The mathematical analysis on the initially recorded timing error data is integrated with an efficient voltage boosting mechanism to propose PREDITOR—a novel low-power error-resilient TPU design paradigm to predict and mitigate timing errors.

This is the first work emphasizing the impacts of undetectable timing errors on the inference accuracy and presents a channel to limit its effects.

Following are the precise contributions of this work:

- A monumental increase of undetected timing errors at higher frequencies and ultra low NTC voltages is observed (Section 4.2).
- PREDITOR—a low-power TPU design paradigm that predicts and mitigates the timing errors over a range of operational cycles using an effective voltage boost mechanism (Section 4.3) is proposed.
- It is demonstrated that PREDITOR tackles up to ∼68% of the *undetectable timing errors*, thereby preserving the inference accuracy of the DNN datasets. PREDITOR offers 3x-5x better performance in comparison to TE-DROP and Modified Razor Flip Flop, with under 3% average loss in accuracy for 5 out of 8 DNN datasets and incurs an area and power overhead of ∼7.7% and ∼2.5% respectively (Section 4.5).

- Finally, it is illustrated that PREDITOR offers up to 87% energy efficiency gain in relative to a TPU operating in the Super-Threshold Computing (STC) realm, while utilizing only 14% of its power (Section 4.5).

4.2 Motivation

In this section, the correlation between the data flow and timing error emergence is investigated, and the concealed opportunity that can be utilized to tackle timing errors in a TPU systolic array is revealed. Section 4.2.1 provides the background of a TPU and elaborates the drawbacks for an NTC TPU operation. Section 4.2.2 introduces the architectural homogeneity of the TPU. The cross-layer methodology explained in Section 4.2.3 is used, to explore the timing error profiles illustrated in Section 4.2.4 and establish the need for a timing error prediction scheme in an NTC TPU.

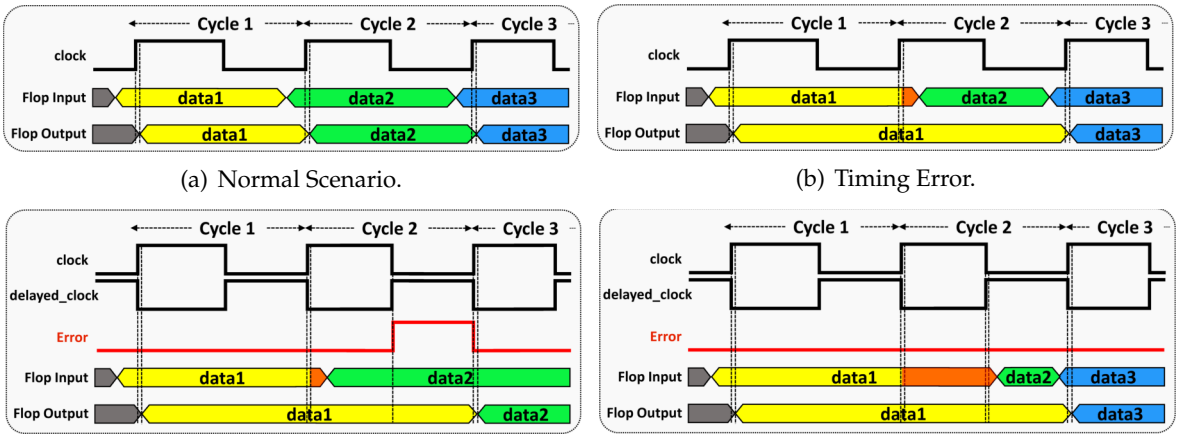
4.2.1 Background and Limitations

Systolic Array based DNN Accelerator

DNNs use multiple layers of computation to obtain the output inference. In each layer, activation matrix is multiplied with the weight matrix. TPU employs a systolic array of 256×256 MAC units to accelerate matrix multiplication [45]. The activation matrix and weight matrix both maintain an 8-bit precision. The weight matrices are pre-loaded into the MACs while the activation flows from left to right in successive clock cycles. The 24-bit precision accumulator output from each MAC moves downstream in each cycle.

Limitations to NTC Performance

Dynamic Scaling of Voltage (DVS) is a common practice used in the modern processing elements to yield significant power savings. The critical voltage is fixed at an optimum point to ensure correct operation of the processing element. However, aggressive scaling leads to an increase in the critical path due to the decrease in clock period of the operation.



(a) Normal Scenario.

(b) Timing Error.

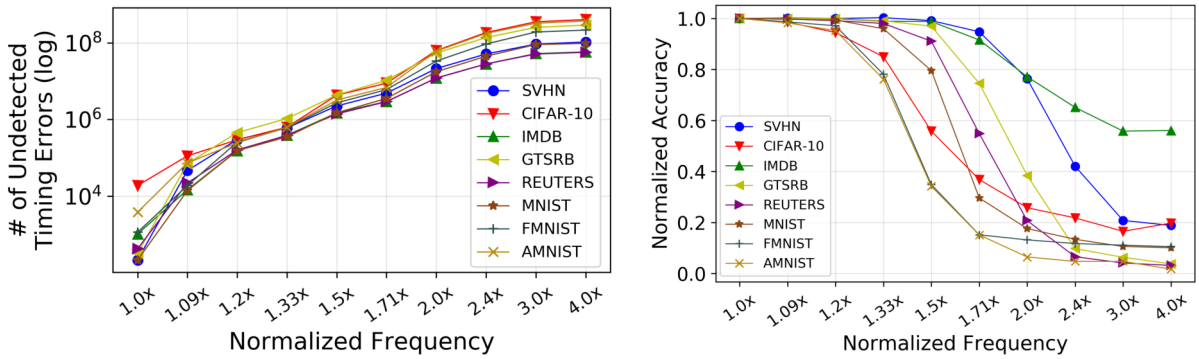
(c) Instant Replay with Razor.

(d) Undetectable Timing Error.

Fig. 4.1: *Figure depicts multiple data latching scenarios as the frequency of operation increases and the timing error detection window diminishes. In Figure 4.1(a) baseline frequency is in operation. Frequency of operation is same for Figures 4.1(b) and 4.1(c) but higher than Figure 4.1(a). Highest frequency of operation is employed in Figure 4.1(d).*

Figure 4.1(a) shows the typical scenario of an operation. The critical path is shorter than the clock period of an operation. In Figure 4.1(b), the frequency of the operation has increased to obtain better performance. During this scenario, the clock period of an operation decreases, leading to the delayed transitioning of the data (i.e., **data1** to **data2**). This phenomenon where the wrong data (i.e., **data1**.) is being latched onto the output instead of the correct data (i.e., **data2**.) is called a timing error [8, 87]. Timing speculation methods like Razor uses a double sampling flop to capture this delayed transitioning in data and employs an instant replay to latch the correct data onto the output [49]. Figure 4.1(c) depicts the operation of a Razor flop, where a delayed clock is employed by the shadow flop to detect the delayed transition. However, when the frequency of operation increases even further, the speculative window becomes too small even for the Razor flop to capture the delayed transition. Figure 4.1(d) elaborates the scenario when the delayed transitions cannot be captured any further. These extremely delayed transitions are the *undetectable timing errors*.

Figure 4.2(a) depicts the increase of the *undetectable timing errors* with an increase of frequency for 8 different DNN datasets. These undetectable errors propagate down the



(a) Undetectable Timing Errors.

(b) Normalized accuracies.

Fig. 4.2: *Figure 4.2(a) shows the rise in undetected timing errors with an increase in the operational frequency. The effects of the increasing timing errors is depicted in Figure 4.2(b), where the classification accuracy drops drastically. This experimentation is performed on the Baseline TPU (Section 4.5.1).*

systolic array resulting in an erroneous output, adversely affecting the system performance [6,67]. Figure 4.2(b) appropriately shows the drop in inferential accuracy as the operational frequency level is bolstered [15]. Especially, in case of DNN computations, where inference accuracy dictates the performance of system, the deterioration of classification accuracy renders the system inept to address the increasing performance needs.

4.2.2 Predictive Systolic Array Dataflow

There exists a strong interdependence between the data flow pattern inside the systolic array and number of timing errors occurring during the systolic array operation. Although the occurrence of timing errors are majorly dependent on the computational delays, the magnitude of timing errors per cycle of operation is dependent on the number of operations during the same period of clock cycles. Due to the disparate number of operations during every active execution cycle, the scope of timing errors varies but closely follows a foreseeable pattern. To further investigate this property, an exhaustive cross-layer methodology is employed, discussed next.

4.2.3 Methodology

A MAC unit is synthesized at NTC, using 15-nm FinFET library from NanGate [69]. VARIUS-NTV models are used to model the PV at NTC for FinFET [70]. For a conservative estimate, PV-induced delays in randomly chosen 2% of the gates in the circuit is considered. The in-house Statistical Timing Analysis (STA) tool to investigate the delay distribution of the sensitized paths for different inputs to the MAC unit. Additionally, the in-house TPU systolic array simulator is used to simulate the working of a TPU. Different levels of high degree timing error inducing input vectors are used with the TPU simulator to investigate the operational flow and timing error intricacies.

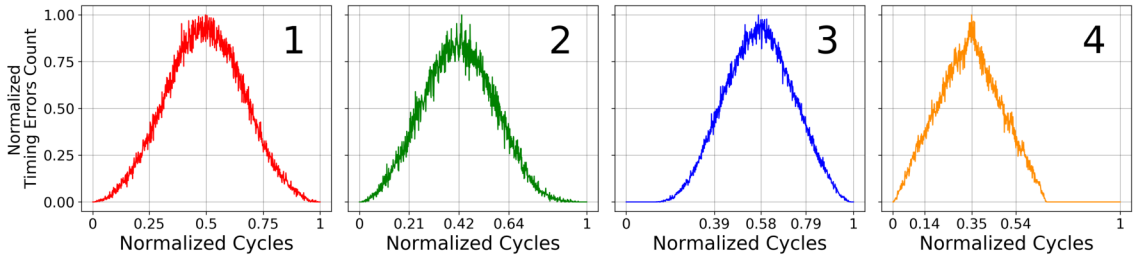


Fig. 4.3: *Figure 4.3 depicts the various timing error profiles during the high frequency operations of the TPU.*

4.2.4 Results and Significance

Figure 4.3 demonstrates the extensively observed timing error count variances in a systolic array operation. The Y-axes are normalized to the highest timing error in the respective plots. X-axes are normalized to the total number of operational cycles. *Plots(1-3)* in Figure 4.3 shows an exponential increase/decrease in the timing errors and *Plot 4* depicts a relatively linear rise/fall. The operational cycle in which the highest timing errors occur varies across the plots in Figure 4.3. For example, in *Plot 2* of Figure 4.3, highest timing error occurs at 42^{th} cycle, whereas for *Plot 3*, highest timing error occurs at 58^{th} cycle.

The results indicate a disparity in the timing error profiles, but all the plots have a symmetrical nature. *Plots 1-3* of Figure 4.3, *nearly exhibit a normal distribution curve charac-*

teristic, while *Plot 4* resembles a *triangular curve*. The symmetrical nature of the profiles is mainly attributed to the uneven number of MAC units being active, during each clock cycle of a systolic array operation. Utilizing this predictable timing error occurrence model, a prediction strategy is devised to significantly lower the timing errors in a TPU. With this insight, the scheme—PREDITOR is proposed, to mitigate up to *68%* of the timing errors by boosting the operating voltage during the high timing error occurrence cycles.

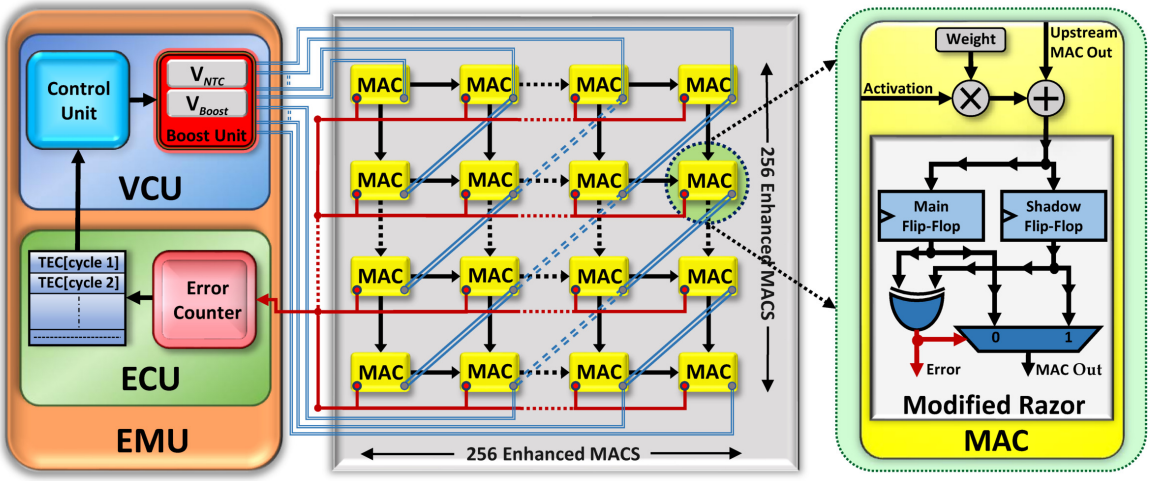


Fig. 4.4: *Each MAC in the systolic array is enhanced with an MRFF. EMU comprises of ECU and VCU. ECU collects and stores timing error count from each cycle. VCU queries the timing error information from ECU to predict the operational clock cycles and boost the operating voltage during the predicted cycle interval.*

4.3 Design

In this chapter, an *Analytical PREDIcTion based ErrOR Resilient TPU (PREDITOR)*, a novel low-power design paradigm is proposed to enhance the error resilience of an NTC TPU using mathematical analysis of the detectable timing errors. Section 4.3.1 outlines the design overview. Section 4.3.2 through Section 4.3.4 elaborates the components of PREDITOR.

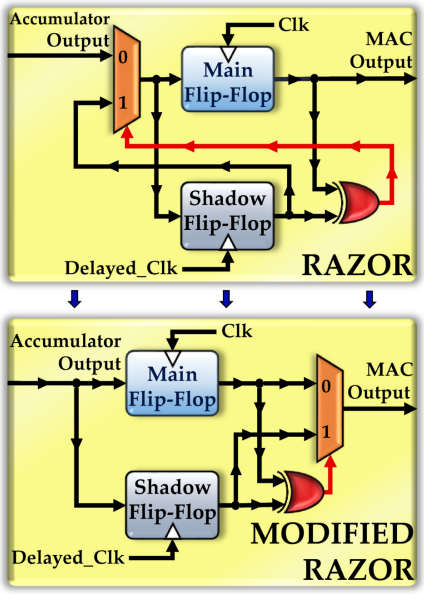
4.3.1 Design Overview

Figure 4.4 depicts the top level overview of PREDITOR. The prime enhancement of PREDITOR is the Error Management Unit (EMU). The main components of an EMU are Error Collection Unit (ECU) and Voltage Control Unit (VCU). Each MAC unit is augmented with a Modified Razor Flip Flop (MRFF) to detect and mitigate timing errors during the non-boosted cycles of operation. MRFF captures a timing error using a shadow latch and delivers an *error* signal to the ECU. Additionally, MRFF also provides the corrected value to the downstream MAC unit. ECU records the number of timing errors occurring during every operational clock cycle. VCU profiles and analyzes the timing error information from ECU once in every two clock cycles, to predict the operational clock cycle intervals which yields maximum timing errors. VCU will then boost the operating voltage of the MACs during the predicted intervals, to ensure an error free operation. Since the data flows in the systolic array as a diagonal wavefront, MACs along the diagonal (i.e., left bottom to right top.) will be active/inactive at the same time [45]. Hence, VCU will only boost the voltage of the MACs which are computationally active, thereby introducing a minuscule energy overhead.

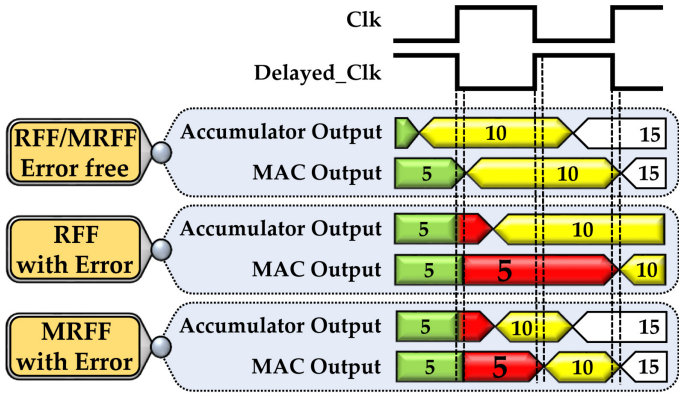
4.3.2 Modified Razor Flip Flop (MRFF)

MRFF detects and corrects timing errors without using any additional operational clock cycles. In a MAC unit, multiplier is sensitized only to the periodically changing activation input, as the pre-loaded weights remain unchanged throughout the entire systolic array operation. Additionally, the multiplier operation being computationally prolonged compared to the accumulation operation. It thus opens up a *timing aperture*, which can be exploited to override the previously transmitted erroneous value with the updated upstream MAC output. Figure 4.5(a) depicts the changes in the schematic between a regular Razor flip flop (RFF) and MRFF.

Figure 4.5(b) demonstrates the error handling capabilities of an RFF and MRFF for a column-wise systolic array operation. As depicted in Figure 4.5(b), whenever an RFF detects a timing error due to the delayed manipulation of the output data, an instant re-



(a) Razor Flip Flop to Modified Flip Flop.



(b) MRFF detects and corrects Timing Errors within a clock period.

Fig. 4.5: *Figure 4.5(a) depicts the conversion of Razor Flip Flop to Modified Razor Flip Flop by altering the multiplexer input/output connections. The timing error correction capability of MRFF is shown in Figure 4.5(b).*

play is initiated, which requires an additional clock cycle to correct the erroneous value. However, an MRFF will detect and correct the timing error within the same clock cycle, by opportunistically using the *timing aperture*.

However, MRFF cannot correct timing errors when the computational delays are beyond detection (Section 4.2.1). Hence, VCU will utilize the timing error information stored in ECU to predict the range of operational cycles to mitigate timing errors which is explained in Section 4.3.4.

4.3.3 Error Collection Unit (ECU)

ECU contains a 16-bit Error Counter and a content-addressable memory. During each operational clock cycle, ECU captures the timing errors from individual MAC units and these error signals are accumulated using the 16-bit Error Counter. Timing Error Count (TEC) (i.e., the total number of timing errors.) obtained during every clock cycle is stored in the consecutive locations of a content-addressable memory and successively updated at

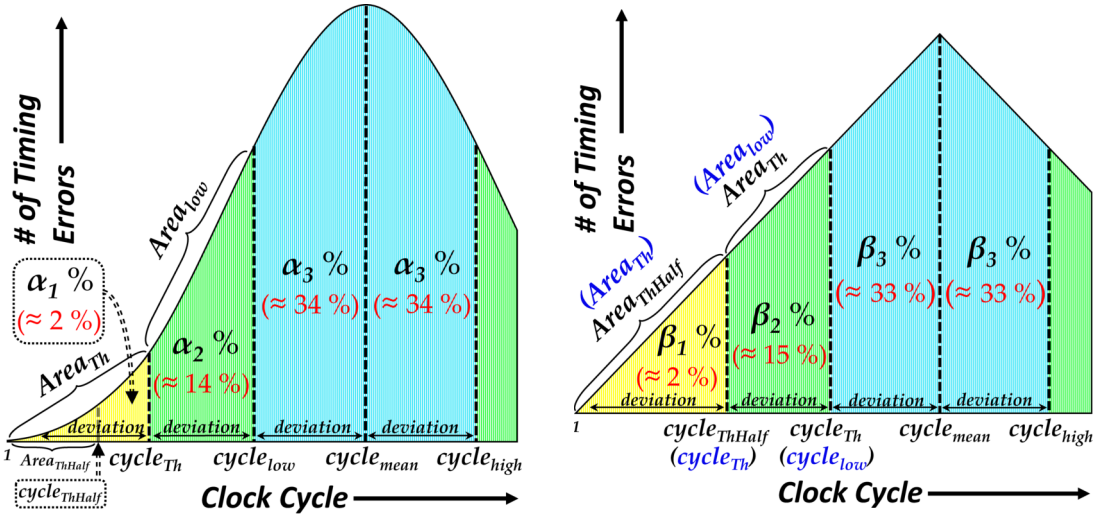
the end of every operational clock cycle.

4.3.4 Voltage Control Unit (VCU)

VCU houses the Control Unit (CU) and the Boost Unit (BU). CU computes the range of operational clock cycles for voltage boosting, by analyzing the cycle-wise TEC from ECU. BU boosts the supply voltage of MACs for the clock cycle interval predicted by the CU. Voltage boosting aids in mitigating both detectable and undetectable timing errors.

Control Unit (CU)

CU is responsible for predicting the operational clock cycle intervals yielding maximum timing errors. Based on the findings in Section 4.2.4, an algorithm modeled on the Normal Distribution Curve characteristic is developed. Since ∼68% area of a Normal Distribution curve is covered between the first standard deviations from the mean, by mathematical analysis, ∼68% of the timing errors are identified to be occurring within the ∼33% operational clock cycles centered around the mean clock cycle. Hence, PREDITOR targets to mitigate ∼68% of *undetectable timing errors*.



(a) Normal Distribution.

(b) Triangular Distribution.

Fig. 4.6: *Representative Timing Error Profiles.*

Algorithm 2 Boost Cycles Prediction Algorithm

$dim \leftarrow systolic\_array\_dimension$
$cycle_{ops} \leftarrow current\_operational\_clock\_cycle$
$total\_num\_cycles \leftarrow ((3 * dim) - 2)$
$cycle_{mean} \leftarrow (total\_num\_cycles \div 2)$
$cycle_{high} \leftarrow (4/3)\ cycle_{mean}$
$cycle_{low} \leftarrow (2/3)\ cycle_{mean}$
$cycle_{Th} \leftarrow (1/3)\ cycle_{mean}$
$cycle_{ThHalf} \leftarrow (1/2)\ cycle_{Th}$
procedure AREA_CALCULATION($init\_cycle, final\_cycle$)
    $Area \leftarrow 0$
    for $iterator \leftarrow (init\_cycle)$ *to* $(final\_cycle)$ **do**
        $Area \leftarrow Area + TEC[iterator]$
    end for
    return $Area$
end procedure
procedure CURVE_PREDICTION($cycle_{ops}$)
    if $(cycle_{ops} = cycle_{Th})$ **then**
        $CF \leftarrow\ Area_{Th} / Area_{ThHalf}$
        if $CF > ND_{thresh}$ **then**
            $Curve\_Dist_{coeff} \leftarrow ND_{coeff}$
            $boost\_initiate \leftarrow 1$
        else if $CF < ND_{thresh}$ & $CF > TD_{thresh}$ **then**
            $Curve\_Dist_{coeff} \leftarrow TD_{coeff}$
            $cycle_{low} \leftarrow cycle_{Th}$
            $cycle_{Th} \leftarrow cycle_{ThHalf}$
            $boost\_initiate \leftarrow 1$
        else
            $boost\_initiate \leftarrow 0$
        end if
    end if
end procedure
procedure BOOST_INTERVAL_COMPUTE($cycle_{ops}$)
    if $cycle_{ops} = cycle_{low} + 1$ **then**
        if $Area_{low} < (Curve\_Dist_{coeff} * Area_{Th})$ **then**
            $cycle_{Th} \leftarrow\ cycle_{Th} + 1$
            $cycle_{low} \leftarrow\ cycle_{low} + 1$
        else
            $deviation \leftarrow (cycle_{low} - cycle_{Th})$
            $cycle_{high} \leftarrow (cycle_{low} + (2 * deviation))$
        end if
    end if
end procedure

Figure 4.6(a) and 4.6(b) show the representative area-wise Normal and Triangular Distribution of a timing error profile. The boost cycles prediction procedures are elaborated in Algorithm 2. Based on the experimental analysis, it is determined that a rapid rise in the rate of timing errors results in the timing error profile exhibiting a Normal Distribution curve characteristic and a slower rise results in Triangular Distribution curve characteristic. Hence, to determine the nature of the timing error profile, a Curve Factor (*CF*) metric is proposed. CF is defined as the ratio of Areas between Threshold clock cycle ($cycle_{Th}$) and Half Threshold clock cycle ($cycle_{ThHalf}$) (line 18 of Algorithm 2) (Figure 4.6(a)). The values of $cycle_{Th}$ and $cycle_{ThHalf}$ are empirically ascertained based on earlier analysis.

To predict the boost intervals, the accumulated areas $Area_{Th}$ and $Area_{low}$ are compared. Areas (i.e., summation of TECs) between specific cycles is computed using an area calculation heuristic (lines 9 - 15 of Algorithm 2.). For example, $Area_{low}$ is summation of TECs from $cycle_{Th}$ to $cycle_{low}$. As new TECs are updated after every clock cycle, a balancing metric, $Curve\_Dist_{coeff}$ is used to dynamically balance the areas at specific clock cycle boundaries (i.e., $cycle_{Th}$ and $cycle_{low}$.). The empirically determined balancing metric for a Normal Distribution curve (ND_{coeff}) is expressed in equation 4.1. The cycles are adjusted dynamically (lines 35 - 36 of Algorithm 2) in case, the areas are unbalanced.

$$ND_{coeff} = \frac{\alpha_2}{\alpha_1} \tag{4.1}$$

For a Triangular Distribution curve, appropriate changes for cycles under consideration are made (lines 23 - 25 of Algorithm 2), as timing error counts at lower cycles needs to be analyzed. The empirically determined balancing metric for a Triangular Distribution curve (TD_{coeff}) is defined as:

$$TD_{coeff} = \frac{\beta_2}{\beta_1} \tag{4.2}$$

A very low *CF* evades the need for voltage boosting (line 28 of Algorithm 2). Whenever there is alteration in the central cycle (*Plot 3* of Figure 4.3), an increase in the timing error rate (not shown in Algorithm 2) is employed to dynamically calibrate the change.

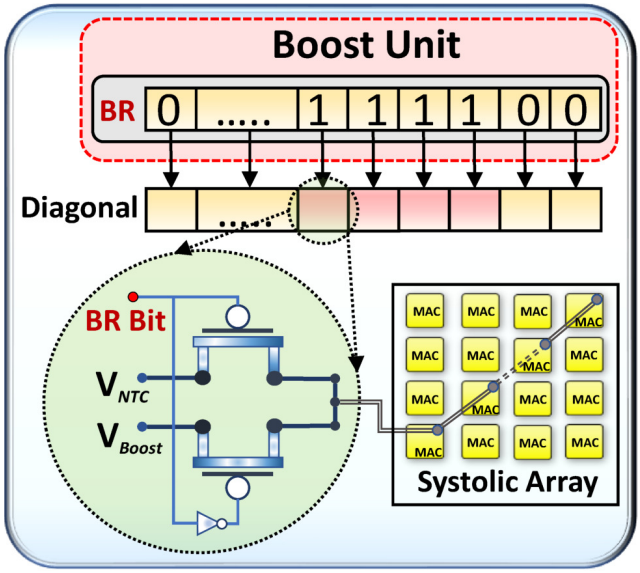


Fig. 4.7: *Interaction of BU with MACs along the diagonal.*

Algorithm 3 Voltage Boost Algorithm

```
1: procedure VOLTAGE_BOOST(cycle_ops, boost_initiate)
2:     if boost_initiate = 1 then
3:         if (cycle_ops ≥ cycle_low) || (cycle_ops ≤ cycle_high) then
4:             supply_voltage ← V_Boost
5:         else
6:             supply_voltage ← V_NTC
7:         end if
8:     end if
9: end procedure
```

Boost Unit (BU)

BU boosts the operating voltage of the MAC units to mitigate detectable and undetectable timing errors. As depicted in Figure 4.7, BU houses the Boost Register (BR), where each bit of this 511-bit register corresponds to a series of active MAC units along the diagonally active wavefront. A boosting technique proposed in [105] is employed by BU, where supply voltage to MAC units has two rails, V_{NTC} and V_{Boost}. V_{NTC} represents near-threshold voltage, set at 0.45V and V_{Boost} is the boost voltage, set at 0.6V. BU utilizes the procedure in Algorithm 3 to boost the supply voltage. Incorporating the booster infrastructure in [105], it is observed that switching between V_{NTC} and V_{Boost} can be procured

within a single cycle of operation. BU boosts the supply voltage to V_{Boost} and switches to V_{NTC} at the end of each boost cycle. To maintain a trade-off between boosting cycle intervals and the voltage boosting energy overhead, bits in BR are set empirically. This action is performed to lower the energy footprint in case, the magnitude of timing errors is significantly high in number. In such a scenario, MRFF will continue to handle the detectable timing errors. The experimental results and area/power overheads are discussed in Section 4.5.

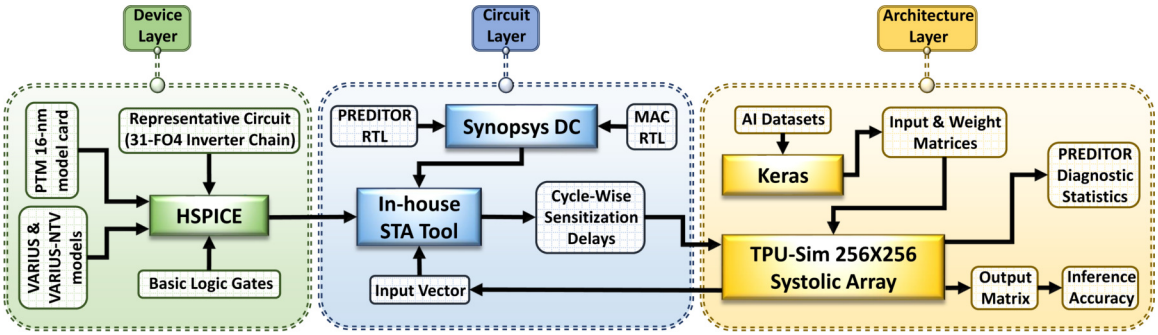


Fig. 4.8: *Cross Layer Methodology.*

4.4 Methodology

In this section, the comprehensive cross-layer methodology (as shown in Figure 4.8) used to implement the proposed design is described. The cross-layer methodology will aid in evaluating the proposed design's capabilities across DNN applications.

4.4.1 Device Layer

In order to estimate gate delay distributions, HSPICE models of basic logic gates (viz., NOR, NAND and Inverter) based on 16nm Predictive Technology Model are simulated. VARIUS-NTV model is considered to incorporate the impact of PV at NTC [8]. The impact of FinFETs is modeled using VARIUS-TC model [76]. To affirm sensitized path delay in a MAC unit, the delays of basic gates is employed in the circuit layer (Section 4.4.2).

4.4.2 Circuit Layer

A systolic array is implemented in Verilog RTL description and enhanced with PREDITOR components. RTLs are synthesized using Synopsys Design Compiler, to estimate area and power overheads. The sensitized delays are obtained for the MAC array using the in-house STA tool by employing real dataset driven inputs. By utilizing libraries of delay distributions for basic logic gates from HSPICE simulations (Section 4.4.1), STA tool provides the delays of sensitized paths in the MAC circuit.

Datasets	
Name	**Layer Architecture**
SVHN [77]	CONV: (32, 32, 3)x(32, 32, 32)x(32, 32, 32)x(14, 14, 64)x(14, 14, 64)x(5, 5, 128)x(5, 5, 128), FC: 512x512x10
GTSRB [78]	CONV: (3, 48, 48)x(32, 48, 48)x(32, 46, 46)x(64, 23, 23)x(64, 21, 21)x(128, 10, 10)x(128, 8, 8), FC: 2048x512x43
Reuters [79]	FC: 2048x256x256x46
IMDB [80]	CONV: 400x(400x50)x(398, 256), FC: 256x1
MNIST [73]	FC: 784x256x256x10
CIFAR-10 [81]	CONV: (32, 32, 3)x(32, 32, 32)x(32, 32, 32)x(16, 16, 64)x(16, 16, 64)x(8, 8, 128)x(8, 8, 128), FC: 2048x512x10
FMNIST [82]	FC: 784x256x512x10
AMNIST [83]	CONV: (20, 25, 1)x(20,25,128)x(20,25,64), FC: 32000x256x128x40

Table 4.1: *List of DNN datasets.*

4.4.3 Architecture Layer

The in-house cycle-accurate TPU systolic array simulator developed using C++, based on the TPU architectural details provided in [45] is employed for matrix multiplication. To accurately model timing errors resembling real-time sensitized path delays in MAC units, the STA tool (Section 4.4.2) is incorporated into the TPU Simulator. A real-life inference engine is replicated by interfacing Keras [84] with the TPU simulator. Using Keras and employing TensorFlow in the backend multiple DNN applications as mentioned in Sec-

tion 4.5.3 are trained. Table 4.1 provides the size and layer information for the 8 DNN datasets. A filter size of *(3,3)* and default stride of *1* is used for all datasets. Additionally, *padding* is also added to the input feature map. The DNN datasets include representations from different Neural Network domains (Image Recognition, Speech, Audio etc.). The diverse range of datasets aids in exploring the efficacy of PREDITOR in handling the timing violations, arising due to the wider set of variations in the activation patterns. The trained models have varying input sizes and utilize the state-of-art layers like dropout and batch normalization. The trained model weights and the activations from each layer are extracted into 256×256 8-bit integer matrices. The TPU simulator performs the matrix multiplication operation for each pair of activation and weight matrices. The resulting output matrices are combined together to determine the inference accuracy.

4.5 Experimental Results

In this section, the effectiveness of different timing error resilient schemes are evaluated by employing a TPU in NTC operating conditions. The baseline operation condition of the NTC TPU is (0.45V, 67.5MHz) to ensure error-free systolic array operations. Section 4.5.1 describes the different comparative schemes. Section 4.5.2 presents the error resilience of PREDITOR. Section 4.5.3 elaborates the inference accuracies. Section 4.5.4 presents the energy efficiency of PREDITOR. Section 4.5.5 discusses the area and power overheads.

4.5.1 Comparative Schemes

- **Baseline TPU (B-TPU) :** This technique does not employ any timing speculation methodologies and propagates the erroneous values down the systolic array computation stages [50].
- **TE-DROP (TED) :** In this scheme, MAC encountering timing error recomputes the correct value by borrowing a clock cycle from the downstream MAC. Downstream MAC effectively annuls its operation and procures the recomputed upstream MAC output onto the next stage [15].

- **MRFF :** This scheme exploits the *timing aperture* to drive the delayed output onto the downstream MAC. Delayed output beyond detection, results in an erroneous value being propagated down the systolic array.

- **PREDITOR (PRED) :** This is the proposed scheme which uses the timing error information, obtained using timing speculation mechanism to predict and mitigate eminent timing errors by boosting operating voltage of the MAC units for a definite period of operation (Section 4.3).

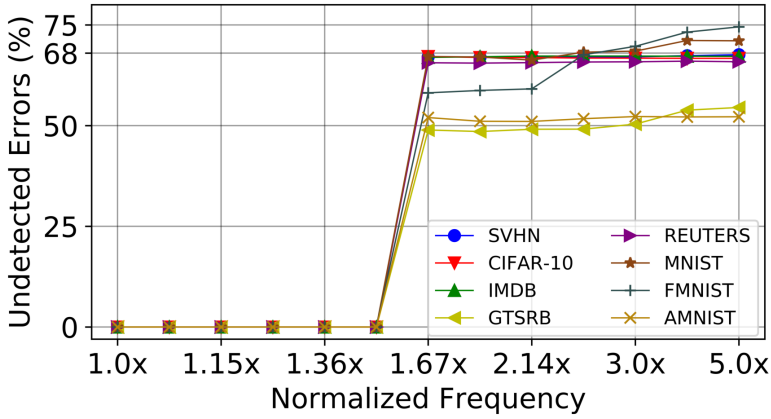


Fig. 4.9: *Percentage of undetected timing errors mitigated by PREDITOR for 8 different datasets.*

4.5.2 Error Resilience

Figure 4.9 demonstrates the number of undetected timing errors mitigated by PREDITOR when the TPU is operated at higher frequencies compared to the baseline frequency of operation. The Y-axis represents the percentage of undetected timing errors effectively handled by PREDITOR. The operating voltage is kept constant at 0.45V for all frequencies denoted by the X-axis. The X-axis is normalized to the baseline frequency of operation. Up to 1.44x the baseline frequency, detectable timing errors are effectively handled by MRFF. Hence, the effect of timing errors on the inference accuracy is negligible, as explained in

Section 4.5.3. PREDITOR's voltage boosting mechanism becomes eminent after 1.44x the baseline frequency, thereby correcting the undetectable timing errors. Meanwhile, ∼68% of *undetectable timing errors* are mitigated for six out of eight DNN datasets, as evidenced in Figure 4.9, validating the expectation elaborated in Section 4.3.4.

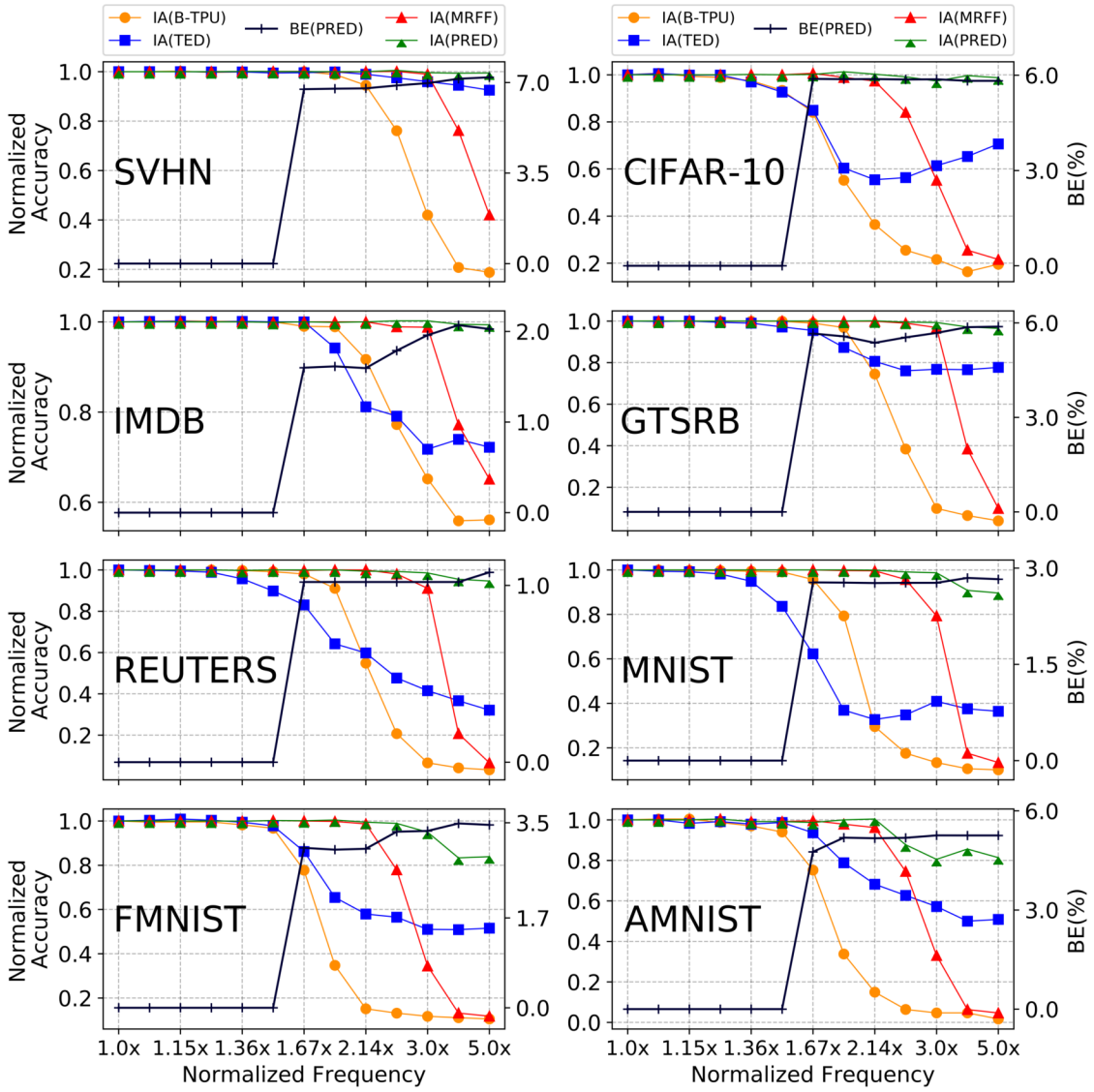


Fig. 4.10: *Normalized Inference Accuracy (IA) of different comparative schemes and Voltage Boost Energy (BE) of Preditor for 8 DNN datasets at various normalized frequencies.*

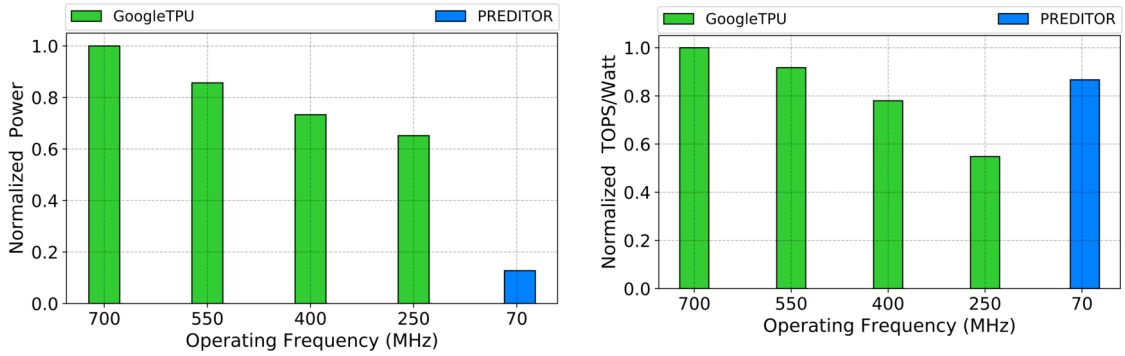
4.5.3 Inference Accuracy and Voltage Boost

Figure 4.10 depicts the variations in normalized inference accuracies for 8 different datasets with various error resilient schemes. Y-axes are normalized to error free inference accuracy at the baseline frequency for all datasets (Error free accuracy for datasets are SVHN: 0.94 [77], CIFAR-10: 0.77 [81], IMDB: 0.89 [80], GTSRB: 0.97 [78], REUTERS: 0.80 [79], MNIST: 0.98 [73], FMNIST: 0.89 [82], AMNIST: 0.92 [83])). Figure 4.10 also depicts the Voltage Boost Energy (BE) of PREDITOR across all frequencies of operation. BE is computed as percentage of energy consumed by an NTC TPU operating at baseline conditions, without any voltage boosting mechanism.

Consistent error resiliency is provided by all schemes up to 1.36x the baseline frequency. As the performance is increased, PREDITOR surpasses B-TPU, TED and MRFF by providing an appreciably better accuracy. PREDITOR mitigates more timing errors at higher frequencies due to a superior prediction engine. The number of undetected timing errors during non-boosted cycles for AMNIST and FMNIST are relatively high, resulting in a drop of accuracy for PREDITOR. A large number of MAC operations being bypassed due to higher timing errors contributes to a deterioration of accuracy for TED. MRFF encounters a sudden fall in accuracy at higher frequencies due to an increase in undetected timing errors. Hence, PREDITOR enhanced NTC TPU contributes to an accuracy loss of only 3% in five datatsets, when operated up to 5$\times$ the baseline frequency.

Voltage boosting from PREDITOR is not utilized until 1.67x the baseline frequency due to a lower number of timing errors which are corrected by MRFF. As frequency is scaled beyond 1.67x, nearly all datasets witness an identical rise in BE, as voltage boosting across a period of operational cycles becomes imminent. BE for four out of eight DNN datasets (SVHN, CIFAR-10, GTSRB and AMNIST) is relatively higher at $\sim$7%, compared to other datasets, as significant number of timing errors occurring during initial operational cycles compels PREDITOR's prediction mechanism to boost a relatively higher number of operational cycles. However, number of boosted cycles are relatively small compared to the entire systolic array operation cycles.

4.5.4 Is NTC TPU worth it ?



(a) Power Consumption at different frequency points. (b) Tops/Watt comparison at different frequency points.

Fig. 4.11: *Power and performance comparison of PREDITOR with GoogleTPU.*

Figures 4.11(a) and 4.11(b) presents the average power consumption and energy efficiency, measured in Tera Operations Per Second (TOPS)/Watt of a GoogleTPU for 8 DNN datasets, as it scaled from STC to NTC region. The Figures also demonstrate the power consumption and energy efficiency of PREDITOR operating at NTC. Power consumption and TOPS/Watt metrics at different frequencies is normalized to that of the GoogleTPU operating at STC (i.e., 700 MHz.). A steady decline in power is noted for GoogleTPU as the operating voltage is reduced along with the operating frequency. Power consumption of PREDITOR is significantly lower than GoogleTPU as it is operated in the NTC region. As the GoogleTPU is scaled below the STC region, timing errors appear in the system, leading to a deterioration in the performance [15]. However, an NTC TPU equipped with PREDITOR can effectively handle timing errors for more than 3× its baseline frequency (Sections 4.5.2 and 4.5.3), thereby providing an efficient performance per unit consumption. *PREDITOR working at NTC delivers up to* 87% *energy efficiency gain of a GoogleTPU operating at STC* (Figure 4.11(b)), *while consuming only* 14% *of its power* (Figure 4.11(a)). *Hence, PREDITOR validates to be a low-power and error-resilient design paradigm, offering a high performance in an NTC TPU.*

4.5.5 Hardware Overheads

The area and power overhead incurred by PREDITOR are ∼7.7% and ∼2.5% respectively. Area overhead in PREDITOR is due to inclusion of the MRFF into each MAC unit and EMU into the TPU systolic array. Power overheads obtained are in comparison to the error-free operation of a baseline NTC TPU.

CHAPTER 5

RECLAIMING THE RELIABILITY OF A NEAR-THRESHOLD TENSOR PROCESSING UNIT BY DELAY FAULT DETECTION AND MITIGATION

5.1 Background and Contributions of This Work

The emergence of Deep Neural Networks (DNNs) and their growing usage in diverse applications has led to a rapid advancement in the development of Application Specific Integrated Circuit (ASIC) architectures for the Artificial Intelligence (AI) computing realm. A Tensor Processing Unit (TPU) is one such ASIC, developed by Google and has been deployed in their datacenters for the inference phase of the DNN computation [68]. With the rapid deployment of AI hardware at the edge, there is a tremendous need to investigate these circuit-architectures in the Low-Power Computing (LPC) region [106].

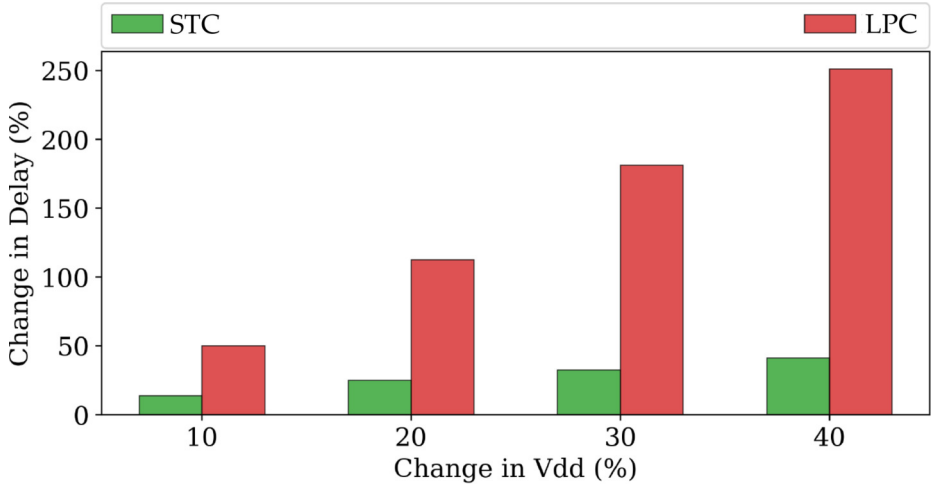


Fig. 5.1: *Delay sensitivity for power supply at STC vs LPC. HSPICE simulations shown for a 31 fan-out-of-four (FO4) inverter chain at the 14nm multi-gate technology node [107].*

Figure 5.1 demonstrates a key challenge in realizing AI hardware at the edge: a massive delay variance with voltage at LPC in comparison to Super Threshold Computing

(STC). The figure shows a FO4 inverter delay characteristics at these two regions, when the supply voltage is varied by the same percentage variation of the respective voltage domains. For example, a 40% voltage variation can cause a 45% delay variation at STC, and a huge 250% delay variation at LPC. Due to these extreme sensitivities, the omnipresent manufacturing Process Variation (PV) in the devices can realize gate delays up to 20× of the nominal values in the LPC region [87]. Collectively, a small set of PV-affected gates, in combination with a minute variation in operating condition (e.g., voltage), can completely alter the critical path of the circuit and protract the combinational delay, which subsequently leads to frequent timing violations. These delay faults are termed as Low-Power Faults (LP-faults) in this paper.

LP-faults may remain benign and exhibit the correct operation at nominal voltages. However, they are exposed and cause frequent timing faults at LPC. The process of frequency guard-banding which works efficiently at STC, becomes highly ineffective at LPC due to the extreme variation in delays. In a tightly pipelined architecture such as TPU with a large number of interconnected Multiplier-and-Accumulate (MAC) units, detection of an LP-fault in an individual MAC unit can be a formidable challenge. Furthermore, as these faults manifest only after fabrication and at certain operating conditions, detecting and correcting these faults become even more challenging. In this chapter, the damage an LP-fault can induce on a DNN inference operation is analyzed.

Interestingly, many modern datasets used in DNN computations exhibit a plethora of zero weights [12,13], as well as, zero activation elements. In conjunction with the perceived resilience of DNN software from errors, it is intuitive to expect that DNN inference may be inherently tolerant to LP-faults [67]. However, the rigorous cross-layer analysis utilized in this chapter reveals otherwise. For example, an otherwise innocuous zero result from a MAC can lead to non-zero outcome under an LP-fault, causing a havoc in the inference accuracy. Not only is it found that inference accuracy can drop dramatically from LP-faults, the results show extreme unpredictability in these inference drops, demonstrating a critical need to rigorously analyze and mitigate LP-faults for successful deployment of

these AI hardware accelerators at the edge.

Razor is a popular technique which uses a double sampling flip-flop to detect a timing error in a pipelined circuit and employs an instruction replay to recompute the erroneous data [49]. However, replaying an instruction in a TPU pipeline requires stalling of the entire systolic array operation, incurring a massive loss in throughput for such a data-parallel architecture. *This is the first work to introduce a post-fabrication fault detection technique to identify the faulty MAC units in a TPU.*

The specific contributions of this chapter are as follows:

- The impact of delay faults in a systolic array of MAC units is explored. The problem an LP-fault can pose in transforming a zero output from a multiplier to an incorrect non-zero value is investigated (Sections 5.2.2 and 5.2.3). Additionally, this research also shows how a subset of zero computations in a DNN matrix multiplication magnifies this threat (Section 5.2.3).
- STRIVE—a low-overhead faulty MAC detection technique for a TPU systolic array (Section 5.3.5), to identify the PV-affected MAC units and timing error resilience techniques to mitigate the effect of LP-faults, is introduced (Sections 5.3.4 and 5.3.6).
- Finally, it is demonstrated that STRIVE incurs less than 1% loss in inference accuracy for 6 DNN benchmarks in a TPU affected by a gate level fault rate of 1% (Section 5.5.2). Additionally, STRIVE gives 1.8× and 1.3× better performance per unit power than Fault-Aware Pruning [108].

5.2 Motivation

In this section, a post fabrication phenomenon which poses a severe threat to the error resilience of DNNs is uncovered. Moreover, a demonstration is performed to uncover the threat posed by a multiplier unit, when an expected zero computational output results in an unpredictable non-zero value. Sections 5.2.1 and 5.2.2 provide a background of the TPU and LP-faults, respectively. Sections 5.2.3 and 5.2.4 elaborate the results and the significance of the demonstration.

5.2.1 TPU Systolic Array

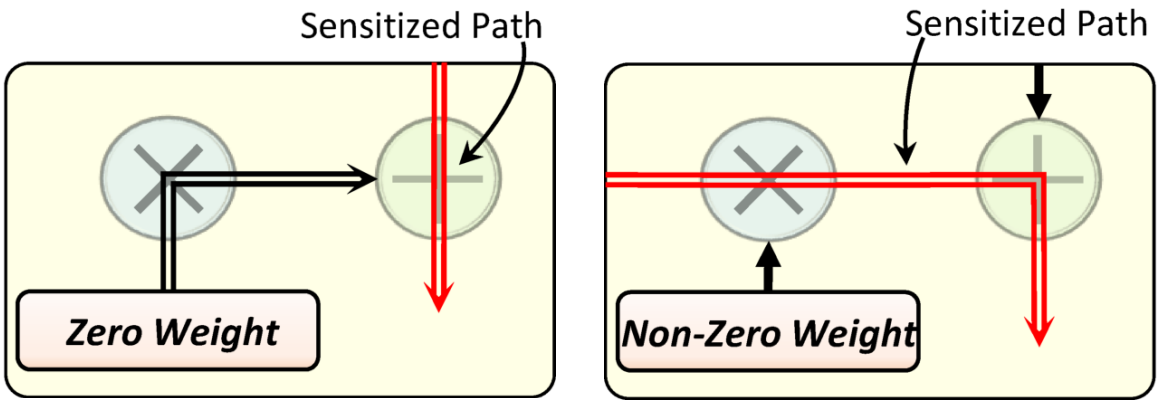
DNNs utilize multiple layers of computations during the inference stage. The multiple layers of DNNs are translated into a matrix multiplication of the activation input, with the weight matrix. A TPU employs a systolic array of $n \times n$ MAC units to expedite the matrix multiplication operations. Figure 5.5 shows a TPU–the yellow array, where each yellow box is a MAC unit comprising a multiplier and an accumulator. Activation inputs are stored in a unified buffer and subsequently streamed across the MAC array, while the weight matrices are pre-loaded into the MAC units. The activation inputs and stationary weights maintain an 8-bit precision.

5.2.2 Impact of PV at Low-Power: LP-faults

PV sensitivities experienced at STC are severely exacerbated at LPC. Therefore, a small effect of PV in a post fabrication circuit can produce a substantially large delay variation, when the operating region is switched from STC to LPC. Although the actual variation in physical dimensions (i.e., $gate$ length or t_{ox} length) remains identical, the observed variation will be in the transformed or elongated delays. Hence, any sensitized circuit path that exceeds the guard-banded clock period leads to a timing violation. These faults are termed as ***LP-faults*** or ***Delay Faults***. LP-faults can be completely hidden at STC, as the timing guard-band essentially covers up the relatively smaller delay variations (Figure 5.1). As the operating region is moved to LPC, the voltage and frequency are lowered appropriately, and a suitable timing guard-band is applied. However, as the delay variations are significantly higher at LPC, the prolonged combinational delay due to PV will exceed the timing guard-band and trigger an LP-fault. Furthermore, as factors leading to LP-faults emanate during fabrication, their effects can be perceived only during the real-time working environment.

Threats due to LP-faults

Effect on Inference Accuracy: In a TPU systolic array (at every clock cycle), the output of



(a) Zero Weight—Masks LP-fault.

(b) Non-zero Weight—Exposes LP-fault.

Fig. 5.2: *Sensitized paths in a MAC unit for zero and non-zero weights, respectively.*

each MAC unit, is added to the resulting activation and weight product of the consecutive downstream MAC unit. Hence, the accumulated product from the lowermost row forms a single element of the output matrix. However, as the systolic array is operated at LPC, the effects of PV in a MAC unit can instigate a delay fault and produce an erroneous output. As the PV-affected gates are distributed across the systolic array, more and more MAC units will be rendered faulty. Consequently, a large magnitude of LP-faults will increase the propagation of erroneous outputs and eventually incorrect values will be stored in the output matrix (later shown in Figure 5.8). Therefore, inference accuracy processed using the erroneous values will be significantly lowered.

Significance of Zero Activation: Figures 5.2(a) and 5.2(b) compare the sensitized paths in a MAC unit between a zero weight and a non-zero weight. A zero weight in the MAC unit drives the output of the multiplier to zero for the entirety of the matrix multiplication operation, thus sensitizing the second accumulator input path (i.e., output from the upstream MAC) to the output. Eventually, the upstream MAC output is forwarded to the MAC unit in the immediate lower row. Hence, a zero weight masks the LP-faults. However, for a non-zero weight, the circuit path from the activation input will be sensitized. *Consequently, even for a zero activation input, a prolonged delay needed to stabilize the faulty multiplier output can eventually expose an LP-fault, thereby resulting in a non-zero output.* Such computations are termed as *Fault Prone* zero computations.

The motivational data to demonstrate the serious effects of LP-faults is presented by employing a rigorous cross-layer methodology as described in Section 5.4.

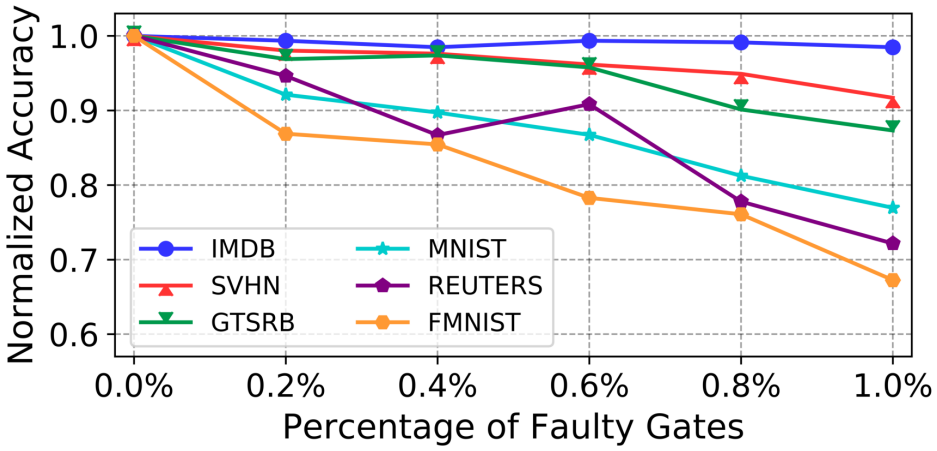


Fig. 5.3: *Increasing number of faulty gates increases the magnitude of LP-faults, thereby deteriorating the inference accuracy.*

5.2.3 Results

Figure 5.3 depicts a drop in the inference accuracy when the percentage of faulty gates increases in a TPU systolic array, for 5 out of 6 DNN benchmarks. The increase in the LP-fault level challenges the inherent timing error tolerance of DNNs [67], thereby dwindling the DNN inference accuracy. An outlier benchmark is IMDB, where the significant number of zero weights sensitizes the accumulator paths to the output (case of Figure 5.2(a)), consequently reducing the damaging effects of LP-faults.

Figure 5.4 shows the percentage of zero computations and *Fault Prone* zero computations for 6 different benchmarks. From Figure 5.4, it is evident that even though more than 80% of the computations involve zero computations, more than 20% of the zero computations are *Fault Prone* in 5 out of the 6 DNN benchmarks. Therefore, even a zero computation in a PV-affected multiplier can pose a considerable threat and contribute in lowering the inference accuracy.

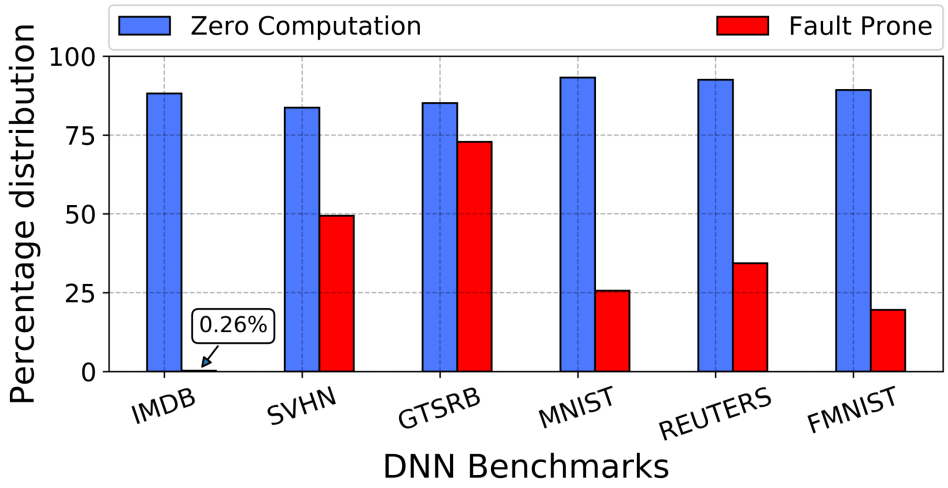


Fig. 5.4: *Zero computations and Fault Prone zero computations elaborated as a percentage of total computations for 6 different DNN benchmarks.*

5.2.4 Significance

The findings demonstrate that *the effects of PV at LPC can lead to significant deterioration of the DNN inference accuracy*. Additionally, the study is this work shows that even a predictable zero computation is not safe from the threat of an LP-fault. Since the phenomenon leading to a delay fault is born during fabrication, identical chips can exhibit different variations in the sensitized paths. As AI Edge computing is migrating more towards LPC, the emergence of delay faults can severely hamper the DNN predictions. *Hence, a runtime mechanism is needed to identify the faulty MAC units when a systolic array is operated at LPC and an efficient design paradigm to reduce the effect of LP-faults*. Since a zero activation input can be identified due to the redundant bit pattern and the output of a MAC unit for a zero activation is predictable, such characteristics can be doctored for the benefit. With this premise, in the next section, the proposed scheme—STRIVE, to detect and handle LP-faults affected MAC units is explored.

5.3 Strive Design

In this section, STRIVE—the novel design paradigm to detect and mitigate the effect of delay faults in an LPC TPU is discussed. STRIVE uses a low-overhead *post-fabrication*

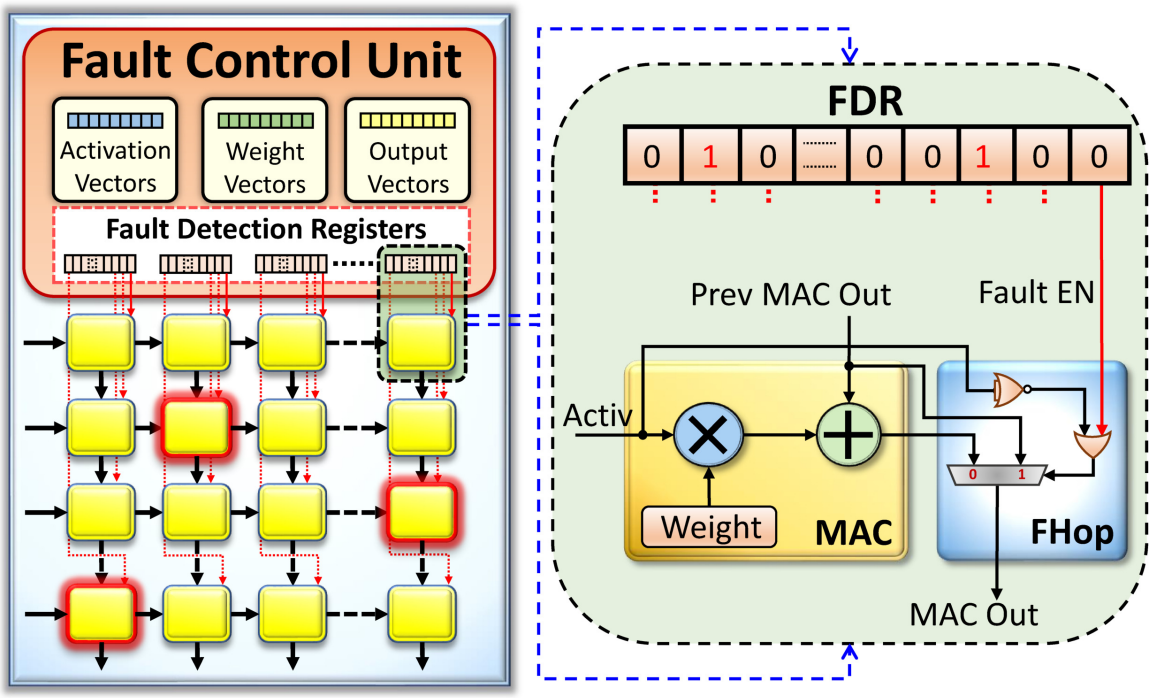


Fig. 5.5: *Design block and dataflow of STRIVE.*

faulty MAC detection technique to identify the error-prone MACs. Furthermore, by inferring the location of faulty MACs, a *position-aware timing error mitigation* scheme to tackle an LP-fault is devised and employed. The challenges and overview of STRIVE are described in Sections 5.3.1 and 5.3.2. The threat posed by an LP-fault is explained in Section 5.3.3. Sections 5.3.4 through 5.3.6 elaborate the design components in detail.

5.3.1 Challenges

In this section, the problems which need to be addressed by STRIVE to effectively counter the threat posed by LP-faults are highlighted.

1. A faulty MAC unit is susceptible to *Fault Prone* zero computation. Hence, a MAC unit needs to be equipped to handle this scenario (addressed in Section 5.3.4) .

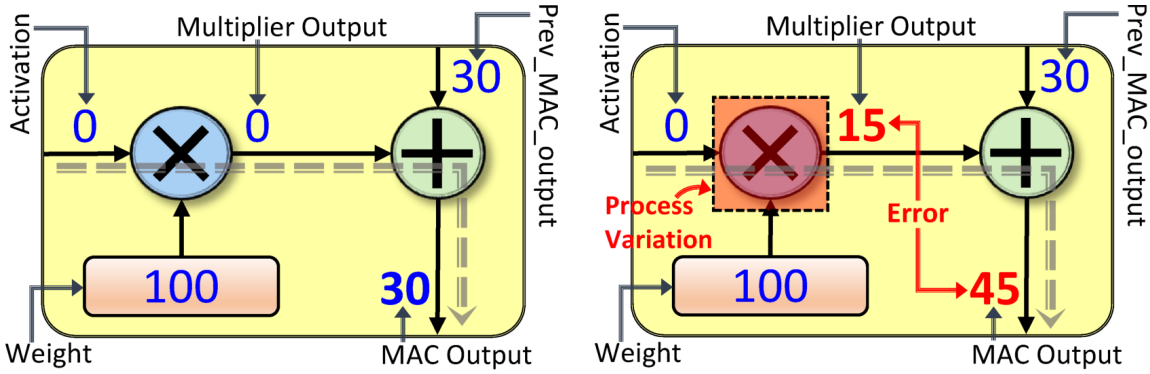
2. Diagnosing faulty MAC units spread across the TPU systolic array (addressed in Section 5.3.5).

3. Reclaiming the DNN inference accuracy from an LPC TPU housing PV-affected MAC units (addressed in Sections 5.3.5 and 5.3.6).

5.3.2 Design Overview

Figure 5.5 depicts the top-level overview of STRIVE. The LPC TPU is enhanced with a Fault Control Unit (FCU) and each MAC unit is augmented with Fault Hop (FHop). Initially, the FCU generates the activation and weight matrices, and the output map. Later, the FCU compares the TPU-generated output matrix and the output map to deduce the location of PV-affected MACs (Section 5.3.5). The faulty MAC locality is later exploited by the FCU, to tackle the LP-faults, using FHop (Section 5.3.5).

Additionally, an alternate technique Fault Hop Time-Borrow (FHop-TB), to mitigate the effects of an LP-fault, by enhancing the design of FHop, is discussed (Section 5.3.6).



(a) Fault free MAC. (b) PV-affected MAC.

Fig. 5.6: *Working of a PV-free and PV-affected MAC.*

5.3.3 Illustrative example for an LP-fault

Figures 5.6(a) and 5.6(b) describe the operations of a fault-free and faulty MAC, respectively. For a fault-free MAC (Figure 5.6(a)), the multiplier concludes its computation within the clock period and delivers an error-free output, thereby leading to a correct output from the MAC unit (i.e., ***30***). However, the significant delay stemming from the

PV-affected multiplier in a faulty MAC (Figure 5.6(b)), instigates an LP-fault. Hence, an incorrect value will be processed at the multiplier output (i.e., ***15*** in Figure 5.6(b)) and an erroneous value (i.e., ***45***) will be delivered to the next logic stage. So, a faulty MAC unit needs to be protected from a zero activation input, which is discussed next.

5.3.4 Fault Hop (FHop)

In this section, FHop is introduced. As a zero activation input produces a zero computation, the MAC operation for a MAC unit can be entirely skipped. The MAC unit is augmented with a NOR gate, an OR gate and a multiplexer (MUX), as demonstrated in Figure 5.5. The MUX output is controlled either by the NOR gate (viz., the zero activation input) or the *Fault EN* signal from FCU (discussed in Section 5.3.5). Thus, for a *zero activation input or when a Faulty EN signal is set*, the erroneous MAC operation is bypassed and the accumulator input is directly presented to the MAC output through the MUX. Hence, *FHop prevents a possible Fault Prone zero computation and aids in the identification of faulty MACs* (discussed in Section 5.3.5).

5.3.5 Fault Control Unit (FCU)

FCU houses the Fault Detection Registers (FDRs) along with the fault detection vectors. One FDR is dedicated to each column of the systolic array (Figure 5.5). Each bit of the FDR is labeled as a *Fault EN* signal and maps to a corresponding MAC unit in that column. The FCU operates in two modes: (a) *Fault Detection Mode*, to determine the faulty MACs, and (b) *Fault Resilient Mode*, which is the nominal operating mode of the TPU.

In *Fault Detection Mode*, all the bits of the FDR are reset to zero, to skip a MAC operation only for a zero activation input. The FCU later sets the FDR bits for all the faulty MACs. Hence, during the *Fault Resilient Mode*, the errant MAC operations are also skipped.

Next, the two operating modes of the FCU are discussed.

Fault Detection Mode

For illustration purposes, the detection technique is demonstrated for a 3×3 systolic array, as the same technique will be scaled up to any dimension of the systolic array.

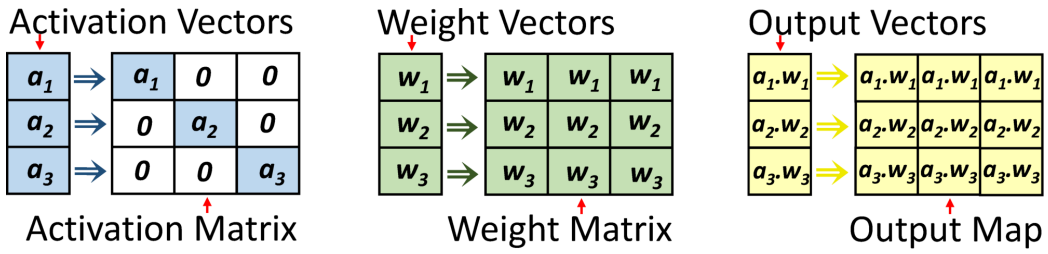


Fig. 5.7: *Development of input matrices and output map—correct outputs—by FCU.*

Matrix and map—correct outputs—generation

Figure 5.7 depicts the low-overhead matrix/map generation methodology. The FCU generates the activation matrix by allocating the individual activation vectors along the diagonal and zero value for the non-diagonal entries, thereby creating a diagonal matrix. For the weight and output matrix, all the elements in a row are assigned the same weight and output vector. However, each row is allocated a different vector.

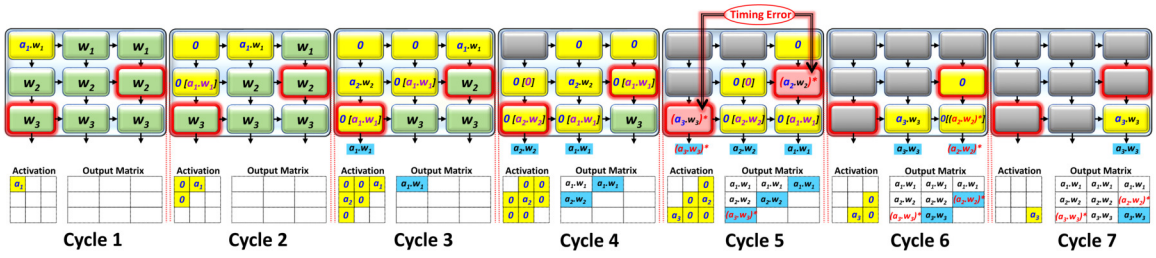


Fig. 5.8: *Green MACs are yet to receive the activation, yellow MACs are in operation and faulty MACs are highlighted with a dark red outline. Red MACs denote the occurrence of a timing error. Grey MACs have completed their execution. Only the final operation from each MAC is shown on the yellow MACs for space constraints. "0[$a_n.w_n$]" operation on a yellow MAC indicates that the activation input is "0" and accumulator input "$a_n.w_n$" will be forwarded to the next stage.*

Faulty MAC detection

Systolic Array Operation with Faulty MACs : Figure 5.8 demonstrates the cycle-wise accurate matrix multiplication between the activation and weight matrices for the fault detection. The systolic array is pre-loaded with the weight matrix, while the activation matrix is transposed and streamed to the individual rows (i.e., from left to right). Activation table and the Output Matrix table shown below the systolic array in Figure 5.8, coherently maps the activation input being streamed to the corresponding MAC units and the output accumulated from the last row of the systolic array in the respective clock cycles.

- In Cycle 1, activation a_1 is multiplied with weight w_1 in the only active MAC unit.
- In Cycle 2, zero activation values are streamed to the first and second row. The activation a_1, from Cycle 1, is forwarded to the successive column (i.e., second column) of the first row. As the MAC units are enhanced with FHop, the active MAC unit in the second row encountering the zero activation will skip the MAC operation and forward the upstream MAC output (i.e., $a_1.w_1$) to the downstream MAC unit.
- Nominal operation continues in Cycles 3 and 4, as non-zero activation has not reached the faulty MACs.
- However, in Cycle 5, activation inputs a_2 and a_3 reach the faulty MAC units in rows two and three, respectively. As the non-zero activation inputs are multiplied by the respective weights, the extended combinational delay will trigger a timing error and erroneous values (i.e., $(a_2.w_2)^*$ and $(a_3.w_3)^*$) are generated. The erroneous value from the faulty MAC unit in the third row (i.e., $(a_3.w_3)^*$) is stored in the output matrix.
- In Cycles 6 and 7, the erroneous value $(a_2.w_2)^*$ along with other output entries are accordingly forwarded and stored in the output matrix.

Detection of Faulty MACs : Figure 5.9 presents the process of detecting the Faulty MACs, using the output matrix. FCU compares the respective entries of the output matrix with the output map (Figure 5.7) and determines the location of faulty MACs. FCU thus sets the

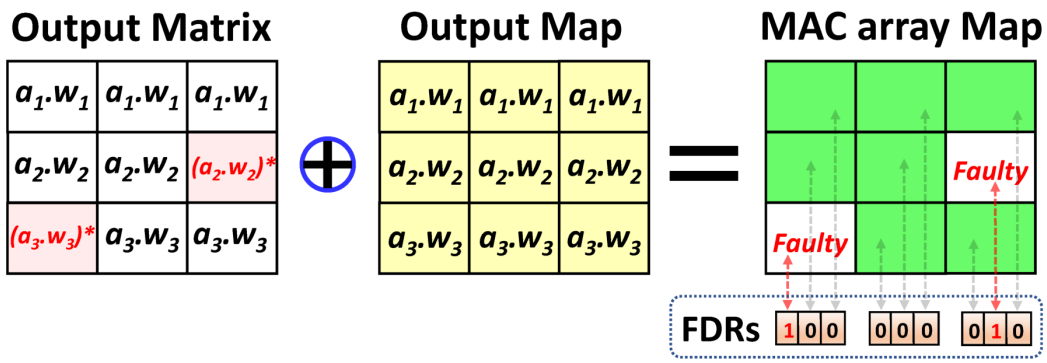


Fig. 5.9: *Comparing the entries of the Output Matrix and the Output Map yields the locality of the Faulty MACs.*

bits of the FDR targeting the faulty MACs in each column (Figure 5.5). An entire systolic array operation is utilized by STRIVE for the errant MACs detection process.

Fault Resilient Mode

In the Fault Resilient Mode, as the faulty MACs are identified, the set *Fault EN* signal effectively skips the faulty MAC computation. Even though the skipping operation generates a loss in precision for the output matrix entries, the overall inference accuracy is not relatively affected due to the algorithmic level error tolerance of DNNs [67]. However, increase in the number of faulty MACs will eventually lead to a drop in DNN inference accuracy. To address this caveat, the design of FHop is enhanced with a *time-borrow* feature to achieve a superior performance, discussed next.

5.3.6 Fault Hop Time-Borrow (FHop-TB)

Figure 5.10 presents FHop-TB—an enhanced variant of FHop that utilizes the combinational delay disparity between the multiplier unit and the accumulate unit to prevent the propagation of corrupted values. FHop-TB uses a Time-Borrow (TB) flop to capture the delayed output of the faulty MAC and direct it to the next logic stage within the same clock cycle.

The accumulation process utilizes less than 50% of the clock cycle. For a faulty MAC, 50% of the clock cycle is borrowed from its downstream MAC (i.e., the downstream MAC

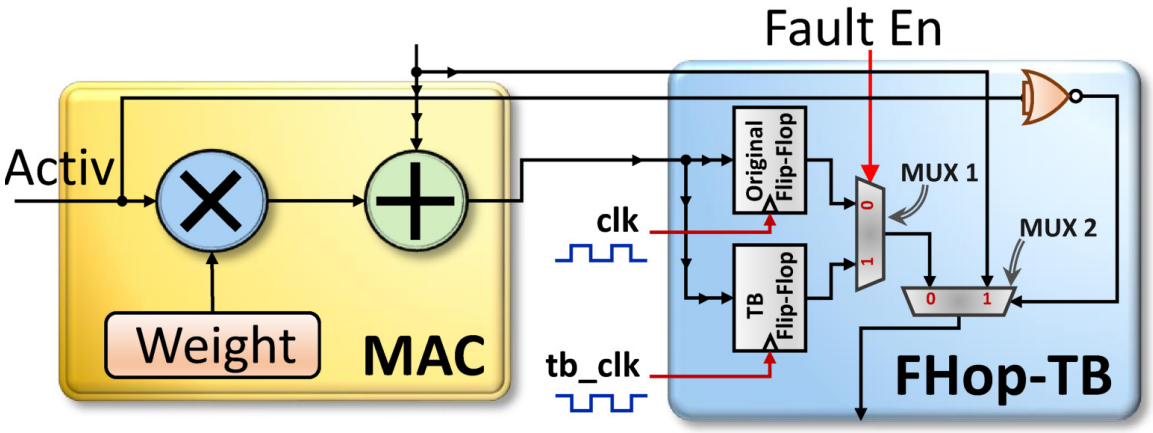


Fig. 5.10: *Detection and correction of timing errors by FHop-TB, using the Time-Borrow technique.*

will be performing its own multiplier operation during this period) to procure the correct MAC output and ensure that the correct output is provided to the downstream MAC for its accumulation process. Thus, the TB latch is driven by a delayed clock (i.e., 50% shift from the system clock) to perform the *Time Borrowing* operation. In a faulty MAC, the output of the shadow latch is sensitized to the MUX 1 output; else, the output of the original latch is delivered to the MUX 1 output. For a zero activation, MUX 2 bypasses the upstream MAC output to the downstream MAC unit, irrespective of a faulty/non-faulty MAC unit. Therefore, the output of MUX 2 can switch between MUX 1 output and upstream MAC output.

5.4 Methodology

The extensive cross-layer methodology employed in this research allows the combination of a functional simulation of a DNN inference task (thus, allowing precise estimation of inference accuracy) with a holistic power-timing characteristics spanning three layers: device, circuit and architecture.

HSPICE simulations are performed on basic logic gates (e.g., NOR, NAND and Inverter) using the 16-nm predictive technology model to measure their delay distributions [107]. VARIUS-NTV is employed to implement the impacts of with-in die PV at LPC [8].

Benchmarks		Error-free Accuracy
Name	**Architecture**	
IMDB [80]	CONV: 400x(400x50)x(398, 256), FC: 256x1	0.89
SVHN [77]	CONV: (32, 32, 3)x(32, 32, 32)x(32, 32, 32)x(14, 14, 64)x(14, 14, 64)x(5, 5, 128)x (5, 5, 128), FC: 512x512x10	0.94
GTSRB [78]	CONV: (3, 48, 48)x(32, 48, 48)x(32, 46, 46)x(64, 23, 23)x(64, 21, 21)x(128, 10, 10)x (128, 8, 8), FC: 2048x512x43	0.97
MNIST [73]	FC: 784x256x256x10	0.98
REUTERS [79]	FC: 2048x256x256x46	0.80
FMNIST [82]	FC: 784x256x512x10	0.89

Table 5.1: *List of DNN benchmarks and their error-free accuracy.*

The FinFET attributes are incorporated using the VARIUS-TC model [76]. The Verilog RTL of the TPU systolic array augmented with the design components are synthesized using Synopsys Design Compiler. The synthesized netlists are utilized by the in-house Static Timing Analysis (STA) tool along with the libraries of delay distributions to generate the sensitized path delays of the MAC unit for all the benchmark driven inputs. Cadence SoC Encounter is used to place and route the design and measure the area, power, and wirelength overheads.

The in-house cycle-accurate TPU systolic array simulator, modeled on the detailed TPU architecture [68], is utilized. To accurately simulate a timing violation, the combinational delays for a PV-affected MAC unit and a nominal MAC unit, developed from the STA tool are integrated into the TPU simulator. Initially, the DNN benchmarks are trained by interfacing the TPU simulator with Keras (running tensorflow in the background) [84]. Table 5.1 lists the DNN benchmarks along with their error-free accuracies. Activation inputs from each layers are streamed across the weight matrices from the trained model for matrix multiplication. The output matrices are appropriately combined to obtain the inference accuracy.

5.5 Experimental Results

This section examines the efficacy of STRIVE at LPC operating conditions–a typical use case for an edge system deployed for an inference task. The baseline operation is set to (0.45V, 67.5MHz) and guarantees an error-free execution for an LPC TPU (i.e., TPU without any faulty MACs). Section 5.5.1 introduces the comparative schemes. Sections 5.5.2 and 5.5.3 elaborate the inference accuracies and energy efficiency of the schemes. Section 5.5.4 discusses the overheads of STRIVE.

5.5.1 Comparative Schemes

- **Fault-Aware Pruning (FAP) :** This scheme prunes all the weights of faulty MAC units [108]. The weights are implemented for multiple precision values to bypass the faulty multiplier in the MAC unit. FAP does not have an error detection scheme as proposed in this chapter, and therefore *presumes to have an oracular knowledge of the faulty MACs*. As expected for an edge system deployed for inference in the field, retraining of weights is not expected, and thus consistently model no retraining for both FAP and the schemes, STRIVE and STRIVE-TB.

- **STRIVE :** This is the proposed technique, which utilizes the locality of the PV-affected MAC or a zero activation input to skip the erroneous MAC operation. FCU enables the *Fault EN* signal for the respective error-prone MAC to restrict the erroneous data from latching on to the accumulator output. FHop is used to perform the bypassing operation (Figure 5.5).

- **STRIVE-TB :** This scheme is an upgraded variant of STRIVE, which uses FHop-TB (Figure 5.10). The error-free output from a faulty MAC unit is captured by the Time-Borrow Flop using the delayed clock, and the *Fault EN* signal enables the forwarding of this correct data to the next stage.

5.5.2 Inference Accuracy

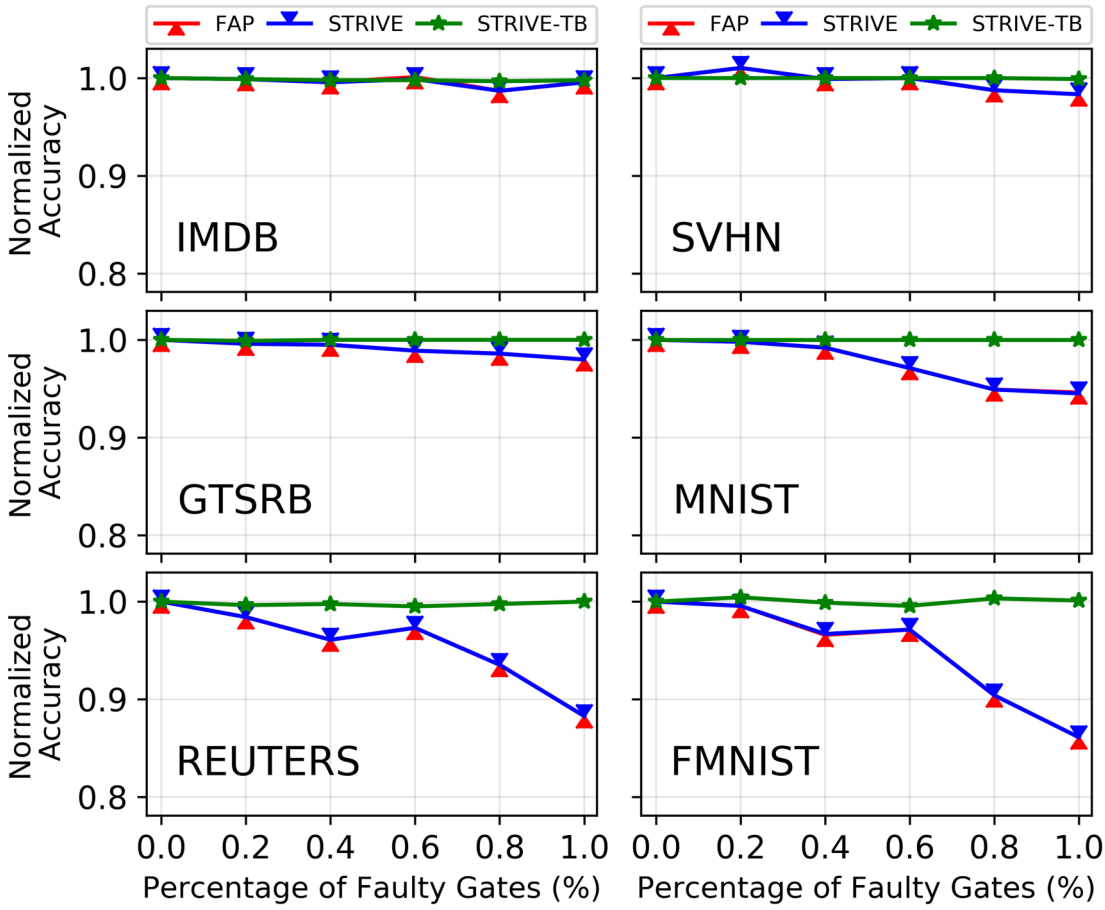


Fig. 5.11: *Normalized Inference accuracies of FAP, STRIVE and STRIVE-TB across 6 DNN benchmarks for different percentages of faulty gates (***under low-power***) affected in a TPU systolic array. The Y-axis values are normalized to the corresponding error-free accuracy at the baseline LPC TPU operation.*

Figure 5.11 depicts the normalized inference accuracy for the three comparative schemes, as the percentage of faulty gates are increased in the TPU. All the schemes are operated at the baseline voltage and frequency. The X-axis represents the percentage of faulty gates in the TPU. STRIVE and FAP are able to offer modest error resilience for 4 out of 6 DNN benchmarks up to 0.6% fault rate. For REUTERS and FMNIST, the extreme data-delay variance between the activation sequences causes significant timing errors, thereby dropping the inference accuracy for STRIVE and FAP. The significant number of zero activation computations aids STRIVE in retaining the inference accuracy as the percentage of faulty

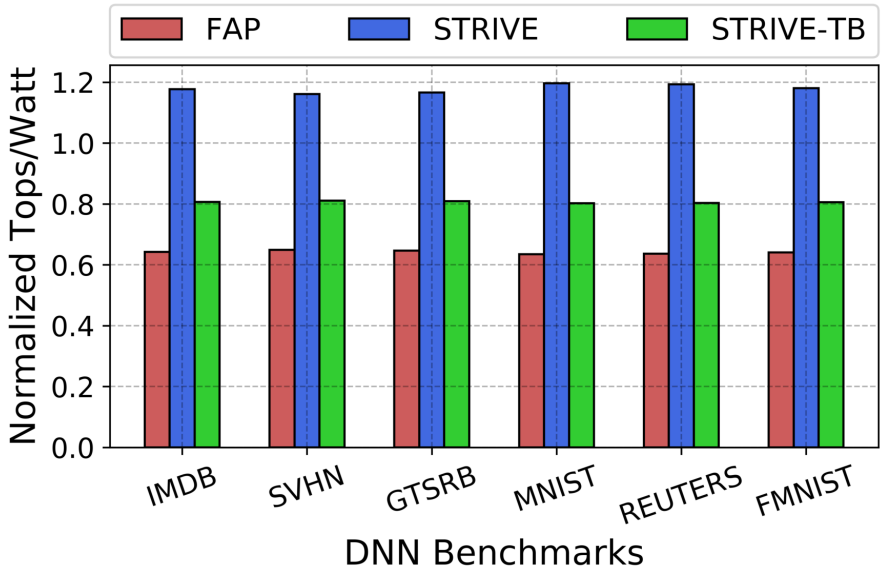


Fig. 5.12: *Energy Efficiency comparison of FAP, STRIVE and STRIVE-TB (higher is better).*

gates are increased. However, STRIVE-TB incurs less than 1% loss in inference accuracy for up to 1% gate fault rate in a TPU, as it is able to capture the delayed data using the *Time-Borrow* approach. *Overall, STRIVE-TB vastly outperforms FAP, demonstrating remarkable resilience in retaining inference accuracy under LP-faults.*

5.5.3 Energy Efficiency

Figure 5.12 presents the energy efficiencies of the comparative schemes measured using the Tera Operations Per Second (TOPS)/Watt metric. TOPS measure will be the same for all the schemes at the corresponding frequency of operation. The TOPS/Watt is normalized to that of the LPC TPU, operated at the baseline operating conditions. FAP has a lower energy efficiency due to its larger power consumption in comparison to STRIVE and STRIVE-TB. STRIVE boasts a higher energy efficiency due to its low overhead operation compared to STRIVE-TB. STRIVE and STRIVE-TB offers an average 1.8× and 1.3× better performance per unit power, compared to FAP. Hence, *STRIVE is an energy efficient design paradigm, able to extract superior performance even from an error-prone TPU.*

5.5.4 Implementation Overheads

STRIVE incurs overheads due to the inclusion of FCU and the combinational logic for FHop or FHop-TB. Area overhead added by STRIVE and STRIVE-TB is $\sim$ 1.8% and $\sim$ 6%. STRIVE and STRIVE-TB incurs a power overhead of $\sim$ 2% and $\sim$ 5%, and a wire-length overhead of $\sim$ 1.1% and $\sim$ 3.5%, respectively.

CHAPTER 6

Conclusion

This dissertation presents architectural enhancements to improve the performance and reliability of the TPU operating at a lower voltage. Minor maneuvering of the circuit connection in a processing unit unveils significant improvements in fault resilience. Exploiting the predictable dataflow brings about noticeable reduction in power consumption. Additionally, foreseeable datatflow pattern aids in predicting the impending timing errors and mitigate them using a statistical based approach. The homogeneous architecture of the systolic array with a minor architectural implement facilitates the localization of faulty elements using tailored inputs.

Increase in the processing workloads in real-time DNN applications calls for a DNN accelerator capable of delivering high classification accuracy while efficiently meeting the energy requirements of the system. This dissertation demonstrates EFFORT— a high-performance energy-efficient novel design paradigm for a TPU, operating at NTC. EFFORT utilizes the disparate delay profiles of a multiplier and accumulator to detect a timing error emanating due to increased frequency and corrects it using a minor architectural manipulation. Later, EFFORT exploits the predictable dataflow pattern in the systolic array to simultaneously clock gate the MAC units along the diagonal to lower the consumption of dynamic power. The efficient design of the clock gating scheme significantly optimizes the number of clock gating components. Hence, EFFORT efficiently detects and tackles timing errors while reducing the power consumption of the TPU. EFFORT delivers up to 2.5$\times$ increase in performance with a minimum drop in accuracy and consumes 6% - 27% less power in comparison to recently proposed schemes. Additionally, EFFORT gives between 1.06$\times$ and 1.35$\times$ superior performance per unit power against representative timing error resilient schemes. EFFORT facilitates the migration of computing platform towards NTC, thereby paving the way for an energy efficient AI computing realm.

The meteoric rise in the DNN computations demands a low-power error resilient DNN accelerator design paradigm capable of delivering quintessential classification accuracy. This dissertation proposes PREDITOR—an energy efficient high performance novel design for an NTC TPU. The timing error profiles are statistically analyzed and the range of operational cycles producing significant timing errors are computed. The voltage levels of the MACs are slightly boosted over these clock cycles to mitigate the impending timing errors. MRFF is utilized by PREDITOR to tackle the detectable timing errors throughout the entire operation. PREDITOR tackles up to $\times$68% of the undetectable timing errors, thereby preserving the inference accuracy of the DNN datasets. PREDITOR offers 3$\times$–5$\times$ better performance in comparison to other error resilience schemes, with under 3% average loss in accuracy. Additionally, PREDITOR offers up to 87% energy-efficiency gain in relative to a TPU operating at STC.

This dissertation also highlights the impact of PV in a TPU systolic array under low-power operation. This dissertation also demonstrates STRIVE—an energy efficient paradigm to identify and nullify the effect of LP-faults, and reclaim the performance from a TPU systolic array affected by faulty MAC units. STRIVE incurs minimum loss in inference accuracy for a TPU infested by a gate level fault rate of 1%. Additionally, STRIVE delivers 1.8$\times$ and 1.3$\times$ better TOPS/watt compared other fault mitigation scheme. The lower overhead and the efficient fault detection aids in STRIVE being deployed efficiently in the EDGE AI platform.

In conclusion, this dissertation presents methodologies to enhance performance and reclaim reliable computing a NTC, thereby promoting the adoption of LPC into the nominal computing realm. The works in this dissertation aim to induce more research efforts in investigating and highlighting the circuit-architectural innovations with minor overhead and significant improvements in performance. Overall, this dissertation aims to provide a worthwhile contribution to academia and the semiconductor industry.

REFERENCES

[1] "Artificial intelligence is booming—so is its carbon footprint," https://www.bloomberg.com/news/articles/2023-03-09/how-much-energy-do-ai-and-chatgpt-use-no-one-knows-for-sure#xj4y7vzkg.

[2] R. G. Dreslinski, M. Wieckowski, D. Blaauw, D. Sylvester, and T. N. Mudge, "Near-threshold computing: Reclaiming moore's law through energy efficient integrated circuits," *Proc. of the IEEE*, vol. 98, no. 2, pp. 253–266, 2010.

[3] D. Markovic, C. C. Wang, L. P. Alarcon, T.-T. Liu, and J. M. Rabaey, "Ultralow-power design in near-threshold region," *Proceedings of the IEEE*, vol. 98, no. 2, pp. 237–252, 2010.

[4] P. Friedberg, Y. Cao, J. Cain, R. Wang, J. M. Rabaey, and C. J. Spanos, "Modeling within-die spatial correlation effects for process-design co-optimization," in *ISQED*, 2005, pp. 516–521.

[5] B. Stine, D. Boning, and J. Chung, "Analysis and decomposition of spatial variation in integrated circuit processes and devices," *IEEE Tran. on Semicond. Manufac.*, vol. 10, no. 1, pp. 24–41, Feb 1997.

[6] U. Karpuzcu, N. S. Kim, and J. Torrellas, "Coping with parametric variation at near-threshold voltages," *IEEE Micro*, vol. 33, no. 4, pp. 6–14, July 2013.

[7] S. Borkar, T. Karnik, and V. De, "Design and reliability challenges in nanometer technologies," in *Proceedings of the 41st annual Design Automation Conference*, 2004, pp. 75–75.

[8] U. R. Karpuzcu, K. B. Kolluru, N. S. Kim, and J. Torrellas, "Varius-ntv: A microarchitectural model to capture the increased sensitivity of manycores to process variations at near-threshold voltages," in *DSN*, 2012, pp. 1–11.

[9] N. Pinckney, K. Sewell, R. Dreslinski, D. Fick, T. M. udge, D. Sylvester, and D. Blaauw, "Assessing the performance limits of parallelized near-threshold computing," in *DAC*, 2012, pp. 1143–1148.

[10] H. Kaul, M. Anders, S. Hsu, A. Agarwal, R. Krishnamurthy, and S. Borkar, "Near-threshold voltage (ntv) design—opportunities and challenges," in *Proc. of DAC*, June 2012, pp. 1149–1154.

[11] V. De, S. Vangal, and R. Krishnamurthy, "Near threshold voltage (ntv) computing: Computing in the dark silicon era," *IEEE Design & Test*, vol. 34, no. 2, pp. 24–30, 2016.

[12] B. Reagen, P. Whatmough, R. Adolf, S. Rama, H. Lee, S. K. Lee, J. M. Hernández-Lobato, G.-Y. Wei, and D. Brooks, "Minerva: Enabling low-power, highly-accurate deep neural network accelerators," in *Proc. of ISCA*, vol. 44, no. 3. IEEE Press, 2016, pp. 267–278.

[13] Y.-H. Chen, T. Krishna, J. S. Emer, and V. Sze, "Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks," *JSSC*, vol. 52, no. 1, pp. 127–138, 2016.

[14] X. Wang, R. Hou, B. Zhao, F. Yuan, J. Zhang, D. Meng, and X. Qian, "Dnnguard: An elastic heterogeneous dnn accelerator architecture against adversarial attacks," in *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*, 2020, pp. 19–34.

[15] J. Zhang, K. Rangineni, Z. Ghodsi, and S. Garg, "Thundervolt: Enabling aggressive voltage underscaling and timing error resilience for energy efficient deep neural network accelerators," 2018.

[16] N. Chandramoorthy, K. Swaminathan, M. Cochet, A. Paidimarri, S. Eldridge, R. Joshi, M. Ziegler, A. Buyuktosunoglu, and P. Bose, "Resilient low voltage accelerators for high energy efficiency," in *2019 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, 2019, pp. 147–158.

[17] J. Yu, A. Lukefahr, D. Palframan, G. Dasika, R. Das, and S. Mahlke, "Scalpel: Customizing dnn pruning to the underlying hardware parallelism," in *ACM SIGARCH Computer Architecture News*, vol. 45, no. 2. ACM, 2017, pp. 548–560.

[18] E. Ozen and A. Orailoglu, "Snr: S queezing n umerical r ange defuses bit error vulnerability surface in deep neural networks," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 20, no. 5s, pp. 1–25, 2021.

[19] B. Salami, O. S. Unsal, and A. C. Kestelman, "On the resilience of rtl nn accelerators: Fault characterization and mitigation," in *2018 30th International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD)*. IEEE, 2018, pp. 322–329.

[20] S. Kim, P. Howe, T. Moreau, A. Alaghi, L. Ceze, and V. S. Sathe, "Energy-efficient neural network acceleration in the presence of bit-level memory errors," *IEEE Transactions on Circuits and Systems I: Regular Papers*, no. 99, pp. 1–14, 2018.

[21] F. Libano, B. Wilson, J. Anderson, M. Wirthlin, C. Cazzaniga, C. Frost, and P. Rech, "Selective hardening for neural networks in fpgas," *IEEE Transactions on Nuclear Science*, vol. 66, no. 1, pp. 216–222, 2018.

[22] S. Ghodrati, H. Sharma, S. Kinzer, A. Yazdanbakhsh, K. Samadi, N. S. Kim, D. Burger, and H. Esmaeilzadeh, "Mixed-signal charge-domain acceleration of deep neural networks through interleaved bit-partitioned arithmetic," *arXiv preprint arXiv:1906.11915*, 2019.

[23] C. Mackin, H. Tsai, S. Ambrogio, P. Narayanan, A. Chen, and G. W. Burr, "Weight programming in dnn analog hardware accelerators in the presence of nvm variability," *Advanced Electronic Materials*, vol. 5, no. 9, p. 1900026, 2019.

[24] A. Shafiee, A. Nag, N. Muralimanohar, R. Balasubramonian, J. P. Strachan, M. Hu, R. S. Williams, and V. Srikumar, "Isaac: A convolutional neural network accelerator

with in-situ analog arithmetic in crossbars," *ACM SIGARCH Computer Architecture News*, vol. 44, no. 3, pp. 14–26, 2016.

[25] J. K. Eshraghian, S.-M. Kang, S. Baek, G. Orchard, H. H.-C. Iu, and W. Lei, "Analog weights in reram dnn accelerators," in *2019 IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS)*. IEEE, 2019, pp. 267–271.

[26] P. N. Whatmough, I. Darwazeh, D. M. Bull, S. Das, and D. Kershaw, "A robust fir filter with in situ error detection," in *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*. IEEE, 2010, pp. 4185–4188.

[27] L. Yang and B. Murmann, "Sram voltage scaling for energy-efficient convolutional neural networks," in *2017 18th International Symposium on Quality Electronic Design (ISQED)*. IEEE, 2017, pp. 7–12.

[28] A. Di Mauro, F. Conti, P. D. Schiavone, D. Rossi, and L. Benini, "Always-on 674μ w@ 4gop/s error resilient binary neural networks with aggressive sram voltage scaling on a 22-nm iot end-node," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 67, no. 11, pp. 3905–3918, 2020.

[29] P. N. Whatmough, S. K. Lee, D. Brooks, and G. Wei, "Dnn engine: A 28-nm timing-error tolerant sparse deep neural network processor for iot applications," *IEEE Journal of Solid-State Circuits*, vol. 53, no. 9, pp. 2722–2731, Sep. 2018.

[30] P. N. Whatmough, S. Das, and D. M. Bull, "A low-power 1-ghz razor fir accelerator with time-borrow tracking pipeline and approximate error correction in 65-nm cmos," *IEEE Journal of Solid-State Circuits*, vol. 49, no. 1, pp. 84–94, 2013.

[31] D. Xu, C. Chu, Q. Wang, C. Liu, Y. Wang, L. Zhang, H. Liang, and K.-T. Cheng, "A hybrid computing architecture for fault-tolerant deep learning accelerators," in *ICCD*. IEEE, 2020, pp. 478–485.

[32] J. J. Zhang, T. Gu, K. Basu, and S. Garg, "Analyzing and mitigating the impact of permanent faults on a systolic array based neural network accelerator," in *Proc. of VTS*, April 2018, pp. 1–6.

[33] T. Spyrou, S. A. El-Sayed, E. Afacan, L. A. Camuñas-Mesa, B. Linares-Barranco, and H.-G. Stratigopoulos, "Neuron fault tolerance in spiking neural networks," in *Proc. of DATE*, 2021, pp. 743–748.

[34] B. Salami, E. B. Onural, I. Yuksel, F. Koc, O. Ergin, A. Cristal, O. Unsal, H. Sarbazi-Azad, and O. Mutlu, "An experimental study of reduced-voltage operation in modern fpgas for neural network acceleration," in *DSN*, 2020, pp. 138–149.

[35] K. Givaki, B. Salami, R. Hojabr, S. R. Tayaranian, A. Khonsari, D. Rahmati, S. Gorgin, A. Cristal, and O. S. Unsal, "On the resilience of deep learning for reduced-voltage fpgas," in *Proc. of PDNP*. IEEE, 2020, pp. 110–117.

[36] Z. Tang, Y. Wang, Q. Wang, and X. Chu, "The impact of gpu dvfs on the energy and performance of deep learning: An empirical study," in *Proceedings of the Tenth ACM International Conference on Future Energy Systems*, 2019, pp. 315–325.

[37] J. Lee, S. Kang, J. Lee, D. Shin, D. Han, and H.-J. Yoo, "The hardware and algorithm co-design for energy-efficient dnn processor on edge/mobile devices," *TCSI*, vol. 67, no. 10, pp. 3458–3470, 2020.

[38] D.-T. Nguyen, N.-M. Ho, and I.-J. Chang, "St-drc: Stretchable dram refresh controller with no parity-overhead error correction scheme for energy-efficient dnns," in *Proc. of DAC*. New York, NY, USA: ACM, 2019, pp. 205:1–205:6. [Online]. Available: http://doi.acm.org/10.1145/3316781.3317915

[39] S. Koppula, A. G. Orosa, R. Azizi, T. Shahroodi, K. Kanellopoulos, and O. Mutlu, "Eden: Enabling energy-efficient, high-performance deep neural network inference using approximate dram," in *Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitecture*, 2019, pp. 166–181.

[40] W. Jiang, H. Yu, J. Zhang, J. Wu, S. Luo, and Y. Ha, "Optimizing energy efficiency of cnn-based object detection with dynamic voltage and frequency scaling," *Journal of Semiconductors*, vol. 41, no. 2, p. 022406, 2020.

[41] M. E. Elbtity, P. S. Chandarana, B. Reidy, J. K. Eshraghian, and R. Zand, "Aptpu: Approximate computing based tensor processing unit," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 69, no. 12, pp. 5135–5146, 2022.

[42] A. Ruospo, A. Balaara, A. Bosio, and E. Sanchez, "A pipelined multi-level fault injector for deep neural networks," in *2020 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)*. IEEE, 2020, pp. 1–6.

[43] F. S. Hosseini, F. Meng, C. Yang, W. Wen, and R. Cammarota, "Tolerating defects in low-power neural network accelerators via retraining-free weight approximation," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 20, no. 5s, pp. 1–21, 2021.

[44] L.-H. Hoang, M. A. Hanif, and M. Shafique, "Tre-map: Towards reducing the overheads of fault-aware retraining of deep neural networks by merging fault maps," in *2021 24th Euromicro Conference on Digital System Design (DSD)*. IEEE, 2021, pp. 434–441.

[45] N. P. Jouppi, C. Young, N. Patil, D. Patterson, G. Agrawal, R. Bajwa, S. Bates, S. Bhatia, N. Boden, A. Borchers *et al.*, "In-datacenter performance analysis of a tensor processing unit," in *Proc. of ISCA*. IEEE, 2017, pp. 1–12.

[46] M. Orshansky, L. Milor, and C. Hu, "Characterization of spatial intrafield gate cd variability, its impact on circuit performance, and spatial mask-level correction," *IEEE Transactions on Semiconductor Manufacturing*, vol. 17, no. 1, pp. 2–11, 2004.

[47] T. Shabanian, A. Bal, P. Basu, K. Chakraborty, and S. Roy, "Ace-gpu: Tackling choke point induced performance bottlenecks in a near-threshold computing gpu," in *ISLPED*, 2018.

[48] A. Bal, S. Saha, S. Roy, and K. Chakraborty, "Revamping timing error resilience to tackle choke points at ntc systems," in *Proc. of DATE*, 2017, pp. 1020–1025.

[49] D. Ernst, N. S. Kim, S. Das, S. Pant, R. R. Rao, T. Pham, C. H. Ziesler, D. Blaauw, T. M. Austin, K. Flautner, and T. N. Mudge, "Razor: A low-power pipeline based on circuit-level timing speculation," in *Proc. of MICRO*, 2003, pp. 7–18.

[50] P. N. Whatmough, S. Das, D. M. Bull, and I. Darwazeh, "Circuit-level timing error tolerance for low-power dsp filters and transforms," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 21, no. 6, pp. 989–999, 2012.

[51] H. Ji, L. Song, L. Jiang, H. H. Li, and Y. Chen, "Recom: An efficient resistive accelerator for compressed deep neural networks," in *2018 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2018, pp. 237–240.

[52] S. Sanyal, P. Basu, A. Bal, S. Roy, and K. Chakraborty, "Exploring warp criticality in near-threshold gpgpu applications using a dynamic choke point analysis," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 28, no. 2, pp. 456–466, 2019.

[53] L. Zhao, Y. Zhang, and J. Yang, "Aep: An error-bearing neural network accelerator for energy efficiency and model protection," in *2017 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE, 2017, pp. 1047–1053.

[54] P. Basu, H. Chen, S. Saha, K. Chakraborty, and S. Roy, "Swiftgpu: Fostering energy efficiency in a near-threshold gpu through a tactical performance boost," in *Proc. of DAC*, 2016.

[55] P. Chi, S. Li, C. Xu, T. Zhang, J. Zhao, Y. Liu, Y. Wang, and Y. Xie, "Prime: A novel processing-in-memory architecture for neural network computation in reram-based main memory," in *Proceedings of the 43rd International Symposium on Computer Architecture*. IEEE Press, 2016, pp. 27–39.

[56] D. Kim, J. Kung, S. Chai, S. Yalamanchili, and S. Mukhopadhyay, "Neurocube: A programmable digital neuromorphic architecture with high-density 3d memory," in *Computer Architecture (ISCA), 2016 ACM/IEEE 43rd Annual International Symposium on*. IEEE, 2016, pp. 380–392.

[57] G. Gobieski, B. Lucia, and N. Beckmann, "Intelligence beyond the edge: Inference on intermittent embedded systems," in *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, 2019, pp. 199–213.

[58] P. Pandey, P. Basu, K. Chakraborty, and S. Roy, "Greentpu: Predictive design paradigm for improving timing error resilience of a near-threshold tensor processing unit," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 2020.

[59] C. Gao, D. Neil, E. Ceolini, S.-C. Liu, and T. Delbruck, "Deltarnn: A power-efficient recurrent neural network accelerator," in *Proceedings of the 2018 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, 2018, pp. 21–30.

[60] R. J. Shridevi, D. M. Ancajas, K. Chakraborty, and S. Roy, "Tackling voltage emergencies in noc through timing error resilience," in *ISLPED*, 2015, pp. 104–109.

[61] H. Mao, M. Song, T. Li, Y. Dai, and J. Shu, "Lergan: A zero-free, low data movement and pim-based gan architecture," in *2018 51st Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*. IEEE, 2018, pp. 669–681.

[62] S. Yin, S. Tang, X. Lin, P. Ouyang, F. Tu, J. Zhao, C. Xu, S. Li, Y. Xie, S. Wei *et al.*, "Parana: A parallel neural architecture considering thermal problem of 3d stacked memory," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 1, pp. 146–160, 2018.

[63] J.-S. Kim and J.-S. Yang, "Dris-3: Deep neural network reliability improvement scheme in 3d die-stacked memory based on fault analysis," in *Proc. of DAC*. IEEE, 2019, pp. 1–6.

[64] S. Saha, P. Basu, C. Rajamanikkam, A. Bal, K. Chakraborty, and S. Roy, "SSAGA: sms synthesized for asymmetric GPGPU applications," *ACM Trans. Design Autom. Electr. Syst.*, vol. 22, no. 3, pp. 49:1–49:20, 2017.

[65] S. Jain, S. Venkataramani, V. Srinivasan, J. Choi, P. Chuang, and L. Chang, "Compensated-dnn: energy efficient low-precision deep neural networks by compensating quantization errors," in *Proc. of DAC*. IEEE, 2018, pp. 1–6.

[66] F. Nakhaee, M. Kamal, A. Afzali-Kusha, M. Pedram, S. M. Fakhraie, and H. Dorosti, "Lifetime improvement by exploiting aggressive voltage scaling during runtime of error-resilient applications," *Integration*, vol. 61, pp. 29–38, 2018.

[67] X. Jiao, M. Luo, J.-H. Lin, and R. K. Gupta, "An assessment of vulnerability of hardware neural networks to dynamic voltage and temperature variations," in *Proc. of ICCAD*. IEEE Press, 2017, pp. 945–950.

[68] N. Jouppi, C. Young, N. Patil, and D. Patterson, "Motivation for and evaluation of the first tensor processing unit," *IEEE Micro*, vol. 38, no. 3, pp. 10–19, 2018.

[69] NanGate, http://www.nangate.com/?page_id=2328.

[70] S. Sarangi, B. Greskamp, R. Teodorescu, J. Nakano, A. Tiwari, and J. Torrellas, "Varius:a model of process variation and resulting timing errors for microarchitects," *IEEE Tran. on Semicond. Manufac.*, vol. 21, pp. 3 –13, 2008.

[71] P. Pandey, P. Basu, K. Chakraborty, and S. Roy, "Greentpu: Improving timing error resilience of a near-threshold tensor processing unit," in *Proc. of DAC*, 2019, pp. 173:1–173:6.

[72] S. Sanyal, P. Basu, A. Bal, S. Roy, and K. Chakraborty, "Predicting critical warps in near-threshold gpgpu applications using a dynamic choke point analysis," in *2019 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2019, pp. 444–449.

[73] Y. LeCun and C. Cortes, "MNIST handwritten digit database," http://yann.lecun.com/exdb/mnist/, 2010.

[74] J. Albericio, P. Judd, T. Hetherington, T. Aamodt, N. E. Jerger, and A. Moshovos, "Cnvlutin: Ineffectual-neuron-free deep neural network computing," in *2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA)*, June 2016, pp. 1–13.

[75] W. Zhao and Y. Cao, "New generation of predictive technology model for sub-45nm early design exploration," *T. Electron Devices*, vol. 53, no. 11, pp. 2816 –2823, 2006.

[76] S. K. Khatamifard, M. Resch, N. S. Kim, and U. R. Karpuzcu, "Varius-tc: A modular architecture-level model of parametric variation for thin-channel switches," in *ICCD*, 2016, pp. 654–661.

[77] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," in *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011.

[78] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition," *Neural Networks*, no. 0, pp. –, 2012.

[79] "Reuters-21578 dataset," http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html, 2021.

[80] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning word vectors for sentiment analysis." Association for Computational Linguistics, 2011, pp. 142–150.

[81] A. Krizhevsky, "Learning multiple layers of features from tiny images," Tech. Rep., 2009.

[82] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *CoRR*, vol. abs/1708.07747, 2017. [Online]. Available: http://arxiv.org/abs/1708.07747

[83] "Free spoken digit dataset (fsdd)," https://github.com/Jakobovski/free-spoken-digit-dataset, 2021.

[84] F. Chollet *et al.*, "Keras," https://keras.io, 2015.

[85] Y. Long, X. She, and S. Mukhopadhyay, "Design of reliable dnn accelerator with un-reliable reram," 03 2019, pp. 1769–1774.

[86] "Creating an ai can be five times worse for the planet than a car," https://www.newscientist.com/article/2205779-creating-an-ai-can-be-five-times-worse-for-the-planet-than-a-car/.

[87] M. Seok, G. Chen, S. Hanson, M. Wieckowski, D. Blaaw, and D. Sylvester, "Cas-fest 2010: Mitigating variability in near-threshold computing," in *J. Emerg Selec. Topics Cir. Sys*, vol. 1, no. 1, 2011, pp. 42–49.

[88] S. Potluri and C. Diedrich, "Accelerated deep neural networks for enhanced intrusion detection system," in *2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2016, pp. 1–8.

[89] C. Liu, C. Chu, D. Xu, Y. Wang, Q. Wang, H. Li, X. Li, and K.-T. Cheng, "Hyca: A hybrid computing architecture for fault-tolerant deep learning," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 41, no. 10, pp. 3400–3413, 2022.

[90] Y. Ibrahim, H. Wang, J. Liu, J. Wei, L. Chen, P. Rech, K. Adam, and G. Guo, "Soft errors in dnn accelerators: A comprehensive review," *Microelectronics Reliability*, vol. 115, p. 113969, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0026271420308003

[91] P. Hosseinzadeh, Y. Sedaghat, and A. Harati, "Gap: Fault tolerance improvement of convolutional neural networks through gan-aided pruning," in *2022 12th International Conference on Computer and Knowledge Engineering (ICCKE)*, 2022, pp. 294–299.

[92] E. Ozen and A. Orailoglu, "Sanity-check: Boosting the reliability of safety-critical deep neural network applications," in *2019 IEEE 28th Asian Test Symposium (ATS)*, 2019, pp. 7–75.

[93] S. J. Engers, C. Chu, D. Xu, Y. Wang, and F. Chen, "Mocca: A process variation tolerant systolic dnn accelerator using cnfets in monolithic 3d," in *Proceedings of the Great Lakes Symposium on VLSI 2022*. Association for Computing Machinery, 2022.

[94] D. Ji, D. Shin, and J. Park, "An error compensation technique for low-voltage dnn accelerators," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 29, no. 2, pp. 397–408, 2021.

[95] S. Burel, A. Evans, and L. Anghel, "Mozart: Masking outputs with zeros for architectural robustness and testing of dnn accelerators," in *2021 IEEE 27th International Symposium on On-Line Testing and Robust System Design (IOLTS)*, 2021, pp. 1–6.

[96] C. Chu, D. Xu, Y. Wang, and F. Chen, "Canopy: A cnfet-based process variation aware systolic dnn accelerator," in *Proceedings of the ACM/IEEE International Symposium on Low Power Electronics and Design*, ser. ISLPED '22. New York, NY, USA: Association for Computing Machinery, 2022. [Online]. Available: https://doi.org/10.1145/3531437.3539703

[97] E. Ozen and A. Orailoglu, "Architecting decentralization and customizability in dnn accelerators for hardware defect adaptation," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 41, no. 11, pp. 3934–3945, 2022.

[98] F. Sunny, A. Mirza, M. Nikdast, and S. Pasricha, "Crosslight: A cross-layer optimized silicon photonic neural network accelerator," in *2021 58th ACM/IEEE Design Automation Conference (DAC)*, 2021, pp. 1069–1074.

[99] C. Rajamanikkam, R. J. Shridevi, S. Roy, and K. Chakraborty, "Boostnoc: Power efficient network-on-chip architecture for near threshold computing," in *Proc. of ICCAD*, 2016.

[100] S. Kundu, S. Banerjee, A. Raha, S. Natarajan, and K. Basu, "Toward functional safety of systolic array-based deep learning hardware accelerators," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 29, no. 3, pp. 485–498, 2021.

[101] Y. Zhao, K. Wang, and A. Louri, "Fsa: An efficient fault-tolerant systolic array-based dnn accelerator architecture," in *2022 IEEE 40th International Conference on Computer Design (ICCD)*, 2022, pp. 545–552.

[102] G. Hills, D. Bankman, B. Moons, L. Yang, J. Hillard, A. Kahng, R. Park, M. Verhelst, B. Murmann, M. M. Shulaker, H. S. P. Wong, and S. Mitra, "Trig: Hardware accelerator for inference-based applications and experimental demonstration using carbon nanotube fets," in *2018 55th ACM/ESDA/IEEE Design Automation Conference (DAC)*, 2018, pp. 1–10.

[103] M. Ibtesam, U. S. Solangi, J. Kim, M. A. Ansari, and S. Park, "Reliable test architecture with test cost reduction for systolic-based dnn accelerators," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 69, no. 3, pp. 1537–1541, 2022.

[104] T. Soliman, N. Laleni, T. Kirchner, F. Müller, A. Shrivastava, T. Kämpfe, A. Guntoro, and N. Wehn, "Felix: A ferroelectric fet based low power mixed-signal in-memory architecture for dnn acceleration," vol. 21, no. 6, 2022. [Online]. Available: https://doi.org/10.1145/3529760

[105] T. N. Miller, X. Pan, R. Thomas, N. Sedaghati, and R. Teodorescu, "Booster: Reactive core acceleration for mitigating the effects of process variation and application imbalance in low-voltage chips," in *HPCA*, 2012, pp. 1–12.

[106] M. Capra, R. Peloso, G. Masera, M. Ruo Roch, and M. Martina, "Edge computing: A survey on the hardware requirements in the internet of things world," *Future Internet*, vol. 11, no. 4, p. 100, 2019.

[107] W. Zhao and Y. Cao, "New generation of predictive technology model for sub-45 nm early design exploration," *T. Electron Devices*, pp. 2816 –2823, 2006.

[108] J. J. Zhang, K. Basu, and S. Garg, "Fault-tolerant systolic array based accelerators for deep neural network execution," *IEEE Design & Test*, vol. 36, no. 5, pp. 44–53, 2019.

CURRICULUM VITAE

Noel Daniel Gundi

Journal Articles

- Implementing a Timing Error-Resilient and Energy-Efficient Near-Threshold Hardware Accelerator for Deep Neural Network Inference, Noel Daniel Gundi, Pramesh Pandey, Sanghamitra Roy and Koushik Chakraborty, *Journal of Low Power Electronics and Applications (JLPEA)*, vol. 12, no. 2, p. 32, 2022.

- EFFORT: A comprehensive technique to tackle timing violations and improve energy efficiency of near-threshold tensor processing units, Noel Daniel Gundi, Tahmoures Shabanian, Prabal Basu, Pramesh Pandey, Sanghamitra Roy and Koushik Chakraborty, *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 29, no. 10, pp. 1790–1799, 2021.

- Challenges and Opportunities in Near-Threshold DNN Accelerators around Timing Errors, Pramesh Pandey, Noel Daniel Gundi, Prabal Basu, Tahmoures Shabanian, Mitchell Patrick, Koushik Chakraborty and Sanghamitra Roy, *Journal of Low Power Electronics and Applications (JLPEA)*, vol. 10, no. 4, p. 33, 2020.

Conference Papers

- STRIVE: Enabling Choke Point Detection and Timing Error Resilience in a Low-Power Tensor Processing Unit, Noel Daniel Gundi, Zinnia Muntaha Mowri, Andrew Chamberlin, Sanghamitra Roy and Koushik Chakraborty, Accepted for publication in *IEEE/ACM Design Automation Conference (DAC)*, 2023.

- UPTPU: Improving energy efficiency of a tensor processing unit through underutilization based power-gating, Pramesh Pandey, Noel Daniel Gundi, Koushik Chakraborty and Sanghamitra Roy, *IEEE/ACM Design Automation Conference (DAC)*, 2021.

- EFFORT: Enhancing energy efficiency and error resilience of a near-threshold tensor processing unit, Noel Daniel Gundi, Tahmoures Shabanian, Prabal Basu, Pramesh Pandey, Sanghamitra Roy, Koushik Chakraborty and Zhen Zhang, *IEEE Asia and South Pacific Design Automation Conference (ASPDAC)*, 2020.