# TOWARDS INTELLIGENT, SECURE, AND ENERGY-EFFICIENT SYSTEMS-ON-EDGE

**LUIS EDUARDO RUEDA GUERRERO**

**UNIVERSIDAD INDUSTRIAL DE SANTANDER**
**FACULTAD DE INGENIERÍAS FISICOMECÁNICAS**
**ESCUELA DE INGENIERÍA ELÉCTRICA, ELECTRÓNICA Y DE TELECOMUNICACIONES**
**DOCTORADO EN INGENIERÍA: ÁREA INGENIERÍA ELECTRÓNICA**
**BUCARAMANGA**
**2024**

# TOWARDS INTELLIGENT, SECURE, AND ENERGY-EFFICIENT SYSTEMS-ON-EDGE

**LUIS EDUARDO RUEDA GUERRERO**

**A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor in Engineering**

**Advisor:**
**ÉLKIM FELIPE ROA FUENTES**
**INGENIERO ELECTRÓNICO. PhD.**

**UNIVERSIDAD INDUSTRIAL DE SANTANDER**
**FACULTAD DE INGENIERÍAS FISICOMECÁNICAS**
**ESCUELA DE INGENIERÍA ELÉCTRICA, ELECTRÓNICA Y DE TELECOMUNICACIONES**
**DOCTORADO EN INGENIERÍA: ÁREA INGENIERÍA ELECTRÓNICA**
**BUCARAMANGA**
**2024**

## ACKNOWLEDGEMENTS

Primero que todo, agradezco a Élkim, mi director de tesis. Gracias por todas esas charlas que tuvimos. Te admiro.

Agradezco a mis compañeros y amigos del doctorado. Sin ellos, seguramente esta experiencia no habría sido agradable: Javier, Andrés, Héctor, Ckristian, y Juan. Muchas gracias por todas las charlas técnicas y no técnicas que tuvimos. Siempre se aprende algo nuevo con ustedes.

Le agradezco a todos los alumnos con los que me encontré mientras fui profesor de la UIS. En especial agradezco a: David, Laude, Juan Sebastián, Gabriel, Julián, Néstor, Helmunt, Joan, Felipe, Diana, Jose, Alejandro, Ricardo, Andrés, y Edward (espero no se me haya escapado ninguno). De verdad muchas gracias.

Agradezco a mi familia. A mis papás y mis hermanos. Ellos siempre han sido las personas que me han acompañado en todos los procesos de mi vida. Gracias por todo lo que me han dado. Gracias madre por darnos tanto, por dedicar tu vida 100% a nosotros. Gracias padre por presentarme esta rama tan bonita de la ingeniería. Eres mi modelo a seguir. Gracias Marco y Migue. Los amo con todo mi corazón.

Y por último, quiero darle las gracias a Maria Luisa. Amor, eres increíble. Has tenido una paciencia infinita en todos estos años. Sin ti no habría podido terminar el doctorado. Fuiste mi apoyo cuando no creía que iba a ser capaz, así como la que estuvo en esos momentos en los que vi el final de túnel. Siempre te estaré agradecido por todas las cosas que has hecho por mí. Te amo.

*Dedicado a Choko. La muerte va tan segura de ganar que de ventaja nos da una vida.*

# Contents

# List of Figures

# List of Tables

**Page.**

# RESUMEN


**TÍTULO:** TOWARDS INTELLIGENT, SECURE, AND ENERGY-EFFICIENT SYSTEMS-ON-EDGE [*]

**AUTOR:** LUIS EDUARDO RUEDA GUERRERO [**]

**PALABRAS CLAVE:** *System-on-edge*, eficiencia energética, seguridad, computación analógica, variabilidad estocástica, rede neuronales, *machine learning*, computación en memoria, administración de energía, ataques por perturbación transitoria de voltage.

**DESCRIPCIÓN:**

Con miles de millones (incluso billones, según estimaciones) de dispositivos interconectados, el consumo de energía, la gestión de gran cantidad de datos y su seguridad, son algunos de los principales desafíos para las aplicaciones *IoT* (Internet de las cosas). La administración inteligente de la energía, basada en monitores de tensión, es una de las principales soluciones en cuanto a la reducción del consumo de energía. Mientras tanto, la inferencia con sistemas de *deep-learning* surge como una de las formas más efectivas de lidiar con gran cantidad de datos para la toma de decisiones. Al mismo tiempo, la aceleración con *hardware* analógico ha demostrado ser una alternativa prometedora para obtener sistemas de *deep-learning* para aplicaciones *IoT* (*systems-on-edge*-SoE) energeticamente eficientes. La seguridad es otro de los principales desafíos para SoE. Con más nodos conectados, hay más oportunidades para comprometer la seguridad de sistemas completos, lo que podría llevar a la filtración de información sensible o dejar el sistema vulnerable a ataques desde diferentes frentes.

Esta tesis presenta contribuciones en los tres frentes mencionados anteriormente: SoE energeticamente eficientes, SoE para la toma de decisiones y vulneración de seguridad en SoE. Primero, proponemos A-Connect, una novedosa metodología para mejorar la resiliencia de las redes neuronales contra la variabilidad estocástica, como cuando se implementan redes neuronales en aceleradores analógicos imprecisos. Presentamos resultados de simulación aplicando A-Connect a modelos populares de DNN (por ejemplo, LeNet-5 para el conjunto de datos MNIST, AlexNet, VGG-16 y ResNet-20 para el conjunto de datos CIFAR-10, y ResNet-18 para el conjunto de datos CIFAR-100). A-Connect muestra el mejor rendimiento en comparación con otros enfoques *ex-situ*, al tiempo que presenta resultados comparables a métodos *in situ* e híbridos (es decir, utilizando enfoques *ex-situ* e *in situ*) en la literatura.

Luego, proponemos un macro para *Machine Learning* (ML) con computación en memoria (CIM)


[*] Tesis de Doctorado

[**] Facultad de Ingenierías Físico-Mecánicas. Escuela de Ingenierías Eléctrica, Electrónica y de Telecomunicaciones. Director: Elkim Felipe Roa Fuentes. PhD.

usando memoria SRAM, con un amplio rango de frecuencia y alta eficiencia energética para SoE multi-modo, que utiliza un enfoque de co-diseño de software-hardware con la ayuda de la metodología A-Connect. También presentamos un *datapath* completamente analógico, y de señal mezclada, que incorpora no solo operaciones MAC, sino también operaciones de ML comúnmente utilizadas dentro del dominio analógico (por ejemplo, ReLU, normalización, memoria). Las simulaciones presentadas en un nodo tecnológico CMOS de 180 nm muestran que los resultados del macro propuesto están cerca de los macros en 65 nm del estado del arte. Además, mostramos estimaciones de rendimiento para un diseño en 28 nm que sitúan al macro analógico propuesto por encima del rendimiento absoluto del estado del arte.

Continuamos con la propuesta de monitores de voltaje de múltiples niveles de ultra bajo consumo para estrategias de administración de energía de granularidad fina en una tecnología CMOS de 180 nm. También demostramos experimentalmente cómo estos monitores de voltaje podrían usarse en una estrategia real de gestión de energía en un sistema en chip (SoC) con un microcontrolador RISC-V. Al tener múltiples niveles para los umbrales de voltaje, es posible habilitar tres modos de energía diferentes que utilizan un suministro de voltaje más bajo: activo, *sleep* y *deep sleep*. En comparación con investigaciones anteriores que no consideran los efectos de baja temperatura al usar ramas de alta impedancia, este trabajo logra un bajo consumo de corriente en dichas condiciones.

Finalmente, exploramos mecanismos de vulneración de seguridad no convencionales en ataques por *hardware*. Presentamos nuestro trabajo sobre ataques por perturbación transistoria del voltaje de alimentación. Como contribución, logramos incluir la red de suministro de energía de un SoC en el enfoque clásico de violación de restricciones de tiempo, lo que nos permitió obtener una relación analítica entre el potencial de una perturbación de voltaje para inyectar una falla en un sistema y los parámetros de la forma de onda de la perturbación (por ejemplo, duración, amplitud). Anticipamos que nuestro trabajo permitiría un modelo de falla del sistema para cualquier forma de onda de perturbación, incluso aquellas generadas por algoritmos genéticos o redes neuronales.

# ABSTRACT

**TITLE:** TOWARDS INTELLIGENT, SECURE, AND ENERGY-EFFICIENT SYSTEMS-ON-EDGE [*]

**AUTHOR:** LUIS EDUARDO RUEDA GUERRERO [**]

**KEYWORDS:** System-on-edge, energy-efficiency, security, analog computation, stochastic variability, neural networks, machine learning, computation-in-memory, power management, voltage glitching attacks.

**DESCRIPTION:**

With billions (even trillions, according to estimations) of devices interconnected, power consumption, big data management, and security are some of the main challenges for IoT applications. Intelligent power management, based on supply monitors, is one of the main solutions in regard to lower power consumption. Meanwhile, inference on the edge with deep-learning systems arises as one of the most effective ways to deal with big data for decision-making. At the same time, to obtain energy-efficient deep-learning systems-on-edge (SoE), analog hardware acceleration has shown to be a promising alternative. Security is another one of the main challenges for SoE. With more connected nodes, more opportunities to break whole systems' security. The latter could lead to the filtration of sensitive information or leave the system vulnerable to attacks from different fronts.

This thesis presents contributions on the three fronts mentioned above: energy-efficiency SoE (power consumption), big data for decision-making SoE, and security infringement on SoE. First, we propose A-Connect, a novel methodology to improve neural network resilience against stochastic variability when deploying neural networks in imprecise analog accelerators. We present simulation results applying A-Connect to popular DNN models (e.g., LeNet-5 for the MNIST dataset, AlexNet, VGG-16, and ResNet-20 for the CIFAR-10 dataset, and ResNet-18 for the CIFAR-100 dataset). A-Connect shows the best performance when compared to other *ex-situ* approaches while having comparable results to *in situ* and hybrid (i.e., using *ex-situ* and *in situ* approaches) methods in the literature.

Then, we propose a wide frequency range and high energy efficiency CIM SRAM-based ML macro for multi-mode SoE, that uses a co-design software-hardware approach with the help of the A-Connect methodology. We also present an end-to-end analog datapath that incorporates not only MAC operations but commonly used ML operations within the analog domain (e.g.,

[*] PhD Thesis

[**] Facultad de Ingenierías Físico-Mecánicas. Escuela de Ingenierías Eléctrica, Electrónica y de Telecomunicaciones. Advisor:Elkim Felipe Roa Fuentes. PhD.

ReLU, normalization, memory). The simulations in a 180nm CMOS technology node show that the proposed macro's results are close to state-of-art macros in 65nm. Furthermore, we show performance estimations for a 28nm design that put the proposed analog macro above absolute state-of-art performance.

We continue with the proposal of ultra-low-power multi-level voltage monitors for multi-mode fine-grained power management strategies in a 180nm CMOS technology. We also show experimentally how these voltage monitors could be used in a real power management strategy in a system-on-chip (SoC) with a RISC-V MCU core. By having multi-level voltage thresholds we enable three different power modes that used lower voltage supply: active, sleep, and deep-sleep. In comparison to previous research that neglected considering the low-temperature effects when using large impedance branches, this work achieves a low current consumption in such conditions.

Finally, we explore unconventional security infringement mechanisms in hardware-based attacks. We present our work on voltage glitching attacks. As a contribution, we manage to include the power delivery network of an SoC in the classical timing constraint violation approach. Doing so allows us to obtain an analytical relation between the potential of a voltage glitch to inject a fault into a system and the glitch waveform parameters (e.g., duration, amplitude). We anticipate that our work would permit a system's fault model for any glitch waveform, even those generated by genetic algorithms or neural networks.

# 1. PROJECT OVERVIEW

## 1.1. Introduction

The Internet of Things (IoT) is among the most used buzzwords by technology companies and users. Still, it is a phenomenon we are experiencing, which has changed and will continue to shape how we live and interact with everything around us, from people to machines. IoT is the interconnection of physical devices, vehicles, smart buildings, embedded electronic systems, sensors, etc., that allows sharing, exchanging, collecting, and processing information between these objects. Although the term was first used by technological pioneer Kevin Ashton in 1999[1], the concept of combining computers with communication networks to monitor and control machines has been around for decades. For example, in 1970, the systems to monitor the energy consumption in buildings using telephone lines were already commercially available[2], and in 1990, with the progress of wireless networks, the machine-to-machine (M2M) communication was born. For the first time, a device (a toaster) was controlled (on and off) using commands sent through the internet protocol (IP).

There are tons of IoT devices in the market already. For example, sensors to measure vital signals while doing any physical activity (wearable systems)[3], or energy-saving systems for smart buildings or homes[4]. Any imaginable application can be transformed into an IoT system, turning this industry into one of the most promissory markets for the near future. Recent projections estimate that hundreds of thousands of IoT devices will be interconnected by 2025[5], with applications ranging from the healthcare sector (including wearable devices)[6] to entire connected cities (smart cities)[7]. In the same

[1] Kevin ASHTON et al. "That 'internet of things' thing". In: *RFID journal* 22.7 (2009), pp. 97–114.

[2] P. T. *Sensor monitoring device*. US Patent 3,842,208. Oct. 1974.

[3] S. M. R. ISLAM et al. "The Internet of Things for Health Care: A Comprehensive Survey". In: *IEEE Access* 3 (2015), pp. 678–708.

[4] J. PAN et al. "An Internet of Things Framework for Smart Energy in Buildings: Designs, Prototype, and Experiments". In: *IEEE Internet of Things Journal* 2.6 (Dec. 2015), pp. 527–537.

[5] *The Internet of Thinks: An Overview*. Accessed: 2016-10-17. Oct. 2015.

[6] P. A. LAPLANTE et al. "The Internet of Things in Healthcare: Potential Applications and Challenges". In: *IT Professional* 18.3 (May 2016), pp. 2–4.

[7] A. ZANELLA et al. "Internet of Things for Smart Cities". In: *IEEE Internet of Things Journal* 1.1 (Feb.

way, an economic growth of billions of dollars is expected for the year 2025[5]. Although many companies in the industry do not agree with exact numbers, with others not even optimistic about the hype around this movement, the forecasts show substantial growth and influence of this industry over society in general.

Due to its massive development and rapid growth, experts foresee this industry to face significant challenges, from the technical point of view, as well as ethical and social[5], creating different research branches focused on the solution of these problems[8]. There are three main challenges of great interest to this work:

**Creation of knowledge and big data:** In an IoT world, extraordinary quantities of unprocessed and noisy data are continuously collected. Developing techniques that allow us to convert this data into usable knowledge, and make proper inferences for decision-making, is crucial. Deep neural networks (DNNs) are foreseeing as one of the most promising alternatives to solve this problem[9].

**Security:** At the same time, users need to trust that their information (which in many cases could be very sensitive, e.g., healthcare industry) is safe from any external intruder who may steal it or perform an attack to the system. Devices and servers with low-security levels could be targeted as access points by these hackers, ending in catastrophic and lethal consequences.

**Energy-efficiency:** In the past, devices with power consumption on the order of micro-Watts ($10^{-6}$W) were considered ultra-low power. That has shifted for today and future systems-on-edge standards. Perhaps with tens of millions of interconnected devices, this power range is still acceptable (will be equivalent to consuming tens of Watts), but if we are expecting interconnections on the order of trillion devices, with each of them consuming on the order of micro-Watts, the total power consumption will be on the order of Mega-Watts ($10^{6}$W). On the other hand, tons of these systems-on-edge are expected to be in areas where continuous access will be difficult (e.g., within a patient's body). Hence, these

2014), pp. 22–32.

8 J. A. STANKOVIC. "Research Directions for the Internet of Things". In: *IEEE Internet of Things Journal* 1.1 (Feb. 2014), pp. 3–9.

9 H. LI et al. "Learning IoT in Edge: Deep Learning for the Internet of Things with Edge Computing". In: *IEEE Network* 32.1 (Jan. 2018), pp. 96–101.

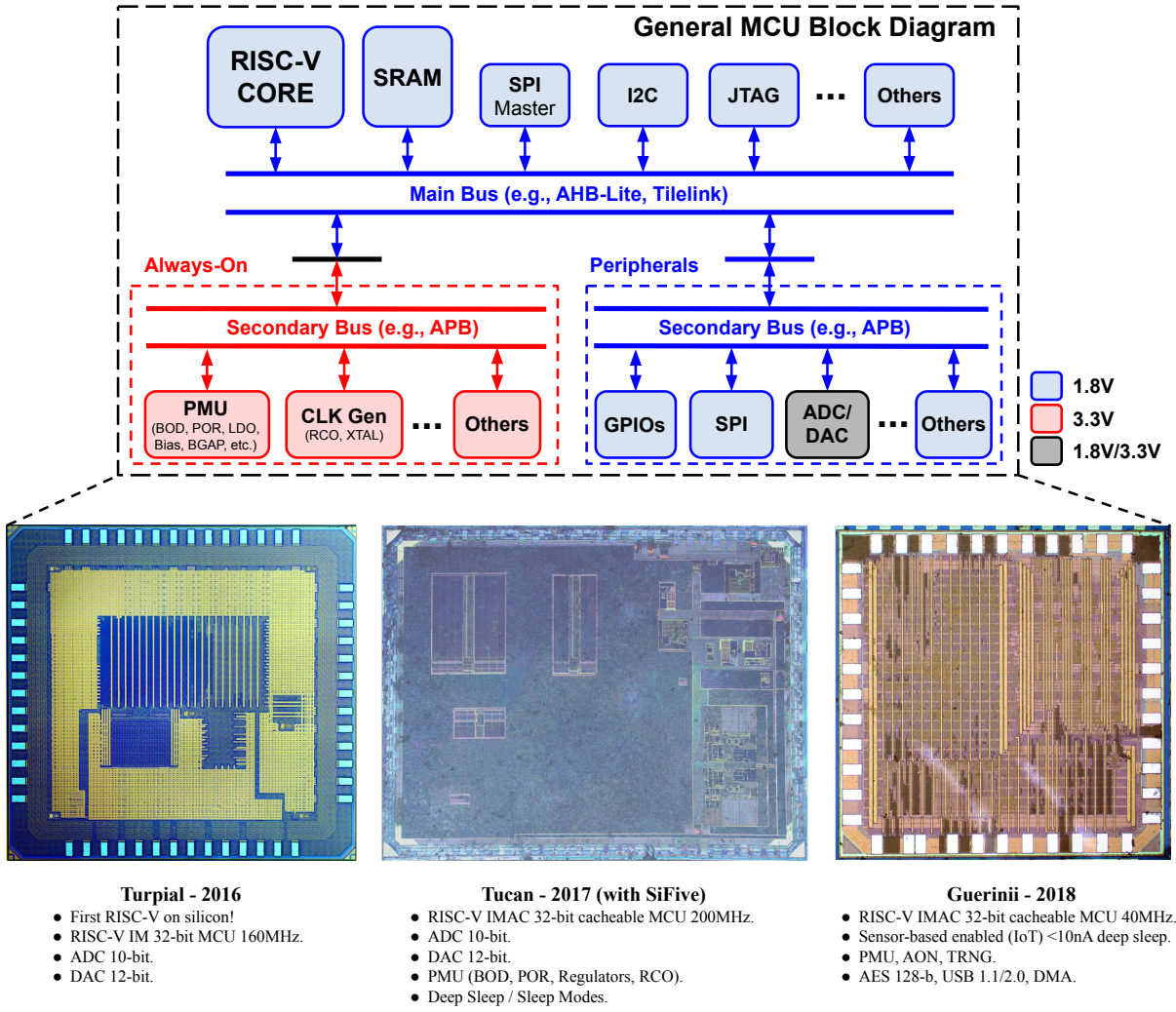


Figure 1. Three MCU generations: Turpial (first ever RISC-V on silicon), Tucan (in collaboration with SiFive), and Guerinii (IoT platform).

systems need a long life span, translating to ultra-low power consumption. For these systems-on-edge, nano-Watts ($10^{-9}$W) or even pico-Watts ($10^{-12}$W) range are needed.

The idea behind this thesis is to devise possible solutions to these challenges and to envision a new wave of problems regarding intelligent, secure, and energy-efficient systems on edge.

## 1.2. A RISC-V based SoC platform for systems-on-edge

To talk about this work, it is necessary to talk about its foundation and the required infrastructure for its realization. The conception of the ideas for this thesis came from

the work that has been done in the research group OnChip for the last seven years, in which I have been one of the leaders: the creation of an SoC platform for systems-on-the-edge, better known as the internet of things (IoT)[10,11,12].

IoT platforms capable of processing big data for decision-making or providing a secure interface when sharing information while keeping low-cost, high-speed, and low-power features are necessary to fulfill the market needs. The problem arises when licensed hardware increases the costs and restricts the process of modifying platform cores for different purposes, such as enhancing performance and adapting it to other IoT applications.

OnChip has adopted an open-source hardware (OSH) platform based on the instruction set architecture RISC-V[13] to solve the aforementioned problem. OnChip is a pioneer in the OSH community and the first-ever to fabricate on silicon a functional RISC-V based microcontroller[14,15,16].

Through the course of seven years, the RISC-V community has grown at an accelerated pace ($>$300 members), attracting huge names within its ranks, such as[17]: Google, Nvidia, Samsung, Western Digital, among others. In the same period, OnChip has designed and fabricated three generations of RISC-V based microcontrollers for IoT applications. Figure 1 shows the three generations of microcontrollers units (MCUs), their key characteristics, as well as a general block diagram of the platform:

- **Turpial[10,11]:** Turpial is the first silicon-proven 32-bit microcontroller with a RISC-V instruction set in the world. It is a 32-bit microcontroller fabricated in a 130nm

[10] C. DURAN et al. "A 32-bit RISC-V AXI4-lite bus-based microcontroller with 10-bit SAR ADC". in: *2016 IEEE 7th Latin American Symposium on Circuits Systems (LASCAS)*. Feb. 2016, pp. 315–318.

[11] C. DURAN et al. "A system-on-chip platform for the internet of things featuring a 32-bit RISC-V based microcontroller". In: *2017 IEEE 8th Latin American Symposium on Circuits Systems (LASCAS)*. Feb. 2017, pp. 1–4.

[12] C. DURAN et al. "An Energy-Efficient RISC-V RV32IMAC Microcontroller for Periodical-Driven Sensing Applications". In: *2020 IEEE Custom Integrated Circuits Conference (CICC)*. 2020, pp. 1–4.

[13] Andrew WATERMAN et al. *The RISC-V Instruction Set Manual, Volume I: User-Level ISA, Version 2.1*. Tech. rep. UCB/EECS-2016-118. EECS Department, University of California, Berkeley, May 2016.

[14] EENEWS EUROPE. *RISC-V MCU grown in Colombia*. Sept. 2016.

[15] NAVA WHITEFORD. *A COMPLETELY OPEN MICROCONTROLLER*. Oct. 2016.

[16] BRIAN BENCHOFF. *OPEN-V, THE FIRST OPEN SOURCE RISC-V MICROCONTROLLER*. Nov. 2016.

[17] RISC-V FOUNDATION. *Members at a Glance*. 2020.

CMOS technology, mounted through AXI4-Lite and APB buses for communication. The microcontroller contains a 10-bit SAR ADC, a 12-bit DAC, an 8-bit GPIO module, a 4kB-RAM, an SPI AXI slave interface for output verification, and an SPI APB slave interface for checking the correct behavior of the APB bridge. The RISC-V and SPI AXI master interface (used for programming the device and checking the data flowing through all the slaves) control all peripherals. We reported a total power density of 167$\mu$W/MHz. The area for this RISC-V microcontroller was around 2.1mm × 2.1mm.

- **Tucan**[12]**:** Tucan is our second 32-bit RISC-V IMAC-based microcontroller (MCU) in a 180nm CMOS technology. We developed this microcontroller in collaboration with the silicon valley company SiFive. It features a low-energy always-on (AON) subsystem extending minimum-energy (ME) adaption by including peripherals. Reported work on ME computing for low-power applications was focused on tracking the microprocessor ME voltage supply. However, using low-power systems requires accounting for regulator losses, voltage monitors, biasing, peripheral, clock sources, and start-up energies to adapt the correct ME supply to different operation modes. We developed an MCU with low-energy clock sources and voltage monitors that enabled 32.768kHz to 55MHz operation and power-gate the MCU into three power states adjusted to work at the ME supply operation. Measured start-up energies using integrated RC-based oscillators showed restarting energies down to 6pJ, which is 1000X less than the energy required in MCUs that apply crystal oscillators. AON peripherals enabled the MCU for low-duty-cycle sensor node applications.
- **Guerinii:** Guerinii is our third and last 32-bit RISC-V IM-based microcontroller unit. It is an MCU shielded against supply voltage attacks while apprising SoC interoperation. The MCU peripherals comprise a 145nW multi-level brown-out detector, repurposed as a supply glitch detector, to identify different amplitude glitches and security instances. Such instances include a 10.8pJ/bit true random seed generator block and a 3.58pJ/bit 256b advanced encryption standard (AES) substitution box (Sbox) accelerator.

Turpial was our starting point, the one that helped us understand the difficulties of designing microcontrollers, as well as the responsible for us gaining international attention. Because of Turpial, the collaboration with SiFive was possible: Tucan. With Tucan, we started to test ideas for power management strategies for energy-efficient systems focused on IoT. With Tucan, we also investigated conventional security mechanisms for SoE, such as the generation of secure keys with the design of a true random number generator (TRNG)[18,19]. Then we designed Guerinii, an MCU thought for IoT applications. In Guerinii, we continued developing power management strategies with even lower energy consumption. In terms of security, we expanded our portfolio: in addition to continuing with our work in TRNGs, we also tested other conventional mechanisms with physical unclonable functions (PUFs)[20], as well as our first custom instruction-based accelerator for advanced encryption standard AES-256[21,22].

Having in mind all the research the OnChip group has done, we wanted to portray in this thesis some of my contributions through the years. In particular, we wanted to include my work on designing crucial blocks that enable the development of different power management strategies for systems-on-edge, as well as my work on understanding unconventional mechanisms to infringe SoE security, such as power supply glitching. We even wanted to go further, thinking about the next generations of MCUs. Inspired by the custom instruction-based AES acceleration, we also investigate how to apply traditional computer architecture techniques to mixed-signal machine learning accelerators for decision-making intelligent SoEs. Considering all of the above, the following section presents this thesis' goals and the outline of this work.

[18] Juan CARTAGENA et al. "A fully-synthesized TRNG with lightweight cellular-automata based post-processing stage in 130nm CMOS". in: *2016 IEEE Nordic Circuits and Systems Conference (NORCAS)*. 2016, pp. 1–5.

[19] Hector GOMEZ et al. "Low-cost TRNG IPs". In: *IET Circuits, Devices & Systems* 14.7 (Oct. 2020), pp. 942–946.

[20] Javier ARDILA et al. "A Stable Physically Unclonable Function Based on a Standard CMOS NVR". in: *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*. 2020, pp. 1–4.

[21] Ckristian DURAN et al. "AES Sbox Acceleration Schemes for Low-Cost SoCs". In: *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*. 2021, pp. 1–5.

[22] Ckristian DURAN et al. "A 10pJ/bit 256b AES-SoC Exploiting Memory Access Acceleration". In: *IEEE Transactions on Circuits and Systems II: Express Briefs* 69.3 (2022), pp. 1612–1616.

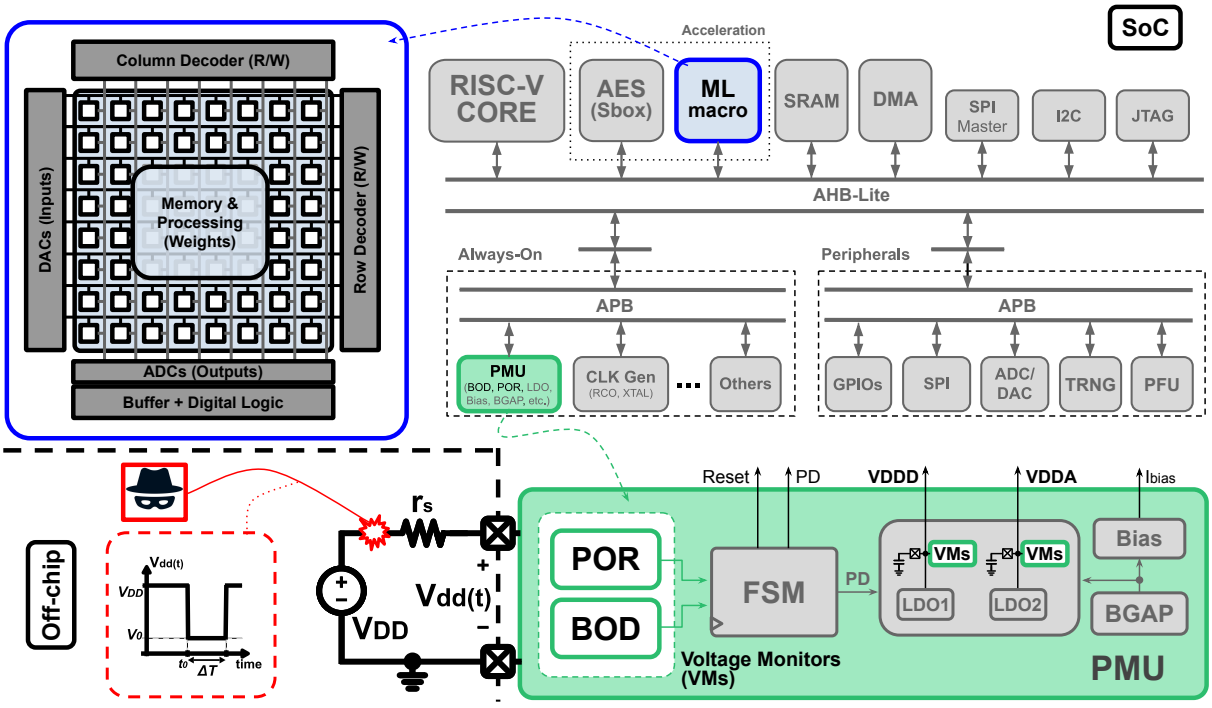


Figure 2. Thesis contributions summary: a) feasibility of applying in mixed-signal accelerators traditional computer architecture and ML techniques; b) mixed-signal power management strategies for energy-efficient SoE; c) impact of power supply glitching as a way to infringe SoE security.

## 1.3. Dissertation Goals and Outline

### 1.3.1. Project Goal

To devise solutions in unconventional domains (analog-mixed signal) to some of the conventional problems regarding IoT challenges, as well as to envision the rise of new challenges in secure and efficient systems-on-edge. In security, attack actions on already implemented SoCs will be performed to study their nature and to be able to propose new ways to counter them. In regards to energy efficiency, power-management schemes will be tested on already implemented SoCs to study their effectiveness, as well as to propose new power-management strategies. Finally, to further improve the energy efficiency of systems-on-edge with decision-making capabilities, machine-learning accelerator architectures in the analog-mixed signal domain will be explored.

### 1.3.2. Dissertation Outline

We decided to dissect the main goal in the following three objectives, which helped us to accomplish the completion of this thesis:

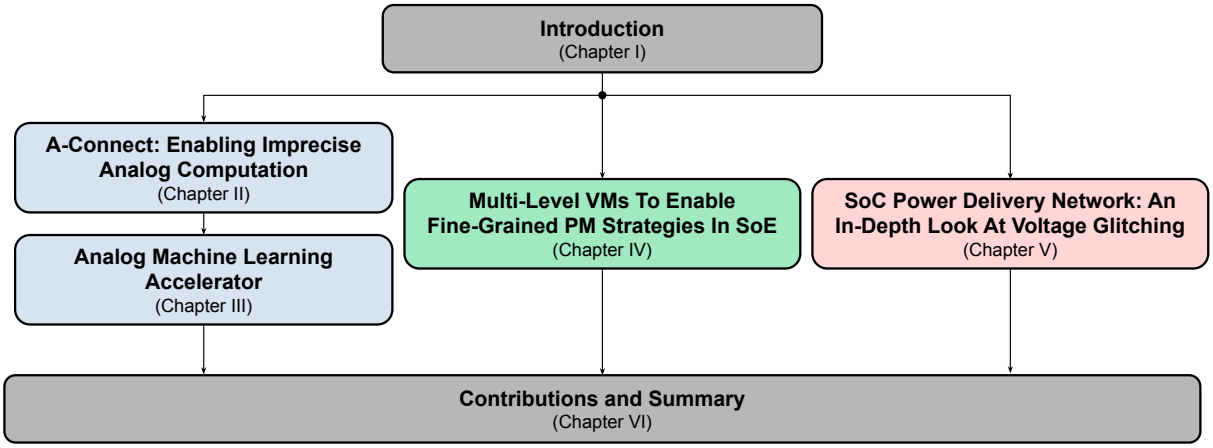


Figure 3. Dissertation roadmap.

- To explore the feasibility of applying in mixed-signal accelerators traditional computer architecture and machine learning techniques. To improve the energy efficiency of systems-on-edge with decision-making capabilities, machine-learning accelerator architectures in the analog-mixed signal domain will be explored.
- To devise mixed-signal power management strategies for energy-efficient systems-on-edge. Power-management schemes will be tested on already implemented SoCs to study their effectiveness, as well as to propose new power-management strategies.
- To study the impact of power supply glitching as a way to infringe systems-on-edge security. Attack actions on already implemented SoCs will be performed to study their nature, in order to be able to propose new ways to counter them.

During the course of the project, the objectives were transformed into milestones and now constitute both the contributions of my work and the chapters of this book. I present in Figure 2 the main contributions of my thesis, placing special emphasis on how these contributions fit into the research work of the OnChip group shown before. To conclude this introduction, Figure 3 shows the roadmap of this thesis dissertation. I give a brief abstract of each chapter in the following paragraphs so that the reader can have an idea of what can be found in this work:

**Chapter I: "A-Connect: Enabling Imprecise Analog Computation"**

*Objective: to explore the feasibility of applying in mixed-signal accelerators traditional*

*computer architecture and machine learning techniques*

Analog-based neural network accelerators outperform digital-based accelerators in energy efficiency by trading accuracy. Analog computation is susceptible to hardware stochastic variability, incurring limited signal-to-noise and aggravating for compact and low-power applications. Chapter 2 introduces A-Connect, an *ex-situ* statistical methodology to improve analog neural network resilience against stochastic variability. Our methodology achieves state-of-art performance in analog environments with heavy stochasticity levels by injecting noise during the neural network forward propagation and considering the same injected noise during the backward propagation. Furthermore, we developed a Keras/Tensorflow library with fully-connected and convolutional layers versions using our training methodology, which can be coupled easily to standard machine learning platforms. We present simulation results applying A-Connect to popular DNN models, like LeNet-5 for the MNIST dataset, AlexNet, VGG-16, and ResNet-20 for the CIFAR-10 dataset, and ResNet-18 for the CIFAR-100 dataset. When validating the CIFAR-10 or CIFAR-100 recognition tasks, the results with the A-Connect methodology showed an improvement over the baseline model of around 15 to 68 percentage points for the median accuracy at a 70% of stochastic variability. The deviation of the results with A-Connect is around 20X lower than the baseline at this level of stochasticity. A-Connect also showed the best performance when compared to other *ex-situ* approaches while having comparable results to *in situ* and hybrid (i.e., using *ex-situ* and *in situ* approaches) methods in the literature. We anticipate that the A-Connect methodology could enable emergent memory technologies, such as ReRAM and PCM, for accurate computation-in-memory applications.

**Chapter III: "Analog Machine Learning Accelerator"**

Once we tackled the problem of hardware stochastic variability in analog-based neural network accelerators with the A-Connect methodology, we continue with our hardware implementation proposal in chapter 3: a wide frequency range and high energy efficiency CIM SRAM-based ML macro for multi-mode systems-on-edge. The proposed analog macro can perform at high energy efficiency by following two principles: avoiding data conversion by staying in the same physical domain (i.e., current) and the use

of simplified and low-area circuits by using co-design software strategies that mitigate stochastic and deterministic errors (i.e., the A-Connect methodology in chapter 2). We propose an end-to-end analog datapath that incorporates not only MAC operations but commonly used ML operations within the analog domain, such as ReLU and scaling (the latter enabled normalization operations), as well as memory capabilities for pipeline execution. Since all analog operations in our macro are current-based, we implement a wideband current mirror that enables a wide range of operating frequencies while improving energy efficiency. The simulation results, presented in a 180nm CMOS technology node, show that the analog macro performed at a wide range of frequencies (200kHz-15MHz) over an ultra-low and broad range of current levels (i.e., 1nA to 100nA biasing) while maintaining a relatively similar energy efficiency (760-1076 1b-TOPS/W). When compared to other works, the proposed macro's results were compatible with state-of-art macros in 65nm. Furthermore, we showed performance estimations for a 28nm design that put the proposed analog macro above absolute state-of-art performance. To our knowledge, our work is the only study investigating multi-mode ML accelerators performing efficiently at different current levels and clock rates.

**Chapter IV: "Multi-Level Voltage Monitors to Enable Fine-Grained PM Strategies in SoE"**

*Objective: to devise mixed-signal power management strategies for energy-efficient systems-on-edge*

In chapter 4, we present ultra-low-power multi-level voltage monitors for multi-mode fine-grained power management strategies. Simulation results over PT variations, as well as measurements at nominal temperature, showed a robust performance within the industrial temperature range from -40°C to 125°C and a wide supply rise and fall times, ranging from 1us to 1s. In the first version of the POR (POR1), we obtained a current consumption of 7$\mu$A. In the second version (POR2), the POR had a nominal current consumption of 19nA. Both PORs had up to 3 different voltage threshold levels. In regards to the BOD, we presented an architecture with low-temperature slew compensation for low-power applications, multiple voltage threshold levels, and current consumption of 200nA. We also showed experimentally how these voltage monitors

could be used in a real power management strategy. By having multi-level voltage thresholds we enabled three different power modes that used lower voltage supply: active, sleep, and deep-sleep. According to measurements, the SoC had an 8mA current consumption at 16MHz in active mode, 27.5$\mu$A at 32.768kHz in sleep mode, and 530nA at 32.768kHz in deep-sleep mode. In comparison to previous research that neglect to consider the low-temperature effects when using large impedance branches, this work achieved a low current consumption even by considering these temperature effects. Current consumption, programmability, and reduced area make the proposed voltage monitors enablers of different fine-grained power management schemes.

**Chapter V: "System-on-Chip Power Delivery Network: An in-Depth Look at Voltage Glitching"**

*Objective: to study the impact of power supply glitching as a way to infringe systems-on-edge security*

As we stated before, the group OnChip has been investigating conventional security mechanisms for system-on-edge in the past, from the generation of secure keys and unclonable functions (e.g., TRNG and PUF circuits) to the acceleration of the encryption of data through AES. We wanted to investigate in this work a more unconventional way to cause fault injections within SoCs that did not involve software-based attacks. Chapter 5 presents our work on voltage glitching, one of the most researched fault injection mechanisms in systems-on-chip (SoC) at a hardware level. The easiness of execution and permanent availability of an external pin for the power supply make glitching injection one of the preferred methods for security tampering. Previous works have provided experimental evidence demonstrating that voltage glitching fault injections cause time constraint violations. However, there is still a lack of understanding of the voltage glitching nature, which has prevented obtaining a direct link between the glitch characteristics and the likelihood of the glitch injecting a fault into a system. In Chapter 5, we include the power delivery network in the timing constraint violation approach. We derive expressions and analyses relating glitch waveform parameters (e.g., duration and amplitude) with the fault injection potential of that voltage glitch. We present simulation results and measurements of over 4500 experiments attacking an

in-house RISC-V MCU across multiple glitch voltage amplitudes and MCU's operating frequencies, supporting our findings. We foresee that the analyses and results in this chapter will allow designers to fully characterize SoC against voltage glitching fault injection without restricting the characterization to only squared pulse glitches. For example, our approach could permit a system's fault model for any glitch waveform, even those generated by genetic algorithms or neural networks.

# 2. A-CONNECT: ENABLING IMPRECISE ANALOG COMPUTATION

## 2.1. Introduction

The new wave of neural networks in the last decade placed computation-in-memory (CIM) architectures as one of the most effective approaches to solve the energy cost problem of data movement (the von Neumann bottleneck[23]). To further increase the energy efficiency, recent works have combined CIM architectures with analog processing/memory technologies that act like the synapses of the neural network[24] (Figure 4(a) shows some examples of these technologies). Figure 4(b) shows the common crossbar analog-based CIM architecture. Each memory/processing unit (squares with different shades of red) is typically composed of an analog-based non-volatile memory (NVM) technology, e.g., resistive random-access memory (ReRAM)[25,26], and phase-change memory (PCM)[27]. There is also a recent interest in hybrid approaches that use SRAM extended cells (more than six transistors) to execute analog operations within the same cell[28,29,30].

The energy-efficiency performance achieved by the latest works in analog-based ac-

[23] John BACKUS. "Can Programming Be Liberated from the von Neumann Style? A Functional Style and Its Algebra of Programs". In: *Commun. ACM* 21.8 (Aug. 1978), 613–641.

[24] Hsinyu TSAI et al. "Recent Progress in Analog Memory-Based Accelerators for Deep Learning". In: *Journal of Physics D: Applied Physics* (June 2018).

[25] S. YIN et al. "Monolithically Integrated RRAM- and CMOS-Based In-Memory Computing Optimizations for Efficient Deep Learning". In: *IEEE Micro* 39.6 (Nov. 2019), pp. 54–63.

[26] C. XUE et al. "Embedded 1-Mb ReRAM-Based Computing-in- Memory Macro With Multibit Input and Weight for CNN-Based AI Edge Processors". In: *IEEE JSSC* 55.1 (Jan. 2020), pp. 203–215.

[27] W. KIM et al. "Confined PCM-based Analog Synaptic Devices offering Low Resistance-drift and 1000 Programmable States for Deep Learning". In: *2019 Symposium on VLSI Technology*. June 2019, T66–T67.

[28] P. SRIVASTAVA et al. "PROMISE: An End-to-End Design of a Programmable Mixed-Signal Accelerator for Machine-Learning Algorithms". In: *ACM/IEEE 45th Annual ISCA*. June 2018, pp. 43–56.

[29] M. KANG et al. "An Energy-Efficient VLSI Architecture for Pattern Recognition via Deep Embedding of Computation in SRAM". in: *IEEE ICASSP*. May 2014.

[30] X. SI et al. "A Twin-8T SRAM Computation-in-Memory Unit-Macro for Multibit CNN-Based AI Edge Processors". In: *IEEE JSSC* (Jan. 2020).

[31] Thomas DALGATY et al. "In Situ Learning Using Intrinsic Memristor Variability via Markov Chain Monte Carlo Sampling". In: *Nature Elect.* (Jan. 2021).

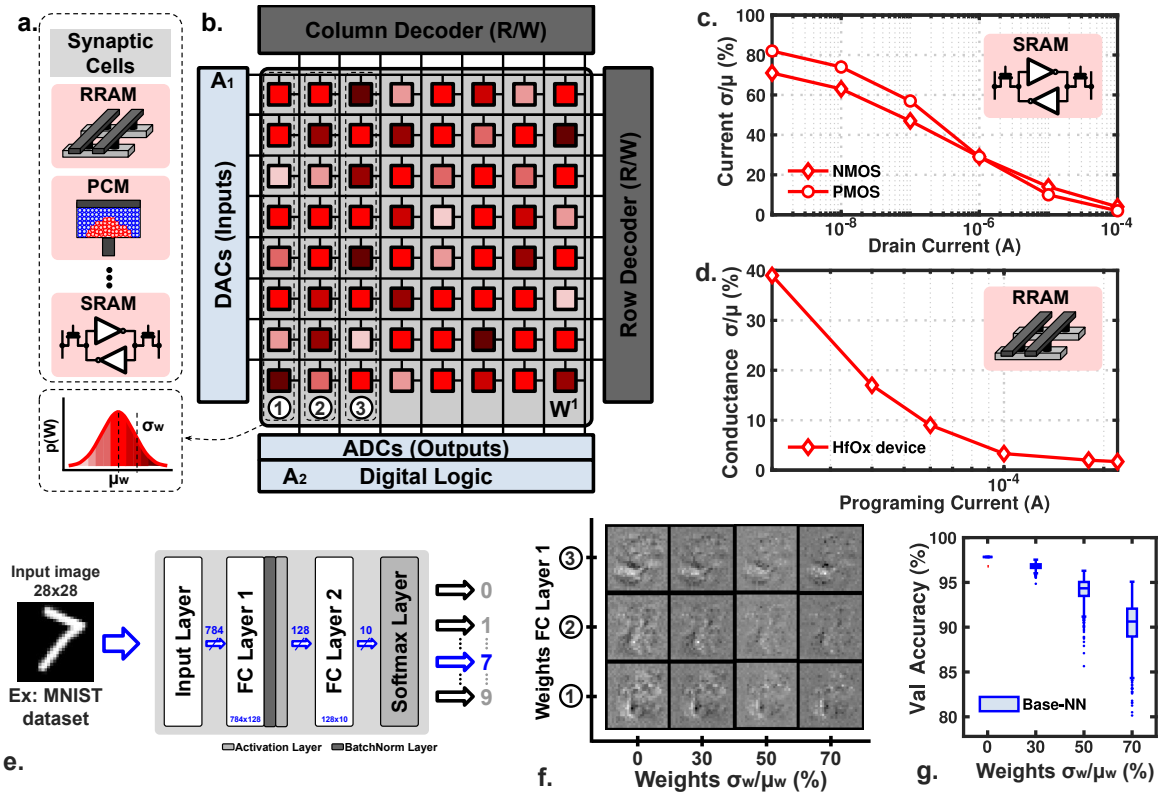


Figure 4. Computation in Memory (CIM) with analog memory technologies. a) Examples of analog synaptic cells. b) Analog-based CIM architecture with probability density function of the stochastic variability in synaptic cells (bottom-left corner). c) Drain current stochastic variability values (standard deviation) for the smallest devices in a commercial CMOS 180nm technology (data obtained from Monte Carlo simulations). d) Programmed conductance stochastic variability for a hafnium-dioxide-based (HfOx) random access memory (data obtained from [31]). e) Example of a two fully-connected layers neural network for the MNIST handwritten dataset. f) Effect of the analog synaptic cells stochastic variability on the first three neurons weights of the first neural network layer (see b. and e.). The weights are in float32 precision format. g) Effect of the analog synaptic cells stochastic variability on the neural network (see e.) test accuracy.

celerators outperforms their digital counterparts by factors ranging between 10X to 100X[32,33,34], which is one of the reasons for the latent interest in this field. On the other hand, the performance in terms of accuracy is generally better in digital accelerators than in analog ones. Digital computation can achieve high signal-to-noise and distortion ratio, while analog computation is very susceptible to hardware non-idealities.

[32] H. VALAVI et al. "A 64-Tile 2.4-Mb In-Memory-Computing CNN Accelerator Employing Charge-Domain Compute". In: *IEEE JSSC* (2019).

[33] W. KHWA et al. "A 65nm 4Kb Algorithm-Dependent Computing-in-Memory SRAM Unit-Macro with 2.3ns and 55.8TOPS/W Fully Parallel Product-Sum Operation for Binary DNN Edge Processors". In: *IEEE ISSCC*. Feb. 2018.

[34] D. MIYASHITA et al. "Time-Domain Neural Network: A 48.5 TSOp/s/W Neuromorphic Chip Optimized for Deep Learning and CMOS Technology". In: *IEEE A-SSCC*. Nov. 2016.

Commonly, analog circuit non-idealities fall into two distinctive groups: deterministic and stochastic (random) non-idealities. In regards to deterministic non-idealities in analog accelerators for inference, one of the most prominent is the non-linearity of the synaptic cells[24]. Fortunately, recent works have demonstrated that if the actual non-linear model (or an approximation) of the synapse is used instead of the ideal multiply-accumulate operation, it is possible to achieve almost complete immunity in terms of performance[35].

Stochastic variability, either temporal (noise) or spatial (mismatch), is the other type of analog non-ideality that worsens signal-to-noise and distortion ratio. Although neural networks have shown some intrinsic immunity to noisy environments[24], such as analog accelerators, either the performance is still not comparable to digital counterparts, or the techniques employed are impractical in many cases. The effect of stochastic variability is even worse for smaller and ultra-low power consumption synaptic cells, which are fundamental towards more compact and energy-efficient DNN accelerators.

Figures 4(c)-(d) show the spatial stochastic variability in two synaptic cell analog/hybrid technologies: Figure 4(c) shows the drain current stochasticity with respect to the absolute drain current value in a commercial 180nm CMOS technology; Figure 4(d) shows the stochasticity in the conductance value in a ReRAM technology (hafnium-dioxide-based RAM) with respect to the current employed for programming a single cell[36]. It is possible to see how on different synaptic cell technologies, the stochastic variability increases when working in the low-power regime. Other works using ReRAM technologies as synaptic cells show similar behaviour of the stochastic variability with respect to the current through the devices[37,38,39,40], while the stochasticity gets worse

[35] Hyungjun KIM et al. “Deep Neural Network Optimized to Resistive Memory with Nonlinear Current-Voltage Characteristics”. In: *JETCS* 2 (July 2018).

[36] Thomas DALGATY et al. “In Situ Learning Using Intrinsic Memristor Variability via Markov Chain Monte Carlo Sampling”. In: *Nature Elect.* (Jan. 2021).

[37] Bin GAO et al. “Ultra-Low-Energy Three-Dimensional Oxide-Based Electronic Synapses for Implementation of Robust High-Accuracy Neuromorphic Computation Systems”. In: *ACS Nano* (June 2014).

[38] Qiangfei XIA et al. “Memristive Crossbar Arrays for Brain-Inspired Computing”. In: *Nature Materials* 18.4 (Mar. 2019), pp. 309–323.

[39] Shinhyun CHOI et al. “Data Clustering using Memristor Networks”. In: *Scientific Reports* 5.1 (May 2015).

[40] S. AMBROGIO et al. “Statistical Fluctuations in HfOx Resistive-Switching Memory: Part I - Set/Reset

with more advanced CMOS technology nodes (e.g., 7nm and beyond)[41,42,43]. These technological walls limit the achievable energy-efficiency of an analog accelerator.

Figure 4(e) shows an example with two fully-connected layers for the MNIST handwritten digit classification problem, which will help understand the effect of the synaptic cells' stochastic variability on the learned parameters of a neural network. Figure 4(f) shows the rearranged floating-point 32-bits precision weights corresponding to the first three neurons of the first layer. From this figure, it is possible to distinguish some patterns from the learned weights. These patterns are less identifiable when adding more stochastic variability, to the point that they are almost lost when considering high stochasticity levels ($\sigma_w / \mu_w \geq$30%).

To quantify the effect of the synaptic cells' stochastic variability in the neural network performance, Figure 4(g) shows the test accuracy of the trained neural network in Figure 4(e). The performance results were obtained using floating-point 32-bits precision weights. The results show that the stochastic variability has a crucial impact on the network accuracy, with a decrease of around eight percentage points (from 98% to 90%) in the median values for the floating-point 32-bits precision weights when using a stochasticity level of 70%. When considering the deviation of the results, the accuracy levels can get as low as 75%.

Towards enabling energy-efficient and compact imprecise analog DNN accelerators, such as ReRAM-based accelerators, this work focuses on increasing the accuracy resilience to synaptic cells' stochastic variability. This work makes the following contributions:

- We introduce A-Connect, an *ex situ* statistical training methodology to mitigate analog computation stochastic variability in neural networks, like the ones caused by mismatch and noise in the synaptic cells. We provide simulation results where

Variability". In: *IEEE Trans. on Electron Devices* (2014).

41 Qiang HUO et al. "Physics-Based Device-Circuit Cooptimization Scheme for 7-nm Technology Node SRAM Design and Beyond". In: *IEEE Transactions on Electron Devices* 67.3 (2020), pp. 907–914.

42 Sourav DE et al. "Neuromorphic Computing with Fe-FinFETs in the Presence of Variation". In: *2022 International Symposium on VLSI Technology, Systems and Applications (VLSI-TSA)*. 2022, pp. 1–2.

43 N. RONCHI et al. "A Comprehensive Variability Study of Doped $HfO_2$ FeFET for Memory Applications". In: *2022 IEEE International Memory Workshop (IMW)*. 2022, pp. 1–4.

popular DNNs are deployed in an imprecise environment (e.g., analog hardware).[i]

- We developed a library with the A-Connect methodology[44]. The library is available for Keras/TensorFlow, and contains the A-Connect modified versions of fully-connected and convolutional layers.

## 2.2. Related Work

In regards to stochasticity in analog accelerators, it is possible to divide research works into two categories: those that try to exploit the inherent stochastic behavior of analog computation and those that try to mitigate this effect. Ensemble learning[45] and extreme learning machines (ELMs)[46] are examples of the former. In ensemble learning, the idea is to achieve better accuracy performance by using several learners (possibly inaccurate). Bagging[47] and Boosting[48] are two examples of ensemble learning algorithms. There have been works focusing on the implementation of weak classifiers to accelerate ensemble learning algorithms at hardware level[49], as well as algorithmic proposals applying Boosting techniques[50]. Along with ensemble learning, extreme learning machines are the other type of neural networks that can exploit stochasticity. ELMs are feedforward neural networks with single or multiple hidden layers with parameters that do not need to be tuned, nor updated, and in many cases randomly assigned. Because of this capability, ELMs are a perfect fit for imprecise analog ac-

[44] Luis E. RUEDA G et al. *A-Connect for TensorFlow*. [Online] Available: https://github.com/onchipuis/A-Connect. 2021.

[45] D. OPITZ et al. "Popular Ensemble Methods: An Empirical Study". In: *Journal of Artificial Intelligence Research* (Aug. 1999).

[46] Guang-Bin HUANG et al. "Extreme Learning Machine: Theory and Applications". In: *Neurocomputing* 70.1-3 (2006), pp. 489–501.

[47] Leo BREIMAN. "Bagging Predictors". In: *Mach. Learn.* (Aug. 1996).

[48] Robert E. SCHAPIRE. "The Strength of Weak Learnability". In: *Mach. Learn.* 5.2 (July 1990), 197–227.

[49] J. ZHANG et al. "In-Memory Computation of a Machine-Learning Classifier in a Standard 6T SRAM Array". In: *IEEE JSSC* (Apr. 2017).

[50] Z. WANG et al. "Error Adaptive Classifier Boosting (EACB): Leveraging Data-Driven Training Towards Hardware Resilience for Signal Inference". In: *IEEE TCAS-I* (2015).

[i] Although we only provide evidence for spatial stochasticity mitigation, our methodology can be used for temporal stochastic variability as well.

celeration, exploiting device-to-device mismatch and noise[51,52,53]. The problem with these accelerators is that they can only execute ELM algorithms and not other types of artificial neural networks.

Instead of exploiting stochasticity and limit the acceleration to ELM algorithms, there are works where the complete focus is the mitigation of synaptic cell's stochastic variability. The idea is to match analog accelerators performance with their digital counterparts. Although the chosen technology in the majority of works found in this respect is the ReRAM-based (memristor) synaptic cell, their contributions and conclusions can be generalized to any analog accelerator. Different mitigation approaches can be distinguished in these works: *ex situ*[54,55,56,57,58,59,60], *in situ*[61,62], or a combination of both

[51] Y. CHEN et al. "A 128-Channel Extreme Learning Machine-Based Neural Decoder for Brain Machine Interfaces". In: *IEEE TBioCAS* (2016).

[52] O. RICHTER et al. "Device Mismatch in a Neuromorphic System Implements Random Features for Regression". In: *IEEE BioCAS*. 2015.

[53] A. TRIPATHI et al. "Analog Neuromorphic System Based on Multi Input Floating Gate MOS Neuron Model". In: *IEEE ISCAS*. May 2019.

[54] A.F. MURRAY et al. "Enhanced MLP performance and fault tolerance resulting from synaptic weight noise during training". In: *IEEE Transactions on Neural Networks* 5.5 (1994), pp. 792–802.

[55] Y. LONG et al. "Design of Reliable DNN Accelerator with Un-reliable ReRAM". in: *DATE*. 2019.

[56] Sanjay KARIYAPPA et al. "Noise-Resilient DNN: Tolerating Noise in PCM-Based AI Accelerators via Noise-Aware Training". In: *IEEE Trans. on Electron Devices* (2021).

[57] Z. HE et al. "Noise Injection Adaption: End-to-End ReRAM Crossbar Non-ideal Effect Adaption for Neural Network Mapping". In: *ACM/IEEE DAC*. 2019.

[58] Y. ZHU et al. "Statistical Training for Neuromorphic Computing using Memristor-based Crossbars Considering Process Variations and Noise". In: *DATE*. 2020.

[59] Vinay JOSHI et al. "Accurate deep neural network inference using computational phase-change memory". In: *Nature Communications* 11.1 (May 2020).

[60] Julian BÜCHEL et al. "Network Insensitivity to Parameter Noise via Parameter Attack During Training". In: *International Conference on Learning Representations (ICLR)*. 2022.

[61] M. HU et al. "Memristor Crossbar-Based Neuromorphic Computing System: A Case Study". In: *IEEE TNNLS* (2014).

[62] A. MOHANTY et al. "Random Sparse Adaptation for Accurate Inference with Inaccurate Multi-Level RRAM Arrays". In: *IEEE IEDM*. 2017.

(hybrid)[63,64,65,66,67,68,69,70]. In the *ex situ* approaches, the neural network parameters are trained using a software-based model of the accelerator, while *in situ* approaches use the actual hardware where the neural network will be deployed. In terms of inference performance, *in situ* methods can achieve better results than *ex situ* since the former does consider the actual behavior of the synaptic cells. The problem with the *in situ* approach is the cost associated with its implementation (e.g., high-resolution data-converters, complex feedback control)[64], which makes it impractical in some cases. In this respect, *ex situ* approaches may be preferred since they avoid the hardware overhead that comes with *in situ* training. We will focus our attention on *ex situ* training approaches. Among the different works related to *ex situ* training, we have identified at least four different subclasses that are not mutually exclusive:

**Parameter noise-injection training:** The idea in *ex situ* approaches is to consider the synaptic cells' stochastic variability through statistical training, which is commonly achieved by injecting some signal corruption (e.g., noise) into the training data or directly over the neural network parameters (e.g., weights, biases). With this, the neural network is regularized, which makes it robust against stochastic non-idealities. There are two types of parameter noise-injection techniques: additive and multiplicative.

*Additive noise-injection:* This technique directly adds noise to the NN parameters (e.g., $W = W_0 + \Delta W$) during training (forward propagation phase) to mitigate stochastic variability[57 ,59]. The results in these works showed that additive noise-injection is very

[63] Fabien ALIBART et al. "Pattern Classification by Memristive Crossbar Circuits Using ex situ and in situ Training". In: *Nature Comm.* (June 2013).

[64] B. LIU et al. "Vortex: Variation-Aware Training for Memristor X-Bar". In: *ACM/EDAC/IEEE DAC*. 2015, pp. 1–6.

[65] L. CHEN et al. "Accelerator-Friendly Neural-Network Training: Learning Variations and Defects in RRAM Crossbar". In: *DATE*. 2017.

[66] A. BANAGOZAR et al. "Robust Neuromorphic Computing in the Presence of Process Variation". In: *DATE*. 2017.

[67] G. CHARAN et al. "Accurate Inference With Inaccurate RRAM Devices: A Joint Algorithm-Design Solution". In: *IEEE JXCDC* (2020).

[68] S. D. PYLE et al. "Leveraging Stochasticity for In Situ Learning in Binarized Deep Neural Networks". In: *Computer* 52.5 (2019), pp. 30–39.

[69] Ziqi MENG et al. "Digital Offset for RRAM-based Neuromorphic Computing: A Novel Solution to Conquer Cycle-to-cycle Variation". In: *DATE*. 2021.

[70] Ming-Guang LIN et al. "D-NAT: Data-Driven Non-Ideality Aware Training Framework for Fabricated Computing-In-Memory Macros". In: *IEEE Journal on Emerging and Selected Topics in Circuits and Systems* 12.2 (2022), pp. 381–392.

effective in the mitigation of stochastic variability due to its regularization effect. Still, there are several problems when training with this method. Additive noise injection does not capture the real nature of many of the synaptic cells, where their stochasticity level depends on the weight values, i.e., multiplicative noise nature[56 ,70] (e.g., PCM drift and programming noises, SRAM shot and thermal noises, the mismatch between devices, etc.). Training with additive noise will inject too much noise in small weight values while injecting too little in large weight values, which may not fully prepare the trained model for inferring in a multiplicative noise environment. The work in[56] shows that training with multiplicative noise improves the network resilience when compared to additive noise methods.

*Multiplicative noise-injection:* This technique injects noise to the network by multiplying the parameters with a noisy signal (e.g., $W = W_0 \cdot (1 + \epsilon)$), which captures better the nature of several synaptic cells, as stated above. One of the first works implementing multiplicative noise-injection (and noise-injection in general) dates back to 1994[54]. The authors used this technique in a multilayer perceptron (MLP), and found that when using this technique *the dependence of the outputs on the weights is evenly distributed across the weight set* (i.e., NN regularization). In more recent years, several works applying multiplicative noise-injection have emerged[55 ,56 ,58 ,60]. In particular, the device-variation-aware (DVA) training methodology in[55], called in[56] as multiplicative noise training (MNT), demonstrated updated versions of multiplicative noise-injection in popular and recent DNNs. The DVA/MNT methodology injects noise to the layer's weights during the forward propagation training stage using a normal distribution ($\epsilon \sim \mathcal{N}(0, \sigma^2)$),[ii] creating a regularization effect. Still, they failed to indicate what should be the procedure during back-propagation. In section 2.4, we will show how our methodology outperforms the DVA/MNT one and why A-Connect could be seen as an improved version of DVA/MNT, and multiplicative noise-injection techniques in general.

**Modifications to training hyperparameters:** Instead of injecting noise directly to the NN parameters, some works have chosen to modify training hyperparameters to take into account stochastic variations. As an example, the variation-aware training in[64]

[ii] Or the appropriate distribution according to the synaptic cells' nature.

included an L2-norm penalty to the cost function, based on the expected stochastic variability using a linearization of a log-normal distribution. This penalty on the cost function has been shown to be ineffective for large neural networks architectures in[58]. Instead, the authors of[58] used noise-injection within several operations of simple NN layers (they modified the multiplication, addition, softplus, and sigmoid operations), but also modified the cost function directly (not as a penalty). They only demonstrated results for small NN (two-layer FC on the MNIST handwritten dataset), perhaps due to the complex modifications proposed to the NN layers.[iii]

**Adversarial attacks training:** A recent work investigated adversarial attacks to mitigate stochastic variability by perturbing the parameters of the neural network during training[60]. The latter is different to classical adversarial training methods where the main objective is to attack the input space. Although their algorithm is effective in producing models that are more robust to parameter noise, their adversarial training method alone is not always better than DVA/MNT (as demonstrated by their experiments, where they called this method as 'Forward-Noise'). Their best results were obtained when combining their method with DVA/MNT, which outperformed other adversarial training algorithms[71,72], as well as other training methods, such as Dropout[73] and DVA/MNT.

**Other works:** There have been works related to spiking neural networks (SNNs) which try to mitigate the stochasticity problem from a statistical training perspective as

[71] Dongxian WU et al. "Adversarial Weight Perturbation Helps Robust Generalization". In: *NeurIPS*. 2020.

[72] Yaowei ZHENG et al. "Regularizing Neural Networks via Adversarial Model Perturbation". In: *CVPR*. 2021.

[73] Nitish SRIVASTAVA et al. "Dropout: A Simple Way to Prevent Neural Networks from Overfitting". In: *JMLR* (Jan. 2014).

[iii] The authors in[58] consider the local process variation as correlated, unlike our work (and the majority of multiplicative noise-injection papers) where we consider all process variation as uncorrelated. With the former, the noise-injection during training is more complex as shown in[58].

well[74,75,76,77]. Although the nature of SNN is different from conventional artificial neural networks, we have found that some of their contributions are related to our work. As an example, in[75] regularization methods (such as DropConnect[78]) were found to be successful in modelling stochastic behaviour of synaptic cells. According to our work, it does make sense that these types of methods are effective in SNNs due to the binary nature of their layers' outputs. Including stochastic variability, either temporal or spatial, may cause a spiking neuron to fire at a given time but not at another. The use of DropConnect is a natural fit for spiking events since it can effectively model what would happen in an SNN accelerator.

### 2.3. The A-Connect Methodology

In this section, we present A-Connect, a methodology to mitigate stochastic variability present in neural network analog accelerators. We also show the intuition behind A-Connect, using classical machine learning theories. Then, we extend the A-Connect methodology for non-normal distributions. Finally, we calculate a global coefficient of variation for the A-Connect stochasticity model.

#### 2.3.1. A-Connect to Mitigate Stochastic Variability

Consider a generic deep neural network like the one presented in Figure 5(a), where we include batch-normalization and activation layers after every hidden layer in the network (the output layer is an activation layer itself, e.g., softmax layer). In general, for the $k$-th hidden layer, $a_{(k-1)}$ represents its input (output from the preceding activation layer), $W_{(k)}$ the synaptic weights of the layer, and $a_{(k)}$ the output of the subsequent activation layer. In an ideal digital implementation, the weights $W_{(k)}$ are unmodified

[74] D. QUERLIOZ et al. "Bioinspired Programming of Memory Devices for Implementing an Inference Engine". In: *Proceedings of the IEEE* (2015).
[75] Emre O. NEFTCI et al. "Stochastic Synapses Enable Efficient Brain-Inspired Learning Machines". In: *Frontiers in Neuroscience* 10 (June 2016).
[76] N. ZHENG et al. "Learning in Memristor Crossbar-Based Spiking Neural Networks Through Modulation of Weight-Dependent Spike-Timing-Dependent Plasticity". In: *IEEE Transactions on Nanotechnology* (2018).
[77] X. SHE et al. "Improving Robustness of ReRAM-based Spiking Neural Network Accelerator with Stochastic Spike-timing-dependent-plasticity". In: *IJCNN*. 2019.
[78] Li WAN et al. "Regularization of Neural Networks Using Dropconnect". In: *ICML*. 2013.

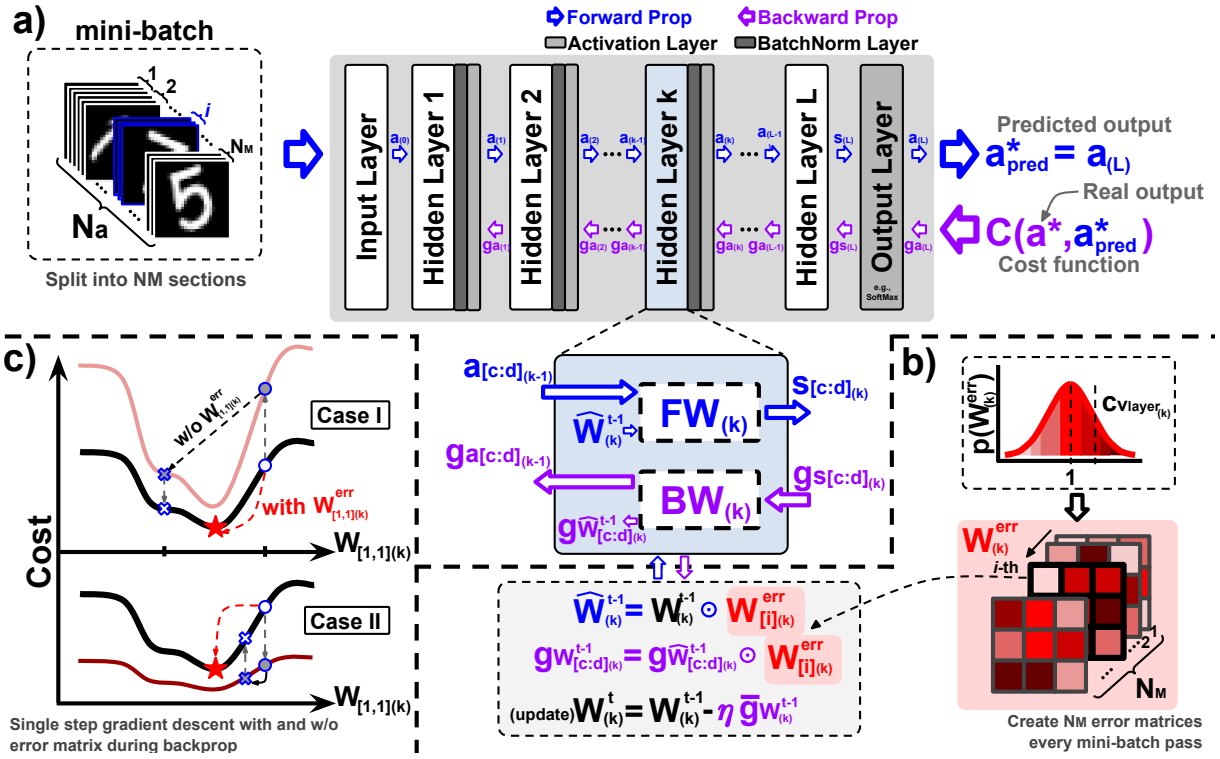


Figure 5. The A-Connect methodology: a) A generic DNN showing training forward and backward propagation paths; b) Creation and injection of error matrices/masks during forward propagation, and inclusion of the same error matrix during backward propagation; c) Possible cases during weight update using a gradient descent algorithm with and without considering the error matrices.

by the hardware executing the neural network, but in an analog implementation, the weights will deviate from their ideal values due to hardware non-idealities, namely, stochastic variability. We modeled this deviation as represented in Figure 5(b), by multiplying (element-wise) the actual weights $W_{(k)}$ with several error masks $W^{err}_{(k)}$. We use these error masks during forward and backward propagation. Algorithm 1 shows the steps followed during training with A-Connect:

- **Forward Propagation:** during forward-propagation, a mini-batch of inputs $a_{(0)}$ are passed through the network. The weights $W^{t-1}_{(k)}$ (as well as the biases $b^{t-1}_{(k)}$), are multiplied element-wise by error matrices/masks randomly selected using a probability distribution. Supposing the behaviour of the synaptic cells follow a normal distribution ($W^{err}_{(k)} \sim \mathcal{N}(1, c^2_{v_{\text{layer}}})$),[iv] this mask will have a mean equal to 1 (ideal values), and $c_{v_{\text{layer}}} \in (0, 1]$, which represents the standard deviation

[iv] It is common to represent a physical quantity with a normal distribution, but if the actual distribution is known, using this distribution instead may lead to better results.

(or coefficient of variation) relative to the absolute mean of the layer's weights ($c_{v_{\text{layer}}} = \sigma_W / \mu_W$).[v] The algorithm creates an $N_M$ number of different error matrices. The mini-batch is then split into $N_M$ sections and only one error matrix is used per section. We show in section 2.4.1, that even for a number of 2 different error masks ($W^{err}_{(k)}$ and $b^{err}_{(k)}$) per mini-batch, A-Connect is effective. In general, between 2 to 16 different error masks are necessary, depending on the implementation.

- **Backward Propagation and Parameter Update:** with A-Connect, the parameters update can be done using a gradient descent algorithm (GD) by back-propagating gradients of the cost function $C(a^*, a^*_{pred})$ with respect to the layer's parameters (see Figure 5(a)). The gradients of the cost function with respect to the weights and biases are multiplied by the error masks $W^{err}_{(k)}$ and $b^{err}_{(k)}$, respectively, to obtain the proper gradients for the parameters update (e.g., $g_{W^{t-1}_{(k)}} = g_{\widehat{W}^{t-1}_{(k)}} \odot W^{err}_{(k)}$). Finally, when back-propagating the gradients to the preceding layers ($g_{a_{(k-1)}}$), the masked weight matrix ($\widehat{W}^{t-1}_{(k)} = W^{t-1}_{(k)} \odot W^{err}_{(k)}$) is used instead of the weight matrix ($W^{t-1}_{(k)}$). The gradients for each parameter are averaged over the training examples in each mini-batch for the parameters update (e.g., $W^{t}_{(k)} = W^{t-1}_{(k)} - \eta \overline{g}_{W^{t-1}_{(k)}}$, see Figure 5(b)).

### 2.3.2. Intuition Behind the A-Connect Methodology

The A-Connect methodology inherits the regularization properties that make multiplicative noise-injection methods to perform well under stochastic variability[54]. Still, the A-Connect methodology has two main contributions and differences with respect to other multiplicative noise-injection techniques (as shown in Algorithm 1): the use of multiple error matrices (instead of only one), and the inclusion of such error matrices during backward propagation.

**Multiple Error Matrices/Masks:** Although previous works using multiplicative noise-injection training have shown strong results against stochasticity by using one single

[v] The total stochasticity of the weights' values ($\sigma_W$) is a combination of the spatial ($\sigma_{W_{\text{mis}}}$) and temporal ($\sigma_{W_{\text{noise}}}$) stochastic variabilities: $\sigma_W = \sqrt{\sigma^2_{W_{\text{mis}}} + \sigma^2_{W_{\text{noise}}}}$.

**Algorithm 1:** SGD training with A-Connect to take into account device stochastic variability. $C$ is the cost function for the mini-batch, $L$ is the number of layers. ($\odot$) denotes element-wise multiplication. $\mathbf{FW_{(k)}}$ and $\mathbf{BW_{(k)}}$ are the forward and backward operations of the $k$-th layer, respectively (e.g., fully-connected, convolution, etc.). BN and BNback, indicate batch-normalization and batch-normalization backpropagation, respectively. Act and ActBack, indicate activation layer forward and backpropagation, respectively.

**Input:** a mini-batch of $N_a$ inputs and targets $(a_{(0)}, a^*)$, previous parameters $W^{t-1}$ (weights), $b^{t-1}$ (biases), $\theta^{t-1}$ (batch-normalization parameters) and learning rate $\eta$. The weights are initialized using Glorot technique[79].

**Output:** updated parameters $W^t$ and $b^t$.

**Forward propagation:**

**for** $k = 1$ *to* $L$ **do**

$W^{err}_{(k)} \sim \mathcal{N}(1, \sigma^2_{\text{layer}} = c^2_{v_{\text{layer}_{(k)}}})_{[1:N_M][\text{weight\_size}]}$

$b^{err}_{(k)} \sim \mathcal{N}(1, \sigma^2_{\text{layer}} = c^2_{v_{\text{layer}_{(k)}}})_{[1:N_M][\text{bias\_size}]}$

`# Split mini-batch into` $N_M$ `sections`

$N'_a = \lfloor N_a / N_M \rfloor$

**for** $i = 1$ *to* $N_M$ **do**

$c = (i-1) \cdot N'_a + 1$

$d = i \cdot N'_a$

`# One error matrix per section`

$\widehat{W}^{t-1}_{[i]_{(k)}} \leftarrow W^{t-1}_{(k)} \odot W^{err}_{[i]_{(k)}}$

$\widehat{b}^{t-1}_{[i]_{(k)}} \leftarrow b^{t-1}_{(k)} \odot b^{err}_{[i]_{(k)}}$

$s_{[c:d]_{(k)}} \leftarrow \mathbf{FW_{(k)}}\left(a_{[c:d]_{(k-1)}}, \widehat{W}^{t-1}_{[i]_{(k)}}\right) + \widehat{b}^{t-1}_{[i]_{(k)}}$

**end**

$s_{b_{(k)}} \leftarrow \text{BN}(s_{(k)}, \theta^{t-1}_{(k)})$

$a_{(k)} \leftarrow \text{Act}(s_{b_{(k)}})$

**end**

**Backward propagation:**

`Compute` $g_{a_{(L)}} = \frac{\partial C}{\partial a_{(L)}}$ `knowing` $a_{(L)}$ `and` $a^*$

**for** $k = L$ *to* $1$ **do**

$(g_{s_{b_{(k)}}}) \leftarrow \text{ActBack}(g_{a_{(k)}}, s_{b_{(k)}})$

$(g_{s_{(k)}}, g_{\theta_{(k)}}) \leftarrow \text{BNback}(g_{s_{b_{(k)}}}, s_{(k)}, \theta^{t-1}_{(k)})$

$(g_{a_{(k-1)}}, g_{\widehat{W}^{t-1}_{(k)}}) \leftarrow \mathbf{BW_{(k)}}\left(g_{s_{(k)}}, a_{(k)}, \widehat{W}^{t-1}_{(k)}\right)$

$g_{\widehat{b}^{t-1}_{(k)}} \leftarrow g_{s_{(k)}}$

**for** $i = 1$ *to* $N_M$ **do**

$c = (i-1) \cdot N'_a + 1$

$d = i \cdot N'_a$

`# Same error matrices as in forward prop.`

$g_{W^{t-1}_{[c:d]_{(k)}}} \leftarrow g_{\widehat{W}^{t-1}_{[c:d]_{(k)}}} \odot W^{err}_{[i]_{(k)}}$

$g_{b^{t-1}_{[c:d]_{(k)}}} \leftarrow g_{\widehat{b}^{t-1}_{[c:d]_{(k)}}} \odot b^{err}_{[i]_{(k)}}$

**end**

**end**

**Parameter update:**

`Compute` $\overline{g} = \frac{1}{N_a} \sum_{i=1}^{l=N_a} g_{[i]}$

$\theta^t_{(k)} \leftarrow \theta^{t-1} - \eta \overline{g}_{\theta^{t-1}_{(k)}}$

$W^t_{(k)} \leftarrow W^{t-1}_{(k)} - \eta \overline{g}_{W^{t-1}_{(k)}}$

$b^t_{(k)} \leftarrow b^{t-1}_{(k)} - \eta \overline{g}_{b^{t-1}_{(k)}}$

error matrix[54,55,56,60], it is possible to obtain a higher network regularization, and better performance, if more error matrices are used, as we will show experimentally in section 2.4. The DropConnect work showed a similar result when concluding that

one of the key components in their method was the selection of one mask per training example, since a single mask per mini-batch did not regularize the model enough[78]. In a general implementation of A-Connect, it would be possible to use one error matrix per image in the mini-batch ($N_M = N_a$), but for large NN, there are practical issues due to the computational time required for training.

As well, it is possible to understand the A-Connect using multiple error matrices methodology as an ensemble learner, similar to[80] where the Dropout method[73] is analyzed from the ensemble learning theory. In ensemble learning, the idea is to use several learners to achieve better performance. However, ensemble learning might be computationally expensive since many learners are needed to obtain an accurate result. A-Connect is an ensemble learner since several networks can be obtained and trained by randomly modifying the layers' parameters during the training process. The networks created during training try to mimic the actual synaptic distribution due to stochastic variability. In short, instead of having many learners, with A-Connect it is like statistically training many networks that are sampled from a distribution, using a Monte Carlo method.

**Error Matrices/Masks During Backpropagation:** Because our methodology uses a multiplicative noise-injection approach, we can treat the values on the error masks as constants during forward and backward propagation of a mini-batch. Hence, if the weights used during forward propagation are $\widehat{W} = W \odot W^{err}$, the gradient of the cost function with respect to the actual weights $W$ is $g_W = g_{\widehat{W}} \odot W^{err}$. In this way, A-Connect updates the weights with the same proportion they were modified by the error masks.

Figure 5(c) shows two possible cases during a weight update (using a gradient descent algorithm) with and without considering the error value applied to the weight during forward propagation. Suppose that the cost $C$ is a function of the weight $W_{[1,1](k)}$ as represented by the black line (same for both cases). Now, let's suppose that during forward propagation, we use a single error matrix $W^{err}_{(k)}$ and multiply the weight $W_{[1,1](k)}$ by an error value $W^{err}_{[1,1](k)}$. If during backpropagation the algorithm does not multiply the gradient of the weight by the error value, it would be like scaling the cost function

[80] Kazuyuki HARA et al. "Analysis of Dropout Learning Regarded as Ensemble Learning". In: *ICANN*. 2016.

by $1/W^{err}_{[1,1](k)}$, which in return scales the gradients by the same amount. For "case I" ($W^{err}_{[1,1](k)} < 1$) this would effectively increase the learning rate and lead to drastic weight updates, which could be a problem if the current weight value is close to the ideal value. On the contrary, for "case II" ($W^{err}_{[1,1](k)} > 1$), the learning rate would be reduced and lead to mild weight updates, which could be a problem if the current update is far from the ideal value.

Now, when using many error matrices (obtained from a normal distribution) the effect of the errors will cancel each other when averaging to obtain the weight gradient across a mini-batch. In theory, using this approach would allow us to have similar results in both cases: when the error matrices are considered, and when they are not considered during backpropagation. The problem with this approach is that the number of error matrices would be prohibitively high, making it impractical for training. We show in the experiments in section 2.4 that the major effect of considering the error matrices during backward propagation is not only on the network median accuracy performance, but on the accuracy deviation, which is diminished (in the majority of cases) when the NN is tested across different accelerators (e.g., Monte Carlo simulations).

### 2.3.3. A-Connect using a log-Normal Distribution

In this section, we will consider the case when A-Connect is trained with a log-normal distribution, to investigate how to apply our methodology to memory technologies such as ReRAM. In general, the actual resistance of a ReRAM follows a log-normal distribution of the form $R = R_0 e^{\theta}$, with $R$ being the analog representation of the actual weights in a ReRAM memory, $R_0$ the target resistance, and $\theta \sim \mathcal{N}(0, \sigma^2)$ a normal distributed random variable with mean zero and standard deviation $\sigma \in (0, 1]$[81,69].

In these type of asymmetrical distributions, the mean and the median are not the same. Therefore, when applying a log-normal distribution to obtain the error masks ($W^{err}$ and $b^{err}$) in Algorithm 1, the mean (target) value of the layer's parameter will be modified by a deterministic factor $\mu_r = E[e^{\theta}] = \sqrt{e^{\sigma^2}}$. The latter is particularly troublesome when using batch normalization, because the calculated mean and deviation of the layer's

[81] C. MA et al. "Go Unary: A Novel Synapse Coding and Mapping Scheme for Reliable ReRAM-based Neuromorphic Computing". In: *DATE*. 2020.

output will not correspond to the values obtained during training.

Suppose a NN (trained using A-Connect with a specific level of stochasticity $\sigma_0$) is deployed in a ReRAM with a measured stochastic variability of $\sigma_1$. Because of the difference between the trained and the real deviation, the mean value of the NN's parameters and layer's output will be modified by the factor $\mu_{r_1} = \sqrt{e^{\sigma_1^2}}$. The performance of the NN will be greatly affected even for a deviation of 10% on the ratio $\mu_{r_1}/\mu_{r_0}$.

Fortunately, a correction factor is enough to compensate for the error introduced since it is deterministic. There are two ways of compensating for this error: the straightforward solution is to multiply all the NN's weights and biases by the correction factor $\rho = \sqrt{e^{\sigma_0^2 - \sigma_1^2}}$. The latter would imply a complete modification of the NN's parameters. A more practical solution is to modify the batch normalization parameter only. A batch normalization layer performs the following operations (using the same notation as Algorithm 1):

$$\hat{s}_{0(k)} = \frac{s_{0(k)} - \mu_{B0}}{\sigma_{B0}} \longrightarrow s_{b_{0(k)}} = \gamma \hat{s}_{0(k)} + \beta \tag{1}$$

with $s_{0(k)}$ being the output of the forward operation of the *k*-th layer, $\mu_{B0}$ the trained mean, $\sigma_{B0}$ the trained deviation, and $\gamma$ and $\beta$ are the learned parameters during training. The sub-index $0$ indicates that the A-Connect methodology used a stochasticity of $\sigma_0$. The idea with the correction factor is to obtain the same $s_{b_{0(k)}}$ after the batch normalization. Therefore, when deploying the NN obtained with A-Connect at a stochastic variability of $\sigma_0$, the new batch normalization becomes:

$$\hat{s}_{1(k)} = \frac{s_{1(k)} - \mu_{B1}}{\sigma_{B1}} \longrightarrow s_{b_{1(k)}} = \gamma \hat{s}_{1(k)} + \beta \tag{2}$$

Since $s_{1(k)} = s_{0(k)}/\rho$, only by applying $\mu_{B1} = \mu_{B0}/\rho$ and $\sigma_{B1} = \sigma_{B0}/\rho$, we obtain $\hat{s}_{1(k)} = \hat{s}_{0(k)}$, hence $s_{b_{1(k)}} = s_{b_{0(k)}}$, and the NN would be compensated.

### 2.3.4. Stochasticity Model - Coefficient of Variation Calculation

The stochasticity level ($\sigma$), used in the error matrices $W^{err}$ in Algorithm 1, represents the stochastic variability of an entire NN's layer ($\sigma_{\text{layer}}$). In this subsection, we will define how the stochasticity of a layer is related to that of a synaptic cell ($\sigma_c$) by using

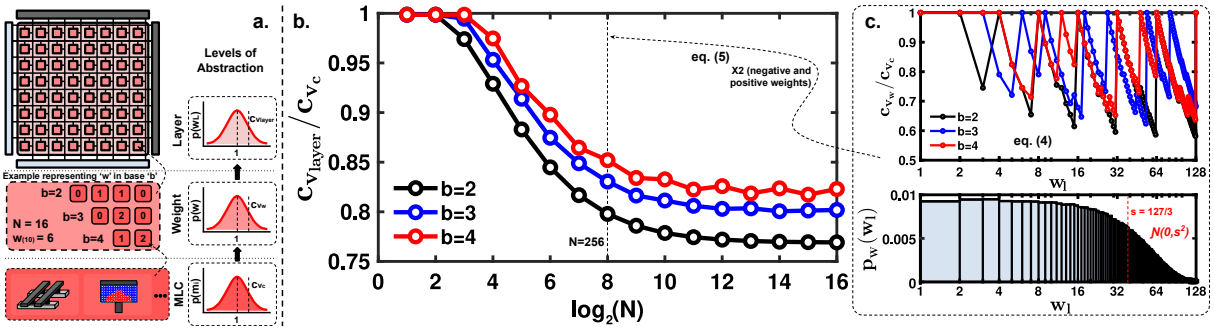


Figure 6. Coefficient of variation: a) $c_v$ at different abstraction levels (device, weights, NN's layer); b) $c_{v_{\text{layer}}}/c_{v_c}$ against the number of weight quantization levels $N$ for different base-systems (i.e., different b-level MLCs); c) $c_{v_w}/c_{v_c}$ and $p_W(w_l)$ used to calculate the $c_{v_{\text{layer}}}/c_{v_c}$ at $N = 256$ in (b), using eqs. (3) to (5).

the coefficient of variation[vi] ($c_v$) at each level of abstraction (i.e., layer, weight, cell). Considering a bit level representation of the NN layers' parameters (i.e., weights and biases), and that these parameters may be stored in synaptic multi-level cells (MLC), the coefficient of variation for a layer ($c_{v_{\text{layer}}}$) is different from that of a weight ($c_{v_w}$), which in turn is different from the variability of an MLC ($c_{v_c}$), as shown in the levels of abstraction hierarchy in Figure 6.a.

In a general sense, a number can be represented in a positional numeral system as $w_{(b)} = [d_{n-1} \dots d_1 d_0]_{(b)}$, with $d_i \in [0, b-1]$. In this system, $n$ is the number of digits that represents the number, and $b$ is the base of the numeral system, or equivalently, the number of levels of the MLC. The number $w_{(b)}$ in base $b$ can be represented in the decimal system as:

$$w_{(10)} = \sum_{i=0}^{n-1} m_i \rightarrow m_i = \begin{cases} d_i b^i & : \text{Positional System} \\ d_i & : \text{Unary} \end{cases} \tag{3}$$

Since each MLC follows an independent random distribution, the coefficient of variation of a single weight is (the weight being represented with an $n$ number of MLCs, with each

[vi] As defined in section 2.3.1, the coefficient of variation is the ratio between the standard deviation and the mean of a variable.

MLC having b-levels):

$$c_{v_w}^2 = \frac{\sum_{i=0}^{n-1} m_i^2}{\left(\sum_{j=0}^{n-1} m_j\right)^2} \cdot c_{v_c}^2 \tag{4}$$

with $c_{v_c} \in (0,1]$. Hence, it is possible to calculate the coefficient of variation for any given weight value with any numeral system ($w_{l_{(b)}}$) using eqs. (3) and (4).

The next step is to obtain the layer's coefficient of variation ($c_{v_{\text{layer}}}$). For the latter, it is important to consider the probability distribution of the weights and biases since the NN's parameter values are not uniformly distributed. Dividing the problem into the NN's layers, the $c_{v_{\text{layer}}}$ can be expressed as a function of the different $c_{v_{w_l}}$ obtained from eqs. (3) and (4). Therefore, the layer's coefficient of variation is calculated as:

$$c_{v_{\text{layer}_k}}^2 = \sum_{l=0}^{N-1} p_W(w_l) \cdot c_{v_{w_l}}^2 \tag{5}$$

where $p_W(w_l)$ is the discrete probability function of the quantized weights ($w_l$) in the *k*-th layer, and $N$ is the number of weights (or biases) quantization levels (i.e., for a base $b$, $N = b^n$ for the positional system, and $N = (b-1) \cdot n + 1$ for the unary system).

As an example, Figure 6.b shows the layer's coefficient of variation ($c_{v_{\text{layer}}}$) normalized to that of the cell's ($c_{v_c}$), using eq. (5). For more clarity, we provide the steps used to obtain $c_{v_{\text{layer}}}/c_{v_c}$ for the case of $N = 256$ (see Figure 6.c):

1. We used the positional system described in eq. (3), with $b = \{2,3,4\}$.
2. We obtained the weight's coefficient of variation $c_{v_w}$ normalized to $c_{v_c}$ for half of all possible weights $w_{l(10)} = \{0,1,\ldots,127\}$.[vii] Notice how the maximum peaks ($c_{v_w}/c_{v_c} = 1$) occur whenever $w_{l(b)} = [d_{n-1}\ldots d_1 d_0]_{(b)}$ has a single one of its digits $d_i$ active and the rest of its digits to zero (i.e., for any base, the peaks occur at $w_{(b)} = d_i b^i$). On the other hand, there are $(b-1)$ distinguishable minimum peaks between weights $b^{i-1}$ and $b^i$, which correspond to weights represented using all the $d_0$ to $d_{i-1}$ digits to non-zero values (i.e., for any base, the minimum

[vii] We only show half of the weights in Figure 6.c since we used a sign-magnitude representation; negative numbers have the same $c_{v_w}/c_{v_c}$ behavior as their positive counterpart, e.g., $c_{v_w}/c_{v_c}$ is the same for weights -37 and 37.

peaks occur at $w_{(10)} = (d_{i-1}+1)\cdot b^i - 1)$. We can conclude that for any value of $w_l$, $c_{v_w} \leq c_{v_c}$,[viii] reaching its maximum value ($c_{v_w} = c_{v_c}$) when only one device is used to represent a weight, and getting lower as more devices are used to represent the weights.

3. In the final step, we multiplied the $(c_{v_{w_l}}/c_{v_c})^2$ to their corresponding probability value to obtain the points for $N = 256$ in Figure 6.b. The discrete probability function $p_W(w_l)$ used is a discretized approximation of a normal distribution $\mathcal{N}(0, s^2)$, with $s = (N/2-1)/3$,[ix] as shown in Figure 6.c (the x-axis is in log-scale).

Note: The tendency in $c_{v_{\text{layer}}}/c_{v_c}$ (Figure 6.b) is a consequence on the behavior of the minimum values achievable by $c_{v_w}/c_{v_c}$ (Figure 6.c). The $(b-1)$ minimums between the $b^i$ and $b^{i-1}$ weights tend to finite values when increasing the number of levels $N$, and consequently, the number of digits (devices) $n$. The reader can verify that these values (from eq. (4)) tend to:

$$\left(\frac{c_{v_w}}{c_{v_c}}\right)^2_{\text{min(j)}} = \left(\frac{j}{j+1}\right)^2 + \left(\frac{b-1}{j+1}\right)^2 \cdot \left(\frac{1}{b^2-1}\right)$$

with $j = \{1, \ldots, b-1\}$ for $b \geq 3$, and $j = 0$ for $b = 2$. Hence, the layer's coefficient of variation will tend to a value in between the maximum and minimum of $c_{v_w}$, or:

$$\left(\frac{c_{v_w}}{c_{v_c}}\right)_{\text{min(j)}} < \frac{c_{v_{\text{layer}}}}{c_{v_c}} < 1$$

Finally, by using equations (3) to (5) one can obtain the stochasticity levels for the layers ($c_{v_{\text{layer}}} \rightarrow \sigma_{\text{layer}}$), for the weights ($c_{v_w} \rightarrow \sigma_w$), and for the cells ($c_{v_c} \rightarrow \sigma_c$), depending on the probability distribution followed by the cells. For example, for a variable with a normal distribution $\mathcal{N}(1, \sigma^2)$, the coefficient of variation is $c_v = \sigma$; for a log-normal distribution $r = e^\theta$, with $\theta \sim \mathcal{N}(0, \sigma^2)$, $c_v = \sqrt{e^{\sigma^2} - 1}$.

---

[viii] This is a direct consequence of eq. (4), since $\sum_{i=0}^{n-1} m_i^2 \leq \left(\sum_{j=0}^{n-1} m_j\right)^2$.

[ix] It is common that the layer's parameters of a NN follow a normal distribution (e.g.,[55] ).

## 2.4. Experimental Results

In this section, we present the experiment results to validate the A-Connect methodology. For this reason, we developed a library with the A-Connect methodology[44].[x] The library is available for Keras/TensorFlow, and contains the A-Connect modified versions of fully-connected and convolutional layers. We used the Google Colab (Tesla K80 GPU) and Kaggle (Tesla P100 GPU) platforms to perform all the tests in this section.[xi]

### 2.4.1. A-Connect in Deep Neural Networks

In this set of experiments, we provide evidence on the A-Connect effectiveness in DNNs. We used three popular architectures: LeNet-5[82] (trained in the MNIST dataset), AlexNet[83], VGG-16[84], and ResNet-20[85] (the last three trained in the CIFAR-10 dataset[86]), and ResNet-18[85] (trained on the CIFAR-100 dataset[86] ).

**General training conditions:** We used stochastic gradient descent with momentum of 0.9 (SGDM) as the training method for the experiments in this section. The training images for both datasets (i.e., MNIST and CIFAR-10)[xii] were divided into mini-batches of 256 images each, with data-shuffling every epoch. The activation layer implemented was ReLU. We used quantization-aware training modules provided by TensorFlow (based on[87]) in conjunction with the A-Connect training methodology to quantize all our layers' parameters (i.e., weights and biases) and outputs to 8-bits.

---

82 Y. LECUN et al. "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* (1998).

83 Alex KRIZHEVSKY et al. "ImageNet Classification with Deep Convolutional Neural Networks". In: *NIPS* (Jan. 2012).

84 Karen SIMONYAN et al. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. Ed. by Yoshua BENGIO et al. 2015.

85 Kaiming HE et al. *Identity Mappings in Deep Residual Networks*. 2016. arXiv: `1603.05027 [cs.CV]`.

86 A. KRIZHEVSKY. "The CIFAR-10 and CIFAR-100 datasets". In: *https://www.cs.toronto.edu/ kriz/cifar.html/* (2009).

87 Benoit JACOB et al. "Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2018.

---

x Also available as a pip package: *pip install aconnect*

xi All the experiments performed in this chapter can be found in `https://github.com/onchipuis/Tests_A-Connect`.

xii Both MNIST and CIFAR-10 datasets consist of 50000 training images and 10000 test images. We divided the original training sets into 40000 training-only images, and 10000 validation images.

**Specific training conditions:** Due to memory limitations in Google Colab and Kaggle, we made some modifications on the DNN architectures for the experiments in this subsection. Below, we indicate specific training conditions for each DNN:

- **LeNet-5:** The model was trained with a learning rate of 0.01, and 20 epochs.
- **AlexNet**: the AlexNet architecture uses two fully connected layers of 4096 outputs before the output layer. We replaced these layers with two layers of 1024 outputs plus one layer of 512 ousputs. We used a resize layer as the first layer of the model to upscale the 32x32 images from the CIFAR-10 dataset to 227x227. The model was trained with an initial learning rate of 0.01 (decaying every 20 epochs), and 100 epochs.
- **VGG-16:** the VGG-16 uses three FC layers before the output layer (2-FC layers of 4096 outputs in cascade, followed by an FC layer of 10 outputs). We replaced these layers with 2-FC layers of 256 and 10 outputs, respectively. We used several data augmentation techniques, such as normalization, random flip, random translation, and random zoom. The model was trained during 50 epochs with a learning rate using exponential decay, with an initial rate of 0.1, decaying by half every 30 epochs. We also used a pre-trained model,[xiii] using the same training conditions as before but during 90 epochs.
- **ResNet-20 and ResNet-18:** we used the ResNet version in[85], implementing the residual blocks with two 3x3 convolutional layers, with a batch-normalization layer after the addition.[xiv] Both models (ResNet-20 and ResNet-18) were trained through 120 epochs with an initial learning rate of 0.1 (decaying by a factor of ten at epochs number 30, 60, and 100) without data augmentation. We also used a pre-trained model, trained through 200 epochs, with an initial learning rate of 0.1 (decaying by a factor of ten at epochs number 80, 120, and 160), and with data augmentation techniques such as normalization, random flip, random translation, and random zoom.

[xiii] We started from a model trained in ImageNet.

[xiv] We also tried with the full pre-activation version ResNet-20 (the final version in[85] ), but the results were worse (e.g., around 5 percentage points lower at 70% of stochasticity with A-Connect). The lack of a batch-normalization layer in the main signal path might be one of the possible causes.

Table 1. DNNs training conditions with A-Connect methodology.

| | LeNet-5 | AlexNet | VGG-16 | ResNet-20 | ResNet-18 |
|---|---|---|---|---|---|
| **Dataset** | **MNIST** | | **CIFAR-10** | | **CIFAR-100** |
| **Method** | | | **SGDM** | | |
| **Mini-batch size** | | | **256** | | |
| **Quant. (act./weights)** | | | **8b/8b** | | |
| **#Images (train./val./test)** | | | **40k/10k/10k** | | |
| **Initial LR** | 0.01 | 0.01 | 0.1 | 0.1 | 0.1 |
| **Decaying LR** | no | yes | yes | yes | yes |
| **Epochs** | 20 | 100 | 50 | 120 | 120 |
| **Data Augmentation** | no | no | yes | no | no |
| **Kernel Regularizer** | no | no | no | L2(1e-4) | L2(1e-5) |
| **Transfer Learning** | no | no | yes | no | no |
| **Pre-trained model** | no | no | yes | yes | yes |
| **#Error Matrices (FC/Conv)** | 2/2 | 8/8 | 2/2 | 2/8 | 2/4 |
| **#Trainable params.** | 0.06M | 14.8M | 14.9M | 0.27M | 11.2M |

### Effect of the number of error masks/matrices

We performed several experiments to determine the effect of the number of error matrices on the neural networks accuracy and training time. We varied the number of error matrices up to 32 for LeNet-5, and up to 16 for AlexNet, VGG-16, ResNet-20, and ResNet-18. We performed 100 Monte Carlo simulations per experiment.

The results in Figure 7 show that the improvement on the test accuracy and the interquartile range almost settled after using 2 different error matrices per mini-batch. The intuition behind this improvement (e.g., more than 3 percentage points when training at a 70% of stochasticity) is that using at least 2 different error matrices, gives the necessary regularization for A-Connect to be effective.

The experiments also show the training time against the number of error matrices used, as well as a comparison with the baseline NN training time. It is possible to see that our algorithm incurs an additional training time that increases linearly with the number of error matrices.[xv]

For the remaining experiments, we decided to use the best accuracy-time to train trade-off for the LeNet-5, VGG-16, and ResNet-18 experiments in Table 2 (i.e., 2 error matrices for LeNet-5 and VGG-16, and 4 error matrices for ResNet-18). On the other

[xv] The linear increment in time is a direct consequence on the code implementation shown in Algorithm 1. Because the mini-batch is split into $N_M$ sections (there are $N_M$ error matrices), we apply a for-loop in order to use one error matrix per section. Since only the multiply-accumulation of multiple images is performed in the GPU (parallel), when the mini-batch is split into more sections (more error matrices), then, more calls are needed to the GPU. The algorithm then has the time complexity of a for-loop, or $\mathcal{O}(N_M)$, making it linearly dependent on the number of error matrices.

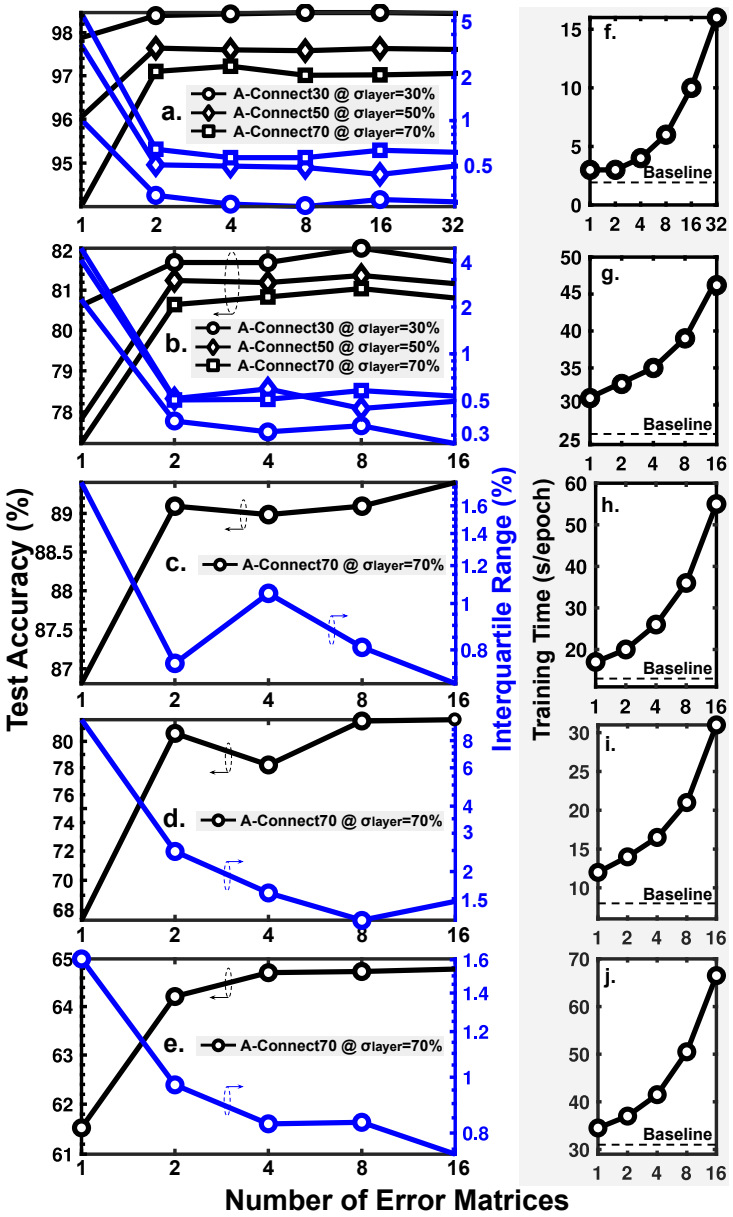


Figure 7. Effect of the number of error matrices on the DNNs performance. Test accuracy for: a) LeNet-5 on MNIST; b) AlexNet on CIFAR-10; c) VGG-16 on CIFAR-10; d) ResNet-20 on CIFAR-10; e) ResNet-18 on CIFAR-100. Training Time for: f) LeNet-5; g) AlexNet; h) VGG-16; i) ResNet-20; j) ResNet-18.

hand, we used the number of matrices that performed best in terms of accuracy (and accuracy deviation) for the AlexNet and ResNet-20 experiments (i.e., 8 error matrices). Table 1 summarizes the final set of training conditions for the DNN used in this chapter.

### Results

Table 2 summarises the results obtained for the experiments of A-Connect in DNNs

Table 2. Test accuracy rates on the MNIST, CIFAR-10, and CIFAR-100 datasets using popular DNNs: LeNet-5, AlexNet, VGG-16, ResNet-20, and ResNet-18. The median (M) and the interquartile range (IQR) are presented for each experiment (M/IQR). The best results (M-wise) for each simulated error ($\sigma_{\text{layer}}$) are highlighted.

| Dataset | DNN | Sim. Error ($\sigma_{\text{layer}}$) | Baseline | A-Connect | | | Baseline | A-Connect | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | 30% | 50% | 70% | | 30% | 50% | 70% |
| MNIST | LeNet-5[a] | 0% | **98.9%/0.0%** | **98.9%/0.0%** | 98.8%/0.0% | 98.4%/0.0% | 98.8%/0.0% | **99.3%/0.0%** | 98.9%/0.0% | 98.5%/0.0% |
| | | 30% | 88.7%/10.7% | 98.5%/0.2% | **98.6%/0.1%** | 98.3%/0.2% | 87.7%/10.4% | **98.8%/0.3%** | 98.7%/0.1% | 98.2%/0.2% |
| | | 50% | 61.7%/18.5% | 97.7%/0.6% | **98.2%/0.3%** | 98.0%/0.3% | 56.0%/18.43% | 96.2%/2.3% | **98.0%/0.5%** | 97.7%/0.5% |
| | | 70% | 46.7%/17.1% | 96.3%/1.5% | **97.6%/0.6%** | 97.4%/0.5% | 30.1%/13.9% | 78.7%/10.9% | 93.8%/3.6% | **96.4%/1.4%** |
| CIFAR-10 | AlexNet[b] | 0% | **84.1%/0.0%** | 82.3%/0.0% | 81.7%/0.0% | 80.6%/0.0% | 84.1%/0.0% | **85.1%/0.0%** | 83.0%/0.0% | 80.0%/0.0% |
| | | 30% | 79.8%/4.1% | **81.4%/0.2%** | 81.0%/0.2% | 80.1%/0.3% | 82.7%/0.8% | **84.1%/0.5%** | 82.2%/0.7% | 79.4%/0.7% |
| | | 50% | 71.9%/11.4% | 79.4%/0.6% | **80.0%/0.4%** | 79.8%/0.5% | 75.4%/3.2% | 80.6%/1.4% | **80.7%/1.6%** | 78.0%/2.0% |
| | | 70% | 65.3%/14.3% | 75.5%/1.4% | 77.7%/0.8% | **80.0%/0.6%** | 53.7%/8.1% | 72.6%/3.2% | **76.9%/2.0%** | 74.9%/3.1% |
| | VGG-16[c] | 0% | 92.0%/0.0% | **92.7%/0.0%** | 92.6%/0.0% | 92.6%/0.0% | 92.0%/0.0% | 92.3%/0.0% | 92.6%/0.0% | **92.9%/0.0%**[f] |
| | | 30% | 90%/1.0% | 91.5%/0.4% | 91.7%/0.5% | **92%/0.4%** | 90%/0.9% | 91.2%/0.4% | 91.7%/0.4% | **92.4%/0.2%**[f] |
| | | 50% | 83.9%/3.0% | 88.9%/0.9% | 89.9%/0.7% | **90.7%/0.6%** | 81.4%/3.6% | 87.5%/1.4% | 89.4%/0.8% | **91.3%/0.5%**[f] |
| | | 70% | 74.0%/6.9% | 83.3%/2.4% | 86.4%/1.8% | **89.1%/0.7%** | 53.6%/13.1% | 72.2%/5.9% | 81.7%/2.4% | **88.0%/0.9%**[f] |
| | ResNet-20[d] | 0% | **91.6%/0.0%** | 90.1%/0.0% | 89.3%/0.0% | 85.6%/0.0% | **91.6%/0.0%** | 91.1%/0.0% | 87.1%/0.0% | 83.6%/0.0%[g] |
| | | 30% | 74.8%/7.3% | 87.2%/0.6% | **87.7%/0.4%** | 84.3%/0.9% | 75.2%/8.3% | **87.1%/1.3%** | 85.7%/0.6% | 82.6%/0.7%[g] |
| | | 50% | 41.3%/13.5% | 80.0%/2.5% | **84.7%/0.8%** | 82.3%/1.6% | 33.2%/13.3% | 72.5%/5.6% | **81.2%/1.5%** | 80.0%/1.1%[g] |
| | | 70% | 20.7%/7.7% | 64.4%/7.0% | 78.7%/2.6% | **81.6%/1.2%** | 12.3%/3.9% | 29.6%/10.9% | 64.8%/6.8% | **72.9%/3.1%**[g] |
| CIFAR-100 | ResNet-18[e] (Top-1) | 0% | 70.3%/0.0% | 71.7%/0.0% | **71.8%/0.0%** | 68.4%/0.0% | 70.3%/0.0% | 71.7%/0.0% | **71.8%/0.0%** | 68.4%/0.0% |
| | | 30% | 64.9%/0.5% | 69.0%/0.4% | **69.9%/0.5%** | 66.7%/0.8% | 64.9%/1.7% | 68.7%/0.4% | 70.2%/0.3% | **70.3%/0.3%** |
| | | 50% | 54.8%/4.6% | 63.9%/1.2% | **66.6%/0.7%** | 65.3%/1.1% | 51.0%/5.6% | 61.3%/1.6% | 65.8%/0.7% | **67.6%/0.5%** |
| | | 70% | 36.2%/8.2% | 52.9%/3.2% | 59.7%/2.0% | **64.7%/0.8%** | 25.2%/8.0% | 43.1%/4.1% | 54.7%/1.7% | **61.4%/1.0%** |
| | ResNet-18[e] (Top-5) | 0% | 90.4%/0.0% | 90.8%/0.0% | **90.9%/0.0%** | 90.8%/0.0% | 90.4%/0.0% | 90.8%/0.0% | **90.9%/0.0%** | 90.8%/0.0% |
| | | 30% | 87.8%/0.9% | 89.1%/0.3% | **89.6%/0.2%** | 87.6%/0.4% | 87.8%/1.2% | 89.2%/0.3% | 89.7%/0.3% | **90.3%/0.2%** |
| | | 50% | 81.6%/3.4% | 86.0%/0.7% | **87.5%/0.5%** | 86.7%/0.7% | 79.0%/4.4% | 84.8%/1.0% | 86.9%/0.6% | **88.7%/0.4%** |
| | | 70% | 65.2%/9.1% | 78.1%/2.5% | 82.7%/1.4% | **86.2%/0.6%** | 54.1%/10.6% | 72.1%/3.2% | 79.1%/1.5% | **84.5%/0.7%** |
| Sim. Error Distribution | | | | Normal | | | | log-Normal | | |

**Training Times:**
[a] Baseline: 2s/epoch; A-Connect: 5s/epoch. Log-Normal distribution trained up to 50 epochs.
[b] Baseline: 26s/epoch; A-Connect: 39s/epoch.
[c] Baseline: 13s/epoch; A-Connect: 19s/epoch.
[d] Baseline: 8s/epoch; A-Connect: 21s/epoch.
[e] Baseline: 31s/epoch; A-Connect: 41.5s/epoch.
[f] Trained with 100 epochs more, with an initial learning rate of 0.02 and restarted at 50 epochs. No random zoom data augmentation.
[g] Used transfer learning from the model trained at 70% stochasticity with normal distribution. Used 60 epochs, an initial learning rate of 1e-2, decreased to 1e-3 at epoch 30.

Table 3. Equivalent cell's stochasticity ($\sigma_c$) for different layer's stochasticity ($\sigma_{\text{layer}}$) and MLC levels (the NN's parameters used 8-bit quantization). The equivalencies are presented for cells' stochastic variability following normal/log-normal distributions.

| Sim. Error ($\sigma_{\text{layer}}$) | Equivalent $\sigma_c$ | | | |
|---|---|---|---|---|
| | 1bit MLC | 2bit MLC | 4bit MLC | 8bit MLC |
| **30%** | 37.5% / 37.1% | 35.3% / 35% | 33.3% / 33.1% | 30% |
| **50%** | 62.5% / 60.6% | 58.8% / 57.6% | 55.5% / 54.8% | 50% |
| **70%** | 87.5% / 82.9% | 82.4% / 79.3% | 77.8% / 76% | 70% |

using Monte Carlo simulations.[xvi] Table 3 shows the equivalency between the layers' stochasticity ($\sigma_{\text{layer}}$, used during training and simulation) and cells' stochasticity ($\sigma_c$,

[xvi] Monte Carlo simulations with 100 runs (1000 for LeNet-5 and AlexNet) were performed to test the accuracy resilience of the trained networks against stochastic variations (we used the same 10000 test images for each of the 100 Monte Carlo runs). In each run, an error matrix is created per layer and then multiplied (element-wise) with the layers' parameters (weights and biases). The elements of the error matrices were sampled randomly either from a normal distribution $\mathcal{N}(1, \sigma^2_{\text{layer}})$, or from a log-normal distribution $e^{\theta}$, where $\theta = \mathcal{N}(0, \sigma^2_{\text{layer}})$, and $\sigma^2_{\text{layer}}$ represents the layers' stochastic variation, different from the cells' stochastic variation $\sigma_c$ (see section 2.3.4).

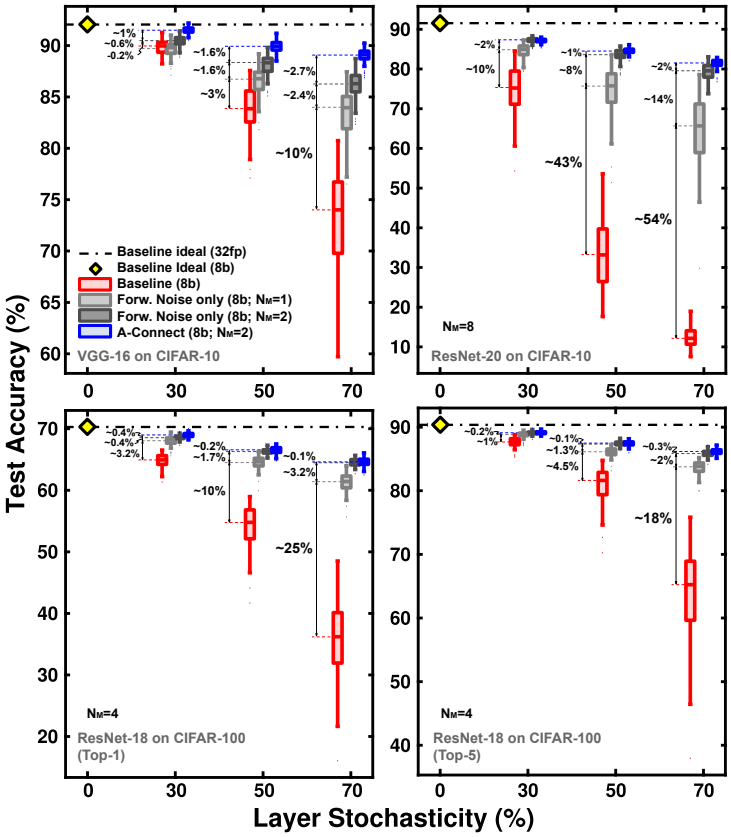


Figure 8. Accuracy resilience against layers' stochasticity ($\sigma_{\text{layer}}$) of NNs. Comparison between NN baseline (red), trained with forward-noise only and single error matrix (light gray), trained with forward-noise only and multiple error matrices (dark gray), and the trained version with the A-Connect methodology (blue): a) VGG-16 on CIFAR-10; b) ResNet-20 on CIFAR-10; ResNet-18 on CIFAR-100 c) top-1, and d) top-5.

following either normal or log-normal distribution) for parameters quantized at 8-bit, as described in section 2.3.4 (e.g., a $\sigma_{\text{layer}} = 70\%$ is equivalent to a $\sigma_c = 79.3\%$ for 2-bit MLC and using a log-normal distribution).

As expected, A-Connect shows an improvement in the neural network median accuracy and IQR compared to the baseline. With a 70% of stochasticity on the layers ($\sigma_{\text{layer}}$), the improvement of the median accuracy over the baseline oscillated around 15 to 68 percentage points (e.g., 78.6% with A-Connect at a $\sigma_{\text{layer}} = 70\%$ using a normal distribution compared to a baseline of 20.7% in ResNet-20). The performance boost with A-Connect is even more notorious when looking at the deviation of the results, where the IQR can be around 20X lower when using A-Connect (e.g., 0.6% with A-Connect at a $\sigma_{\text{layer}} = 70\%$ using a normal distribution compared to a baseline 14.3% in AlexNet).

For better visualization, Figure 8 shows the benefits of the A-Connect methodology in terms of accuracy. We compare the ideal 32-bit floating-point (32fp), the 8-bit baseline, the 8-bit multiplicative noise-injection during forward propagation (only) with a single error matrix and with several error matrices, and the 8-bit A-Connect implementations across multiple layers' stochasticity levels. We can compare the effectiveness of multiplicative injection with only one error matrix during training (as in DVA/MNT method[55,56]), as well as the major improvement in both accuracy and deviation when using multiple error matrices, similar to the results obtained in Figure 7. In the majority of the results, the fully A-Connect methodology (i.e., when including the error matrices during backpropagation) shows the best performance in terms of accuracy and deviation. The results show that the inclusion of multiple error matrices during forward and backward propagation not only improves the median accuracy, but it has a major impact on the deviation of the results.

### 2.4.2. Comparison with the DVA/MNT Method

In this subsection, we implemented the DVA/MNT method[55],[xvii][56] for comparison with the A-Connect methodology. The results presented in Table 4 corroborate the improvement of A-Connect over DVA/MNT. A-Connect is better than DVA/MNT in all the experiments performed, except for ten cases (out of 74) where DVA/MNT is better by $\leq$1%, where nine of them are at 0% stochasticity, and one at 30% of stochasticity. The accuracy improvement with A-Connect is more remarkable for higher stochastic variability ($\sigma_{\text{layer}}$ $\geq$50%). Furthermore, A-Connect presents less deviation in the accuracy results, where the IQR is 2X to 8X less when using A-Connect.

Two main differences made A-Connect perform better than the DVA/MNT methodology: first, DVA/MNT only considers the error injection during the forward propagation step, as a difference to A-Connect, that also takes it into account during back-propagation (see Algorithm 1). And second, DVA/MNT only uses a single error injection matrix per

[xvii] The baseline accuracy for AlexNet version in[55] applied to the CIFAR-10 dataset is 86.8%. According to the data provided by the authors, when using the DVA method in combination with dynamical fixed point data representation, it is possible to achieve 80.7% average accuracy in the CIFAR-10 dataset at $\sigma_{\text{layer}}$ =50% stochasticity.

Table 4. Test accuracy rates on the MNIST, CIFAR-10, and CIFAR-100 datasets using popular DNNs: LeNet-5, and AlexNet. Comparison between A-Connect and device-variation-aware (DVA), or multiplicative noise training (MNT), method proposed in[55,56]. The median (M) and the interquartile range (IQR) are presented for each experiment (M/IQR). The best performances are highlighted in colors.

| Dataset | DNN | Sim. Error ($\sigma_{\text{layer}}$) | DVA/MNT (Trained with $\sigma_{\text{layer}}$@ 30%) | A-Connect (Trained with $\sigma_{\text{layer}}$@ 30%) | DVA/MNT (Trained with $\sigma_{\text{layer}}$@ 50%) | A-Connect (Trained with $\sigma_{\text{layer}}$@ 50%) | DVA/MNT (Trained with $\sigma_{\text{layer}}$@ 70%) | A-Connect (Trained with $\sigma_{\text{layer}}$@ 70%) |
|---|---|---|---|---|---|---|---|---|
| MNIST | LeNet-5[a] | 0% | **98.9%/0.0%** | **98.9%/0.0%** | 98.6%/0.0% | **98.8%/0.0%** | **98.5%/0.0%** | 98.4%/0.0% |
| | | 30% | 98.0%/1.6% | **98.5%/0.2%** | 98.1%/0.5% | **98.6%/0.1%** | 98.0%/0.4% | **98.3%/0.2%** |
| | | 50% | 94.3%/4.9% | **97.7%/0.6%** | 96.3%/2.2% | **98.2%/0.3%** | 96.8%/1.5% | **98.0%/0.3%** |
| | | 70% | 87.9%/10.8% | **96.3%/1.5%** | 93.7%/4.8% | **97.6%/0.6%** | 95.1%/3.3% | **97.4%/0.5%** |
| CIFAR-10 | AlexNet[b] | 0% | **82.7%/0.0%** | 82.3%/0.0% | **82.7%/0.0%** | 81.7%/0.0% | 80.0%/0.0% | **80.6%/0.0%** |
| | | 30% | 80.3%/1.4% | **81.4%/0.2%** | **81.3%/0.8%** | 81.0%/0.2% | 78.3%/1.8% | **80.1%/0.3%** |
| | | 50% | 76.3%/3.8% | **79.4%/0.6%** | 79.2%/1.7% | **80.0%/0.4%** | 76.5%/3.3 | **79.8%/0.5%** |
| | | 70% | 69.4%/5.9% | **75.5%/1.4%** | 75.7%/3.0% | **77.7%/0.8%** | 76.4%/3.8 | **80.0%/0.6%** |
| | VGG-16[c] | 0% | 92.2%/0.0% | **92.7%/0.0%** | 92.2%/0.0% | **92.6%/0.0%** | 90.9%/0.0% | **92.6%/0.0%** |
| | | 30% | 89.8%/0.8% | **91.5%/0.4%** | 90.6%/0.7% | **91.7%/0.5%** | 89.3%/0.9% | **92.0%/0.4%** |
| | | 50% | 83.3%/3.5% | **88.9%/0.9%** | 86.8%/1.7% | **89.9%/0.7%** | 86.4%/1.9 | **90.7%/0.6%** |
| | | 70% | 74.6%/7.4% | **83.3%/2.4%** | 80.7%/3.9% | **86.4%/1.8%** | 84.0%/3.1 | **89.1%/0.7%** |
| | ResNet-20[c] | 0% | **91.2%/0.0%** | 90.1%/0.0% | **90.7%/0.0%** | 89.3%/0.0% | **86.3%/0.0%** | 85.6%/0.0% |
| | | 30% | 85.0%/2.2% | **87.2%/0.6%** | 86.4%/1.6% | **87.7%/0.4%** | 81.3%/3.2% | **84.3%/0.9%** |
| | | 50% | 69.4%/7.8% | **80.0%/2.5%** | 75.8%/7.0% | **84.7%/0.8%** | 72.2%/8.4 | **82.3%/1.6%** |
| | | 70% | 46.5%/14.1% | **64.4%/7.0%** | 58.2%/11.7% | **78.7%/2.6%** | 65.7%/12.2 | **81.6%/1.2%** |
| CIFAR-100 | ResNet-18[d] (Top-1) | 0% | **71.8%/0.0%** | 71.7%/0.0% | 71.5%/0.0% | **71.8%/0.0%** | 66.6%/0.0% | **68.4%/0.0%** |
| | | 30% | 68.2%/0.9% | **69.0%/0.4%** | 68.9%/0.7% | **69.9%/0.5%** | 64.0%/1.0% | **66.7%/0.8%** |
| | | 50% | 61.7%/2.1% | **63.9%/1.2%** | 64.61.2%/1.7% | **66.6%/0.7%** | 61.5%/1.7 | **65.3%/1.1%** |
| | | 70% | 49.1%/3.7% | **52.9%/3.2%** | 57.0%/2.4% | **59.7%/2.0%** | 61.4%/1.9 | **64.7%/0.8%** |
| | ResNet-18[d] (Top-5) | 0% | **90.7%/0.0%** | 90.4%/0.0% | 90.3%/0.0% | **90.9%/0.0%** | 87.7%/0.0% | **90.8%/0.0%** |
| | | 30% | 88.9%/0.4% | **89.1%/0.3%** | 88.8%/0.4% | **89.6%/0.2%** | 86.0%/0.6% | **87.6%/0.4%** |
| | | 50% | 84.9%/1.5% | **86.0%/0.7%** | 86.1%/0.9% | **87.5%/0.5%** | 84.1%/1.2 | **86.7%/0.7%** |
| | | 70% | 75.6%/3.4% | **78.1%/2.5%** | 80.6%/1.8% | **82.7%/1.4%** | 83.9%/1.3 | **86.2%/0.6%** |

**Training Times:**
[a] DVA/MNT: 3s/epoch; A-Connect: 5s/epoch.
[b] DVA/MNT: 31s/epoch; A-Connect: 39s/epoch.
[c] DVA/MNT: 13s/epoch; A-Connect: 21s/epoch.
[d] DVA/MNT: 34.5s/epoch; A-Connect: 41.5s/epoch.

training mini-batch, while A-Connect uses more than one error matrix per batch (i.e., 2 for LeNet-5, 8 for AlexNet, 2 for VGG-16, 8 for ResNet-20, and 4 for ResNet-18). Figure 8 visually shows the improvement of A-Connect over these methods, where the DVA/MNT method is represented by the light gray boxes.

### 2.4.3. Comparison with other *ex situ* Methods

In this subsection, we compare A-Connect to other *ex situ* training methodologies. Table 5 shows the comparison between the A-Connect method and four different adversarial attack training methods: adversarial regularization (AR[60]), adversarial regularization with multiplicative noise-injection during training (AR+FN[60]; FN stands for 'forward-noise'), adversarial weight perturbation (AWP[71]), and adversarial model perturbation (AMP[72]). We used the data provided by the experiments in[60] on the fashion

Table 5. Test accuracy rates on the fashion MNIST dataset using the same CNN. The mean (M) and the standard deviation (S) are presented for each experiment (M/S). The best results (M-wise) for each simulated error ($\sigma_{\text{layer}}$) are highlighted.

| Sim. Error ($\sigma_{\text{layer}}$) | Baseline | AR+FN[60] ($\beta_{rob} = 0.1$) | AR[60] ($\beta_{rob} = 0.25$) | AWP[71] ($\epsilon_{pga} = 0.0$) | AMP[72] ($\epsilon = 0.005$) | A-Connect | | |
|---|---|---|---|---|---|---|---|---|
| | | | | | | 30% | 50% | 70% |
| **0%** | 91.85%/0.00% | 91.88%/0.00% | 92.11%/0.22% | **92.35%/0.12%** | 91.34%/0.12% | 91.88%/0.00% | 91.93%/0.00% | 91.06%/0.00% |
| **30%** | 81.10%/4.26% | 91.25%/0.22% | 89.64%/0.65% | 89.88%/0.95% | 83.13%/0.22% | 90.80%/0.43% | **91.47%/0.22%** | 90.83%/0.27% |
| **50%** | 55.26%/9.58% | 89.36%/0.73% | 85.96%/1.87% | 82.59%/3.66% | 60.60%/7.36% | 88.037%/1.06% | 90.30%/0.54% | **90.37%/0.33%** |
| **70%** | 39.06%/7.77% | 84.19%/2.63% | 79.39%/4.40% | 65.38%/8.27% | 36.79%/7.42% | 83.68%/2.39% | 88.74%/0.92% | **89.75%/0.53%** |

* Data taken from[60].

MNIST dataset[88],[xviii] using the same convolutional NN as described in their paper.[xix] The A-Connect methodology is superior than the presented adversarial attack methods on the fashion MNIST dataset, except for the case when there is not stochastic variability (AR and AWP were better than A-Connect by 0.44% and 0.18%, respectively). The biggest difference in the accuracy performance is obtained at 70% of stochasticity, where the A-Connect methodology is 5.56% above the AR+FN method, 10.36% above the AR method, 24.37% above the AWP method, and 52.96% above the AMP method. Although the results with A-Connect alone are better than the adversarial methods shown, these methodologies are not mutually exclusive. In fact, AR+FN is an hybrid between the AR and DVA/MNT method. It could be possible to obtain even better results if the DVA/MNT is replaced by A-Connect in a new AR+AConnect method. The latter is out of the scope of this work.

Other statistical training approaches were reviewed as well, but their reported performance results were inferior in comparison with the DVA/MNT and the AR+FN methods. As an example, in[58] results for one layer and two layers fully-connected neural networks[xx] show 90% and 92% average accuracy with 3% and 1% standard deviation,

88 Han XIAO et al. *Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms*. 2017.

xviii The work in[60] also presented results in CIFAR-10 with ResNet-32, but only reported performance for very low stochastic variability ($\leq$10%), where the difference between the baseline performance and their proposal is minimum ($\leq$0.4%).

xix The CNN is composed of: 2 convolutional layers with kernel size of 4x4, stride of one, no padding, and 64 filters each; the convolutional layers are followed by three fully-connected layers with 256, 64, and 10 outputs, respectively. After each layer, we used a batch-normalization layer, followed by a ReLU layer. After each convolutional layer (the batchnorm and ReLU) the CNN implements a max-pool layer with pool size of 2x2 and stride of 2. The final layer is a softmax layer. Finally, we used 8 error matrices during A-Connect training.

xx Layers' sizes are not stated.

Table 6. Comparison with *in situ* and hybrid methods using VGG-16 over the CIFAR-10 dataset at $\sigma_c = 0.8$[89] using a log-normal distribution.

| | PM[81] | DVA+PM[81] | DigOff[69] | This Work |
|---|---|---|---|---|
| **Method** | *in situ* | hybrid | hybrid | ***ex situ*** |
| **Accuracy** | 81.4% | 87.94% | 88.48% | **88%[a]/ 0.93%**[b] |

[a] Obtained using noise model in section 2.3.4: $b = 4$, $n = 4$, and $N = 256$.
[b] The median (M) and the interquartile range (IQR) are presented (M/IQR).

respectively, in the MNIST handwritten digits dataset at $\sigma_c = 25\%$ device stochasticity. In the same paper, a comparison with the training approach in[64] is also presented for the same neural network architectures, same dataset, and same stochasticity level. The results show 89% with a 3% standard deviation for the one layer NN, and 38% average accuracy with 32% standard deviation for the two layers NN. Finally, and as stated in section 2.2, the results in[56] showed that the MNT method was superior than the additive noise-injection approach[57] ,[59], both in the modeling of the PCM cells' nature and in the accuracy performance in noisy environments.

### 2.4.4. Comparison with *in situ* and Hybrid Methods (log-normal distribution)

In this final set of experiments, we compared A-Connect with state-of-art *in situ* and hybrid (combination of *ex situ* and *in situ* approaches) methods to mitigate stochastic variability. Because these works focused their research in ReRAM devices, we used the log-normal distribution in the A-Connect methodology ($W^{err} \sim e^{\theta}$ with $\theta = \mathcal{N}(0, \sigma^2_{\text{layer}})$, see section 2.3.3). We implemented the VGG-16 model for the CIFAR dataset as in section 2.4.1, but with the modifications implemented for the A-Connect trained at $\sigma_{\text{layer}} = 0.7$ for a log-normal distribution (see the notes in Table 2).

Table 6 shows the comparison of the A-Connect methodology with other methods to mitigate stochastic variability, using either *in situ* approaches only, or *in situ* and *ex situ* approaches. The work in[81] showed the results for a priority mapping (PM) method, where it is necessary to detect the variation of each ReRAM (*in situ*). They also showed

[89] According to section 2.3.4, with an 8-bit parameter quantization using 4-levels (2-bits) MLCs, and with cells' stochastic variability following a log-normal distribution, a stochasticity of 80% at the cell ($\sigma_c$) is equivalent to a stochasticity of 70% for the layer ($\sigma_{\text{layer}}$).

results using PM+DVA to improve the performance of their method. The authors used an unary numeral representation with five 4-level (2-bits) MLCs ($n = 5$, $b = 4$, and $N = 16$). On the other hand, the work in[69] presented a method in which the stochasticity could be mitigated through digital offset compensation (*in situ*). The authors used 8-bits to quantized the NN's parameters with four 4-level MLCs ($n = 4$, $b = 4$, and $N = 256$). We implemented the noise model presented in section 2.3.4,[89] using the same numeral representation that in[69] for a fair comparison. The probability density function of the NN's parameters ($p_W(w_l)$ in eq. (5)) was obtained from the pre-trained parameters distribution.[xxi] The A-Connect methodology showed similar performance than the methods using *in situ* and hybrid approaches (almost half percentage point lower than the best performance in[69] ). The deviation of the data is also very low (IQR of 0.93%) even for such high stochasticity level (the deviation information is not available for other works). These results further motivates the use of A-Connect, which may improve even more when using it in combination with such *in situ* and hybrid approaches.

## 2.5. Conclusion

In this chapter, we have introduced a methodology to improve neural network resilience against stochastic variability when deploying neural networks in imprecise analog accelerators (i.e., synaptic cells). Furthermore, we developed a Keras/Tensorflow library, with versions of fully-connected and convolutional layers using A-Connect. The library can be coupled to standard machine learning platforms in a straightforward manner. We have presented simulation results applying the A-Connect methodology to popular DNNs, such as LeNet-5 for MNIST dataset, AlexNet, VGG-16, and ResNet-20 for the CIFAR-10 dataset, and ResNet-18 for the CIFAR-100 dataset. The experimental evidence compiled in this work showed that the proposed methodology significantly outperforms other *ex situ*, while achieving similar performance than *in situ*, and hybrid approaches to mitigate stochastic variability in the literature.

[xxi] Using A-Connect with the unary representation in[81] gives even better results.

# 3. ANALOG MACHINE LEARNING ACCELERATOR

## 3.1. Introduction

Inference on-the-edge with machine learning (ML) algorithms arises as one of the most effective solutions to handle big data for decision making. For this reason, the number of interconnected devices and the energy requirements to deploy ML algorithms make power consumption one of the main challenges for systems-on-edge (SoE) applications.

We previously discussed that analog computation in memory (CIM) acceleration has shown to be a promising alternative to obtain ultra-low-power ML systems at a circuit level, with energy efficiencies up to 10X-100X better than their digital counterparts[90,91,92]. At a system level, multi-mode SoEs with always-on capabilities (e.g., active mode, sleep mode) is a viable strategy to extend the lifetime of a battery-powered system. The idea is to cut off power-hungry resources, leaving what is strictly necessary to stay in a sleepy state.

Both circuit and system-level strategies can co-exist in what could be called an intelligent SoE. In this type of device, the system can be awakened from its sleep through an ML algorithm responding to an external stimulus (e.g., a voice command). In this case, the ML accelerator should be fully functional in the always-on domain, which means performing at ultra-low power levels. While working in the always-on regime implies limited resources, like lower clock rates ($<$1MHz) and nanoampere biasing currents, once the system awakes (i.e., active mode) the SoE should be able to perform at higher speeds, hence higher currents.

There have been works using separate accelerators for different SoE modes (e.g., one for always-on and one for active mode), implementing a custom design accelerator for

[90] H. VALAVI et al. “A 64-Tile 2.4-Mb In-Memory-Computing CNN Accelerator Employing Charge-Domain Compute”. In: *IEEE JSSC* 54.6 (June 2019), pp. 1789–1799.

[91] Zhewei JIANG et al. “C3SRAM: An In-Memory-Computing SRAM Macro Based on Robust Capacitive Coupling Computing Mechanism”. In: *IEEE JSSC* 55.7 (2020), pp. 1888–1897.

[92] Daniel BANKMAN et al. “An Always-On 3.8 $\mu$ J/86% CIFAR-10 Mixed-Signal Binary CNN Processor With All Memory on Chip in 28-nm CMOS”. in: *IEEE JSSC* 54.1 (2019), pp. 158–172.

the resources available for each mode**?**. Having two separate accelerators impose a great area overhead on intelligent SoEs, which in principle should be as compact as possible. To our knowledge, our work is the only study investigating multi-mode ML accelerators performing efficiently at different current levels and clock rates. This work makes the following contributions:

- An end-to-end analog datapath that avoids data conversion by staying in the same physical domain, the current domain.
- An analog macro that incorporates not only MAC operations but commonly used ML operations within the analog domain, such as ReLU and scaling (the latter enables normalization operations), as well as memory capabilities for pipeline execution.
- Since all analog operations in our macro are current-based, we implement a wide-band current mirror that enables a wide range of operating frequencies while improving the energy efficiency.

This chapter is divided as follows: section 3.2 shows an overview of the proposed analog macro datapath at system level, as well as the detailed implementations of each one of the block and modules that conform the ML macro at transistor level. Section 3.3 reviews the deterministic and stochastic sources of error, as well as the strategies implemented to diminish them. It also shows the Monte Carlo simulation results of each individual macro's block and modules to understand the dominant factors of the total stochastic and deterministic errors. Section 3.4 presents the results of the analog macro's layout, simulation results, and a performance comparison to state-of-art macros at different technology nodes, as well as a performance estimation at a 28nm technology. Finally, section 3.5 shows the conclusion of this chapter.

### 3.2. Computation-in-Memory Analog Macro

The common analog computation-in-memory (CIM) macro datapath has DACs to convert the digital input activations into analog signals, an analog MAC or VMM (vector-matrix multiplication) module, and ADCs to convert the MAC module's output back to the digital domain. Because our proposal is to avoid (as much as possible) analog-to-

digital (AD) and digital-to-analog (DA) conversions for energy efficiency, it is important to include commonly used ML operations, which are executed in-between MAC modules. We decided to include two operations in the analog domain (besides the MAC operation): ReLU, and scaling.

Since typical ML applications involve complex and non-linear tasks, a non-linear function is applied over the output of the MAC operation. We included the ReLU activation function (defined in eq. (6)) in our datapath because of its simplicity, as well as being the most popular activation function in ML applications.

$$y_a = \mathrm{ReLU}(y) = \max(0, y) \tag{6}$$

We included another block in the datapath that can perform scaling operations. The purpose of this block is to enable normalization methods, such as batch normalization (BN)[93]. As an example, consider the MAC operation defined in eq. (7) (considering a fully-connected layer), with a $d$-dimensional layer's input $x = (x^{(1)} \dots x^{(d)})$:

$$y^{(k)} = \mathbf{W} x^{(k)} + b \tag{7}$$

where $\mathbf{W}$ is the layer's weights matrix, and $b$ is the layer's bias vector. It is possible to obtain the output of the BN layer as:

$$\hat{y}^{(k)} = \frac{y^{(k)} - \mu_B^{(k)}}{\sqrt{\left(\sigma_B^{(k)}\right)^2 + \epsilon}} \longrightarrow \mathrm{BN}\left(y^{(k)}\right) = \gamma^{(k)} \cdot \hat{y}^{(k)} + \beta^{(k)} \tag{8}$$

where $\mu_B^{(k)}$ and $\sigma_B^{(k)}$ are the batch mean and the batch standard deviation (obtained during training) in the dimension $k$, respectively, and $\gamma^{(k)}$ and $\beta^{(k)}$ are the BN learned parameters during training ($\epsilon$ is a small value to avoid division by zero). Because the BN layer performs a linear operation, we can fold the BN parameters (as well as the batch mean and standard deviation) back to the layer's parameters. Re-writing eq. (8),

[93] Sergey IOFFE et al. "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". In: *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*. Ed. by David BLEI et al. 2015.

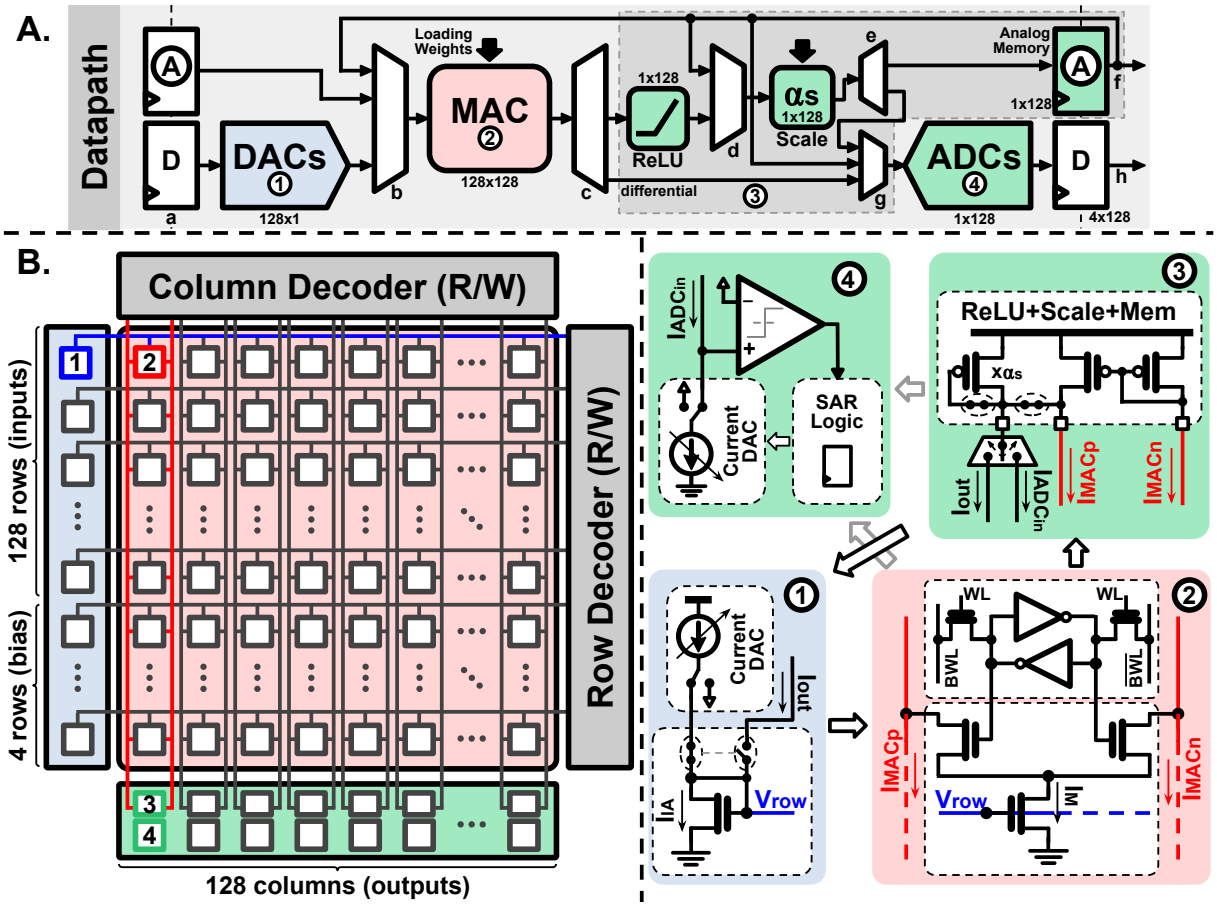


Figure 9. SRAM-based CIM macro (summary). A) Datapath of the CIM macro. B) Top-level architecture of the vector-matrix multiplication CIM macro: B.1) input rows' DAC cell; B.2) synaptic 9T-SRAM cell for MAC operation; B.3) output columns' ReLU+Scaling+Memory cell; B.4) output columns' SAR-ADCs.

we obtain:

$$\mathrm{BN}\left(y^{(k)}\right) = \alpha_s^{(k)} \cdot \left(\mathbf{W}x^{(k)} + b_f^{(k)}\right) \tag{9}$$

where $\alpha_s^{(k)}$ is the scaling factor (per dimension), and $b_f^{(k)}$ is the folded-back bias:

$$\alpha_s^{(k)} = \frac{\gamma^{(k)}}{\sqrt{\left(\sigma_B^{(k)}\right)^2 + \epsilon}} \quad \text{and} \quad b_f^{(k)} = b - \mu_B^{(k)} + \frac{\beta^{(k)}}{\alpha_s^{(k)}} \tag{10}$$

Hence, by using a scaling factor $\alpha_s^{(k)}$, we can effectively enable batch normalization operations.

Figure 9.A shows the analog macro's proposed datapath, including the ReLU and scaling blocks. Supposing that the layer's parameters (as well as the scaling factors) have been loaded, the dataflow is as follows (using the notation on the multiplexers in Figure

9):

a. The first thing to note is that the input of the mixed-signal datapath can be retrieved either from digital or analog memory (i.e., another macro's output stored in the analog memory in a previous cycle).
b. In a second instance, the MAC module can receive one of three different inputs: the digital converted inputs coming from the DACs; another macro's output stored in the analog memory in a previous cycle; the macro's output (same macro) stored in the analog memory in a previous cycle.
c. Then, the MAC's output can either go to the single-ended analog ReLU operation, or directly to a differential ADC.
d. The analog scaling block receives either the ReLU block's output, or the value stored in the analog memory in a previous cycle.
e. The scaling block's output can either be saved in the analog memory, or it can be converted by the ADC's single-ended mode. The scaling operation is executed after the ReLU activation function. The latter is equivalent to apply batch normalization before ReLU (i.e., apply the ReLU function to eq. (9) and with $\alpha_s^{(k)} > 0$):

$$y_a = \mathrm{ReLU}\big(\mathrm{BN}\big(y^{(k)}\big)\big) = \alpha_s^{(k)} \cdot \mathrm{ReLU}(\mathbf{W}x^{(k)} + b_f^{(k)}) \tag{11}$$

f. After saving the scaling block's outputs, the values can be used in the next cycle either by another analog macro, or within the same macro (i.e., by the MAC module, by the scaling block, or by the ADC single-ended).
g. The ADC can take three different inputs: MAC's output (differential mode); the scaling block output (single-ended mode); the value stored in the analog memory in a previous cycle (single-ended mode).
h. After an AD conversion, the macro delivers the digital output to the top-level system for further processing in the next cycle.

Figure 9.B shows an overview of the proposed accelerator macro's hardware arrangement. We use a CIM architecture (MAC and memory cell merged) to reduce power consumption due to data movement. We employ 128 current DACs (Figure 9.B.1) to convert the digital input activations (e.g., 128 pixels) into a current signal, and then

mirror it into an array of cells. The latter allows performing several MAC operations simultaneously over the same input current signal. The MAC cell is conformed by a SRAM cell (binary weights), switches, and the V-to-I transistor (9T-SRAM cell shown in Figure 9.B.2). If the value stored (weight) is a digital '+1' or a digital '-1', the mirrored current $I_M$ will flow to $I_{MACp}$ node or $I_{MACn}$, respectively. By including biases at each column, one can effectively have the output of a MAC operation in eq. (7) (with $x = I_M$).[i]

We also deploy 128 columns for an effective 128x128 binary weight matrix arrangement. As shown in Figure 9.A macro's datapath, after the MAC operation is completed, the output of each column can go to an analog ReLU+Scaling+Memory module (ReScaM). Here the activation function ReLU, as well as an scaling factor $\alpha_s$, is applied to the output current $I_{MAC}$, enabling the operation in eq. (11). The output of the ReLU and scaling operations is saved in a current memory to await for further instructions in the next cycle. Finally, we also include one SAR-ADC per column with differential and single-ended modes, to convert the columns output current to digital in any of the steps specified previously in the macro's datapath in Figure 9.A.

The following sections describe in more detail the modules used for the CIM analog macro, namely: the input DACs, the ReScaM module, and the column's SAR-ADC.

### 3.2.1. Input DACs and the Wideband Current Mirror (WBCM)

Figure 10.a shows the 4-bit current-steering DAC used to convert the layer's digital input activations (IA). We arranged 128 of the IA-DAC cells to drive the CIM macro's rows. We also used 4 rows of the MAC module to represent the layer's bias current (i.e., the $b_f$ value in eq. (11)). The bias cells in Figure 10.b set the currents for the actual layer's biases DAC, which is constructed with four 9T-SRAM cells (see figure 9.B.2). Similarly to how the VMM operation works, all the bias cells' current multiplied by a digital '+1' will be accumulated in the $I_{MACp}$ branch, while the ones multiplied by a digital '-1' will be accumulated in the $I_{MACn}$ branch. The total column's bias current can be calculated as $I_{b_f}[m] = I_b \cdot \sum_{i=0}^{3} b_{f_m}[i]2^i$, with $m \in [0, 127]$ representing the column's

[i] The currents $I_{MACp}$ and $I_{MACn}$ get subtracted in the column.

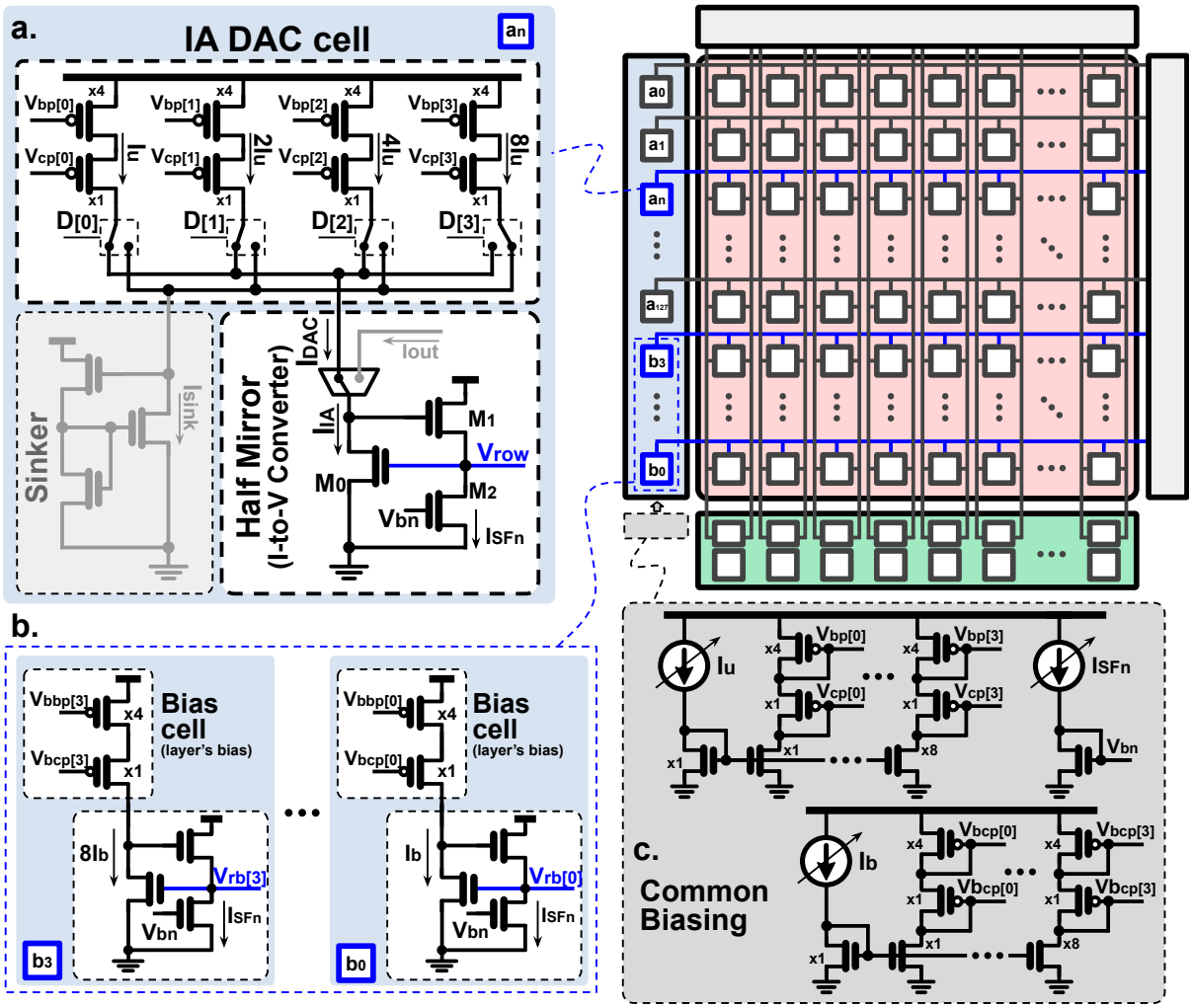


Figure 10. Actual implementation of the DAC cell in Figure 9a.1: a)input activation (IA) DAC cell, b) Layer's bias cells, c) Common biasing used by the macro's IA DAC cells and the layer's bias cells.

number, and $b_{f_m}[3:0]$ is the binary representation of the column's bias value (i.e., $b_{f_m}[i]$ can either be '+1' or '-1').

The arrangement of the current-steering DAC is not the conventional converter where only one unit current ($I_u$) is multiplied through binary-scaled mirrors. As shown in Figure 10.a, we used current mirrors with the same size in the IA DAC cell, but the current through them is binary-scaled with respect to $I_u$ to obtain $I_{DAC}$. We decided to use the latter implementation, rather than the conventional one, to reduce the mismatch variability at the least significant levels of the DAC's output,[ii] while achieving a similar

[ii] Normally the input data is normalized (e.g., by using a batch normalization layer), which adjusts the mean value of the input data to zero. Furthermore, if the activation function used in the previous layer is ReLU, almost half of the input data would be zero.

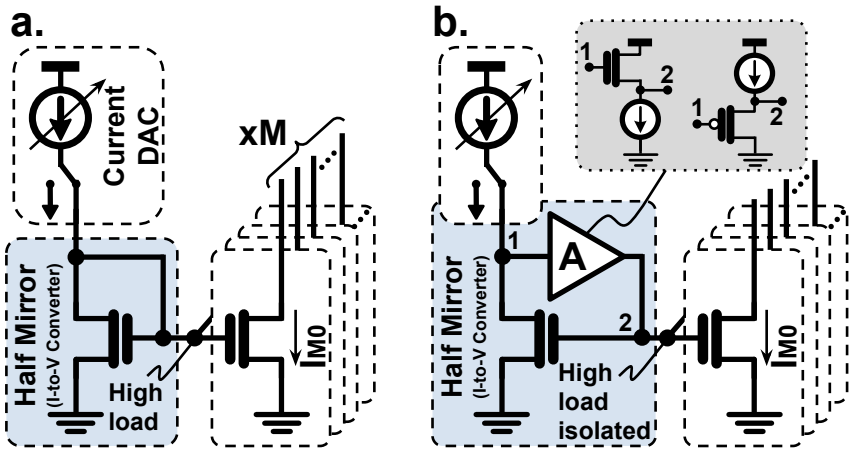


Figure 11. Current mirror accelerator's row driver: a) simple mirrors, b) WBCM.

cell size; instead of using four mirrors scaled to 1X, 2X, 4X, 8X,[iii] we use four mirrors at the same scale of 4X. Figure 10.c shows the current biasing used for the DAC cell rows' drivers.

A crucial element to achieve an analog macro with high energy efficiency is the half mirror in the DAC cells driving the corresponding macro's row. Figure 11.a shows the current-mode implementation for a single row of an accelerator array, used in several ML accelerators' macro to perform the multiply and accumulation (MAC) operation[49,94,95]. The problem with the implementation in Figure 11.a is the restrictive load imposed by the multiple cells of the accelerator's row, which limits the maximum operating frequency and the minimum current level of the application. In this scenario, the DAC current should be sufficient to drive the high-capacitive load. Furthermore, when the input current is zero, and since there is no pull-down mechanism at this node, the only way to discharge the high-capacitive node is through leakage current. For this reason, the simple mirror implementation has an offset current which will be replicated several times across the array[49],[iv] affecting the energy efficiency of the application.

94 Mohammad BAVANDPOUR et al. "aCortex: An Energy-Efficient Multipurpose Mixed-Signal Inference Accelerator". In: *IEEE JESSCDC* 6.1 (2020), pp. 98–106.

95 Yi CHEN et al. "A 2.86-TOPS/W Current Mirror Cross-Bar-Based Machine-Learning and Physical Unclonable Function Engine For Internet-of-Things Applications". In: *IEEE TCAS-I* 66.6 (2019), pp. 2240–2252.

iii The scaling is with respect to a unit transistor size of 890nm/180nm.

iv The work in[49] uses an NMOS transistor controlled by a pulse signal, explicitly pulling down the voltage of this node to reduce power consumption.

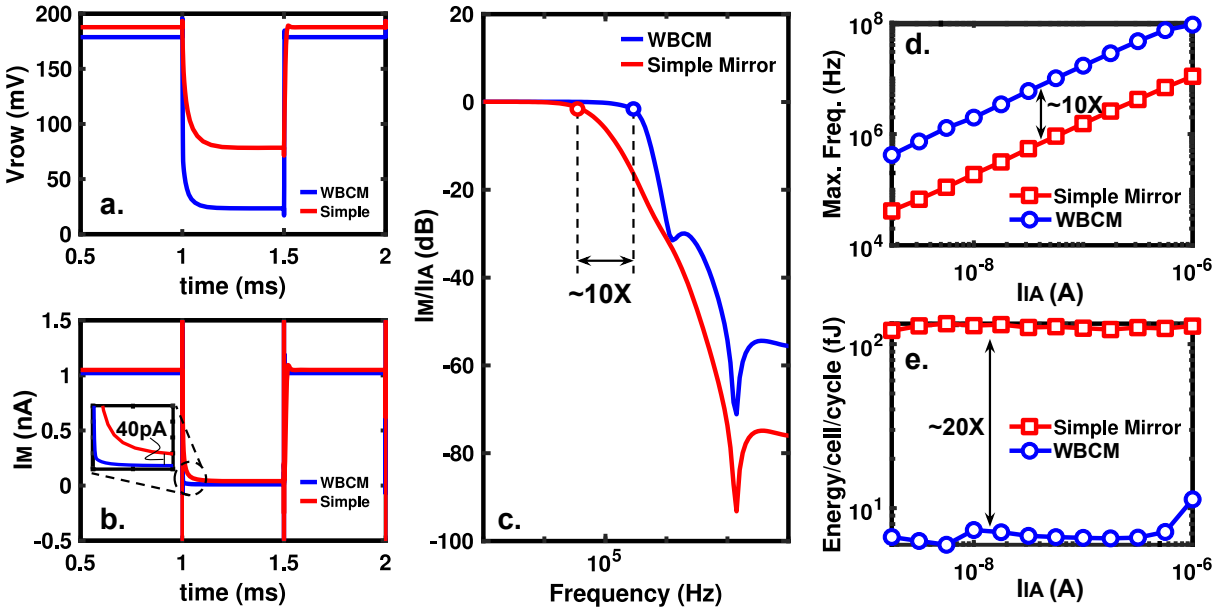


Figure 12. Comparison between the WBCM and simple mirrors behavior for a 0nA-to-1nA pulse input current: a) $V_{row}$ transient response; b) $I_M$ transient response; c) Frequency response of the mirrors' transfer functions; d) Maximum operation frequency against input current levels; e) energy efficiency per cell against input current levels.

We propose to isolate the DAC's output node and the high-capacitive node with a buffer (as shown in Figure 11.b) to improve the frequency response of the mirrors and to allow different current levels. We use a source-follower to drive the high-capacitive node due to its compact implementation. The source-follower may be implemented using the PMOS version, allowing lower voltages at the DAC's output node (node 1 in Figure 11.b) to enhance the dynamic input range of the mirror[96,97,98]. Although this configuration manages to have faster transient responses, the PMOS source-follower version is not strong enough for pull-down purposes, reverberating in the energy efficiency as in the simple mirror case. Hence, the NMOS source-follower is more convenient for this application, allowing lower DAC currents.

To compare the performance of the WBCM against the simple mirrors within an accelerator, a digital periodical pulse was set to control the 4-bits current steering DAC,

[96] J. RAMIREZ-ANGULO. "Low Voltage Current Mirrors for Built-in Current Sensors". In: *IEEE ISCAS*. vol. 5. 1994, 529–532 vol.5.

[97] S.S. RAJPUT et al. "A Current Mirror for Low Voltage, High Performance Analog Circuits". In: *AICPEF* 36.3 (2003), pp. 221–233.

[98] Ying-Chuan LIU et al. "A CMOS Current Mirror with Enhanced Input Dynamic Range". In: *ICICIC*. 2008, pp. 571–571.

making the DAC's current ($I_{DAC}$) vary between 0 LSB and 1 LSB. The latter simulates the transient behavior of the layer's input ($I_{IA} = I_{DAC}$) of a neural network by taking into account that the two most common input values are the two lowest levels. The voltage at the drain of each cell on the accelerator's row (128 cells per row) was set to a constant value. With this setup, one can estimate the behavior and the performance of the accelerator down to a single MAC cell.[v]

As stated before, the WBCM's buffer not only allows to have faster responses but to pull down the high-capacitive node to lower gate voltages (see Figure 12.a), allowing lower synaptic current levels ($I_M$; see Figure 12.b). Figure 12.c shows the effect on the frequency response of the proposed implementation ($I_M/I_{IA}$). Compared to the simple mirror, the WBCM increases in $\sim$10X the bandwidth of the transfer function.

Furthermore, by modifying the bias current of the WBCM's buffer, proportionally to the unit current of the DAC, one can obtain a wide range and wideband accelerator. With this approach, it is possible to maintain the energy efficiency across different current levels and operation frequencies, as shown in Figure 12.d.[vi] The figure illustrates the simulation results obtained for 180nm CMOS technology. The results show an increment of 20X on the energy efficiency per cell and 10X on the maximum operating frequency (Figure 12.e), compared to the simple mirror.

### 3.2.2. Column ReLU, Scaling, and Analog Memory

This section presents the ReScaM (ReLU+Scaling+Memory) module, a multiple purpose block that can sequentially execute three operations over the column output current (obtained within the MAC module): the ReLU activation function, the scaling operation, and finally, current storage, working as an analog register to enable the macro's output for the next cycle.

Prior to the execution of any of the operations in the ReScaM module, it is necessary to subtract the currents $I_{MAC_p}$ and $I_{MAC_n}$ to obtain the actual MAC output $I_{MAC}$ (the output of the columns MAC operation is differential). We use a cascaded WBCM to

[v] This approach only takes into account the energy consumption of the MAC operation.

[vi] We calculated the energy per cell at the maximum operating frequency and using a single pulse cycle. The supply voltage is 1.8V. The buffer's energy is taken into account as well.

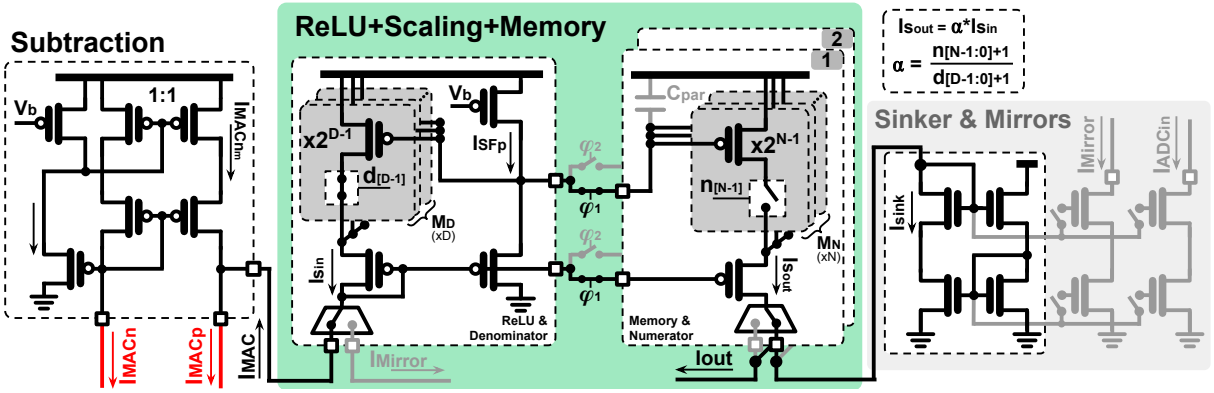


Figure 13. Output columns' current-mode ReLU, scaling, and analog memory (ReScaM) cell.

mirror the $I_{MAC_n}$ current into the $I_{MAC_p}$ branch to successfully subtract the two currents (subtraction cell in Figure 13, $I_{MAC} = I_{MAC_p} - I_{MAC_{n_m}}$).

The ReLU function is applied to the $I_{MAC}$ current after the subtraction cell. The ReLU function (eq. (6)) establishes that every negative input is clipped to zero at the output, while positive values are bypassed to the output. The latter is achieved by making the input current ($I_{s_{in}} = I_{MAC}$) pass through a diode-connected transistor, or through a current mirror, in which only the positive inputs are mirrored, while negative ones are clipped to zero. The ReLU cell within the ReScaM module is implemented as a cascaded WBCM mirror, as shown in Figure 13.

Scaling is applied to the $I_{s_{in}}$ current after the ReLU function. By adjusting the number of transistors in parallel at both ends of the WBCM mirror (the same one used for the ReLU function), the mirror can be repurposed for scaling operations. One can effectively multiply the current $I_{s_{in}}$ by a scaling factor:

$$\alpha_s = \frac{I_{s_{out}}}{I_{s_{in}}} = \frac{n+1}{d+1} \tag{12}$$

where $n$ and $d$ are the number of transistors in parallel ($M_N$ and $M_D$) in the numerator's branch (mirror's output) and in the denominator's branch (mirror's input), respectively (see the numerator and denominator branches in Figure 13). We used 4-bit numbers to represent $n$ and $d$, which can be programmed by the user.[vii]

[vii] In particular, the numerator can be programmed per column. We used four additional rows in the SRAM macro in Figure 9.B to enable storage of these numerator numbers.

Finally, we add memory functionality on the ReScaM module, specifically within the numerator's branch (mirror's output). During the scaling operation, the output current ($I_{s_{out}} = \alpha_s \cdot I_{s_{in}}$) is directed to the sinker block in Figure 13. This is also called the saving state since the output current is being stored as the voltages at the intrinsic gate and drain capacitors of the cascaded transistors on the numerator's branch (mirror's output). During the loading state, the gates of the cascaded transistors are disconnected, while $I_{s_{out}}$ is steered to the output ($I_{out} = I_{s_{out}}$) for the next cycle operation. Two numerator branches are deployed to enable continuous operation of the ReScaM module, i.e., when one branch is in the saving state, the other branch is in the loading state, and vice versa.

There are several non-idealities that affect the proper behavior of the ReScaM module as an analog memory, being the charge injection and leakage charge the ones with the major effects. We decided to use switches with dummy transistors to cancel the charge injection effect, as shown in Figure 14.a. We also used non-overlapping clocks ($\varphi_1 - \varphi_2$ in Figure 13) for both memory units (i.e., the load and save states are non-overlapping). On the other hand, the leakage current was trickier since its effect increases with higher temperatures. The major effect comes from the subthreshold conduction, which becomes the channel current leakage with drain-induced barrier lowering (DIBL), that can be expressed as[99] (for a PMOS transistor):

$$I_c = I_{c_0} \cdot e^{(V_{sg} - |V_{th_p}|)/nV_T} \cdot \left(1 - e^{-V_{sd}/V_T}\right) \cdot \left(e^{\eta V_{sd}/nV_T}\right) \tag{13}$$

where $I_{c_0} = \mu C_{sth} V_T^2 W/L$, $\mu$ denotes the carrier mobility, $C_{sth}$ is the summation of the depletion region capacitance and the interface trap capacitance (both per unit area of the transistor gate), $V_T = kT/q$, $W$ and $L$ denote the transistor width and length, $\eta$ is the DIBL coefficient, and $n$ is the slope shape factor.

We used a low-leakage switch to reduce the effect of subthreshold current by setting $V_{sd} \approx 0$V voltage of the transistor when turned off[100]. The switch shown in Figure

[99] F. FALLAH. "Standby and Active Leakage Current Control and Minimization in CMOS VLSI Circuits". In: *IEICE Transactions on Electronics* E88-C.4 (Apr. 2005), pp. 509–519.

[100] Jiangtao XU et al. "Low-leakage analog switches for low-speed sample-and-hold circuits". In: *Microelectronics Journal* 76 (June 2018), pp. 22–27.

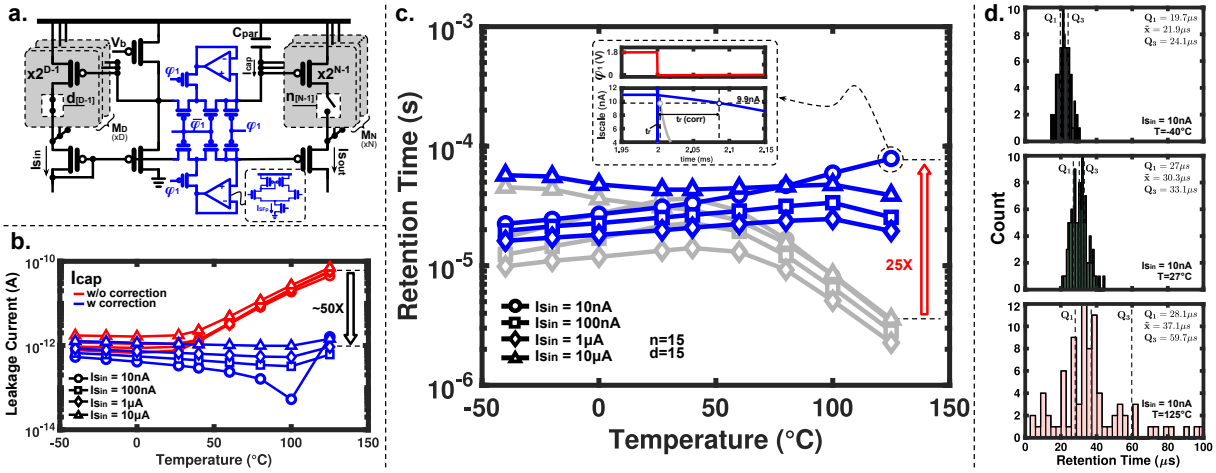


Figure 14. ReScaM module retention time (memory mode): a) Core of ReScaM module showing the low channel leakage switch (in blue); b) Leakage current with and without the correction (low-leakage switch) vs temperature; c) Retention time of ReScaM module vs temperature for different input currents, with and without the low-leakage switch; d) Monte Carlo simulations with process and mismatch variations (100 simulations each temperature).

14.a (in blue) has two stacked PMOS transistors and a subthreshold buffer with its input connected to the storage capacitor ($C_{par}$, the intrinsic transistor's gate capacitor). When the main switch is turned on, the buffer's output is not connected to the main signal path, and the switch works normally. When the main switch is off, the output of the source follower is connected to the internal node of the stacked transistors. We managed to reduce the leakage current at higher temperatures around 25X to 50X, as shown in Figure 14.b.

Figure 14.c shows the retention time of the memory unit against temperature for different current levels.[viii] The retention time is the time during the loading state at which the output current goes down to 90% of the saved current. The simulation results show an improvement on the retention time of around 1.3X for temperatures below 60°C, and 3X to 50X above 80°C, with the lowest retention time being 10$\mu$s at 125°C. We also performed Monte Carlo simulations (Figure 14.d) with process and mismatch variations at -40°C, 27°C, and 125°C for an input current of 10nA.[ix] As expected, the worst results are at 125°C, having a median retention time of 37.1$\mu s$, with the lower quartile at 28.1$\mu s$, and its lowest value at 4.1$\mu s$. The latter would allow operating frequen-

[viii] We used $I_{s_{in}} = 10I_{IA}$ (see Figures 9.B and 10.a).

[ix] This current level presents the highest retention time degradation due to leakage currents.

cies higher than 250kHz when using the ReScaM module as a memory unit, which is agreement with the result shown in Figure 12.d for $I_{IA} = 1$nA.

### 3.3. CIM Analog Macro Non-idealities

The results from chapter 2 allow us to reduce power and area consumption since a great amount of stochastic variability is permissible in exchange of a small detriment in accuracy performance. In a similar fashion, deterministic error can be completely removed by considering it during the NN training stage. Because of the latter, we opted for simplified design choices in many of the CIM analog macro cells and modules, focusing mainly in area reduction and energy efficiency. For example, Including an extra cascode transistor in the synaptic cells would reduce absolute and relative mirror ratio's error, but it would increase the area of the analog macro around 15%. We took this trade-off into consideration and use a co-design approach where errors derived from minimalist hardware design are remedied at software level.

Translating the ideal VMM or MAC operation from eq. 7 to current-mode implementation (i.e., $y = I_{MAC}$, $\mathbf{W} = w_{i,m}$, and $x = I_M$), and considering the output current of a single column, one can obtain (excluding column biases):

$$I_{MAC_m} = I_{MAC_{p_m}} - I_{MAC_{n_m}} = \sum_{i=1}^{N_{row}} \left( w_{p_{i,m}} I_{M_{i,m}} - |w_{n_{i,m}}| I_{M_{i,m}} \right) \tag{14}$$

with $m \in [0, 127]$ representing the macro column's number, $N_{row}$ the number macro's rows, and $w_{p_{i,m}} = \max(0, w_{i,m})$, $w_{n_{i,m}} = \min(0, w_{i,m})$ as the positive and negative layer's weights, respectively. The synaptic cell current $I_{M_{i,m}}$ can be represented as:

$$I_{M_{i,m}} = \left[ (N_{DAC_i} \cdot I_u) \cdot \alpha_{D_i} \right] \cdot \alpha_{w_{i,m}} \tag{15}$$

where $I_{DAC_{0i}} = N_{DAC_i} \cdot I_u$ is the ideal i-th row input activation DAC output, $\alpha_{D_i}$ is the IA DAC gain factor ($\alpha_{D_i} = I_{DAC_i} / I_{DAC_{0i}}$), and $\alpha_{w_{i,m}}$ is the synaptic cell gain factor ($\alpha_{w_{i,m}} = I_{M_{i,m}} / I_{DAC_i}$). Both $\alpha_{D_i}$ and $\alpha_{w_{i,m}}$ gain factors are caused due to non-idealities, either deterministic or stochastic.

Figure 15.a.1 (top) and Figure 15.a.2 (top) show the simulated gain factors $\alpha_{D_i}$ and

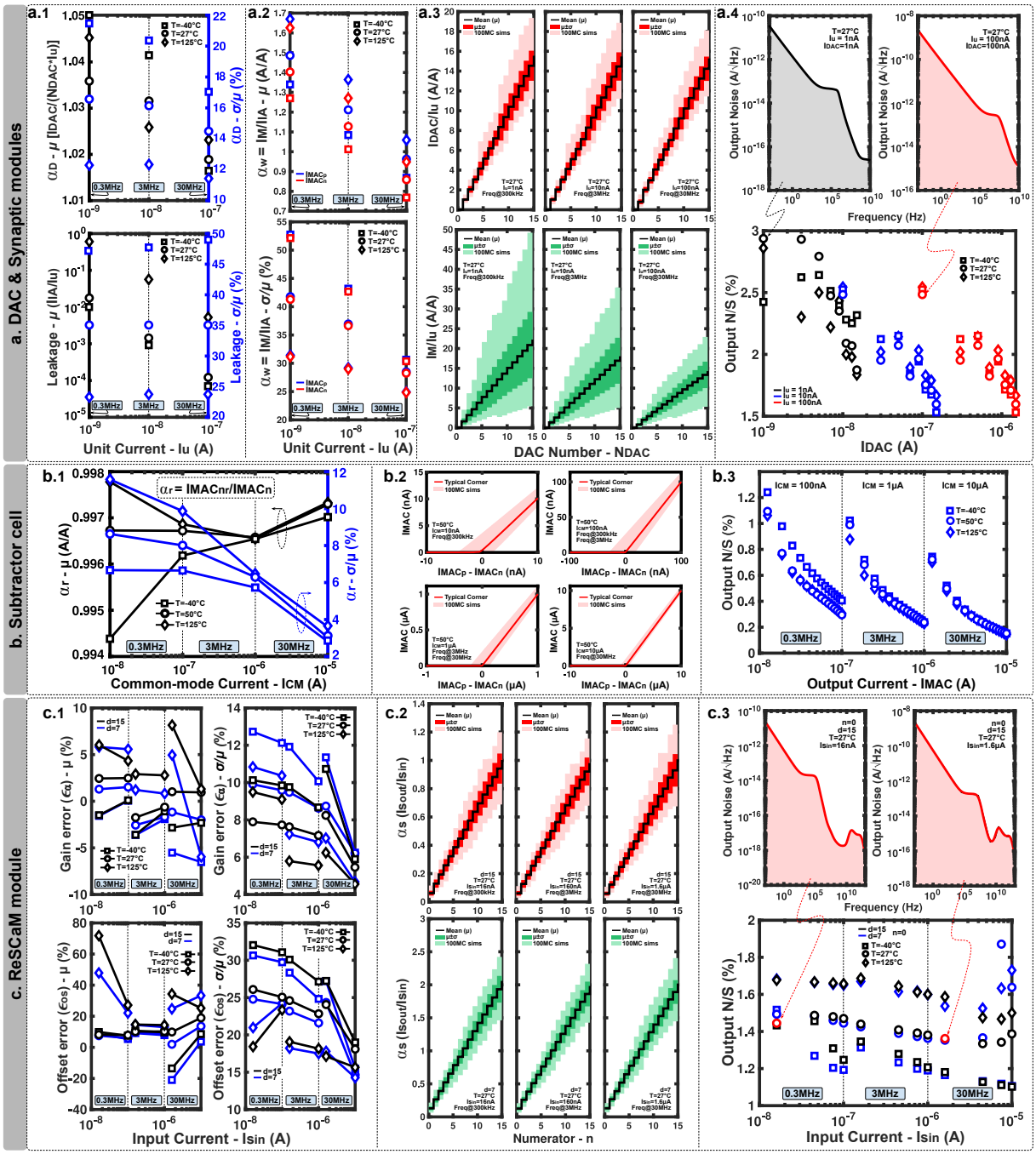


Figure 15. MC simulation results analog macro's modules and cells. a) DAC module and synaptic cell: a.1) DAC gain ($\alpha_D = I_{DAC}/(N_{DAC}I_u)$) mean ($\mu$) and variability ($\sigma/\mu$), and total DAC and synaptic cell offset mean and variability; a.2) synaptic cell gain ($\alpha_w = I_M/I_{DAC}$) mean and variability; a.3) DAC transfer function (top-red), and DAC with synaptic cell transfer function (bottom-green); a.4) output current ($I_M$) noise-to-signal ratio (N/S). b) Subtraction cell MC simulation results: b.1) current ratio ($\alpha_r = I_{MAC_{nm}}/I_{MAC_n}$) mean ($\mu$) and variability ($\sigma/\mu$); b.2) effect of the current ratio variability on the ReLU output current $I_{MAC}$; b.3) output current noise-to-signal ratio (N/S). c) ReScaM module MC simulation results: c.1) scaling factor ($\alpha_s = I_{s_{out}}/I_{s_{in}}$) gain error and offset error: mean ($\mu$) and variability ($\sigma/\mu$); c.2) effect of the scaling factor variability on the ReScaM module transfer function; c.3) output current noise-to-signal ratio (N/S).

$\alpha_{w_{i,m}}$, respectively, across different unit currents $I_u$.[x] Ideally, these should be unitary gain factor, but due to deterministic errors, the absolute values deviate from the ideal one. The main cause of this absolute error is channel length modulation. Since we used cascode mirrors in the current steering DAC (see Figure 10), channel length modulation is not as significant in $\alpha_{DAC_i}$ ($\leq$5%) as it is in $\alpha_{w_{i,m}}$, where the error can go up to 70% for high temperatures at the low current-mode ($I_u = 1$nA).

On the other hand, the absolute error is not as important as the relative error between the mirrored currents flowing to the $I_{MAC_{pm}}$ and $I_{MAC_{nm}}$ branches.[xi] Because the voltage at node $I_{MAC_{pm}}$ is different from that at node $I_{MAC_{nm}}$, the drain voltages of the synaptic cell mirrors vary when steered from one branch to the other, creating the error. Figure 15.a.2 (top) shows both $I_{M_{p_i}}/I_{IA_i}$ and $I_{M_{n_i}}/I_{IA_i}$ ratios ($I_{IA_i} = I_{DAC_i}$), and how they change with respect to the unit current $I_u$. Fortunately, the relative error is not as significant as to be harmful to the accuracy performance of the DNNs trained in this thesis ($1 - I_{M_{p_i}}/I_{M_{n_i}} \leq 10\%$, according to simulations). Even without considering the relative error during the DNNs training stage, the network showed the same performance as with the ideal case.

Leakage current is another important deterministic error for energy efficiency, as well as an important factor for the NN accuracy performance. Leakage current translates at software level to having a small real value when the layers' input activation is zero. Figure 15.a.1 (bottom) shows the leakage current at the output of a single synaptic cell (combination of IA DAC and synaptic cell) relative to $I_u$. As shown in the figure, the error is more notorious at high temperatures. On the other hand, its relative effect can be diminished by increasing $I_u$. The DNNs trained in this thesis showed the same accuracy performance when zero input activations were replaced by real values lower than $0.4$X the least significant discrete value. According to simulations, only high temperatures ($>80°$C) and low current-mode operation would show a degradation in accuracy performance due to leakage effect. Still, it is possible to remove it completely

[x] All simulations in Figure 15 show different current and frequency domains for multi-mode SoE operation. Different $I_u$, $I_{DAC}$, $I_{CM}$, or $I_{s_{in}}$ state a different current-domain of the macro's operation.

[xi] Absolute errors can be translated to linear factors within the MAC operation, which do not affect the accuracy performance behavior of the NN.

by considering during NN's training.

Figure 15.a.1 (blue and right axes) and Figure 15.a.2 (bottom) show the stochastic spatial variability (mismatch) of the DAC module and synaptic cell gain factors, respectively. As expected, the stochasticity in the synaptic cells ($\leq$53%) is greater than the one in the IA DAC modules ($\leq$22%) since the the synaptic cells use smaller devices to enable a more compact CIM macro. In fact, the synaptic cells' stochasticity are the dominant error in the entire system.

Figure 15.a.3 shows the effect of the deterministic and spatial stochastic errors for the DAC's output only (top), and the DAC with the synaptic cell (bottom). The figure presents the transfer function summary for 100 Monte Carlo simulations, highlighting the mean transfer curve, the standard deviation limits, and the maximum and minimum limits. The fan-like shape is characteristic of a multiplicative error (e.g., the mismatch induced error). It can be seen that the error span decreases with higher $I_u$, which allows the analog macro to obtain better accuracy performance by increasing the current-mode of the system. Since the operating frequency is increased proportionally to the current, the analog macro is capable to maintain a similar energy efficiency across different current-modes, as we will show in section 3.4.

Finally, we show the output of the synaptic cell temporal stochastic variability in the form of noise-to-signal ratio (N/S) in Figure 15.a.4. Compared to the spatial stochasticity, the temporal one is almost negligible ($\leq$3%).

Now, to obtain the actual $I_{MAC_m}$ output we used the subtraction cell in Figure 13. We then obtain:

$$I_{MAC_m} = I_{MAC_{p_m}} - \alpha_{r_m} \cdot I_{MAC_{n_m}} \tag{16}$$

We managed to have a very low $\alpha_{r_m}$ deterministic error due to the use of a cascode WBCM mirror, as shown in figure 15.b.1 ($\leq$0.6%). As well, the spatial stochasticity ($\leq$12%) is lower than that of the synaptic cells since we used bigger transistors (16X), considering that is only a single subtraction cell per column. Recalling that the output of the subtraction cell goes directly to the input of the ReScaM module (Figure 13),

$I_{MAC_m}$ experiences the ReLU operation, hence:

$$I_{s_{in}} = \text{ReLU}(I_{MAC_m}) = \max(I_{leak_s}, I_{MAC_m}) \tag{17}$$

where $I_{leak_s}$ is the leakage current at the input od the ReScaM module (this value is only used for power consumption calculations).

Figure 15.b.2 shows the output of the subtraction cell (ReScaM module input) where the ReLU transfer function can be distinguished clearly. The figure presents a summary of 100 Monte Carlo simulations across 4 different current-modes, evidencing the effect of the $\alpha_{r_m}$ spatial stochastic variability. Notice that this effect does not manifest the fan-like shape present in the IA DAC and synaptic cells. The latter is because $I_{MAC_m}$ is the subtraction of two currents with multiplicative errors. Finally, Figure 15.b.3 shows the noise-to-signal ratio of the subtraction cell. As expected, the temporal stochastic variability is lower than the spatial one (10X lower).

The following module in the analog macro's signal path corresponds to the ReScaM module. When applying the scaling factor from eq. (12) with the circuit in Figure 13, the actual scaling factor of the ReScaM module becomes:

$$\alpha_{s_m} = \frac{I_{s_{out_m}}}{I_{s_{in_m}}} = \frac{1}{d+1} \cdot [(1+\epsilon_s) \cdot n + (1+\epsilon_{os})] \tag{18}$$

where $n$ and $d$ are the ideal numerator and denominator of the scaling factor, respectively; $\epsilon_s$ and $\epsilon_{os}$ are the ReScaM gain and offset errors, respectively.[xii]

Figure 15.c.1 shows the deterministic (left) and stochastic (right) variabilities for $\epsilon_s$ (top) and $\epsilon_{os}$ (bottom), for two denominators numbers ($d$): 7 (blue) and 15 (black). These errors include both static and dynamic errors, as well as errors due to the memory retention, as analyzed in section 3.2.2. The most dominant deterministic error corresponds to the settling behavior of the module (dynamic error).[xiii] This error is more noticeable when looking at the offset error in Figure 15.c.1, mainly at lower current-mode

[xii] In principle, $\epsilon_s$ and $\epsilon_{os}$ depend on $n$, $d$, and the input current level $I_{s_{in_m}}$. We decided to use the maximum scalar error values instead of a complex function.

[xiii] In other words, the ReScaM module (and the analog macro in general) can have lower errors at lower operating frequencies at the expense of energy efficiency.

and high temperature, since the leakage current at the output is close to the actual output current at the lowest scaling factor, and because the settling time is larger at lower currents.

Similar to previous modules and cells, Figure 13.c.2 shows the fan-like shape present in the scaling factor transfer function due to deterministic and spatial stochastic errors at different current and frequency modes. And finally, Figure 13.c.3 shows the noise-to-signal ratio, where it can be seen that the temporal stochasticity is $<2\%$.

### 3.3.1. Calculation of the Analog Macro's total Stochasticity

In this subsection we calculate the total stochastic variability caused by all the macro's modules non-idealities contributions. We use the result in the A-Connect methodology chapter (section 2.3) to properly model the stochasticity of the system.

Using equations (14)-(18), the output of the analog macro's signal path (ReScaM module's output) can be expressed as:

$$\begin{aligned} I_{s_{outm}} &= \sum_{i=1}^{N_{row}} (I_{s_{outp_{i,m}}} - I_{s_{outn_{i,m}}}) \\ I_{s_{outp_{i,m}}} &= w_{p_{i,m}} \cdot I_{DAC_{0i}} \cdot \alpha_{D_i} \cdot \alpha_{w_{p_{i,m}}} \cdot \alpha_{s_m} \\ I_{s_{outn_{i,m}}} &= |w_{n_{i,m}}| \cdot I_{DAC_{0i}} \cdot \alpha_{D_i} \cdot \alpha_{w_{n_{i,m}}} \cdot \alpha_{r_m} \cdot \alpha_{s_m} \end{aligned} \tag{19}$$

where the $\alpha_{D_i}$, $\alpha_{w_{i,m}}$, $\alpha_{r_m}$, $\alpha_{s_m}$ are the error sources of the analog macro's modules.

When using the A-Connect methodology proposed in chapter 2, the error matrices model the stochasticity of the layer's weights and biases. The latter is equivalent to refer the stochastic variability to $I_{s_{outp}}$ and $I_{s_{outn}}$ in eq. (19) as follows:

$$\begin{aligned} \sigma_{sp}^2 &= \left|\frac{\partial I_{s_{outp_{i,m}}}}{\partial \alpha_{D_i}}\right|^2 \cdot \sigma_{\alpha_D}^2 + \left|\frac{\partial I_{s_{outp_{i,m}}}}{\partial \alpha_{w_{i,m}}}\right|^2 \cdot \sigma_{\alpha_w}^2 + \left|\frac{\partial I_{s_{outp_{i,m}}}}{\partial \alpha_{s_m}}\right|^2 \cdot \sigma_{\alpha_s}^2 \\ \sigma_{sn}^2 &= \left|\frac{\partial I_{s_{outn_{i,m}}}}{\partial \alpha_{D_i}}\right|^2 \cdot \sigma_{\alpha_D}^2 + \left|\frac{\partial I_{s_{outn_{i,m}}}}{\partial \alpha_{w_{i,m}}}\right|^2 \cdot \sigma_{\alpha_w}^2 + \left|\frac{\partial I_{s_{outn_{i,m}}}}{\partial \alpha_{r_m}}\right|^2 \cdot \sigma_{\alpha_r}^2 + \left|\frac{\partial I_{s_{outn_{i,m}}}}{\partial \alpha_{s_m}}\right|^2 \cdot \sigma_{\alpha_s}^2 \end{aligned} \tag{20}$$

where $\sigma_{sp}$ and $\sigma_{sn}$ are the total stochastic variability of $I_{s_{outp}}$ and $I_{s_{outn}}$, respectively, and $\sigma_{\alpha_D}$, $\sigma_{\alpha_w}$, $\sigma_{\alpha_r}$, $\sigma_{\alpha_s}$ are the analog macro's modules stochasticity. Solving eq. (20), and

using the coefficient of variation ($c_v = \sigma/\mu$), we obtain:

$$\begin{aligned} c_{v_{sp}}^2 &= c_{v_{\alpha_D}}^2 + c_{v_{\alpha_w}}^2 + c_{v_{\alpha_s}}^2 \\ c_{v_{sn}}^2 &= c_{v_{\alpha_D}}^2 + c_{v_{\alpha_w}}^2 + c_{v_{\alpha_r}}^2 + c_{v_{\alpha_s}}^2 \end{aligned} \tag{21}$$

In terms of number of error factors, there is a mismatch between $c_{v_{sp}}$ and $c_{v_{sn}}$, caused by the presence of the mirror current ratio $\alpha_r$ of the subtraction cell. Using the $I_{s_{outn}}$ in eq. (19), as well as considering only the stochasticity of $\alpha_r$, and knowing that $I_{s_{outp}} = I_{s_{CM}} + I_{s_d}/2$ and $I_{s_{outn}} = I_{s_{CM}} - I_{s_d}/2$, we can obtain:

$$\left|\frac{\partial I_{s_{outn}}}{\partial \alpha_r}\right|^2 \cdot \sigma_{\alpha_r}^2 = I_{s_{outn}}^2 \cdot c_{v_{\alpha_r}}^2 \leq \frac{c_{v_{\alpha_r}}^2}{2} \cdot (I_{s_{outp}}^2 + I_{s_{outn}}^2) \tag{22}$$

With the inequality above, we can split the stochastic contribution of $\alpha_r$ between $I_{s_{outp}}$ and $I_{s_{outn}}$ in an equitable way. Then, eq. (21) becomes:

$$c_{v_{sp}}^2 = c_{v_{sn}}^2 = c_{v_{\alpha_D}}^2 + c_{v_{\alpha_w}}^2 + \frac{c_{v_{\alpha_r}}^2}{2} + c_{v_{\alpha_s}}^2 \tag{23}$$

With eq. (23), we can find the layer's coefficient of variation $c_{v_{\text{layer}}}$ used in the A-Connect methodology (see algorithm in section 2.3). Figure 16 shows the relative contributions of each of the error sources to the total macro's stochasticity (or layer's coefficient of variation). As expected, the dominant source of stochasticity comes from the spatial stochastic variability: synaptic cells (between 65% and 75% of the total amount), IA DAC cells (10%-20%), ReScaM module (6%-16%), and subtraction cell (1.5%-5%). On the other hand, the combination of the temporal stochasticity only contributes between 0.5% to 2% of the total macro's stochasticity. Finally, the effect of the current levels and operating frequencies (as well as temperatures) can be distinguished as well. The latter not only allows the analog macro to work in a multi-mode SoE but to achieve better accuracy performance at higher current-modes with the same energy efficiency.

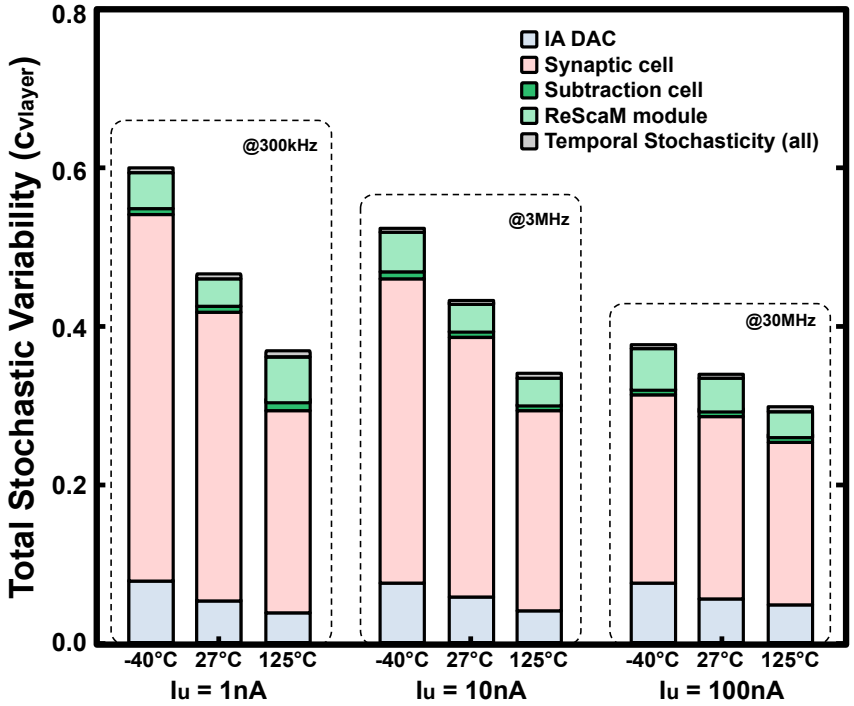


Figure 16. Relative stochasticity contribution from all macro's modules and cells at three different temperatures (-40°C, 27°C, 125°C), and three different current-modes and operating frequencies.

### 3.4. Results

In this section, we will present the simulation results of the ML analog macro accelerator, as well as a comparison with the state of the art.

Figure 17 shows the layout of the analog macro, using a TSMC 180nm CMOS technology node. The entire analog macro occupies an area of 1.09mm x 0.85mm, with the following distribution: 59% SRAM macro, 12% IA DAC cells, 12% OA SAR-ADCs, 10% ReScaM modules, 2% subtraction cells, and 5% biasing. The analog macro is capable of executing 16384 (128x128) MAC operations simultaneously.

To test the performance of the analog macro, we used the LeNet-5 architecture on the MNIST dataset (see Figure 18.a), trained with binary weights (see chapter 2). We only implemented the last FC layer on the analog macro since the simulation time was prohibitively high for larger layers. The last FC layer consists of a vector-matrix multiplication with 84 input activations and 10 output activations (classes). Only 100 images were passed through the network. We then obtained the input of the last FC layer from Tensorflow and quantized these values to use them for the simulations. We

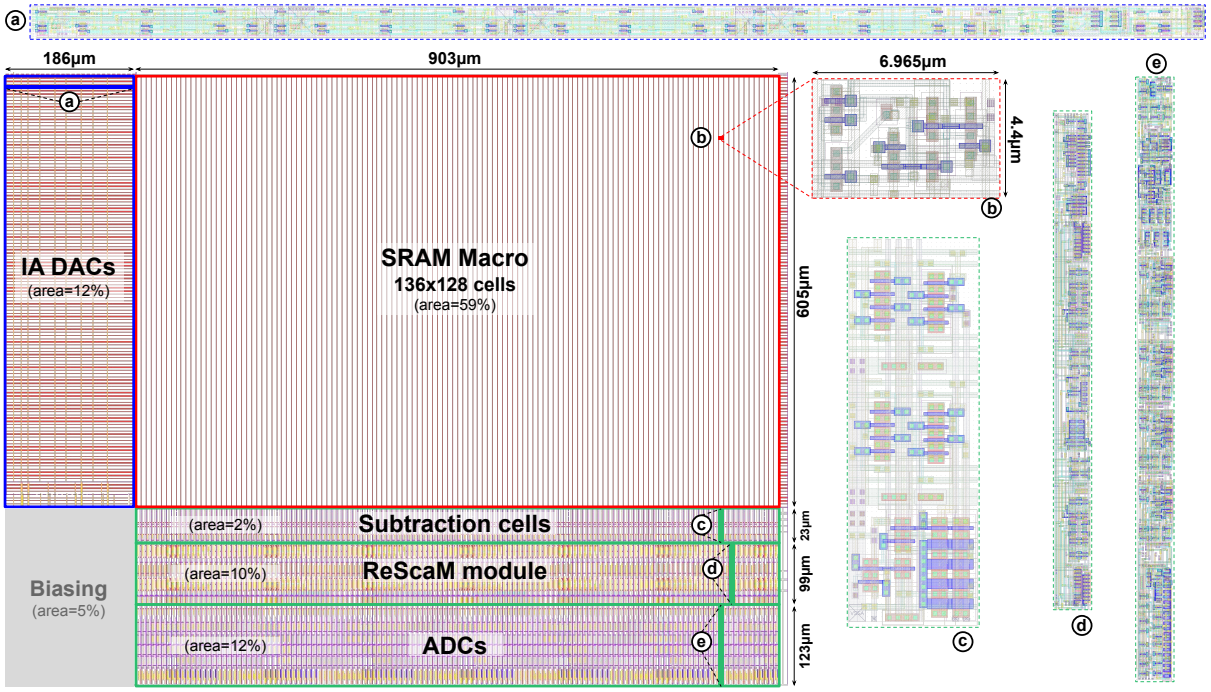


Figure 17. Analog Macro's layout: a) input activation DAC cell; b) synaptic 9T-SRAM cell; c) output column's subtraction cell; d) output activation ReScaM module; e) output column's SAR ADC.

trained the LeNet-5 network using the A-Connect methodology (section 2.3) for a 50% layer stochasticity (based on Figure 16) and binary weights (all layers were trained under these conditions). Finally, we measured the 10 output currents $I_{MAC_p}$ and $I_{MAC_n}$, subtracted their values ($I_{MAC} = I_{MAC_p} - I_{MAC_n}$), and applied the softmax classification layer to obtain the prediction of the neural network.

Figure 18.b (top) shows the 10 FC layer's output currents ($I_{MAC}$) at 27°C, 300kHz, and using 1nA unit current for the IA DACs. The figure shows at each time frame (clock cycle) the expected prediction value (depicted in a circle), and how the corresponding column/output current activates to the highest value among the ten currents (e.g., when the input image is a 6, the 6th column current fires-up). We also present in Figure 18.b (bottom) how the ReScaM module would work, although the last FC layer does not make use of it. It is possible to see the ReLU activation working (i.e., negative current values clamp to zero), as well as the memory function, where the saving and loading states are clearly distinguished. Specifically, in the loading state, the saved output current is retained to be used in the next cycle by another operation (see section 3.2).[xiv]

[xiv] The scaling factor in this simulation was set to one.

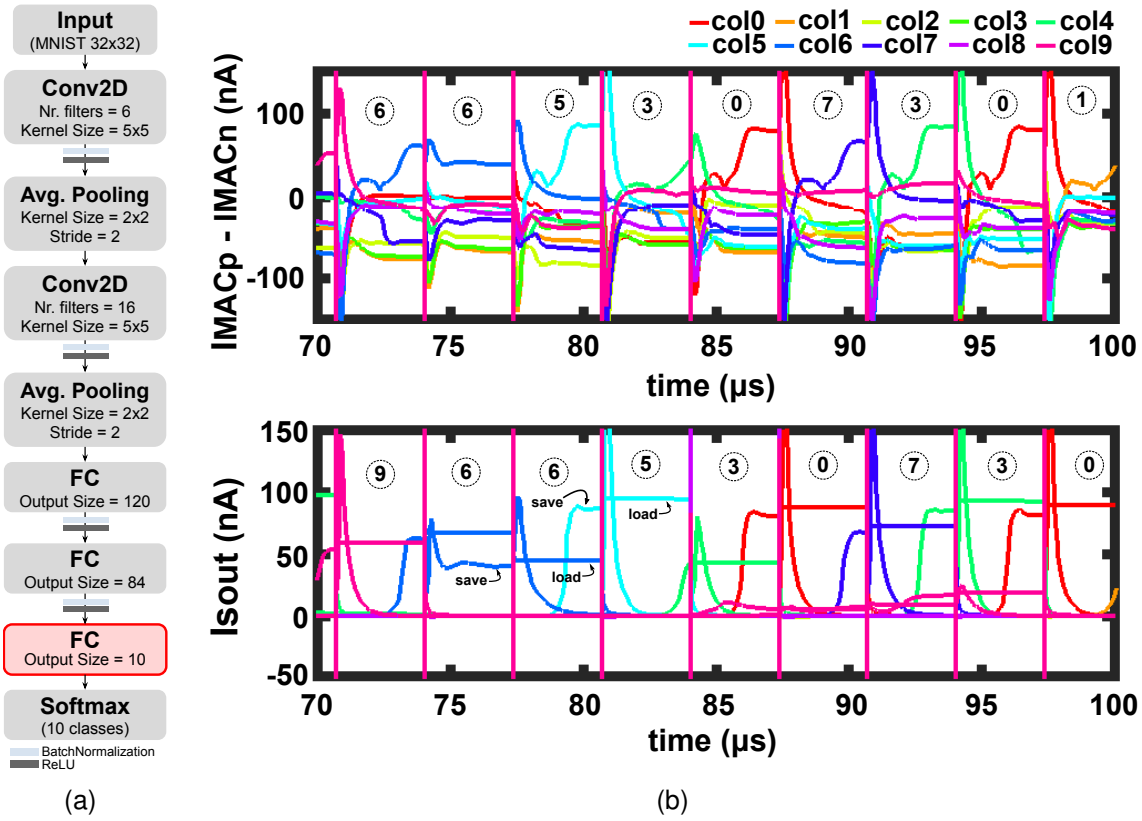


Figure 18. Simulation of a fully-connected layer implemented on the analog macro. a) We used the last FC layer of the LeNet-5 network architecture for the circuit simulation. b) Subtraction of the MAC currents (top) and ReScaM module operation (bottom) using 1nA unit current at 300kHz.

Figure 19 shows the simulation results in terms of energy efficiency (left Y-axes) and accuracy performance (right Y-axes in blue). We simulated across different supply voltages (0.9V-1.8V; X-axes), three different temperatures (-40°C, 27°C, 125°C), different IA DAC unit currents (1nA, 10nA, 100nA), and and across several operating frequencies (200kHz-30MHz), scaled according to the current level. The simulations in the figure are not Monte Carlo simulations, hence, no stochasticity is present. Under these circumstances, and considering that only 100 validation images were used for the simulations, any validation accuracy above 95% is considered a success.

Table 7 shows a comparison between state-of-art SRAM-based ML macros and the summarized macro's simulation results obtained in this work. We chose three different current-frequency levels from the successful results presented in Figure 19 (i.e., accuracies above 95% at 27°C). Because it is difficult to compare results across different

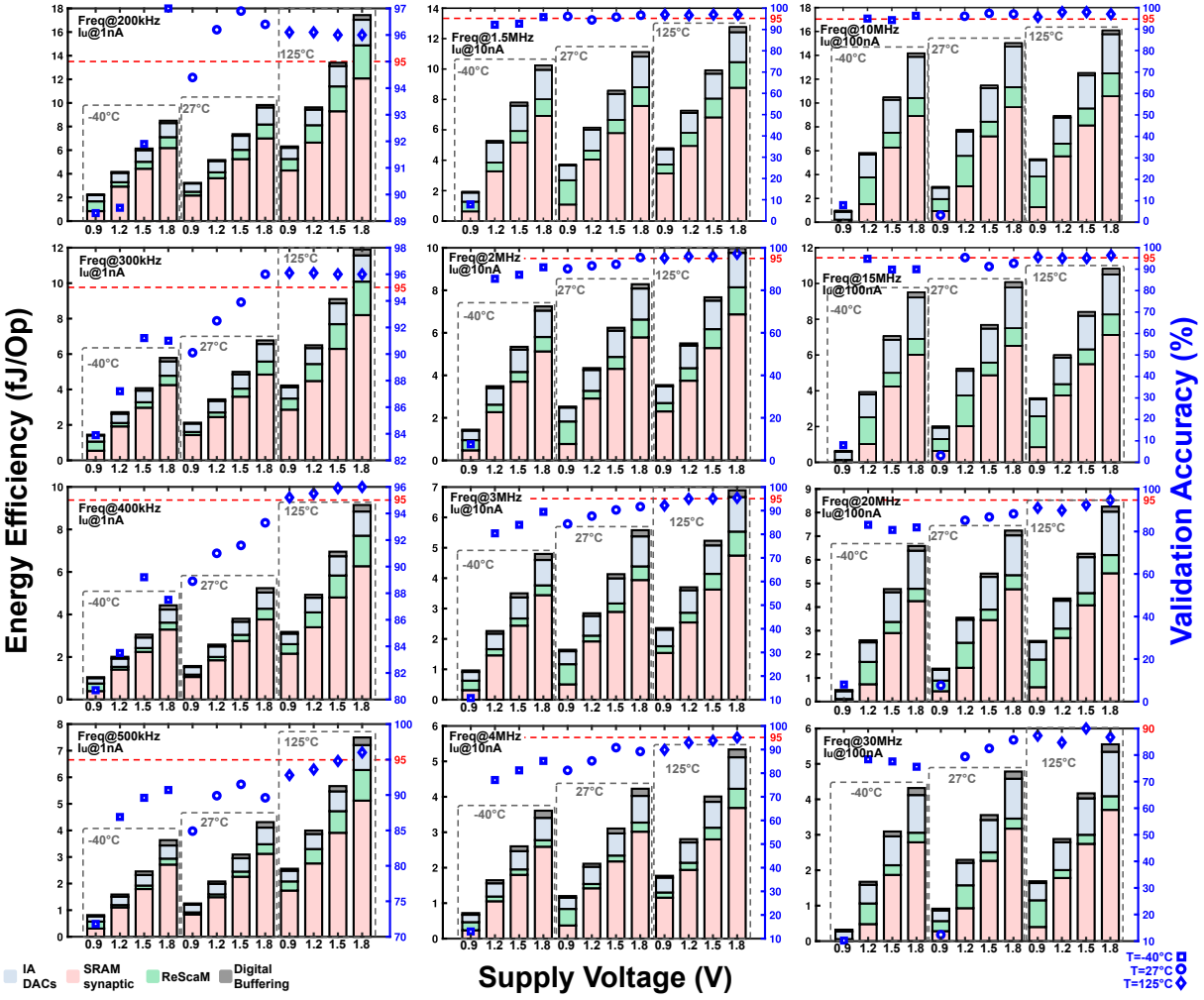


Figure 19. Simulated energy efficiency and validation accuracy for the LeNet-5 last FC layer, on the MNIST dataset. The simulations were performed using different supply voltages (0.9V-1.8V), different IA DAC unit currents (1nA, 10nA, 100nA), at three temperatures (-40°C, 27°C, 125°C), and across several operating frequencies (200kHz-30MHz).

technologies, we used the benchmark proposed in[101], where different works were compared using energy efficiency, area efficiency, throughput, and the information content before the ADC, with the first three metrics scaled to the bit-precision used to represent the input activations and weights (e.g., 1b-TOPS/W = TOPS/W∗IN-precision∗W-precision).

[101] Naresh R. SHANBHAG et al. “Comprehending In-memory Computing Trends via Proper Benchmarking”. In: *2022 IEEE Custom Integrated Circuits Conference (CICC)*. 2022, pp. 01–07.

[102] I. A. PAPISTAS et al. “A 22 nm, 1540 TOP/s/W, 12.1 TOP/s/mm2 in-Memory Analog Matrix-Vector-Multiplier for DNN Acceleration”. In: *2021 IEEE Custom Integrated Circuits Conference (CICC)*. 2021, pp. 1–2.

[103] Yu-Der CHIH et al. “16.4 An 89TOPS/W and 16.3TOPS/mm2 All-Digital SRAM-Based Full-Precision Compute-In Memory Macro in 22nm for Machine-Learning Edge Applications”. In: *2021 IEEE Inter-*

Table 7. Performance comparison between CIM SRAM-based ML macros.

| | Valavi19[90] | Seok20[91] | Papistas21[102] | Chih21[103] | Lee21[104] | This Work | | |
|---|---|---|---|---|---|---|---|---|
| **Technology** | 65nm | 65nm | 22nm | 22nm | 28nm | **180nm** | | |
| **Cell Type** | 8T1C | 8T1C | 18T | 6T+D | 10T1C | **9T** | | |
| **Area (mm$^2$)** | 12.6 | 0.081 | 1.95 | 0.202 | 0.51 | **1.09mm x 0.85mm** (0.9mm x 0.6mm)[a] | | |
| **Mem. Capacity** | 295kB | 2kB | 256kB | 8kB | 36kB | **2kB** | | |
| **Input Precision ($B_{IN}$)** | 1b | 1b | 7b | 4b | 5b | **4b** | | |
| **Weight Precision ($B_W$)** | 1b | 1b | Ternary | 4b | 1b | **1b** | | |
| **Output Precision ($B_{OUT}$)** | 1b | 1b | 6b | 16b | 8b | **analog / 1b-4b** | | |
| **Op. Frequency** | 100MHz | 50MHz | 22.5MHz | 100MHz | 4.17MHz | **200kHz** | **1.5MHz** | **15MHz** |
| **Supply Voltage** | 0.94V | 1V | 0.6V | 0.72V | 0.9V | **1.2V** | **0.9V** | **1.2V** |
| **1b-TOPS/W**[b] | 866 | 671.5 | 20747.2 | 1424 | 5796 | **800**[c] | **1076**[c] | **760**[c] |
| **1b-TOPS** | 18.876 | 1.638 | 64.35 | 52.8 | 6.14 | **0.013** | **0.098** | **0.98** |
| **1b-TOPS/mm$^2$** | 1.498 | 20.2 | 33.28 | 261.4 | 12 | **0.014** | **0.105** | **1.1** |

[a] SRAM macro only (128x128 memory cells).
[b] 1b energy efficiency (1b-TOPS/W $=$ TOPS/W$*B_{IN} * B_W$).
[c] With analog output precision.

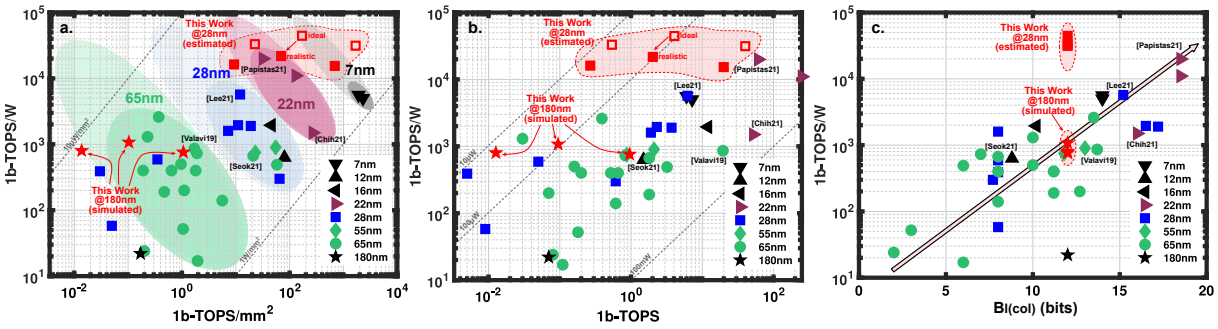


Figure 20. Comparison with other CIM SRAM-based macros (adapted from[101] ): a) 1b-TOPS/W vs. 1b-TOPS/mm$^2$ categorized by technology node; b) 1b-TOPS/W vs. 1b-TOPS; c) 1b-TOPS/W vs. $B_{l(col)}$.

We also included our results in three figures adapted from[101] (see Figure 20), where we can visually compare our work with several macros published in relevant conferences (e.g., ISSCC, VLSI, CICC) in the last two years (the works on Table 7 were included as well). The figures show the behavior of the macros energy efficiency versus the area efficiency (Figure 20.a), versus the macros' throughput (Figure 20.b), and versus the information content before the ADC (Figure 20.c).[xv] Although we used 180nm in our work, the results are compatible with state-of-art macros in 65nm (green cluster),

*national Solid- State Circuits Conference (ISSCC)*. vol. 64. 2021, pp. 252–254.

[104] Jinseok LEE et al. "Fully Row/Column-Parallel In-memory Computing SRAM Macro employing Capacitor-based Mixed-signal Computation with 5-b Inputs". In: *2021 Symposium on VLSI Circuits*. 2021, pp. 1–2.

[xv] According to[101], the information content before the ADC is $B_l(col) = B_{IN} + B_W + log_2(N)$, where $N$ is the number of dot-products per column.

maintaining a relatively similar energy efficiency (760-1076 1b-TOPS/W) across a wide band of operating frequency (200kHz-15MHz), which allows the macro to have a broad range of area efficiency (0.014-1.1 1b-TOPS/mm$^2$) as well as throughput (0.013-0.98 1b-TOPS). In particular, our work stands out from others when looking the clusters in Figure 20.a. The clusters show that increasing either the energy or area efficiency in the same node, would imply the decrement of the other metric (i.e., the energy efficiency is almost inversely proportional with respect to the area efficiency within the same technology node, as shown by the negative slope in the oval clusters) while our proposal maintain the energy efficiency even while increasing the area efficiency.
Furthermore, we also estimated our macro's behavior in a 28nm technology by scaling our results. We used a scaling factor $\lambda^2$ (i.e., $\lambda = 180/28$) for the energy efficiency and the throughput, strategy used in[105]. The area in 28nm is estimated by multiplying the area in 180nm by $1/\lambda^2$, which gives a scaling factor for the area efficiency equal to $\lambda^4$. The ideal results using the scaling factors are presented in Figure 20 as red-empty squares. Because these values are rough estimations, we calculated a more realistic estimation by decreasing 50% the ideal energy-efficiency and throughput values, and by $\sim$70% the area efficiency. Even with the more realistic values, the estimated accelerator performance shows results compatible with 22nm and 7nm technology nodes, which are absolute state-of-art performance, which motivates further research in the proposed direction.

### 3.5. Conclusion

In this chapter, we have proposed a wide frequency range and high energy efficiency CIM SRAM-based ML macro for multi-mode systems-on-edge. The analog macro was able to perform at high energy efficiency by following two principles: avoiding data conversion by staying in the same physical domain (i.e., current), and the use of simplified and low-area circuits by using co-design software strategies that mitigate stochastic and deterministic errors (i.e., the A-Connect methodology presented in chapter 2). We

[105] Jinshan YUE et al. "STICKER-IM: A 65 nm Computing-in-Memory NN Processor Using Block-Wise Sparsity Optimization and Inter/Intra-Macro Data Reuse". In: *IEEE Journal of Solid-State Circuits* 57.8 (2022), pp. 2560–2573.

proposed an end-to-end analog datapath that incorporates not only MAC operations but commonly used ML operations within the analog domain, such as ReLU and scaling (the latter enabled normalization operations), as well as memory capabilities for pipeline execution. The simulation results presented in a 180nm design showed that the analog macro performed at a wide range of frequencies (200kHz-15MHz) over ultra-low and broad range of current levels (i.e., 1nA to 100nA biasing), while maintaining a relatively similar energy efficiency (760-1076 1b-TOPS/W). When compared to other works, the proposed macro's results were compatible with state-of-art macros in 65nm. Furthermore, we showed performance estimations for a 28nm design that put the proposed analog macro above absolute state-of-art performance.

# 4. MULTI-LEVEL VOLTAGE MONITORS TO ENABLE MULTI-MODE FINE-GRAINED POWER MANAGEMENT STRATEGIES IN SYSTEMS-ON-EDGE

## 4.1. Introduction

Multi-mode fine-grained power management system-on-chip (SoC) offer a promising approach for achieving ultra-low power consumption in energy autonomous and battery supplied applications at the edge of Internet-of-Things (IoT). The SoC Power Management Unit (PMU) implements these techniques to obtain a more precise control over power consumption at the subsystem and component level, enabling systems to operate more efficiently.

Fine-grained power management enables the reduction of power consumption by enabling individual components within an SoC to enter low-power modes or be turned off altogether when not in use. This can be accomplished through techniques such as clock gating, power gating, and dynamic voltage and frequency scaling (DVFS)[106]. These techniques allow for the voltage and frequency of individual components to be adjusted based on the current demand for performance. When these techniques are used in subsystems within the SoC, we can refer to a multi-mode fine-grained PM system. In a multi-mode system, different components can be optimized for different power modes. For example, a system may have a high-performance mode for demanding applications (active mode), a low-power mode for battery-sensitive applications (sleep mode), and an ultra-low power mode for idle states (deep-sleep mode). The system can dynamically switch between these modes to meet the current power and performance requirements, reducing power consumption when it is not needed and increasing performance when it is required.

One example of multi-mode fine-grained PM system is shown in Figure 21, considering PMU should be always on in order to administrate power gating of all other domain. Within the system's power management, voltage monitoring is crucial for the proper

[106] Vivek DE. "Fine-grain power management in manycore processor and System-on-Chip (SoC) designs". In: *2015 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. 2015, pp. 159–164.

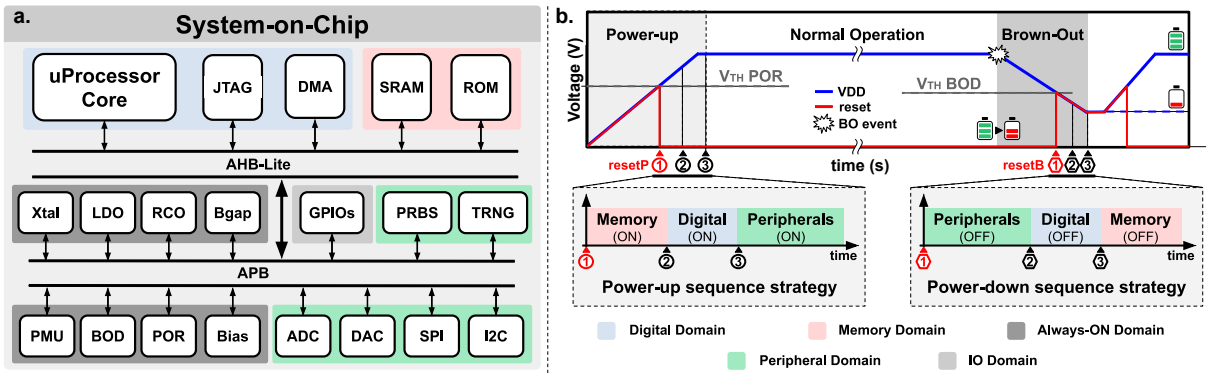


Figure 21. a) Microarchitecture details of the SoC with clearly identified voltage domains. b) Timing sequence of conventional power management strategy at power-up and brown-out event.

behavior of the entire SoC. For example, during the ramp-up of a local supply, memory and registers might have floating states causing undesired initial behavior[107]. To avoid this situation, a common solution is to hold the circuit into a reset state until the local regulator reaches a level that guarantees correct operation. This task is performed by a power-on reset (POR) circuit. During a brown-out event, the PMU can command the memory, digital core, accelerators and peripherals into idle and sleep states to save energy. Power supply is restored by re-enabling individual voltage regulators. The supply voltages are usually constrained to the lowest operational level of the domain without impacting the robustness of its blocks. Upon request, the blocks can be turned-on, individually, through power gating.

Traditional voltage monitor (VM) circuits (i.e., POR and BOD) are designed with a fixed threshold level detection, forcing to design and place different VMs for each domain[107,108,109]. Implementation of a single set of VMs for each voltage domain increases design area, cost (considering its re-design to adjust the threshold level), and power consumption. In this chapter we propose multi-level voltage monitor circuits that can be repurposed for sub-systems that use different voltage domains within the SoC. Our voltage monitors allow more granular control over the behavior of the SoC in the event

[107] H. B. LE et al. "A Long Reset-Time Power-On Reset Circuit With Brown-Out Detection Capability". In: *IEEE Transactions on Circuits and Systems II: Express Briefs* 58.11 (Nov. 2011), pp. 778–782.

[108] S. K. WADHWA et al. "Zero Steady State Current Power-on-reset Circuit with Brown-out Detector". In: *19th International Conference on VLSI Design held jointly with 5th International Conference on Embedded Systems Design (VLSID'06)*. Jan. 2006.

[109] David M. GONZALES. *Low Voltage CMOS Power-on Reset Circuit*. US Patent 9143137. 2015.

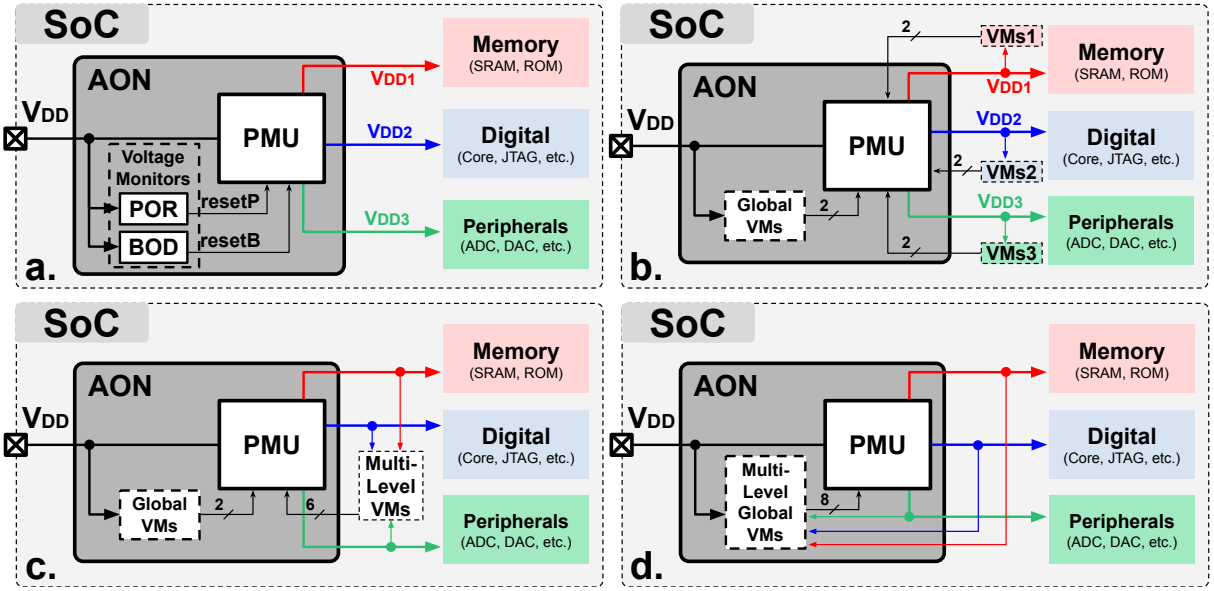


Figure 22. Power management (PM) strategies based on supply voltage monitors (VMs): a. Conventional approach (Global VMs); b. Global and multiple local VMs fine-grained PM; c. Global and single local VMs fine-grained PM; d. Single Global VMs fine-grained PM.

of power supply voltage fluctuations while maintaining ultra low-power consumption. Furthermore, we present design considerations to enable the proper function of such ultra low-power voltage monitors on wider temperature ranges than previous works.

This chapter is divided as follows: section 4.2 shows an overview on the proposed fine-grained power management strategies that could help alleviate the limitations imposed on conventional approaches. Section 4.3 presents the proposed multi-level voltage monitors architectures that enable the proposed fine-grained PM strategies. Section 4.4 shows the results of simulations and measurements of fabricated voltage monitors in a 180nm CMOS technology. It also shows SoC level measurements when using a PM strategy implemented with the proposed VMs. Finally, section 4.5 presents a summary of this chapter.

## 4.2. Fine-Grained Power Management Strategies

In order to circumvent the limitations of the conventional voltage monitoring power management strategy in Figure 22a, fine-grained power management approaches are proposed in this work, all presented in Figure 22b-d. The first of these strategies adopts

a locally-distributed PMU-VMs strategy as Figure 22b shows. The aim is to supply, in an efficient way, every voltage domain sub-systems. Considering the turning-on event, once the global POR ensures a safe start-up of the chip, the PMU provides activation signals for local regulators. The local PORs will then guarantee risk-free turn-on conditions in every domain independently. The advantage of such scheme is the possibility to turn-on multiple domains simultaneously. As well, it is possible to track down brown-out events within each domain separately with local BODs. The latter allows to take individual actions on the domain where the event occurred, without interfering with the operation of unaffected domains.

The obvious disadvantages of the global an multiple local VMs are: larger design area, current consumption, and design cost. In[110], we proposed a modular to tackle these problems. The proposed POR consists of a voltage reference, a voltage detector, and a pulse generator. Instead of placing the entire POR cell to check the target supply, only the voltage detector can be used. At the same time, all the local PORs can reuse the voltage reference block. As a result, low power, reduced area and design costs are possible.

Another possible option for power, area, and cost optimization is to use a single set of local VMs, as shown in Figure 22c. In that case, a mux is required to select the target supply voltage. Requirements in a PMU with one set of VMs imply multiple operation levels. Considering the PMU is in the always-on domain, every time a sub-system is turned-on, the global POR is set to the appropriate voltage threshold. Then, this POR checks the start-up of every domain at a time, in a sequential manner. The disadvantage of such a scheme lies in the inability of power-up different domains at the same time.

Further applications for a multi-level POR can be found if a harvesting system is considered. Zero battery charge implies harvesting system is working to supply a device. In that condition, no additional power consumption is desired, making useful to power off even the PMU. Generally, a near threshold voltage logic is implemented for low

[110] Luis E. RUEDA G. et al. “An Ultra-Low Power Multi-Level Power-on Reset for Fine-Grained Power Management Strategies”. In: *2019 IEEE 10th Latin American Symposium on Circuits Systems (LASCAS)*. 2019, pp. 185–188.

power operations, logic that can be turned on using the lowest voltage level of multi-level POR. Low voltage logic can configure a new POR trigger level, preparing it for switching on PMU when harvesting system achieve a higher supply level.
The main idea of this work is to introduce voltage monitors that can be used in either of the PMU strategies discussed. The following section, describes the proposed architectures.

## 4.3. Voltage Monitor Circuits

This section presents multi-level power-on-resets (POR) and brown-out detector (BOD) architectures that enable different PM strategies as the ones presented in section 4.2. The general architecture of our voltage monitors consists on four functional blocks: an adjustable sub-threshold voltage reference, a voltage detector or comparator, a supply tracker (e.g., voltage divider), and a pulse generator with a counter. In this section we present the sub-threshold voltage reference, followed by the two PORs and the BOD circuits. We present corner simulations for all of the mentioned circuits. The final part of this section shows a discussion regarding the trade-off between power consumption and the achievable temperature range of our circuits.

### 4.3.1. Sub-threshold voltage reference

The voltage reference is based on the low-power sub-threshold reference source proposed in[111]. The circuit consists on a zero-$V_{TH}$ native NMOS device, followed by a group of four stacked PMOS transistors connected as diodes (see Figure 23.a). By connecting the gate of this transistor to a voltage lower than its source voltage, sub-threshold condition is forced in the branch. Since both PMOS and NMOS have the same $V_{gs}$ voltage, and share the same current, due to their complementary nature, it is possible to eliminate the temperature dependency of the reference voltage with correct transistor sizing[111]. Higher $V_{REF}$ may be obtained by stacking more PMOS diode-connected transistors, but it is limited in this design to four due to the leakage

[111] I. LEE et al. “A Subthreshold Voltage Reference With Scalable Output Voltage for Low-Power IoT Systems”. In: *IEEE Journal of Solid-State Circuits* 52.5 (May 2017), pp. 1443–1449.

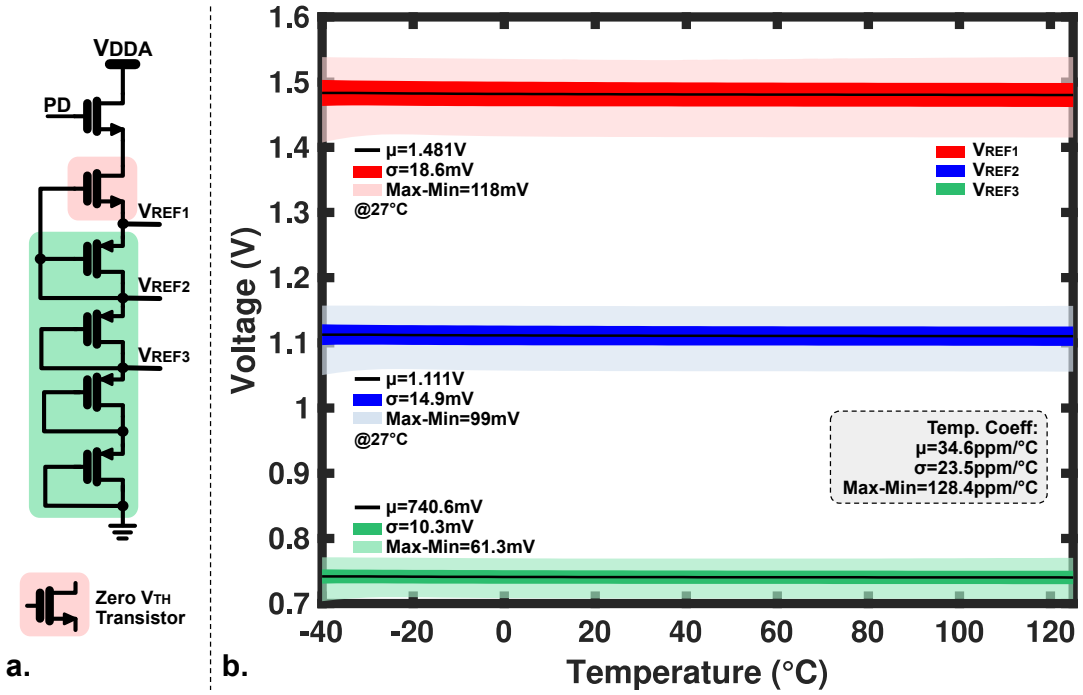


Figure 23. a) Sub-threshold Voltage Reference. b) Voltage reference with 1000 Montecarlo simulation results with process and mismatch variations.

effect of nwell-psub reversed diode.[i]

When process variations result in a slow-slow corner, the leakage current of PMOS transistors may be comparable to the branch's sub-threshold current at low-temperature, degrading the performance. We compensate the temperature coefficient by increasing the current of the PMOS stack, or equivalently, switching-on more native transistors through the TRIM signal in Figure 24. It is important to highlight that extra compensation becomes harder when more PMOS transistors are stacked together because static current drops dramatically.

### 4.3.2. Power-on-Reset

In this subsection, an extended discussion of the power-on-reset circuits introduced in[112,110] (POR1 and POR2, respectively) will be presented with their respective characteristics and differences.

[112] A. AMAYA et al. "A Multi-Level Power-on Reset for Fine-Grained Power Management". In: *28th IEEE PATMOS*. July 2018, pp. 129–132.

[i] This effect is worst at higher temperatures, and with bigger transistors.

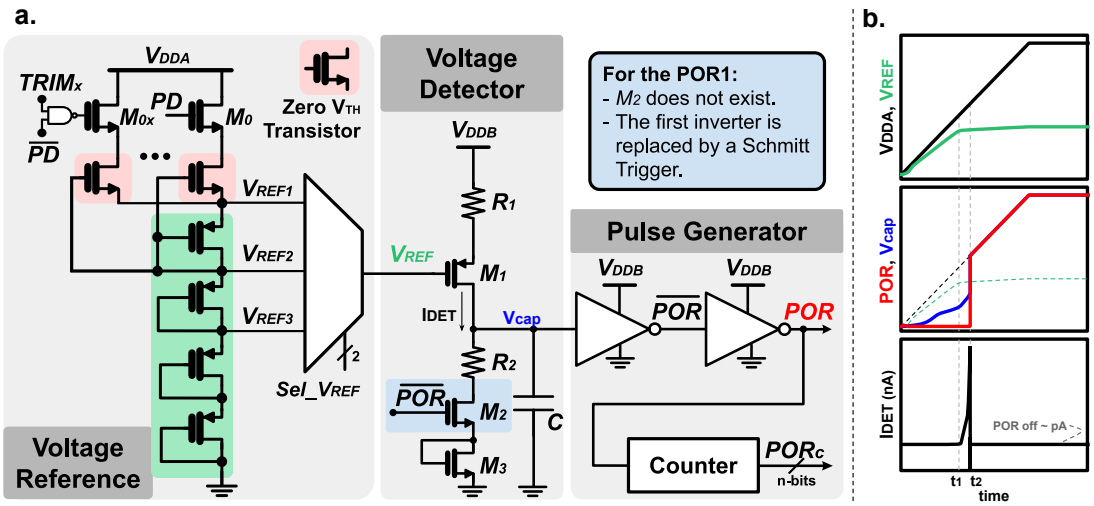


Figure 24. a) Proposed POR circuit composed of a subthreshold voltage reference, a voltage detector and a pulse generator. b) Important circuit signals during a global supply ramp-up (turn-on condition).

**POR1**[112]**:** The proposed POR is a modular circuit composed by three functional blocks as shown in Figure 24. The first block is an adjustable voltage reference, which allows the generation of multiple POR thresholds. The second block, the voltage detector, monitors the supply voltage $V_{DDB}$ and generates a delayed signal that goes to the third block. Finally, the pulse generator sets a logic one, once the capacitor voltage crosses a certain threshold voltage. As an option, the $POR$ signal can drive a counter. The latter provides the possibility of controlling the reset delay time. This feature is useful when circuits, such as oscillators, need a longer time to settle.

At the beginning of operation, when the supply voltage $V_{DDB}$ starts to ramp-up, transistor $M_1$ remains off and the capacitor is discharged. When $V_{DDB} = V_{REF}$, $M_1$ is in weak inversion, and the low current starts to charge capacitor $C$. Once $M_1$ enters strong inversion ($V_{DDB} = V_{REF} + |V_{thp}|$) the detector forces a greater current to charge the capacitor. This charge time depends on the slope of the $V_{DDB}$ ramp, limited by the difference between the current supplied by $M_1$ and the current sinked through the branch formed by $R_2$ and the diode connected transistor $M_3$. In other words, for fast ramps, the time constant of the voltage detector limits the response. Once the voltage at the capacitor reaches the pulse generator input threshold (which for this POR is the high threshold voltage of the schmitt trigger), the $POR$ signal is triggered. Finally, the POR's output may be used to enable a counter. This configuration offers the ability to have

different delayed reset signals (the signal $POR_c$ may have different bits), according to a particular application requirements.

**POR2**[110]**:** The main difference of this POR with respect to POR1 lies in the voltage detector (see Figure 24). When the threshold of the pulse generator is reached, the signal $\overline{POR}$ turns off transistor $M_2$, forcing $C$ to charge completely up to $V_{DDB}$. Turning-off $M_2$ means reducing the quiescent current of the block to the order of nano-amperes. The other noticeable change is the input block of the pulse generator, which in POR1 is a schmitt trigger. The purpose of the schmitt trigger was establishing a different threshold for a brown-out (BO) event. This is not longer needed, since the voltage detector performs this function as well. In the presence of a BO event, the capacitor voltage tries to follow $V_{DDB}$.[ii] It is not until $V_{DDB}$ is close to $V_{REF}$ (which is below the POR threshold) that the voltage at the capacitor starts to differ from $V_{DDB}$, since $M_1$ turns completely off. At this point the voltage capacitor is set to the voltage divider formed by two high impedance branches ($M_1$-$R_1$, and $M_2$-$M_3$-$R_2$). As the voltage capacitor gets lower than $V_{DDB}$, $\overline{POR}$ starts to rise-up, turning $M_2$ on until the capacitor is completely discharged, setting $\overline{POR}$ as a logic one.

Figures 25.a-b show the POR2 threshold voltages for different supply ramps (i.e., $1\mu$s to 1s rising times from 0V to 3.3V), and across process (i.e., slow, typical, and fast) and temperature variations (i.e., -40°C, 27°C, 125°C). In these two plots, three different ranges are distinguishable within each color region,[iii] which correspond to the three different $V_{REF}$ levels of the voltage reference block in Figure 24.

Figure 25.a shows the results for a global POR case, or chip turn-on condition. In this case, the supplies of the reference voltage ($V_{DDA}$) and the rest of the circuitry ($V_{DDB}$) are the same. The first thing to notice is that, for fast ramps (i.e, lower than 1ms), the threshold voltages for the three ranges are 3.3V, or close to that value. The reader might be tempted to conclude that, once the supply reach this value the reset is triggered. In fact, the voltage detector starts to charge the capacitor at the same level as for slow cases ($V_{DDA} = V_{REF} + |V_{thp}|$), but the response is limited by the time

[ii] This condition is restricted by the time constant formed by $R_1$ and $C$ (which filters short duration BO event).

[iii] All corner cases are within these regions.

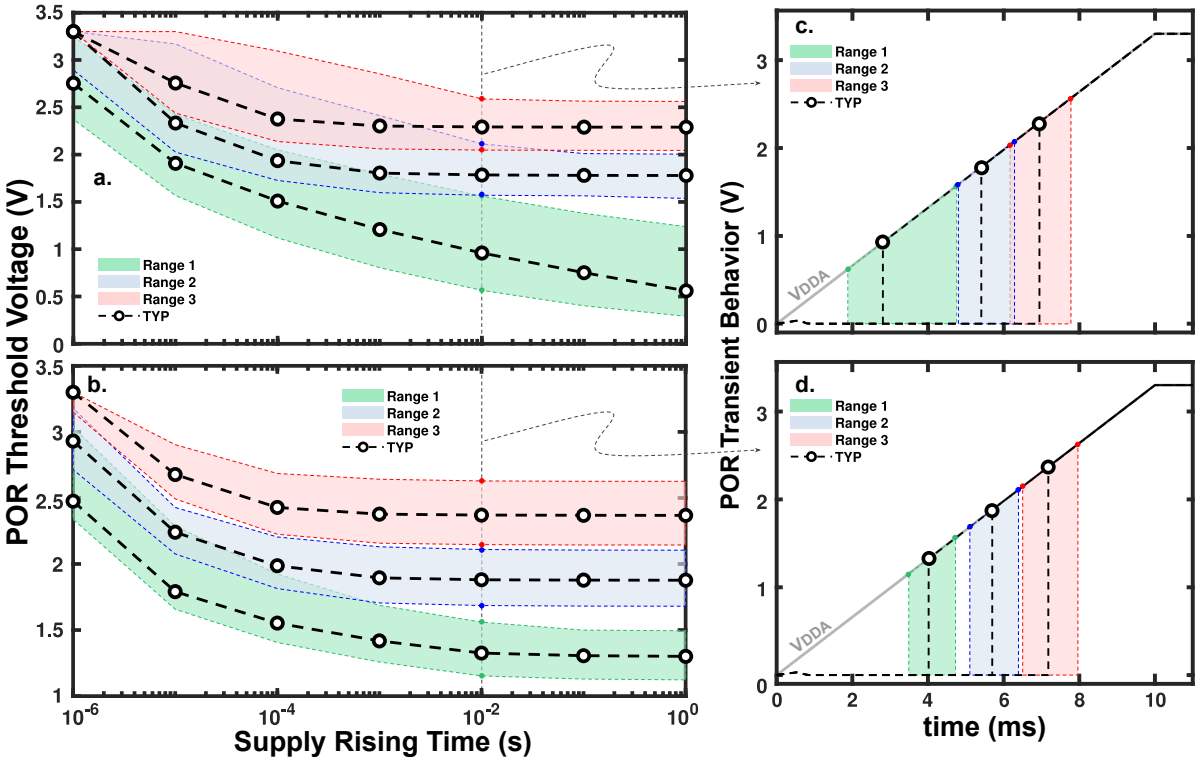


Figure 25. POR voltage thresholds ranges for different supply ramps (i.e., rising times) over process and temperature variations (final supply voltage of 3.3V): a) global POR; b) local POR. POR transient behavior for a rising time of 10ms considering a global POR (c) and a local POR (d).

constant of the voltage detector. The POR will require extra time before the capacitor reaches a value to trigger a reset signal. The second thing to observe appears for ramps slower than 100us; for ranges 2 and 3, the maximum and minimum stabilize to constant values. Meanwhile, for range 1, the lower limit diminish with slower ramps. The reason is because $V_{REF}$ is also in a turn-on condition and its value has not settled at the trigger moment. This is even more critical for the lowest level $V_{REF1}$, as $V_{sg1}$ (in Figure 24) will be sufficient to charge the capacitor at lower values of the supply voltage. This design was aimed for a top level supply voltage of 3.3V, so range 2 and 3 are enough. Even more, they behave as expected with slow ramps, which is in fact, the usual condition for the global POR.

Figure 25.b shows the results for the local POR, or local regulator power-up. In this case, the supply of the reference voltage is the global one ($V_{DDA}$ already settled), and the local supply ($V_{DDB}$) are different. The behavior is very similar to global POR case. The difference are in the limits values, which are much better behaved for all three

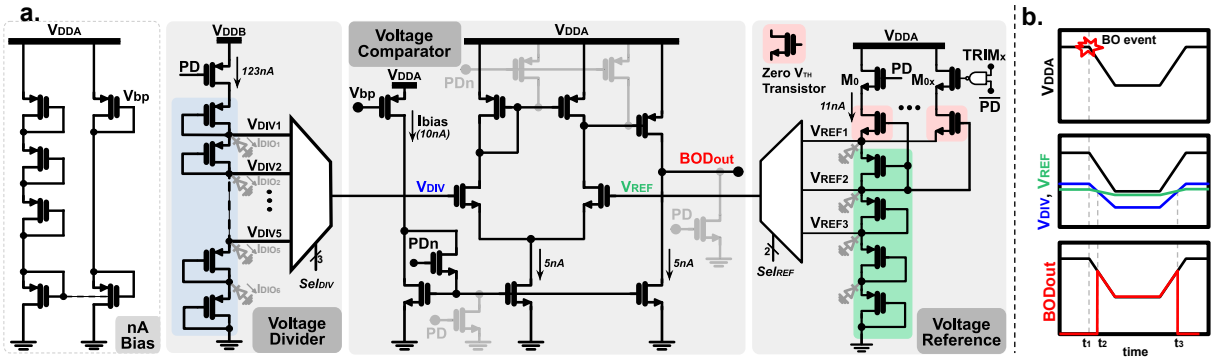


Figure 26. a) Proposed BOD block diagram composed of a voltage divider, a comparator and a subthreshold voltage reference. b) Important circuit signals during a BO event.

ranges. This was expected, since this case uses a steady reference, in contrast with the global case.

Finally, Figures 25.c-d show the $POR$ signal behavior over process and temperature variations, with a 10ms supply rising time for both POR cases: global POR ($V_{DDA} = V_{DDB}$, SoC turn-on condition) and local POR ($V_{DDB} \neq V_{DDA}$), respectively.

### 4.3.3. Brown-Out Detector Circuit[113]

Similar to the POR circuits in previous sections, the proposed BOD architecture is a modular circuit composed of a voltage divider, a voltage reference, and a comparator, as shown in Figure 26.a. The voltage divider has seven stacked 3.3V PMOS transistors connected as diodes, serving as a resistor divider to obtain five different threshold voltage levels. Additionally, a PMOS transistor controlled by the PD signal is used for power-down control. The voltage reference used in the BOD, is the same as the one used in the POR. Finally, the comparator is a two stage OTA with a bias current of 10nA, provided by a nano ampere high temperature compensated current source[114].

When a BO event occurs (see Figure 26.b), the supply voltage and voltage divider start to ramp-down. Although the reference voltage remains constant for most cases, for very pronounced supply decay it starts to ramp-down as well but a slower pace than the

[113] Luis E. RUEDA G. et al. “A Compact Industrial-Grade Multi-Threshold Brown-Out Detector”. In: *2019 26th IEEE International Conference on Electronics, Circuits and Systems (ICECS)*. 2019, pp. 923–926.

[114] J. SANTAMARIA et al. “A Family of Compact Trim-Free CMOS Nano-Ampere Current References”. In: *2019 IEEE International Symposium on Circuits and Systems (ISCAS)*. May 2019, pp. 1–4.

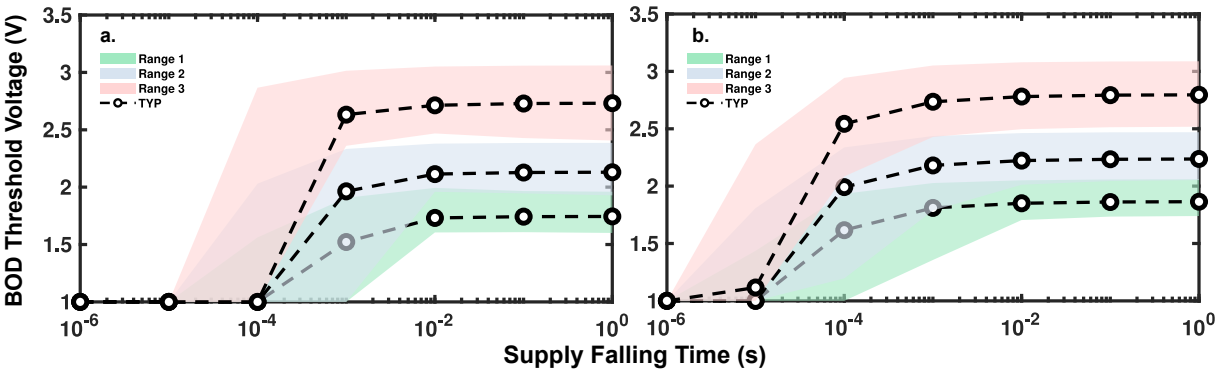


Figure 27. BOD voltage thresholds ranges for different supply ramps over process and temperature variations (from a supply voltage of 3.3V to 1V): a) global case; b) local case.

divider, allowing a future intersection. When the divider voltage is below the reference threshold the comparator detects BO event and sends the reset or warning signal to the PMU to take the corresponding action. When the action is taken, the supply voltage ramps-up again to its nominal value, and the BOD signal is de-activated. The BOD has two multiplexer to select the different reference and divider levels, which translates to different BOD levels. The multiplexers can be used also as trimming devices to mitigate the process and temperature effects.

Figures 27.a-b show the BOD threshold voltages for different supply ramps ($1\mu$s to 1s falling times from 3.3V to 1V) and across process and temperature variations. Several BOD threshold can be obtained depending on the divider-reference output pair chosen ($V_{REF}$-$V_{DIV}$). In this work we will only present three ranges for clarity.[iv] Figure 27.a shows the results for a global BOD (i.e., a BOD that monitors the AON supply), while Figure 27.b shows the results for a local BOD (i.e., a BOD that monitors a domain supply). Although the results for both cases are very similar, the biggest difference is that the local BOD responds faster than the global one for shorter supply falling times ($\sim$10X faster falling times). Because $V_{REF}$ does not ramp-down in the local BOD (only $V_{DIV}$ follows the supply), the BOD's time response is mainly determined by the dynamic characteristics of the OTA (i.e., sensitivity and slew-rate). On the other hand, both $V_{DIV}$ and $V_{REF}$ might be ramping down in the global BOD, delaying the moment at which they cross each other and the comparator detects a BO event. The latter effect

[iv] In our experiments we managed to obtain up to 5 different ranges.

is even more critical at lower temperatures as we will show in the following section.

### 4.3.4. Trade-off Between Power Consumption and Temperature Range

Although the voltage monitors are one of the most important blocks in the power management unit, detecting when the supply voltage goes below or above certain voltage ranges is an auxiliary function but one that is always on. Hence, the VMs should be kept small and "quiet": they should occupy as low area and consume as low power as possible.

One of the major challenges in the design of the VMs proposed in this work is the trade-off between the power consumption and the correct function of the circuits across a wide temperature range (e.g., military range -40°C to 125°C), specially at low temperatures ($\leq$0°C). In particular, the reference and divider blocks could have more diode connected PMOS to reduce their current consumption by increasing the equivalent impedance seen by the supply. The number of stacked PMOS transistors is limited by the leakage effect of the parasitic nwell-psub reversed diodes (see Figure 26) as shown in[111]. With a reversed diode voltage larger than the thermal voltage ($V_D >> V_T$),[v] the reversed diode current is:

$$I_{DIO} = I_s \cdot \left(1 - e^{\frac{-V_D}{V_T}}\right) \approx I_s = qA \cdot \left(\frac{D_n}{N_A L_n} + \frac{D_p}{N_D L_p}\right) \cdot n_i^2 \tag{24}$$

where $A$ is the diode's cross-sectional area; $D_n$ and $D_p$ are the diffusion coefficients of electrons and holes, respectively; $N_D$ and $N_A$ are the n-well donor and p+ acceptor concentrations, respectively; $L_n$ and $L_p$ are the electron and hole diffusion lengths, respectively; and $n_i^2$ is the intrinsic carrier concentration.

More and bigger transistors (i.e., greater $A$) in the stacked configuration increase the cumulative leakage effect of the reversed-diodes. As well, since the diodes voltage gets lower towards the bottom of the transistors stacking, the leakage current caused by the parasitic diodes at the bottom have a greater impact than those at the top. As well, because we are using multiplexers to select between different reference and divider voltages, the transistors of the switches (i.e., Tgate NMOS+PMOS) also contribute with

[v] $V_D > 150$mV at room temperature contributes to a 0.3% error.

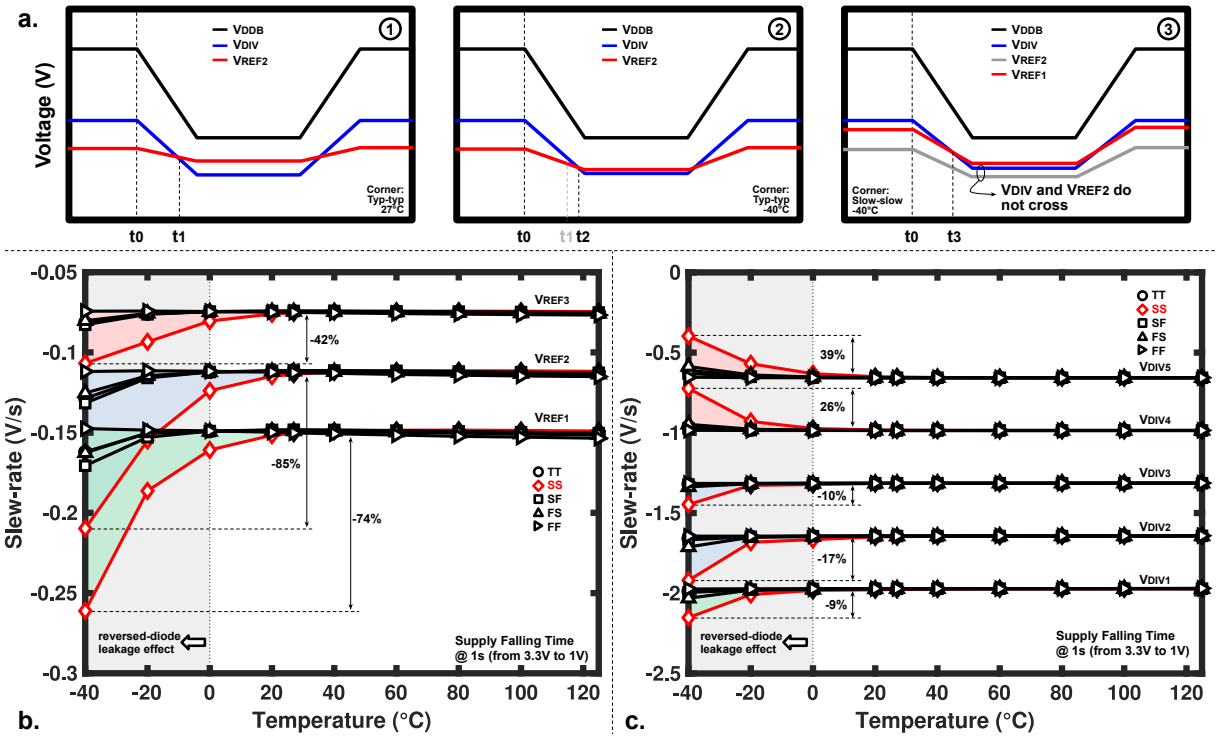


Figure 28. Reversed-diode leakage effect on the BOD: a) BOD threshold voltage scenarios due to leakage; b) voltage reference slew-rate vs temperature for different transistors corners; c) voltage -divider slew-rate vs temperature for different transistors corners.

parasitic reversed diode leakage, as well as channel leakage. Although $n_i^2 \propto e^{-E_g/kT}$, which results in increases in the reversed-diode currents at higher temperatures, the channel subthreshold currents through the main branches are higher than that of the parasitic reversed-diodes (depending on the number of stacked transistors). On the other hand, at lower temperatures the main branch current and the leakage due to reversed-diodes may be comparable, which affects considerably the behavior of the circuits.

As an example, Figure 28.a shows three possible scenarios illustrating the low temperature issue during a BO event in the ultra low-power BOD proposed in this work (i.e., global case). Figure 28.a.1 is the typical case (NMOS and PMOS transistors in typical corners at 27°C) in which both $V_{REF}$ and $V_{DIV}$ cross each other and a BO reset signal is triggered. Figure 28.a.2 shows a low temperature case (-40°C) with NMOS and PMOS in typical corners. In this scenario both $V_{REF}$ and $V_{DIV}$ slew rates are affected by the reversed-diodes leakage and the trigger event occurs at a lower threshold voltage than the one at ambient temperature. Finally, Figure 28.a.3 presents

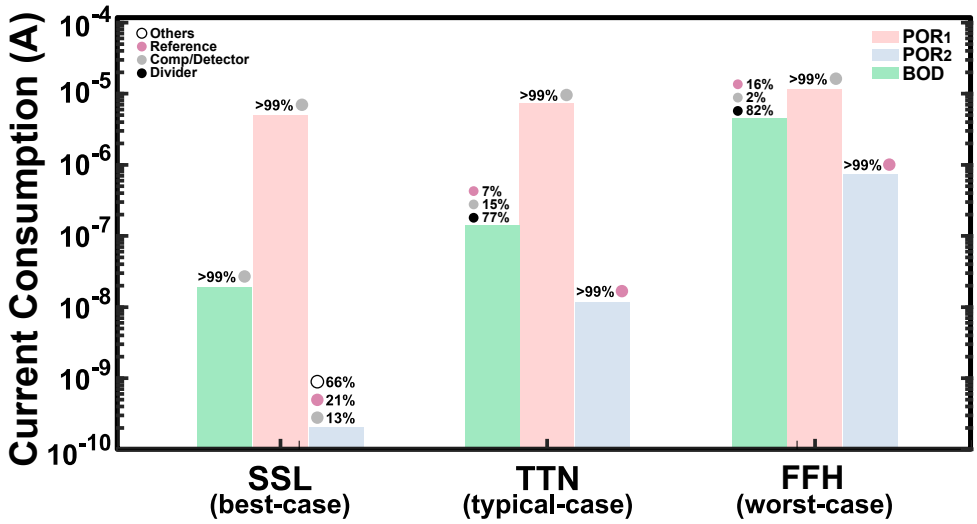


Figure 29. Power consumption of the two POR and the BOD presented in this work. The current breakdown is shown for worst, best and typical cases.

a low temperature scenario but with both NMOS and PMOS transistors in slow corners (SS). Simulations showed that this was the most critical corner, where both $V_{DIV}$ and voltage reference $V_{REF}$ curves do not intercept, therefore preventing the possibility to detect the BO event.

Figures 28.b-c show the slew-rates of the BOD voltage reference and divider taps across different temperatures and for different corners. This figures show that for temperatures lower than 0°C the slew-rates are heavily affected, with the SS corner being the most critical one. For example, for a SS corner and with a BOD configuration selecting $V_{REF2}$ and $V_{DIV2}$, the slew-rates at -40°C are 85% faster at the reference and 17% faster at the divider. Without careful design, the latter may result in the scenario in Figure 28.a.3. With the trimming capability used in this work the low temperature problem can be solved, whether it is in the reference block by modifying its biasing current, or by choosing different voltage reference and divider outputs with the $Sel\_V_{REF}$ and $Sel\_V_{DIV}$ signals. For example, in Figure 28.a.3 scenario, by activating the $V_{REF1}$ output instead of $V_{REF2}$, the interception between the reference signal and $V_{DIV}$ is now possible, hence, letting the BO detection to take place.

Finally, Figure 29 shows the current consumption of the two POR and the BOD presented in this work, showing the best, typical, and worst corners. For the three circuits, it is possible to see that the best case in terms of current consumption occurs at -40°C

with SS corner, while the worst occurs at 125$^{\circ}$C and fast-fast corner. In particular, the variation of the BOD current consumption is greater than an order of magnitude (i.e., 19nA at SSL, and 4.6$\mu$A at FFH). It is clear then that the best case energy-wise coincide with the worst case when considering the BOD performance (with similar results for the PORs), which constitutes a trade-off between the power consumption and temperature range for these types of circuits.

### 4.4. Experimental Results

This section presents the measurements of the proposed voltage monitors. We also present system level measurements when using a power management strategy enabled by the proposed voltage monitors[115].

#### 4.4.1. Voltage Monitors Measurements

Figure 30 shows the measurement setup used in this work: the designed PCB to test the microcontrollers (Figure 30.a), the photos of the two dies with microcontroller-based RISC-V SoCs (Figure 30.b), and examples of the POR and BOD signals when starting-up and during a brown-out event, respectively (Figures 30.c-d). The BOD occupies an area of 150$\mu$m x 70$\mu$m, while the POR occupies an area of 120$\mu$m x 70$\mu$m using a 180nm standard logic technology without additional analog-flavor layers. The reported area includes counters (i.e., one for each voltage monitor) for programmed time delay and additional synthesized digital logic.

Figure 31 shows the measurement results of 9 chips from the two dies presented in Figure 30.b. In order to test the functionality of the voltage monitor cells, power-up and brown-out events in the supply voltage ($V_{DDA}$) were used as test signal. The supply voltage was ramped-up from 0V to 2.4V,[vi] and ramped-down from 2.4V to 0.8V. We used a range of rising and falling times between 3$\mu$s up to 1s, equivalent to power-up slew-rates between 2.4V/s to 0.8V/$\mu$s, and brown-out slew-rates between -0.53V/$\mu$s to

[115] Ckristian DURAN et al. “An Energy-Efficient RISC-V RV32IMAC Microcontroller for Periodical-Driven Sensing Applications”. In: *2020 IEEE Custom Integrated Circuits Conference (CICC)*. 2020, pp. 1–4.

[vi] Due to an implementation error, the maximum supply voltage allowed is 2.4V.

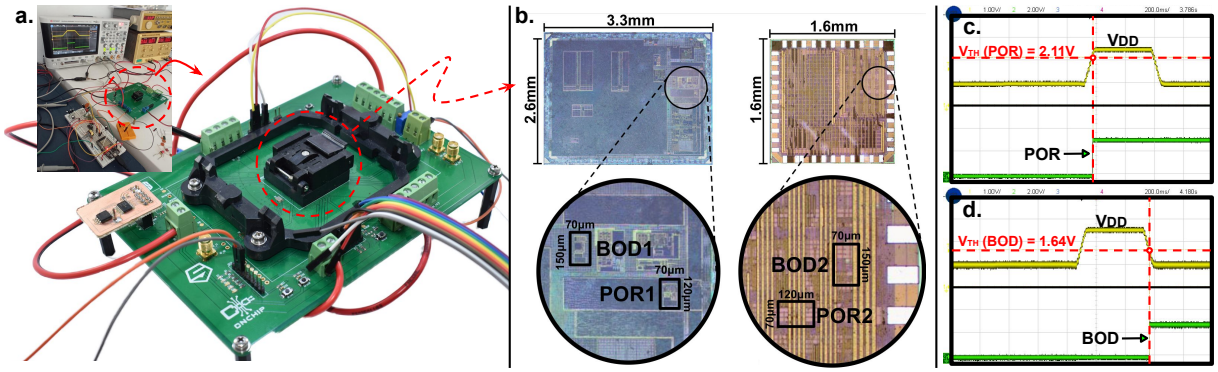


Figure 30. Measurement setup: a) measurement setup and used PCB; b) Photos of the two dies with microcontroller-based RISC-V SoCs in a 180nm CMOS technology; c) POR measured signal after VDD ramp-up; and d) BOD measured signal at a brown-out event.

-1.6V/s. For the measurements we used two different set of performance indicators, the threshold voltage and the time delay, as a difference from Figures 25 and 27 where only the threshold voltages were shown. The time delay is useful for fast supply ramps where the POR or BOD signal is triggered after the supply voltage has been settled.[vii]

Table 8. Comparison with prior work

| | [116] | [117] | [107] | **This work** | |
|---|---|---|---|---|---|
| | | | | **POR1**[112] | **POR2**[110] |
| **Technology** | 0.5um | 0.5um | 180nm | 180nm | |
| **Supply Voltage (V)** | 1.2-3.3 | 1.8-5.5 | 1.8 | 3.3 | |
| **Current (nA)** | 76 | N.A. | 1000 | 7000 | **19** |
| **Supply rise time (s)** | $<10m$ | $<1m$ | $<1.0$ | **$10^{-6}$ - 1.0** | |
| **Temperature (°C)** | – | **-40 - 125** | -20 - 100 | **-40 - 125** | |
| **Area ($\mu m^2$)** | **1900** | 27000 | 12000 | 5700[a] | |
| **Multi-Level** | **Yes** | No | No | **Yes** | |
| | [118] | [108] | [119] | **BOD**[113] | |
| **Technology** | $0.25\mu m$ | 65nm | 180nm | 180nm | |
| **Supply Voltage (V)** | 2.5 | 1.1 | 3.6 | 3.3 | |
| **Current (nA)** | 120 | – | **1** | 200 | |
| **SR$_{V_{DDA}}$ (V/s)** | – | -110k to -44 | – | **-2.3M to -2.3** | |
| **Temperature (°C)** | – | -25 - 105 | 0 - 80 | **-40 - 125** | |
| **Area ($mm^2$)** | – | 0.007 | 0.89 | **0.006**[a] | |
| **Multi Level** | No | No | No | **Yes** | |

[a] Area excluding the counter.

[116] S. U. AY. "A Nanowatt Cascadable Delay Element for Compact Power-on-reset (POR) Circuits". In: *2009 52nd IEEE International Midwest Symposium on Circuits and Systems*. Aug. 2009, pp. 62–65.

[117] R. PRAKASH. "Zero Quiescent Current, Delay Adjustable, Power-on-reset Circuit". In: *2014 IEEE*

[vii] We define the time delay as the time between the moment the supply settles (i.e., end of power-up for the POR, and end of falling event for BOD) and when the POR/BOD signal is triggered, as shown in figure 31.

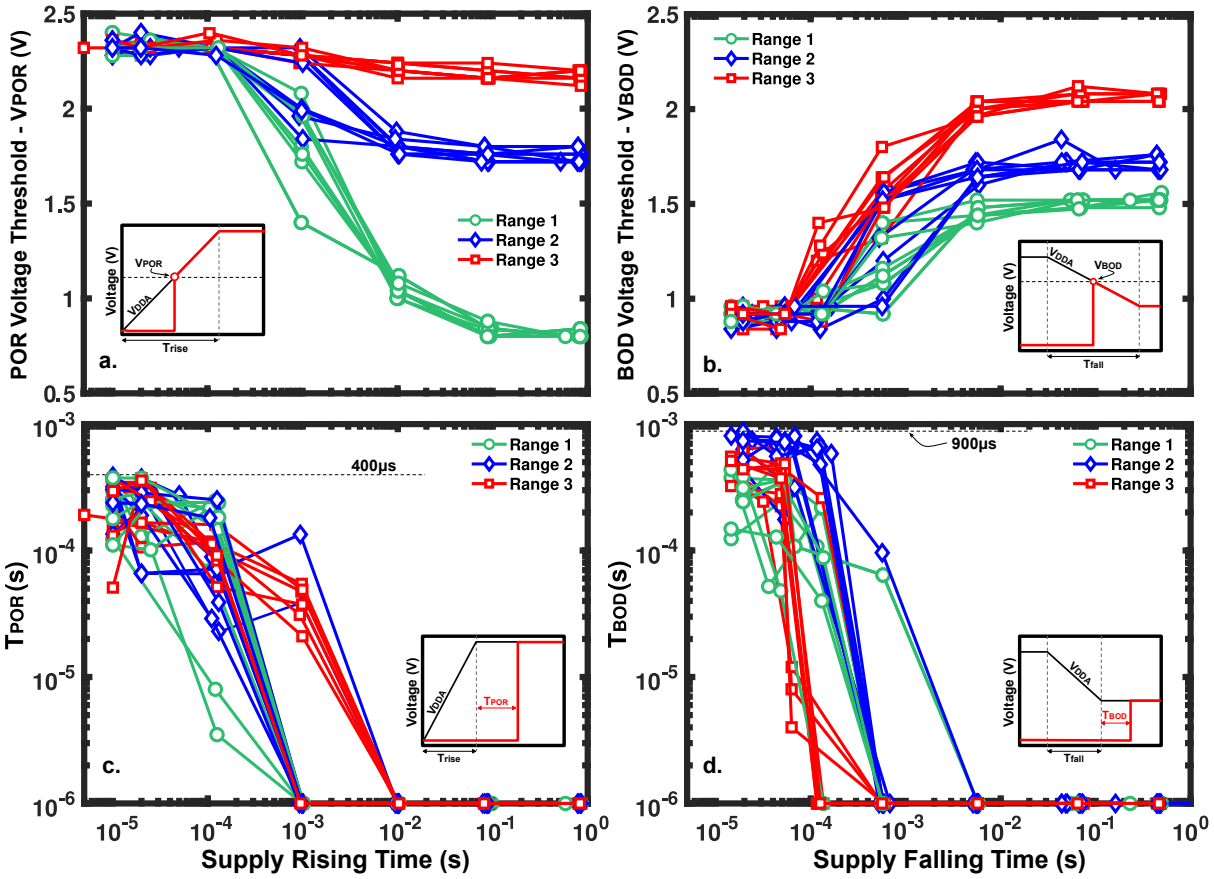


Figure 31. Measurement results: a) POR threshold voltage; b) BOD threshold voltage; c) POR time delay; and d) BOD time delay.

Table 8 shows a comparison with relevant reported works. In regards with the BOD, our proposal has the highest power consumption among those presented (200nA), but it is as well the one with the widest temperature range. For instance, the reported measurements at[119] ($<$1nA) do not include temperature results below 0°C and higher than 80°C. On the other hand, the POR2 quiescent current is around 19nA for typical case, which is the lowest among the works in the table.[viii] Also the proposed PORs and BOD allow the selection multiple levels while having a low area consumption and robust operation, in comparison to fixed number of levels in other works.

*Dallas Circuits and Systems Conference (DCAS)*. Oct. 2014, pp. 1–4.

[118] D.P. GUBBINS. *Brown-out Detector*. US Patent 6,894,544. May 2005.

[119] Inhee LEE et al. "Battery Voltage Supervisors for Miniature IoT Systems". In: *IEEE Journal of Solid-State Circuits (JSSC)* (Nov. 2016).

[viii] It has a maximum current consumption of 1.2$\mu$A, and is caused by the voltage reference block, in a fast corner.

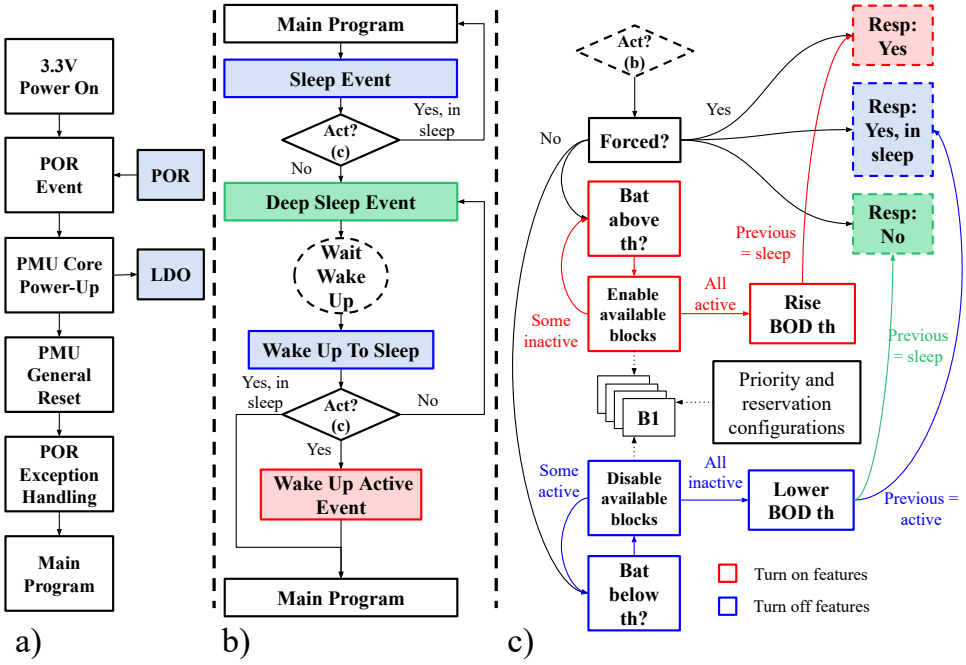


Figure 32. Power management unit main algorithm: a) Initialization algorithm; b) Sleep and deep-sleep operation; c) Block diagram of sleep events.

Measured operation of full SoCs containing our voltage monitors demonstrates the widest range of rising and falling times responses among the works presented in Table 8, as well as memory protection for brown-out events at low-temperature. These capabilities of the proposed circuits extend its implementation in a broader range of power management applications, as we will show in the following section.

### 4.4.2. PM Strategy using the Proposed VMs in a RISC-V Microcontroller[115]

Figure 32 describes the PMU algorithm. When powering up with a 3.3V, the POR issues an event to perform a PMU initialization. The initialization turns LDO on and triggers general resets and exceptions to the core to execute the main program (Figure 32.a). The main program can start the event-driven algorithm from the PMU (Figure 32.b). The main program enables blocks and triggers active mode if the battery level rises above the BOD threshold. Although the PMU regularly drives the power operation, the main program may force operation modes and reserve blocks if necessary. The PMU may trigger operation mode transitions according to the battery supervision of an integrated BOD. The PMU logic has the potential to choose to keep running the main program in sleep mode or to trigger a deep-sleep transition (Figure 32.c). This

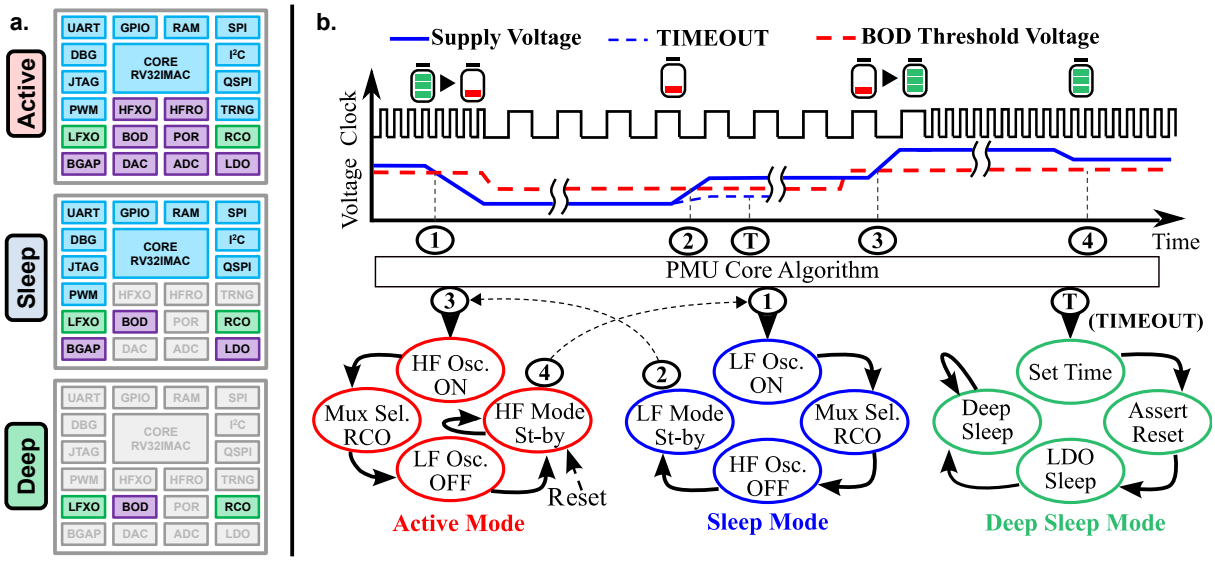


Figure 33. PMU brown-out notification handling scheme.

logic may also set the BOD threshold voltage and withdraw blocks to user-prioritized blocks. In cases where all available blocks are withdrawn, the PMU decreases BOD threshold down to values that require to switch to sleep or deep-sleep mode with selected clock sources and user-prioritized blocks (Figure 33.a).

The proposed PMU logic inside the AON domain integrates programmable state machines to control sleep modes. The BOD notifies the PMU core algorithm to trigger sleep mode sequences depending on the supply voltage level. Figure 33.b illustrates these notifications where graphs 1-4 show different types of battery events. These events compare the supply voltage behavior (blue) with a threshold level from the BOD (red) set by the PMU algorithm. Event (1) happens when the battery-level discharges and triggers a brown-out detection. The PMU lowers the BOD threshold in the algorithm and waits for a voltage rise event (2) while performing energy harvesting. Throughout these events, the MCU enters sleep mode. If it does not raise the supply voltage above the lowered threshold during a timeout, the PMU algorithm commands the MCU to enter deep-sleep mode. Once the desired voltage level is reached, the BOD threshold is set to a higher value accordingly. In this state, the PMU waits for a rise in the supply voltage above the maximum threshold (3). The PMU allows active mode execution once a supply voltage rises to a safe supply value. In active mode, the supply voltage is measured using the maximum BOD threshold in order to keep alert

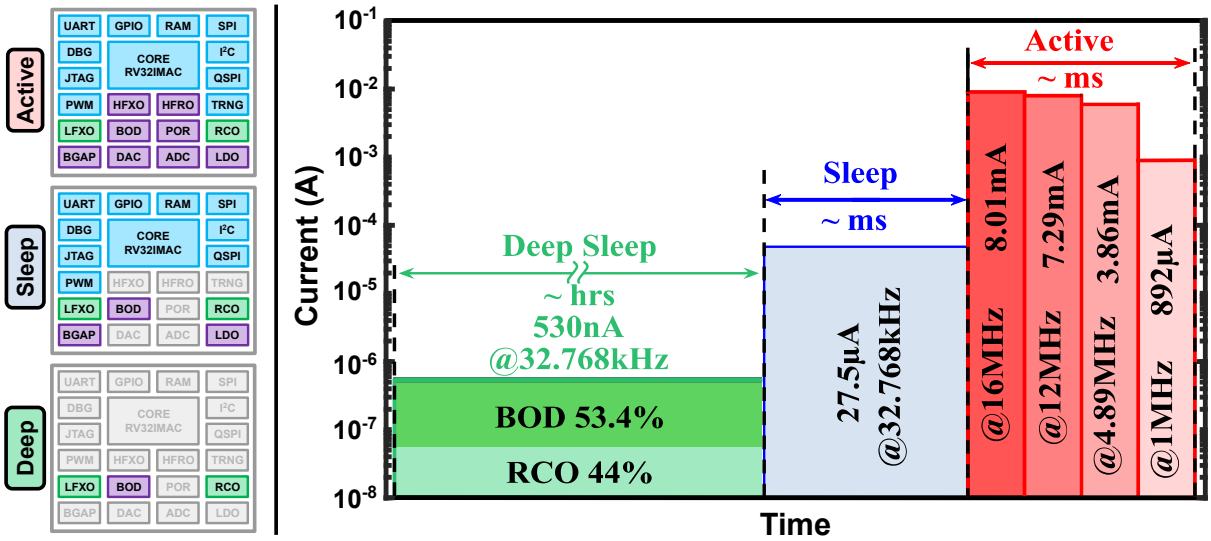


Figure 34. Active, sleep, and deep-sleep modes current consumption measurement.

for a voltage drop due to an additional loading (4).

Figure 34 indicates the measured current consumption at different modes at nominal conditions. In deep-sleep mode, the BOD monitors the battery supply rail drawing 280nA from the 3.3V IO voltage. The RCO oscillator operating as a wake-up timer at 32.768kHz draws 250nA. In sleep mode, the MCU draws a total 27.5$\mu$A from the internal LDO 1.2V-core voltage and the 3.3V AON domain. Figure 34 also indicates a current consumption of four different configured operation frequencies in active mode. A sensor implementing the proposed MCU powered by an SR626 battery will last 5.6 years if every 24h an active and a sleep mode transition occurs during 1ms and 5ms respectively.

### 4.5. Summary

In this chapter, we presented ultra low-power multi-level voltage monitors for multi-mode fine-grained power management strategies. Simulation results over PT variations, as well as measurements at nominal temperature, showed a robust performance within the industrial temperature range from -40°C to 125°C and a wide supply rise and falling times, ranging from 1us to 1s.

In a first version of the POR (POR1), we managed to obtain a current consumption of 7$\mu$A. In the second version (POR2) the POR had a nominal current consumption of 19nA. Both PORs had up-to 3 different voltage threshold levels. In regards to the BOD,

we presented an architecture with low-temperature slew compensation for low power applications, multiple voltage threshold levels, and a current consumption of 200nA.
We also showed experimentally how these voltage monitors could be used in a real power management strategy. By having multi-level voltage thresholds we enabled three different power modes that used lower voltage supply: active, sleep, and deep-sleep. According to measurements, the SoC had an 8mA current consumption at 16MHz in active mode, 27.5$\mu$A at 32.768kHz in sleep mode, and 530nA at 32.768kHz in deep-sleep mode.
In comparison to previous research that neglect to consider the low-temperature effects when using large impedance branches, this work achieved a low current consumption even by considering these temperature effects. Current consumption, programmability, and reduced area, makes the proposed voltage monitors enablers of different fine-grained power management schemes.

# 5. SYSTEM-ON-CHIP POWER DELIVERY NETWORK: AN IN-DEPTH LOOK AT VOLTAGE GLITCHING

## 5.1. Introduction

With billions of interconnected devices, security is one of the main challenges in today's applications (e.g., the internet of things). With more nodes connected, there are more opportunities to breach whole systems' security, leaving the system vulnerable to attacks from different fronts. Because of this, it is critical to maintain high-security levels at every single point in the system.

Although software-based attacks are the most known type of security infringement, hardware attacks (or physical attacks) have become more relevant over the last 15 years because of their effectiveness. Hardware attacks aim to disrupt the correct behavior of a system through physical stress. This stress modifies the operating conditions of the system and provokes a fault injection, which manifests itself as a malfunction at the software level (e.g., corruption of instructions or data).

There are several mechanisms to successfully inject a fault into a system[120]: clock glitching, temperature tampering, voltage underfeeding, electromagnetic pulses, laser pulses, and voltage glitching. Among these mechanisms, voltage glitching is of the most interest for this work. A voltage glitch attack is an externally forced transient power drop in the supply line that occurs at a specific instant, with a duration typically in the tens of ns to the units of $\mu$s. This kind of attack does not require special equipment and can be mounted under low cost and low expertise[121]. The permanent availability of an external pin for a power supply makes it one of the preferred methods for security tampering. Fig. 35 shows one of the most common setups for voltage glitching attacks. Historically, voltage glitching attacks have required physical access to the system to be effective. Investigating these types of fault injection techniques becomes more relevant

[120] Bilgiday YUCE et al. "Fault attacks on secure embedded software: Threats, design, and evaluation". In: *Journal of Hardware and Systems Security* 2.2 (2018), pp. 111–130.

[121] Colin O'FLYNN et al. "ChipWhisperer: An Open-Source Platform for Hardware Embedded Security Research". In: vol. 8622. Apr. 2014.

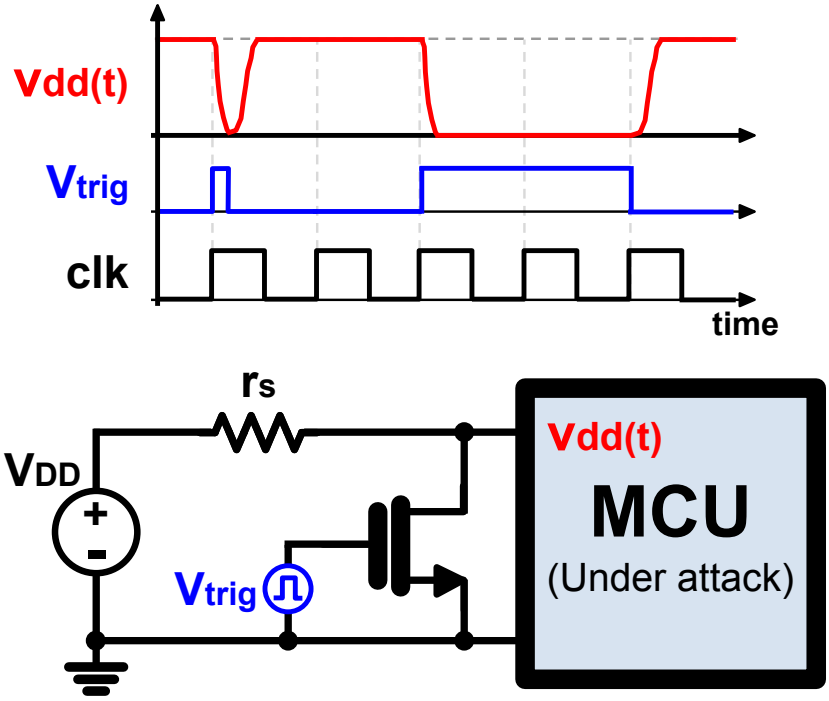


Figure 35. Crossbar glitch attack circuit[121].

when recent works have demonstrated the possibility to gain control over the DVFS (Dynamic Voltage Frequency Scaling)[i] interface of a system-on-chip (SoC) through malicious software[122]. The latter allows the attacker to perform glitch attacks by controlling the clock and supply signals. The authors demonstrated temporal location attacks (i.e., control over the attack's start time and duration), but it may be viable in the future to enable spatial location attacks in SoCs with fine-grained power management units (i.e., SoCs with different voltage and clock domains).

Understanding the nature of voltage glitching fault injection is imperative to develop countermeasures against them. Previous works have presented empirical evidence identifying the cause of voltage glitching fault injection as timing constraint violations[123,124].

[122] Adrian TANG et al. "CLKSCREW: Exposing the Perils of Security-Oblivious Energy Management". In: *26th USENIX Security Symposium (USENIX Security 17)*. Vancouver, BC: USENIX Association, Aug. 2017, pp. 1057–1074.

[123] A. DJELLID-OUAR et al. "Supply voltage glitches effects on CMOS circuits". In: *International Conference on Design and Test of Integrated Systems in Nanoscale Technology, 2006. DTIS 2006.* 2006, pp. 257–261.

[124] Loïc ZUSSA et al. "Power supply glitch induced faults on FPGA: An in-depth analysis of the injection

[i] DVFS is a popular power management technique, where the operating voltage and frequency of the microprocessor is regulated based on its dynamic workload.

These works have neglected the power delivery network (PDN) of the SoC in their analyses. Because of the latter, the timing constraint violation approach (as it has been presented in previous works) does not relate the characteristics of the voltage glitch signal and its potential to inject a fault into the system.

Considering the above, this work makes the following contributions:

- We include the PDN in the timing constraint violation approach, explaining how fault injection voltage glitching is successful. Two cases are identified for effective voltage glitching fault injections when comparing the time response of the PDN and the system's operating frequency. This work shows the analyses for both cases.
- Our analysis evince that when the PDN time response is slower than the system's clock period, voltage glitching has the same effect as underpowering. Simulation results supporting our analysis are presented. In this case, we identify that the obtained glitch duration-amplitude relation has the same behavior as experimental data in previous works.
- This work shows that when the PDN time response is faster than the system's clock period, the fault injection is only possible when the supply voltage rises to the nominal value (at the end of the glitch). In this case, the fault injection always occurs with a minimum glitch duration proportional to a clock's period (or half clock period, if the logic has positive and negative edge-driven circuitry). We present simulation and measurement results to support our findings.
- Our work gives the foundations for a new system's fault characterization approach, leaving aside the common glitch duration-amplitude relationship that only considers squared pulse glitches. We anticipate our results to be the basis for more sophisticated fault injection characterization including arbitrary glitch waveforms, like the ones generated by genetic algorithms[125].

This chapter is organized as follows: section 5.2 summarizes some of the previous papers related to this work. Section 5.3 presents a voltage glitching analysis, including

mechanism". In: *International On-Line Testing Symposium (IOLTS)*. 2013, pp. 110–115.

[125] Claudio BOZZATO et al. "Shaping the Glitch: Optimizing Voltage Fault Injection Attacks". In: *IACR Transactions on Cryptographic Hardware and Embedded Systems* 2019.2 (Feb. 2019), pp. 199–224.

the PDN and the analysis of voltage glitching on memory-based cells. Section 5.4 shows simulation results for a better understanding of the two cases for successful voltage glitching fault injection. Section 5.5 shows measurement results supporting the analysis developed in this work. Finally, section 5.6 presents the conclusions obtained.

## 5.2. Related Works

The main interest in most of the research involving voltage glitching attacks is in the consequences of the fault injection. In other words, investigating the effects that a voltage glitch attack can cause in a system, whether it is a specific cryptographic system[126,127,124], general-purpose processors[125 ,128,129,130], a machine learning accelerator[131], or even in mixed-signal systems[132].

On the other hand, few papers explain why fault injections through voltage glitch attacks are successful. The authors in[123] showed analytically, and through simulations, that voltage glitch attacks cannot inject faults into D-flip-flops. Additionally, they presented simulation evidence relating the voltage glitching fault injections with timing constraint violations. The latter result was confirmed in[124], where timing constraint violations were found to be the main reason behind fault injections in both voltage and clock glitching attacks. They provided experimental results showing that nearly the same faults were injected in an FPGA implementing an advanced encryption standard (AES), independently of the method used for the attacks. Based on these results, several

[126] Alessandro BARENGHI et al. "Fault Injection Attacks on Cryptographic Devices: Theory, Practice, and Countermeasures". In: *Proceedings of the IEEE* 100.11 (2012), pp. 3056–3076.

[127] Honorio MARTÍN et al. "Fault Attacks on STRNGs: Impact of Glitches, Temperature, and Underpowering on Randomness". In: *IEEE Transactions on Information Forensics and Security* 10.2 (2015), pp. 266–277.

[128] Jan Van DEN HERREWEGEN et al. "Fill your Boots: Enhanced Embedded Bootloader Exploits via Fault Injection and Binary Analysis". In: *IACR Transactions on Cryptographic Hardware and Embedded Systems* (Dec. 2020), pp. 56–81.

[129] Colin O'FLYNN. "Fault Injection using Crowbars on Embedded Systems". In: *IACR Cryptol. ePrint Arch.* 2016 (2016), p. 810.

[130] Niek TIMMERS et al. "Controlling PC on ARM Using Fault Injection". In: *2016 Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC)*. 2016, pp. 25–35.

[131] Wenye LIU et al. "Stealthy and Robust Glitch Injection Attack on Deep Learning Accelerator for Target With Variational Viewpoint". In: *IEEE Transactions on Information Forensics and Security* 16 (2021), pp. 1928–1942.

[132] Noemie BERINGUIER-BOHER et al. "Voltage Glitch Attacks on Mixed-Signal Systems". In: *2014 17th Euromicro Conference on Digital System Design*. 2014, pp. 379–386.

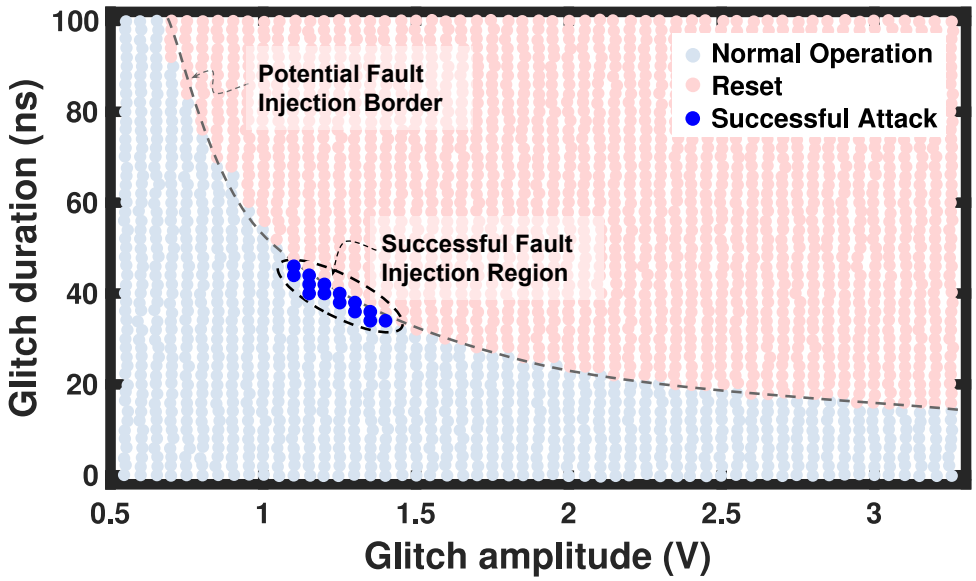


Figure 36. Relationship between glitch duration and glitch amplitude. Data extracted from[120].

works have proposed circuits and strategies for the mitigation or detection of voltage glitching attacks[133,134,135].

In addition, the work presented in this chapter is further motivated by the experimental evidence in[130,120,128], where different voltage glitch amplitudes and glitch durations were applied to inject a fault into an MCU. In particular, the authors in[130] and[120] empirically show the relation between the glitch attack duration and the glitch amplitude. Fig. 36 shows data extracted from[120], where each dot represents a voltage glitch attack with specific duration and amplitude.[ii]

There are four zones that can be distinguished from Fig. 36: the region where the glitch did not affect MCU and continues within normal operation (light blue dots); the

[133] Kamil GOMINA et al. "Power supply glitch attacks: Design and evaluation of detection circuits". In: *2014 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*. 2014, pp. 136–141.

[134] Arvind SINGH et al. "Mitigating Power Supply Glitch based Fault Attacks with Fast All-Digital Clock Modulation Circuit". In: *2019 Design, Automation Test in Europe Conference Exhibition (DATE)*. 2019, pp. 19–24.

[135] Hyung-Min LEE et al. "A Nonvolatile Flip-Flop-Enabled Cryptographic Wireless Authentication Tag With Per-Query Key Update and Power-Glitch Attack Countermeasures". In: *IEEE Journal of Solid-State Circuits* 52.1 (2017), pp. 272–283.

[ii] The authors in[120] performed a sensitivity analysis, using test software on the attacked MCU that runs loops and typical instructions that may be affected. The test software accelerates the detection of hardware weaknesses, and supports finding optimal attack parameters (i.e., glitch duration and amplitude).

region where the glitch was too ”strong” and resulted in a reset of the MCU From the figure; the border between the two mentioned regions (potential fault injection border); and the region where successful attacks where possible. The potential fault injection border delimits the zone where the glitch attacks may be effective (the successful fault injection region is within this border). In this work, we show analytical expressions for the potential fault injection border, explaining why the glitch duration follows an inversely proportional behavior with respect to the glitch amplitude (see section 5.3.1).

## 5.3. Voltage Glitching: including Power Delivery Network and its effects on Memory-based cells

The analyses presented in[124] ,[123] left out significant aspects of the voltage glitching nature within an SoC, as well as circuits non-idealities. In the case of the work in[123], the analysis performed over the SRAM cell only uses small-signal models and omitted non-idealities such as leakage currents (i.e., reversed diodes) and devices mismatch (i.e., offset voltage). On the other hand, the authors in[124] centered their studies on the effects of voltage glitching attacks at gate and transistor levels, omitting the power delivery network of the SoC.

In this chapter, we include the PDN in the classical timing constraint violation approach, obtaining the necessary insight into whether a voltage glitch signal can cause a fault injection or not. We found that because a digital system does not operate continuously but every clock period, two scenarios can be distinguished for effective voltage glitching fault injections, when considering the PDN:

- Case I: the PDN time response is slower than the clock period. In this case, the system can correctly execute several operations during the supply ramping down until the latter reaches a minimum voltage where a fault occurs. We show in section 5.4 that this effect is the same as the one caused by underpowering.
- Case II: the PDN time response is faster (or comparable) than the clock period. In this case, the supply voltage reaches its lower value so fast that there is not even a second clock rising edge for a fault to occur (the system does not execute any operation). The supply voltage goes low, and the digital circuitry stops its op-

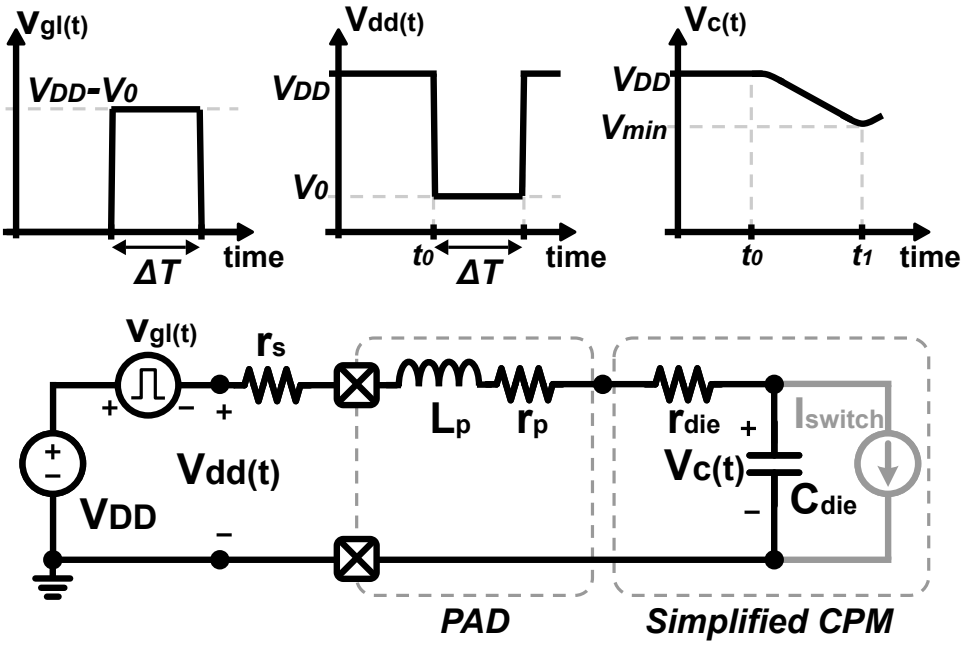


Figure 37. PDN circuit model of a die[136] for a glitch attack characterization. The electrical model of the pad is included as well.

eration without entering a reset state. Depending on the duration of the glitch, the memory-based cells (e.g., flip-flops, latches, SRAM, etc.) preserve the previous state without bit flipping (see section 5.3.2). Hence, the only possibility for a fault injection is during the ramp-up of the supply voltage.

In this section, we show the analysis of the PDN during the supply ramp-down due to a voltage glitch.We also analyze the effect of voltage glitching on memory-based cells. These two analyses are the basis for understanding the two possible fault injection cases. Section 5.4 shows simulation examples for a complete understanding of both cases.

**5.3.1. Voltage Glitch Analysis Including Power Delivery Network** Using the PDN circuit model in Fig. 37[136], a fault injection may occur when a minimum voltage $V_{min}$ stored in the die capacitor ($C_{die}$) is reached. $V_{min}$ is a constant value that sets the minimum voltage for proper MCU performance at a specific operating frequency, as established by the timing constraint violation approach[124].

[136] Emre KULALI et al. “Chip Power Model - A New Methodology for System Power Integrity Analysis and Design”. In: *2007 IEEE Electrical Performance of Electronic Packaging*. 2007, pp. 259–262.

Assuming that the switch current can be neglected during a voltage glitch attack (the discharge current is higher than $I_{switch}$), the transfer function of the system presented in Fig. 37 is given by:

$$H(s) = \frac{1}{L_p C_{die} s^2 + R_{eq} C_{die} s + 1} = \frac{w_0^2}{(s + s_1)(s + s_2)} \tag{25}$$

The damping factor of the system ($\zeta = \frac{R_{eq}}{2}\sqrt{\frac{C_{die}}{L_p}}$) will determine its impulse response, as shown in equation (26), with $R_{eq} = r_s + r_p + r_{die}$, $s_{1,2} = \alpha \mp w_d$, $\alpha = \zeta w_0$, $w_d = w_0\sqrt{|\zeta^2 - 1|}$, and $w_0 = \frac{1}{\sqrt{L_p C_{die}}}$.

$$h(t) = \begin{cases} \frac{w_0^2}{2w_d}(e^{-s_1 t} - e^{-s_2 t}) \cdot u(t) & : \zeta > 1 \\ w_0^2 t e^{-w_0 t} \cdot u(t) & : \zeta = 1 \\ \frac{w_0^2}{w_d} e^{-\alpha t} sin(w_d t) \cdot u(t) & : \zeta < 1 \end{cases} \tag{26}$$

Given a squared pulse voltage glitch, as the one in Fig. 37, with amplitude $V_{DD} - V_0$ and duration $\Delta T$, the minimum voltage $V_{min}$ is:

$$V_{min} = v_c(t_0 + t_1) = V_{DD} + \Delta V\big[m(t_1) - m(t_1 - \Delta T)\big] \tag{27}$$

where $\Delta V$ is the amplitude of the glitch ($\Delta V = V_{DD} - V_0$), and $t_0$ is the moment at which the glitch starts. $m(t)$ depends on the damping factor, and $t_1$ is the time where $V_{min}$ occurs, both depicted in eqs. (28) and (29), respectively. We assume a $\zeta \geq 1.74$

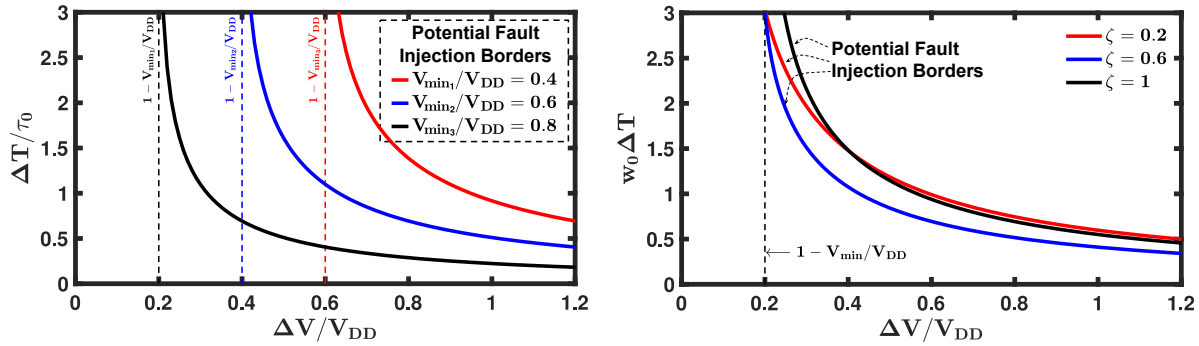


Figure 38. Potential fault injection borders. Normalized voltage glitch duration versus the normalized voltage glitch amplitude, from eq. (27): (a) Special overdamped case ($\zeta \geq 1.74$). Glitch duration normalized to the PDN's time constant. The asymptotes (dashed vertical lines) are equal to $(V_{DD} - V_{min})/V_{DD}$; (b) Critally damped and underdamped cases ($\zeta \leq 1$). Glitch duration normalized to the PDN's resonant frequency.

for the overdamped case[iii] and approximate $1 \leq \zeta < 1.74$ as critically damped.

$$m(t) = \begin{cases} e^{-t/\tau_0} \cdot u(t) & : \zeta \geq 1.74 \\ e^{-w_0 t}\left(1 + w_0 t\right) \cdot u(t) & : \zeta = 1 \\ e^{-\alpha t}\left[\frac{\alpha}{w_d} sin(w_d t) + cos(w_d t)\right] \cdot u(t) & : \zeta < 1 \end{cases} \tag{28}$$

$$t_1 = \begin{cases} \Delta T & : \zeta \geq 1.74 \\ \frac{\Delta T}{1 - e^{-w_0 \Delta T}} & : \zeta = 1 \\ \frac{1}{w_d} tan^{-1}\left[\frac{sin(w_d \Delta T)}{cos(w_d \Delta T) - e^{-\alpha \Delta T}}\right] & : \zeta < 1 \end{cases} \tag{29}$$

When replacing eqs. (28) and (29) in eq. (27), it is possible to obtain an expression for

[iii] Here the resistance and the capacitive reactance are more dominant than the inductive reactance. Assume that the second pole of the system (as given in equation [25]) is at least one decade higher than the first pole ($s_2 > 10 s_1$). The latter sets a damping factor $\zeta \geq 1.74$, where the circuit in Fig. 37 approximates to a first-order RC system, with a time constant $\tau_0 = R_{eq} C_{die}$.

the potential fault injection borders (see curves in Fig. 38). As the name suggests, the curves in Fig. 38 delimits the points where there is a potential risk for a fault injection to occur. Fig. 38a shows the glitch duration normalized to the PDN's time constant ($\Delta T / \tau_0$) against the normalized glitch amplitude ($\Delta V / V_{DD}$) for different $V_{min} / V_{DD}$, and for $\zeta \geq 1.74$. Fig. 38b shows the glitch duration normalized to the PDN's resonant frequency ($w_0 \Delta T$) against the normalized glitch amplitude for the underdamped and critically damped cases ($\zeta \leq 1$).

Regardless of the system's damping factor in all the potential fault injection borders in Fig. 38, the glitch duration has an inversely proportional-like behavior with respect to the glitch amplitude, similar to Fig. 36. The expressions presented in this section analytically explain the behavior between the glitch amplitude and the glitch duration found through physical experiments in[120 ,130].

From measurement results such as the one presented in Fig. 36, not only glitch characteristics can be found (i.e., glitch duration and amplitude), but other useful system information can be obtained through curve fitting of the potential fault injection border, such as the PDN model and $V_{min}$. In fact, in terms of a system's fault characterization, like the ones performed in[120 ,137], using $V_{min}$ as the main parameter (instead of the glitch duration and amplitude) permits consideration of any glitch waveform, not only squared pulses but even glitches with arbitrary waveforms generated by genetic algorithms or neural networks[125].

**5.3.2. Voltage Glitch attacks on Memory-based cells** As stated before, the work in[123] performed an analysis over an SRAM cell omitting circuit non-idealities such as leakage currents (i.e., reversed diodes) and devices mismatch (i.e., offset voltage), concluding that it was not possible to cause fault injection in memory-based (latch) cells.

Fig. 39a shows the non-idealities on the commonly used 6T (six-transistor) SRAM cell. When an SRAM cell is under a voltage glitch attack, like the one presented in Fig. 39b

[137] Thomas TROUCHKINE et al. "Fault Injection Characterization on modern CPUs - From the ISA to the Micro-Architecture". In: *13th IFIP International Conference on Information Security Theory and Practice (WISTP)*. Dec. 2019, pp. 123–138.

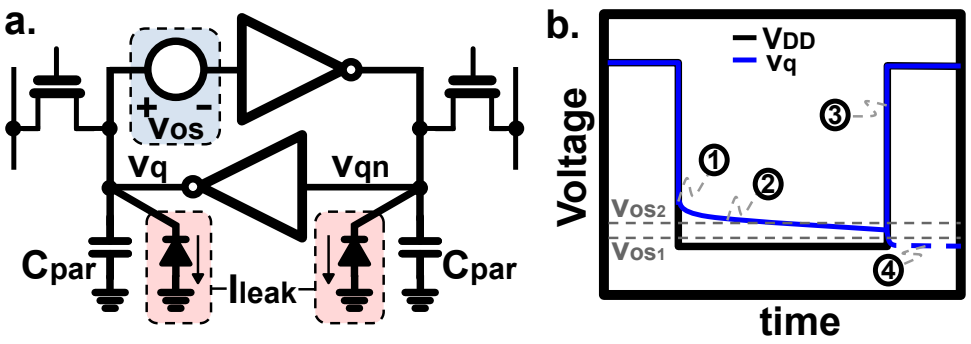


Figure 39. Non-idealities effects in SRAM cell. a) 6T SRAM cell with parasitic capacitors, reversed diodes, and offset voltage. b) Transient behavior of internal SRAM node $v_q$: (1) transistors turned-off; (2) discharge of $C_{par}$ due to leakage currents; (3) case where $v_q$ greater than the offset voltage ($v_{os_1}$); (4) case where $v_q$ lower than the offset voltage ($v_{os_2}$).

(without considering the PDN), it may experience a complete shutdown depending on the glitch duration. Hence, when the glitch is over (supply voltage $V_{DD}$ back to nominal value), the SRAM may flip the stored bit.

The internal node $v_q$ (assuming that $v_q$ node was set to a logic one) follows $V_{DD}$ until the transistors are turned off, and the positive feedback of the latch structure does not work anymore. Due to leakage current ($I_{leak}$, mainly caused by the parasitic reversed diodes of the transistors PN junctions), the voltage stored in the parasitic capacitors ($C_{par}$) starts to discharge until the end of the glitch. When $V_{DDA}$ ramps up and depending on the offset level of the SRAM ($v_{os}$, caused by the devices mismatch), the SRAM will either retain the previously stored value ($v_q > v_{os}$) or flip the bit ($v_q < v_{os}$). This analysis applies to any cell with a latch structure (e.g., SRAM, flip-flops).

Fig. 40 shows the Monte Carlo simulations to evaluate the percentage of flipped bits for different glitch durations. The results are presented for a minimum size transistors SRAM cell and a standard D-Flip-Flop (D-FF) cell. The simulations were performed using a 130nm CMOS technology and for three different temperatures.

According to the Monte Carlo simulations, D-FF cells are less prone to voltage glitching attacks than SRAM cells. The latter is because the transistors used in the D-FF are larger, which results in higher parasitic capacitance (i.e., longer retention time).[iv] The effect of the temperature on the simulation results is also evident, showing an order of

[iv] Bigger transistors mean higher leakage current as well, but the parasitic capacitance effect is dominant.

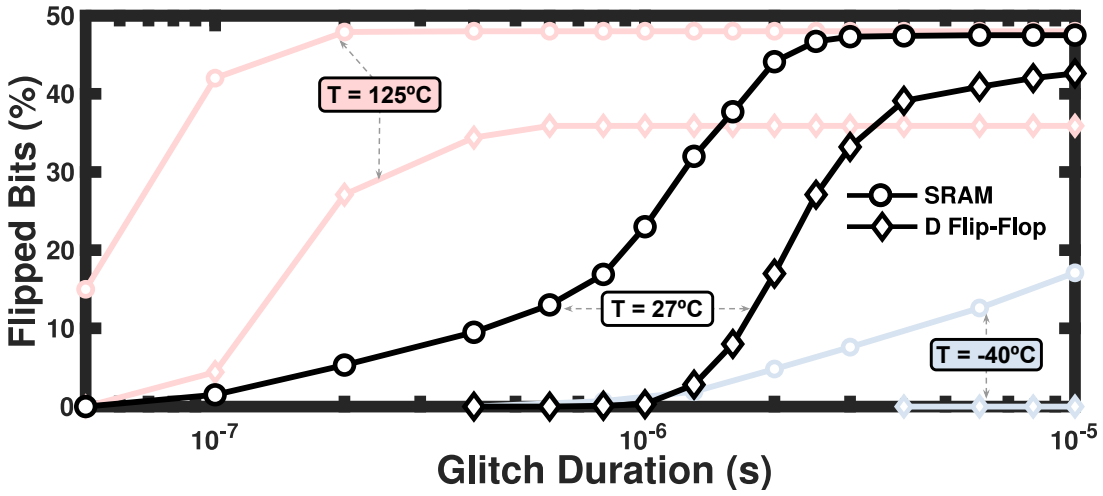


Figure 40. Percentage of flipped bits against glitch duration in a conventional minimum size transistors SRAM cell, and a standard cell D-Flip-Flop using a 130nm CMOS technology node. Montecarlo simulation results of 1000 runs for each point in the graph.

magnitude lower retention time for 125°C, and an order of magnitude higher retention time for -40°C, compared to ambient temperature (27°C).

Flipping bits may put the system in an unknown state depending on how many memory-based cells could be flipped (glitch duration). For this reason, and because it is impossible to establish which cells are attacked,[v] flipping bits on memory-based cells is not the best fault injection technique, though it certainly can give the maximum voltage glitch duration before the attacked system enters a reset or unknown state.

## 5.4. Understanding the relation between the PDN time response and the operating frequency

We used the testbench in Fig. 41 for a complete understanding of the relation between the PDN response under a voltage glitch attack and the system operating frequency, using a CMOS 130nm technology. The testbench depicts an underdamped PDN circuit ($\zeta < 1$), a buffered clock, and two loops formed by combinational circuitry and rising-edge D-FFs. Placing a buffer after the input clock signal emulates the behavior within an SoC, where the clock is also affected by the supply voltage glitches. The latter is in contrast to previous works[123,124] where this effect was not considered.

[v] A fault injection caused by voltage glitching flipped bits is unique to a physical implementation of a system. In other words, it may not be replicable to other physical versions of the same system.

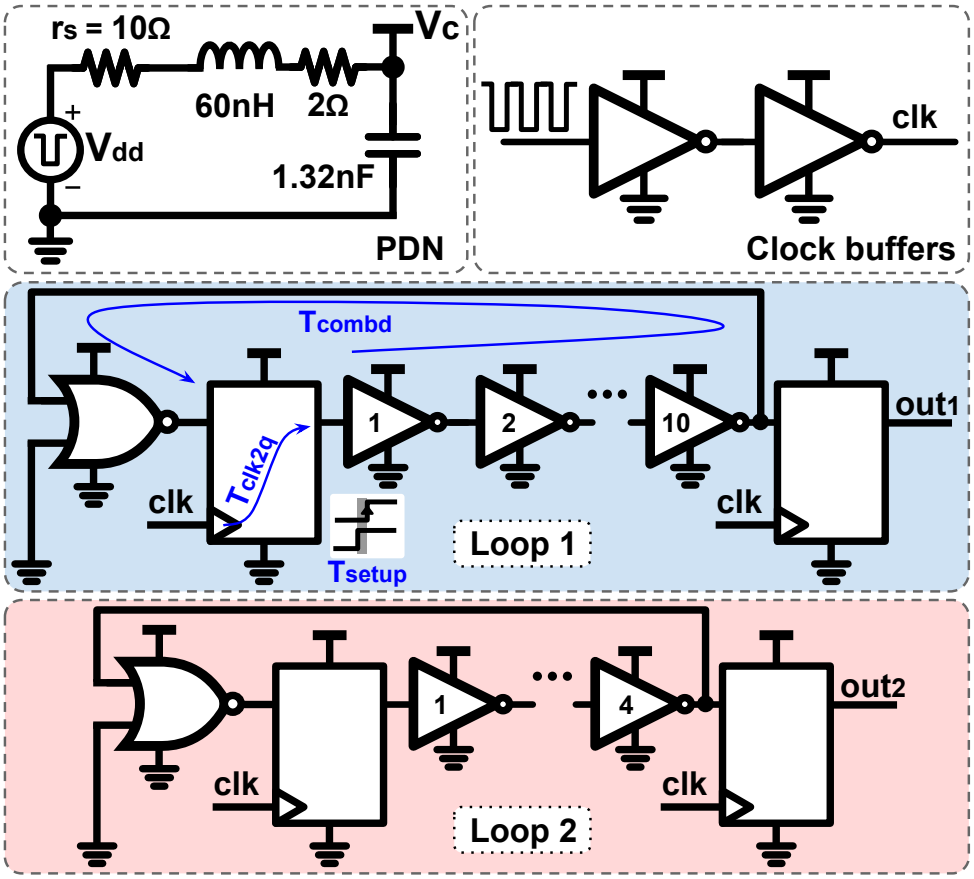


Figure 41. Simulation testbench. The loops are 90° out of phase between each other. A voltage glitch attack can sync both signals.

The main difference between both loops is the number of combinational elements in cascade (ten inverters and one NOR for 'Loop 1'; four inverters and one NOR for 'Loop 2'). The outputs of both loops ($\mathrm{out}_{1,2}$) are periodic signals with twice the period of the system clock, but 90° out of phase to each other. We will consider that a fault is injected if both signals get synced (momentarily or permanently) through supply voltage glitching.

**5.4.1. Underpowering: Timing Constraints vs DC Supply Voltage** The first thing to analyze is the behavior of the system's minimum achievable clock period (maximum operating frequency) as a function of the supply voltage. According to the results presented in[124], the propagation delay of CMOS circuits is inversely proportional to the supply voltage. Because of the latter, a fault injection can occur by decreasing the supply voltage (underpowering), forcing a time constraint violation on the system.

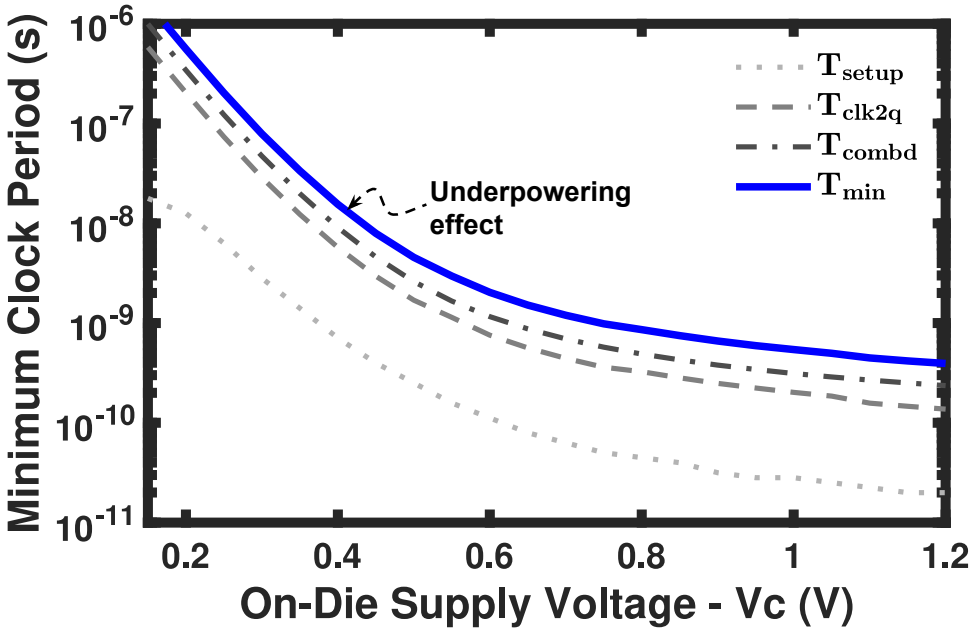


Figure 42. Underpowering: Minimum clock period of the circuit ($T_{min} = T_{combd} + T_{clk2q} + T_{setup}$) over a DC-sweep of the on-die supply voltage.

Fig. 42 shows the minimum clock period of the system over a DC-sweep simulation, where the on-die supply voltage in Fig. 41 ($V_c$) was varied from 150mV to 1.2V. The minimum clock period ($T_{min}$) is calculated as the sum of the combinational circuitry delay propagation ($T_{combd}$), the delay between the output of the D-FF and the clock rising-edge ($T_{clk2q}$), and the D-FF setup time.

In addition, the maximum contribution to the total timing constraint comes from the combinational circuitry. As well, all of the three contributors to the minimum achievable clock period are inversely proportional to the supply voltage. With this result in mind, we will analyze in the following simulations how the result of a voltage glitch attack can be related to the underpower effect on the same system, considering the system's PDN model. Specifically, two cases will be considered: when the PDN time response is slower than the clock period and when the PDN time response is faster (or comparable) than the clock period.

**5.4.2. Case I: PDN Time Response Slower than Clock Period** The clock's frequency ($f_{clk}$) used was 1GHz for the first simulation, which sets $f_{clk} = 10\alpha$ ($\alpha$ is the damping attenuation factor in eq. (26)). Because the time response of the PDN is

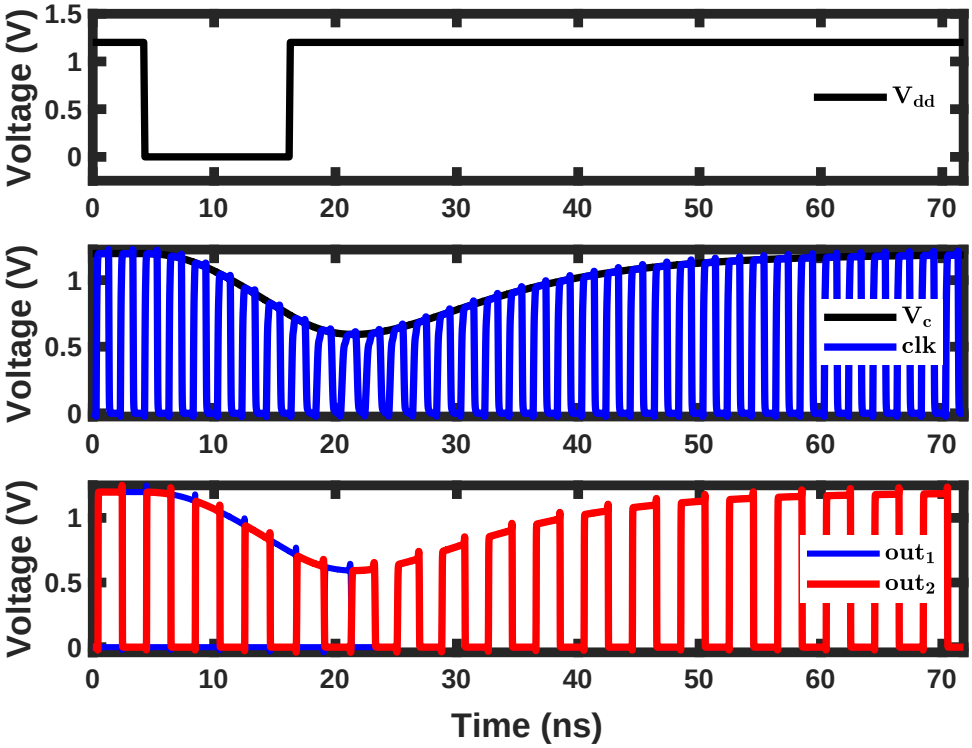


Figure 43. Simulation result at an operating clock frequency of 500MHz.

much slower than the clock period, the system can correctly execute several operations during the $V_c$ ramping down, until the latter reaches a minimum ($V_{min} = \min\{V_c\}$) where a fault occurs. Fig. 43 shows such a case, in which the 90° out of phase output signals ($\mathrm{out}_{1,2}$) get synced after a voltage glitch attack.

We repeated the same simulation presented in Fig. 43 but at different clock frequencies and using different glitch amplitudes, recording the minimum on-die voltage ($V_{min}$) for the minimum glitch duration that could induce a fault. Fig. 44 shows the results of the simulations. The simulated $V_{min}$ follows the same behavior as $f_{clk} = 1/T_{min}$, obtained as a function of the DC supply voltage in Fig. 42, almost independently of the glitch amplitude.[vi] With the latter, we show that voltage glitching attacks cause the same timing constraint violation effect on the system as underpowering, as long as the time response of the PDN is slower than the operating clock period.

Finally, Fig. 45 shows the minimum glitch duration versus the corresponding glitch amplitude that caused fault injections, which demarcates the potential fault injection

[vi] The deviation present in the results is because the $V_c$ voltages at the clock rising edges differ from one glitch amplitude to another.

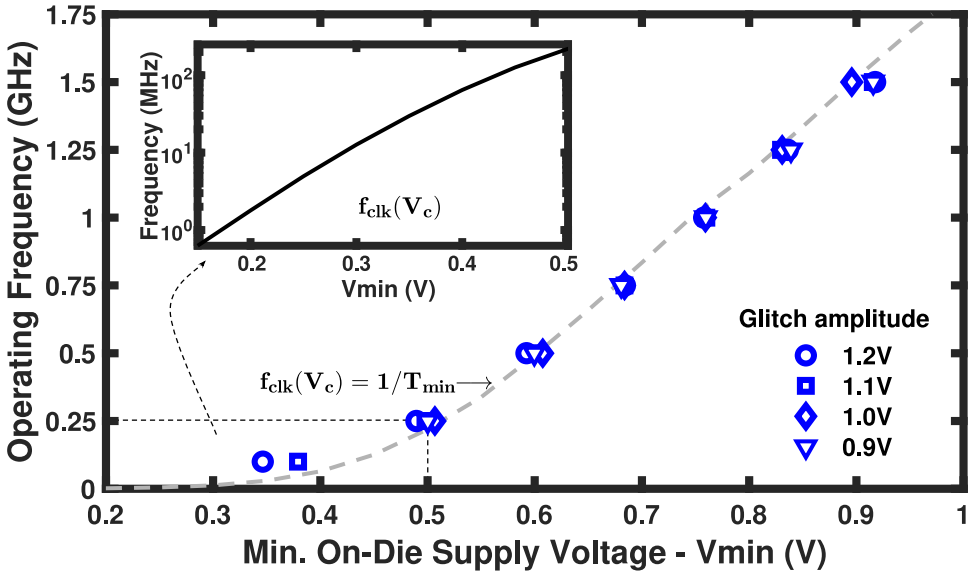


Figure 44. Simulation results of minimum on-die supply voltage ($V_{min}$) after a glitch attack (X-axis) at different operating clock frequencies (Y-axis). Different voltage glitch amplitudes tested.

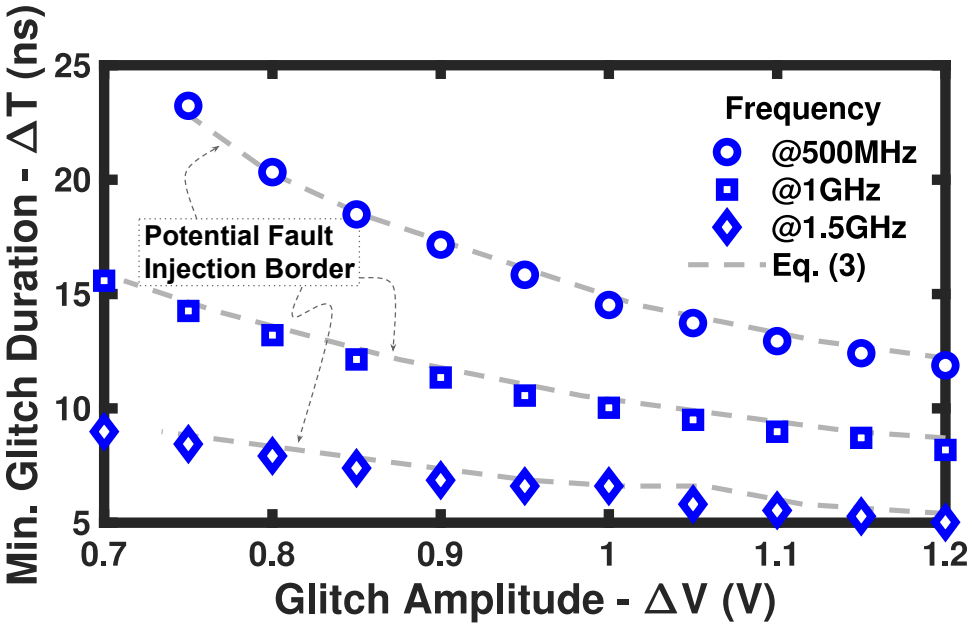


Figure 45. Potential fault injection borders: Simulation results of the minimum glitch duration against glitch amplitude.

borders, as presented in section 5.3.1. We show the results for three different operating frequencies (different $V_{min}$) and compare the simulated results with the expected outcomes when using eq. (27). The expected values are in agreement with the simulations, corroborating the analysis presented in section 5.3.1.

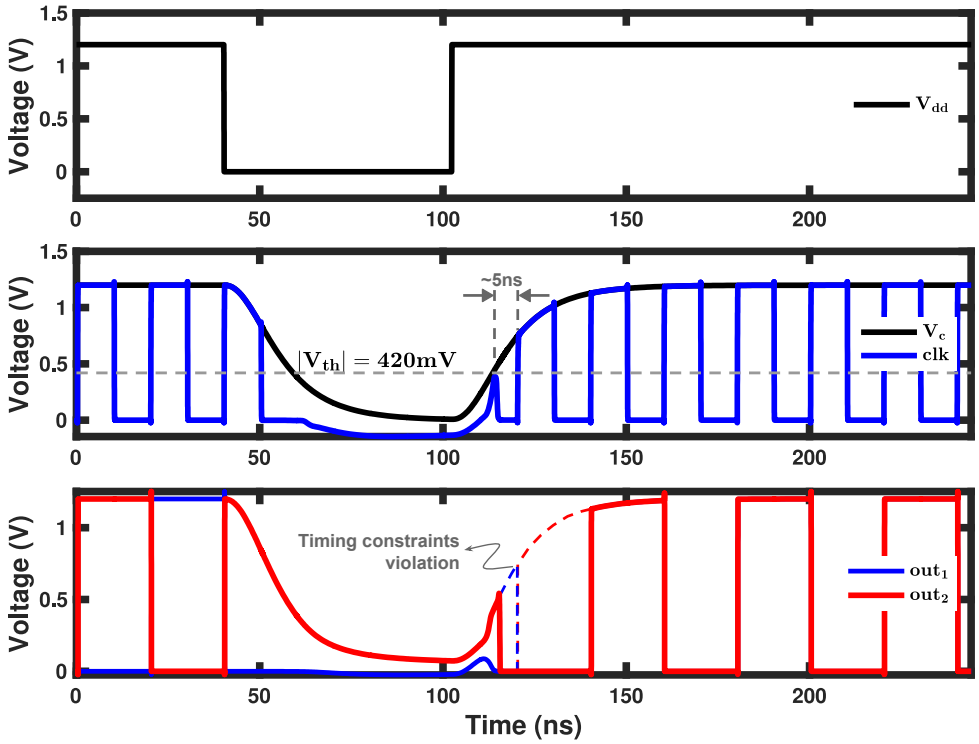


Figure 46. Simulation result at an operating clock frequency of 50MHz.

**5.4.3. Case II: PDN Time Response Faster than Clock Period** The results shown in Fig. 44 follow the same tendency as $f_{clk} = 1/T_{min}$ down to certain frequencies (close to 100MHz) and with $V_c$ around the transistors' threshold voltage ($|V_{th_{\{n,p\}}}| \approx 420mV$). At this point, the underpowering and voltage glitching effects are no longer similar, as the expected $V_c$ from the DC-sweep and the obtained $V_{min}$ value after a glitch attack have a significant deviation (around 100mV at 100MHz in Fig. 44). The latter is because the PDN time response starts to be comparable to the clock period around an operating frequency of 100MHz, and the prediction from Fig. 44 and eq. (27) does not hold any longer.

We carried away a second set of simulations to understand the behavior of a system using a clock period comparable or slower than the PDN time response. Fig. 46 shows the simulation results for a clock frequency $f_{clk} = 50$MHz, or equivalently, $f_{clk} = \alpha/2$. Since the time response of the PDN is faster than the clock period, the supply voltage reaches its lower value so fast that there is not even a second clock rising edge for a fault to occur (the system does not execute any operation). The supply voltage goes low, and the digital circuitry stops its operation without entering a reset state. As shown

in section 5.3.2, depending on the duration of the glitch, the memory-based cells (e.g., flip-flops, latches, SRAM, etc.) preserve the previous state without bit flipping. Hence, the only possibility for a fault injection is during the ramp-up of the supply voltage.

During the supply voltage ramp-up, and before reaching the transistors' threshold voltage, the output of the last inverter in the clock's buffer chain (Fig. 41) starts to rise with $V_c$. Some edge-driven circuitry may count this as a clock rising edge, and some may fail. This is the case in Fig. 46 where the $\mathrm{out}_2$ D-FF recognizes the the clock's edge, while the $\mathrm{out}_1$ D-FF fails.

The latter is sufficient to sync both signals. Still, there is another effect within the supply ramp-up that can cause a fault injection. Since the unintended clock rising edge is very close to the next rising edge and because the supply voltage is still low,[vii] timing constraint violations may occur, which is also the case in the simulation presented in Fig. 46. Furthermore, if the start of the glitch is synchronized with the clock, the glitch duration is always close to a multiple of the clock period (or half a clock period, if there are digital circuitry driven by the clock negative edge) because it is in the vicinity of the clock edges where the fault injection can occur. All of the experiments performed for this second set of simulations showed the same mechanism for fault injection as the one presented in Fig. 46.

As a summary, Fig. 47 shows the obtained minimum glitch duration that can cause a fault injection versus the operating frequency (ranging from 10MHz to 1.5GHz). Two regions are clearly distinguished within the figure, which corresponds to both cases studied in this section. In the right region (case I), it is possible to see how the simulated data follow the behavior of eqs. (27),(28) and (29).[viii] As stated before, this behavior is only achievable if the PDN time response and the clock period are comparable, which for the simulated circuit is around 100MHz. Below 100MHz, the system behaves as in case II, and the glitch duration starts to be a multiple of the clock period (even half clock period), as evidenced by the lines $1/f_{clk}$, $2/f_{clk}$, and $4/f_{clk}$.

[vii] The effective clock period during the ramp-up is 5ns, equivalent to 200MHz. According to Fig. 44, the maximum frequency at 420mV ($V_{th}$) is around 100MHz, hence the timing violation constraints.

[viii] We used the obtained data of the $f_{clk}$ vs. $V_{min}$ relation in Fig. 44 to calculate the operating frequency from a $V_{min}$ caused by a glitch duration $\Delta T$ and glitch amplitude $V_{DD}$.

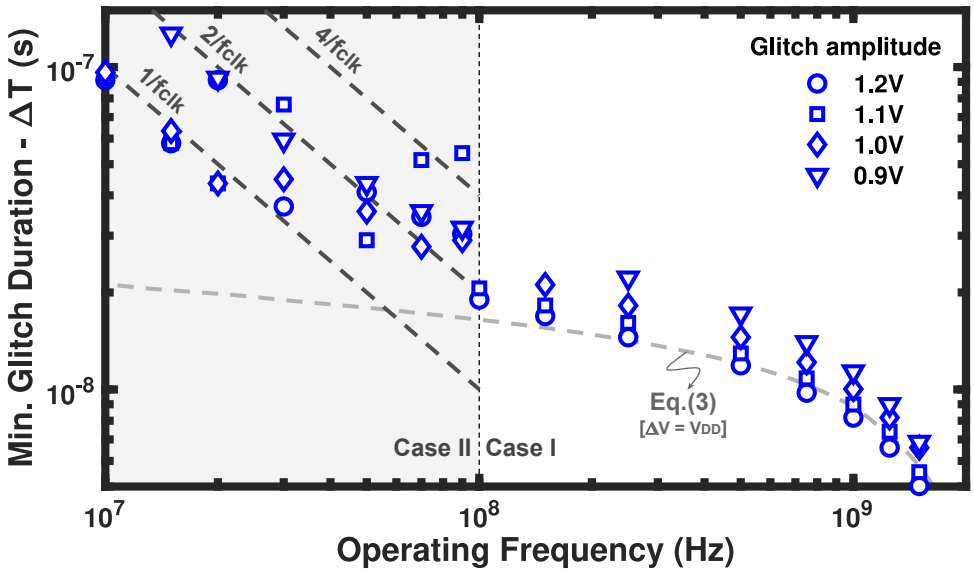


Figure 47. Minimum glitch duration against operating frequency. The cases analyzed in this section are presented to show their behavior.

### 5.5. Experimental Results

In this work, the microcontroller unit (MCU) under test/attack is a 32b RISC-V core[138]. The MCU contains a 4KB SRAM, a 10b ADC, a 12b DAC, 8 GPIO, and two SPI interfaces (master and slave). All the modules are connected using two different buses: AXI4 and APB. The MCU supports a maximal clock frequency of 100 MHz, and the supply core voltage is 1.2V. The core has a two-stage instruction pipeline (i.e., the fetch and execute stages).

In the following set of experiments, we will refer to successful glitch attacks as those that produce a fault injection, such as an undesired jump instruction within the MCU's program. These types of attacks may allow skipping security mechanisms, such as user-password routines. We put the MCU in an infinite loop to emulate a situation where a program awaits indefinitely (e.g., for a user entry). The only possible way to exit the infinite-loop would be through a fault injection, which in our case occurs through a voltage glitch attack.

[138] C. DURAN et al. "A 32-bit RISC-V AXI4-lite bus-based microcontroller with 10-bit SAR ADC". in: *2016 IEEE 7th Latin American Symposium on Circuits Systems (LASCAS)*. 2016, pp. 315–318.

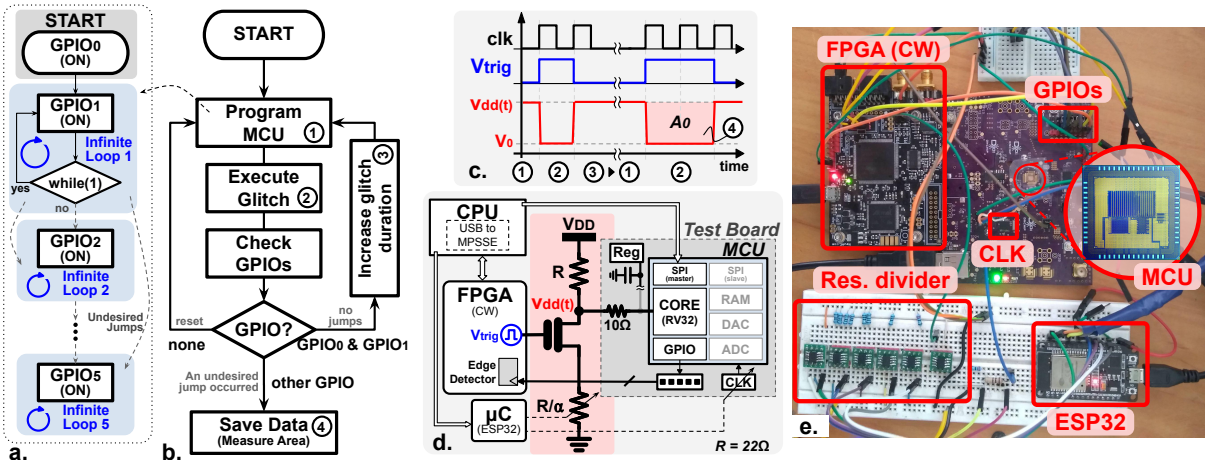


Figure 48. Experiment setup: a) Block diagram of the program written in the MCU's memory; b) Block diagram of the experiment routine; c) Example of the voltage signals during the experiment routine; d) Voltage glitch attack circuit diagram; e) Physical implementation of the experimental setup.

**5.5.1. Experimental Setup** Fig. 48a shows the program written in the MCU's memory. Initially, the GPIO0 goes 'high' while the others are 'low'. Immediately, the program goes into an infinite loop and sets the GPIO1 high. In normal conditions, GPIOs '0' and '1' would be the only ones to be in a high-state indefinitely. The program has other four infinite loops to check the effect of a glitch attack, so if there is an undesired instruction jump (or another type of fault injection, e.g., register corruption), the GPIOs '2' to '5' turn high as well.

Fig. 48b illustrates the routine used to perform the experiments. First, the FPGA is initialized, and the MCU is programmed. A "ready" signal goes out from the MCU, and a glitch attack is executed on the MCU's supply. Then, the routine checks which GPIOs are 'high': if only the GPIOs '0' and '1' are 'high', the glitch duration is increased by one clock signal period in each iteration (see Fig 48c); if any of the other GPIOs between '2' to '5' are 'high', it means a fault injection occurred, and the glitch signal is saved; otherwise, the MCU goes to reset, and the process restarts with the same glitch duration.

We used the circuit diagram in Fig. 48d, which represents the actual experimental setup in Fig. 48e, to execute the routine and the glitch attacks. The experiment routine runs in a computer that controls all the components of the setup. The computer writes the infinite-loops program to the RISC-V MCU's memory (communicates through the

MCU's serial-parallel interface) and controls the FPGA and an external microcontroller (ESP32). The function of the FPGA is to send a trigger signal to activate the glitch attack. The FPGA also senses the GPIOs' states (i.e., 'high' or 'low') through an edge detector. The external microcontroller configures a digitally variable resistor within the glitch generator circuit and a PLL to set the clock frequency of the RISC-V MCU.
The glitch generator circuit in Fig. 48d is composed of a resistor divider with a power NMOS transistor triggered by the glitch signal coming from the FPGA. The output of the glitch generator sets the MCU's supply voltage $v_{dd}(t)$. Since the MCU's test-board has a voltage regulator and coupling capacitors, they were disconnected before injecting the glitch attack. While the FPGA controls the duration of the glitch attack (proportional to an MCU's clock period), the digitally-variable resistor helps to obtain different glitch amplitudes in the MCU supply voltage, following:

$$V_0 = \frac{V_{DD}}{1 + \alpha_r}, \tag{30}$$

where $\alpha_r$ is an integer number between 1 to 15. The $\alpha_r$ binary number triggers the switches to connect or disconnect several resistors in parallel. An additional switch that connects directly to the ground allows a glitch amplitude down to 0V.

**5.5.2. PDN Equivalent Impedance Measurement** For the measurement of the power delivery network equivalent impedance, we used the resistor connected in series in Fig. 48d and measured the voltages at its nodes. Fig. 49 shows the measurement of the magnitude and phase of the PDN impedance at different frequencies.
We used the circuit in Fig. 37 to fit a lumped model, which is also plotted in Fig. 49. Table 9 shows the PDN circuit elements. The calculated damping factor $\zeta$ of the PDN is 0.63, which puts the MCU under attack in the underdamped case analyzed in section 5.3.1. The resonance frequency ($w_0$), the damping attenuation factor ($\alpha$), and the damped resonance frequency ($w_d$) are also shown in Table 9.

**5.5.3. Voltage Glitching Measurement Results** We performed a set of experiments where the MCU was attacked with different glitch amplitudes (using the circuit

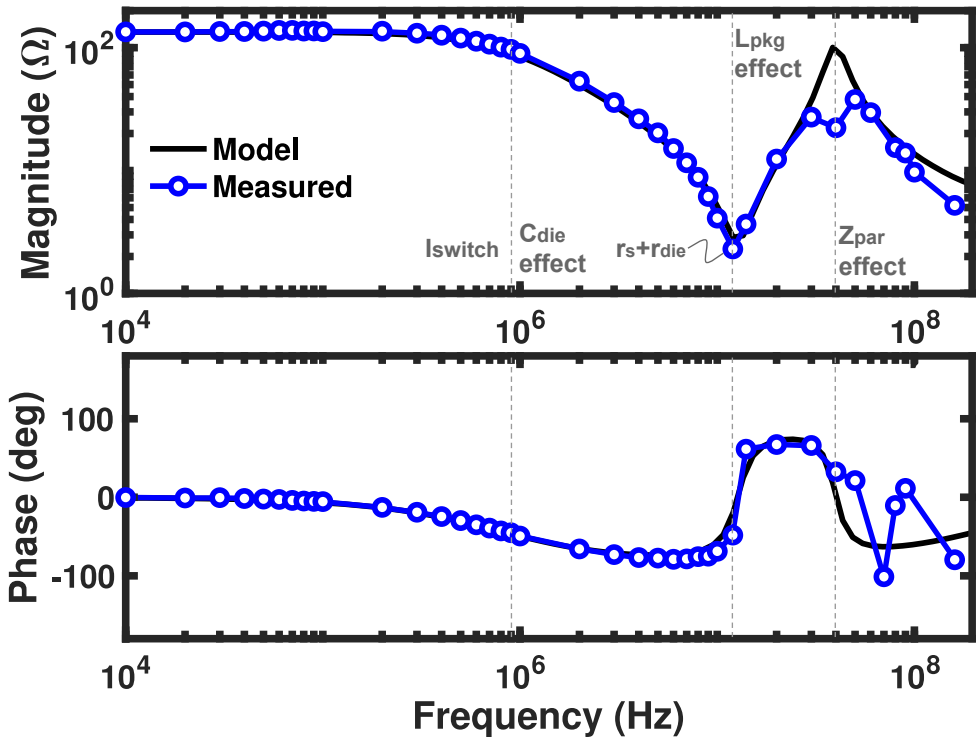


Figure 49. Measured PDN impedance and modeled impedance using the circuit in Fig. 37.

Table 9. PDN circuit equivalent elements (Fig. 37 and eq. (26))

| $C_{die}$ | $R_{eq}$ | $L_{pkg}$ | $R_{switch}$ |
|---|---|---|---|
| 1.32nF | 12Ω | 120nH | 132Ω |
| $\zeta$ | $w_0$ | $\alpha$ | $w_d$ |
| 0.63 | 79.5Mrad/s | 50Mrad/s | 61.8Mrad/s |

in Fig. 48 and equation [30]) while working at different operating frequencies. Only the voltage glitch signal of successful attacks was measured for around 4500 experiments. Fig. 50 shows examples of the applied glitches.[ix]

Fig. 51 shows the measured glitch signal duration against the MCU's operating frequency. According to the results presented in section 5.5.2, the system should behave as in case II for operating frequencies lower (or equal) than 50MHz ($\alpha = 50$Mrad/s), where the PDN time response is faster than the clock period (see section 5.4).

Because the MCU has negative and positive edge-driven flip-flops, the minimum glitch

[ix] Something to notice from these glitches is the ringing present in the waveforms. The latter is due to the resonant frequency formed by the PCB and off-chip components.

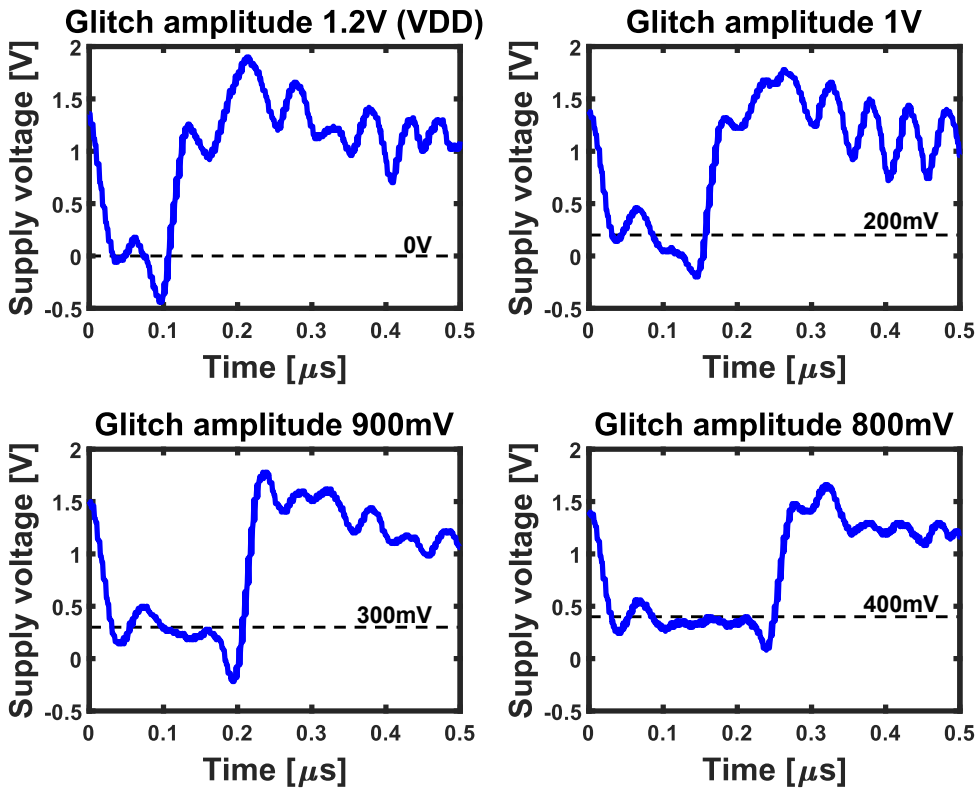


Figure 50. Examples of glitches at the supply voltage during a fault attack at different amplitudes.

duration in Fig. 51 (blue stars) followed a multiple of $0.5/f_{clk}$. The minimum glitch duration starts to deviate from the $0.5/f_{clk}$ value (gets higher, almost settling to a value) when the operating frequency is closer to 50MHz, which is the limit frequency to transition from case II to case I (PDN time response slower than clock period). Unfortunately, the system in Fig. 48e does not support higher frequencies than 50MHz, and case I was not experimentally tested in this work. Nevertheless, remember that in section 5.3.1 we showed how the evidence in[120] ,[130] endorses the analysis presented for case I.

The results in Fig. 51 corroborate the analysis presented in sections 5.3.2 and 5.4,[x] as well as the simulation results shown in Fig. 47, where the minimum glitch duration was close to the clock period, or a multiple of it (the simulated system only had positive edge-driven D-FF).

[x] The section 5.4 analysis can be used to predict what is the minimum possible glitch duration for voltage glitching fault injection, not to establish the most expected duration for a glitch attack. The latter is evidenced by the median values in Fig. 51 (black markers), which differ almost by one (or one and a half) clock period from the minimum duration.

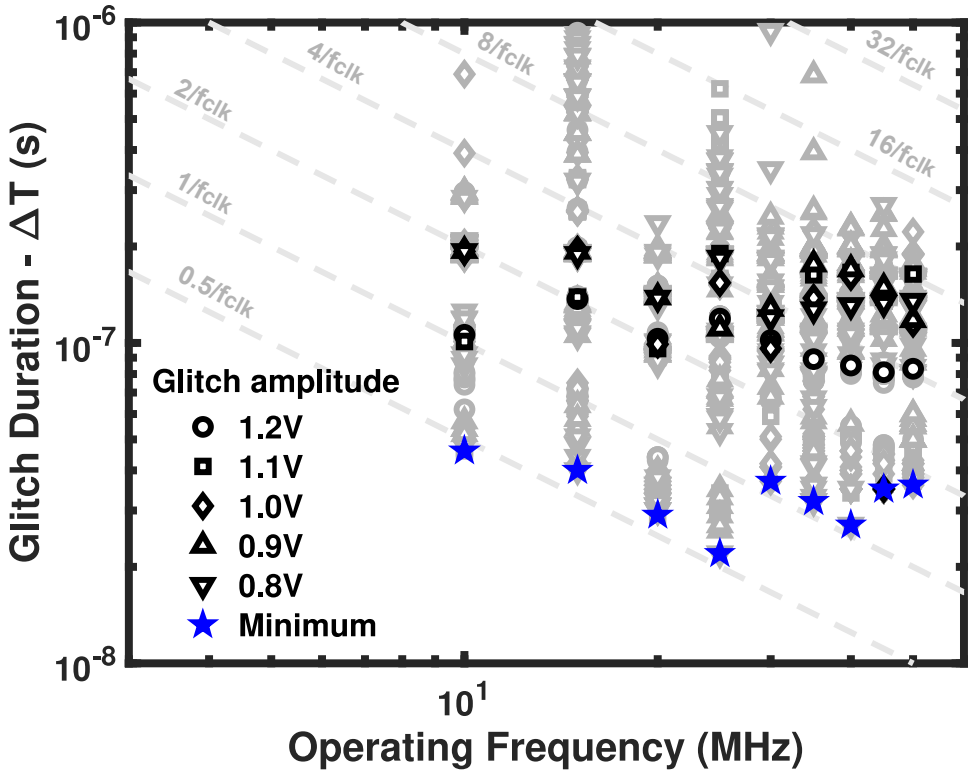


Figure 51. Measured glitch signal duration for successful fault injection. The glitches are grouped by glitch amplitude. There are around 80-100 samples in each frequency and glitch amplitude. Black markers highlight the median at each amplitude, while the blue stars highlight the minimum glitch duration.

### 5.6. Conclusion

In this chapter, we included the power delivery network of an SoC in the classical timing constraint violation approach, which has been historically neglected in the literature. The inclusion of the PDN in the analysis allowed us to establish a relation between the potential of a voltage glitch to inject a fault into a system and the glitch waveform parameters (e.g., duration, amplitude).

We found two scenarios for effective voltage glitching fault injections, depending on the relation between the PDN time response and the system's operating frequency. In the first scenario (case I, PDN time response slower than the clock period), we showed analytically, and through simulations, that voltage glitching attacks cause the same timing constraint violation effect on the system as underpowering. Furthermore, the analytical relationship between the glitch amplitude and the glitch duration (potential fault injection border) obtained in this chapter matches with experimental results behavior

presented in previous works in the literature. By using measurement results and our analysis, an attacker can find useful information about the system through curve fitting of the potential fault injection border. From the victim's point of view and in terms of a system's fault characterization, our analysis may allow a system characterization for any glitch waveform, even glitches generated by genetic algorithms or neural networks. On the other hand, when the PDN time response was faster than the clock period (case II), the supply voltage reached its lower value before executing a single other operation, suggesting that the fault injection is only possible during the supply ramp-up. During the temporary system shutdown due to voltage glitching, we analyzed how memory-based cells (e.g., SRAM and D-FF) retain the previous state if the glitch does not last for a long time. Based on this analysis, we performed a set of simulations where the system under attack was in case II. Additionally, we executed several fault injection (namely, instruction jumps) experiments in an in-house RISC-V MCU, also in case II. The data obtained out of 4500 experiments across multiple glitch voltages and operating frequencies, and the simulation results, showed a pattern in the minimum possible glitch duration, with the latter being a multiple of the clock period (or half of the period for systems with negative and positive edge-driven flip-flops).

# 6. CONTRIBUTIONS AND CONCLUSIONS

Several conclusions raise from the work done in this dissertation. They are compiled and explained in this chapter.

## 6.1. Contributions Summary

The summary of the key contributions of this dissertation is described as:

- A-Connect: a novel methodology to improve neural network resilience against stochastic variability when deploying neural networks in imprecise analog accelerators.
- The development of a public Keras/Tensorflow library, with versions of fully-connected and convolutional layers using A-Connect.
- The proposal of a wide frequency range and high energy efficiency CIM SRAM-based ML macro for multi-mode systems-on-edge.
- The proposal of an end-to-end analog datapath that incorporates not only MAC operations but commonly used ML operations within the analog domain (e.g., ReLU, normalization, memory).
- The proposal of ultra low-power multi-level voltage monitors for multi-mode fine-grained power management strategies.
- The presentation of experiments showcasing how the proposed voltage monitors could be used in a real power management strategy.
- The inclusion of the power delivery network of an SoC in the classical timing constraint violation approach.
- The analytical relation between the potential of a voltage glitch to inject a fault into a system and the glitch waveform parameters (e.g., duration, amplitude).

## 6.2. Conclusions

As described in the Introduction (chapter 1), the project goal was:

*To devise solutions in unconventional domains (analog-mixed signal) to some of*

*the conventional problems regarding IoT challenges, as well as to envision the rise of new challenges in secure and efficient systems-on-edge.*

Since we divided the main goal into three objectives, I decided to present the conclusions of my work by encasing them into one of the three thesis' objectives, as follows:

*To explore the feasibility of applying in mixed-signal accelerators traditional computer architecture and machine learning techniques.*

In regards to improving the energy efficiency of systems-on-edge with decision-making capabilities, we explored machine-learning accelerator architectures in the analog-mixed signal domain. Trying to propose an analog-based ML macro accelerator was a great challenge since analog computation is susceptible to hardware stochastic variability, incurring in limited signal-to-noise and aggravating for compact and low power applications. Hence, before any hardware proposal, we decided to explore a co-design software strategy.

We introduced A-Connect, an *ex situ* statistical methodology to improve analog neural network resilience against stochastic variability, enabling energy-efficient and compact imprecise accelerators. This methodology enables, for instance, emergent memory technologies as ReRAM and PCM for accurate computation-in-memory applications. Furthermore, we developed a Keras/Tensorflow library, with versions of fully-connected and convolutional layers using A-Connect. The library can be coupled to standard machine learning platforms in a straightforward manner. We presented simulation results applying the A-Connect methodology to popular DNNs, such as LeNet-5 for MNIST dataset, AlexNet, VGG-16, and ResNet-20 for the CIFAR-10 dataset, and ResNet-18 for the CIFAR-100 dataset. The experimental evidence compiled in this work showed that the proposed methodology significantly outperforms other *ex situ*, while achieving similar performance than *in situ*, and hybrid approaches to mitigate stochastic variability in the literature. The A-Connect methodology showed an improvement over baseline models of around 15 to 68 percentage points for the median accuracy at a 70% of stochastic variability. The deviation of the results with A-Connect is around 20X lower than the baseline at this level of stochasticity.

Once we tackled the problem of hardware stochastic variability in analog-based neural network accelerators with the A-Connect methodology, we continue with our hardware implementation proposal. We proposed a wide frequency range and high energy efficiency CIM SRAM-based ML macro for multi-mode systems-on-edge. The analog macro was able to perform at high energy efficiency by following two principles: avoiding data conversion by staying in the same physical domain (i.e., current), and the use of simplified and low-area circuits by using co-design software strategies that mitigate stochastic and deterministic errors (i.e., the A-Connect methodology). We proposed an end-to-end analog datapath that incorporates not only MAC operations but commonly used ML operations within the analog domain, such as ReLU and scaling (the latter enabled normalization operations), as well as memory capabilities for pipeline execution. The simulation results presented in a 180nm design showed that the analog macro performed at a wide range of frequencies (200kHz-15MHz) over ultra-low and broad range of current levels (i.e., 1nA to 100nA biasing), while maintaining a relatively similar energy efficiency (760-1076 1b-TOPS/W). When compared to other works, the proposed macro's results were compatible with state-of-art macros in 65nm. Furthermore, we showed performance estimations for a 28nm design that put the proposed analog macro above absolute state-of-art performance. To our knowledge, our work is the only study investigating multi-mode ML accelerators performing efficiently at different current levels and clock rates.

*To devise mixed-signal power management strategies for energy-efficient systems-on-edge.*

Coming back to improving the energy efficiency of systems-on-edge, multi-mode fine-grained power management system-on-chip (SoC) offer a promising approach for achieving ultra-low power consumption in energy autonomous and battery supplied applications at the edge of Internet-of-Things (IoT). The SoC Power Management Unit (PMU) implements these techniques to obtain a more precise control over power consumption at the subsystem and component level, enabling systems to operate more efficiently.
In this work, we presented ultra low-power multi-level voltage monitors for multi-mode

fine-grained power management strategies. Simulation results over PT variations, as well as measurements at nominal temperature, showed a robust performance within the industrial temperature range from -40°C to 125°C and a wide supply rise and falling times, ranging from 1us to 1s. In a first version of the POR (POR1), we managed to obtain a current consumption of 7$\mu$A. In the second version (POR2) the POR had a nominal current consumption of 19nA. Both PORs had up-to 3 different voltage threshold levels. In regards to the BOD, we presented an architecture with low-temperature slew compensation for low power applications, multiple voltage threshold levels, and a current consumption of 200nA. We also showed experimentally how these voltage monitors could be used in a real power management strategy. By having multi-level voltage thresholds we enabled three different power modes that used lower voltage supply: active, sleep, and deep-sleep. According to measurements, the SoC had an 8mA current consumption at 16MHz in active mode, 27.5$\mu$A at 32.768kHz in sleep mode, and 530nA at 32.768kHz in deep-sleep mode. In comparison to previous research that neglect to consider the low-temperature effects when using large impedance branches, this work achieved a low current consumption even by considering these temperature effects. Current consumption, programmability, and reduced area, makes the proposed voltage monitors enablers of different fine-grained power management schemes.

*To study the impact of power supply glitching as a way to infringe systems-on-edge security.*

In terms of security, we decided to focus our efforts on the study of unconventional hardware security infringement in SoC since previous research in our group OnChip studied more conventional software-based attacks. We decided to go this path because the amount of interconnected devices in today's IoT applications is so overwhelming that any front should be protected, even those that appear to be unusual.

Following this line of thought, we investigated power supply glitching, one of the most researched fault injection mechanisms in SoC at a hardware level. The easiness of execution and permanent availability of an external pin for the power supply, make glitching injection one of the preferred methods for security tampering. Although, pre-

vious works have provided experimental evidence demonstrating that voltage glitching fault injections cause time constraint violations, there is still a lack of understanding of the voltage glitching nature. The latter has prevented obtaining a direct link between the glitch characteristics and the likelihood of the glitch injecting a fault into a system.
In our work we managed to include the power delivery network of an SoC in the classical timing constraint violation approach. The inclusion of the PDN in the analysis allowed us to establish a relation between the potential of a voltage glitch to inject a fault into a system and the glitch waveform parameters (e.g., duration, amplitude).
We found two scenarios for effective voltage glitching fault injections, depending on the relation between the PDN time response and the system's operating frequency. In the first scenario (case I, PDN time response slower than the clock period), we showed analytically, and through simulations, that voltage glitching attacks cause the same timing constraint violation effect on the system as underpowering. Furthermore, the analytical relationship between the glitch amplitude and the glitch duration (potential fault injection border) obtained in this paper matches with experimental results behavior presented in previous works in the literature. By using measurement results and our analysis, an attacker can find useful information about the system through curve fitting of the potential fault injection border. From the victim's point of view and in terms of a system's fault characterization, our analysis may allow a system characterization for any glitch waveform, even glitches generated by genetic algorithms or neural networks.
On the other hand, when the PDN time response was faster than the clock period (case II), the supply voltage reached its lower value before executing a single other operation, suggesting that the fault injection is only possible during the supply ramp-up. During the temporary system shutdown due to voltage glitching, we analyzed how memory-based cells (e.g., SRAM and D-FF) retain the previous state if the glitch does not last for a long time. Based on this analysis, we performed a set of simulations where the system under attack was in case II. Additionally, we executed several fault injection (namely, instruction jumps) experiments in an in-house RISC-V MCU, also in case II. The data obtained out of 4500 experiments across multiple glitch voltages and operating frequencies, and the simulation results, showed a pattern in the minimum

possible glitch duration, with the latter being a multiple of the clock period (or half of the period for systems with negative and positive edge-driven flip-flops).

### 6.3. Suggestions for Future Research

This subsection presents some suggestions for future research on the topics we dealt with in this thesis.

- Although we managed to design and simulate the ML macro, we did not fabricate it. One of the major reasons (besides external impediments, such as the pandemic and its consequences) was the complexity required for the communication between the macro/accelerator and the external world, or even a third party. In fact, ML macro/accelerator communication has been one the most studied subjects in analog-based CIM accelerators since it constitutes one of the most challenging data bottlenecks.

  During the execution of the project, we explored two possibilities that were not implemented. The first one was inspired by previous work on the OnChip's group where we accomplished AES acceleration through RISC-V custom instructions. In fact, the analog datapath proposed in this work was conceived from the beginning for such purposes, where each of its instructions would be added to a custom RISC-V MCU. Direct memory access (DMA) would have been a must for this implementation, just as it was for the AES acceleration case. The second was a "straightforward" standalone approach. The idea was to use a FIFO-based strategy where buffer memory was used for proper communication with the exterior. Either way, it was definitely not a simple task for testing purposes since we are talking about a single macro. Even implementing a single layer of a neural network in one macro would require several memory read/write procedures that would have obliged us to use data conversion in each cycle, which would not allow us to test the full potential of the proposed macro. We think that our proposal is better implemented in an accelerator configuration (i.e., multiple macros interconnected), but the latter itself is a great communication challenge.
- We also didn't manage to test the proposed voltage monitors in the different fine-

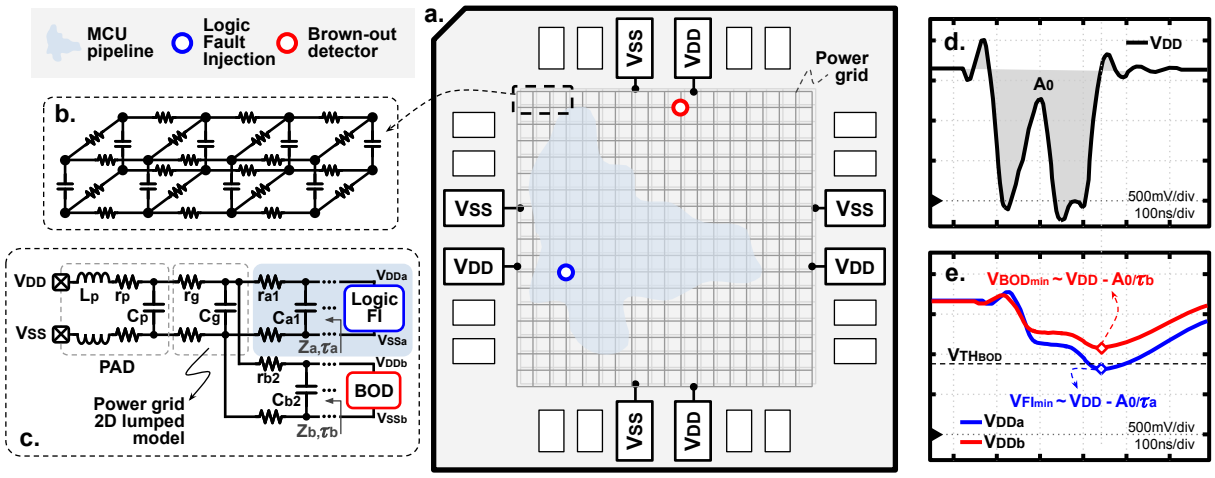


Figure 52. Propagation of voltage glitches through SoC power delivery network: a) SoC showing the PDN grid and the location of the MCU pipeline (light bluw shadow), the location of the BOD (red dot), and where the logic fault injection (FI) occurs (blue dot); b) 3D transmission line model of the supply ($V_{DD}$) and ground planes ($V_{SS}$); c) 1D version of the supply and ground lines for the BOD and the logic FI; d) Arbitrary voltage glitch waveform applied to the SoC's power supply; e) Voltage supply signals at BOD's (red line) and logic FI's (blue line) locations.

grained power management schemes discussed in chapter 4. Mainly, the global voltage monitor with multiple local voltage monitors for the different power domains is the most attractive approach to test, not only for energy-efficiency purposes but for security ones.

One of the main reasons for the low success rate of fault injections in commercial MCUs is the voltage monitors within these systems, namely brown-out detectors (BODs).[139]

Still, there have been studies where it was possible to bypass the voltage monitors. By using genetic algorithms, the authors in[125] were capable of generating arbitrary glitches waveforms, finding the adequate glitch's parameters, customizing the glitches to the attacked MCU. Unfortunately, the authors did not provide an explanation on how they were capable of injecting the faults while bypassing the BOD.

The purpose of Fig. 52 is to describe through an example how a BOD can be

[139] Voltage regulators are also responsible for the low success rate, up to certain extent. For example, there are not so many works with a high power supply rejection (PSR) at the frequencies relevant for a voltage glitch signal (Farid Uddin AHMED et al. "A Brief Overview of On-Chip Voltage Regulation in High-Performance and High-Density Integrated Circuits". In: *IEEE Access* 9 [2021], pp. 813–826). Furthermore, the PSR only reflects the capability of the regulator to reject small signals, while voltage glitches are large signals.

bypassed during a glitch attack, even with the assumption that the BOD is fast enough to respond to the glitch. Fig. 52a shows the hypothetical case of an SoC, its pads, and its power delivery network (power grid). In this hypothetical SoC, we assume the location of the BOD (red circle), the location of the microcontroller's logic (light blue shape), and the location where the logic fault occurs (blue circle). Since the BOD and some parts of the MCU are not physically close, the supply and ground transmission lines may differ enough for fault injection. The circles in Fig. 52a are only a mere representation of possible locations for the BOD and the logic fault. The BOD might be as well at any other position, but then, another part of the logic might be erratic as well.

Understanding how the glitch propagates through different paths within the SoC is not an easy task. Considering the duration of a conventional glitch attack, and the power integrity of the supply ($V_{DD}$) and ground planes ($V_{SS}$), one could use the 3D transmission line model for $V_{DD}$ and $V_{SS}$ presented in Fig. 52b. Just obtaining the actual model of the power grid is a complex task if one considers that all the intersections are not symmetric (i.e., the resistors and capacitors are not the same at every power grid intersection). Furthermore, the multi-metal structure of silicon dies makes the modeling of such transmission lines even more complicated.

Still, a simpler model can give us the means necessary to understand what may happen when a glitch propagates through the chip. Fig. 52c shows the simplified 2D version of the supply and ground lines for two different paths within the SoC: one for the BOD and the other for the logic where the fault occurs. The sections shared by both paths are the supply and ground pads (modeled with a lumped RLC network) and part of the power grid, which corresponds to the top-metal layers interconnections (modeled with a lumped RC network).

By applying an arbitrary glitch waveform to the $V_{DD}$ and $V_{SS}$ nodes, like the one in Fig. 52d, the responses depicted in Fig. 52e could be obtained. The red and blue curves are possible outcomes of the glitch propagated through the power lines down to the BOD and the location of the logic fault, respectively. $V_{TH_{BOD}}$

sets the minimum voltage where the logic works correctly and the BOD sends a reset signal. Hence, Fig. 52e represents the case where the logic sees a voltage drop that leads to a fault injection, but the BOD does not detect any drop.

Finding the proper glitch waveform that can propagate without detection through 3D multi-nodal power lines is a complicated problem. This type of problem is suited for genetic algorithms or neural networks, which can find the adequate glitch for a specific MCU in an offline fashion,[140] and then attack the actual system so that the case presented in Fig. 52e can occur.

There would probably be more than one glitch waveform that could induce a fault injection due to the number of variables involved in such a problem (even in the presence of voltage monitors). For example, it can depend on where the fault (e.g. instruction jump, data corruption) is injected within the MCU. Taking actions against innumerable glitch waveforms is a near impossible task. Instead, it may be possible to predict if a glitch has the potential to inject a fault depending on its integral or its double-time integral.

With the latter in mind, rather than looking at the shape of the glitch in Fig. 52d, one could look at the area ($A_0$) under the glitch, or at the double-time integral ($A_1$, equation [**??**]) of the glitch, depending on the system's damping factor ($\zeta$). According to equation (27), or (**??**), and using the model in Fig. 52c with the time constants $\tau_a$ and $\tau_b$, or the resonant frequencies $w_a$ and $w_b$, it is possible to relate the minimum voltages at the end of the glitch seen by the BOD ($V_{BOD_{min}}$) and the logic where the fault injection occurs ($V_{FI_{min}}$):

$$V_{BOD_{min}} \approx V_{DD} - \kappa_{a,b} \cdot (V_{DD} - V_{FI_{min}}) \tag{31}$$

where $\kappa_{a,b} = \tau_a / \tau_b$ for $\zeta \geq 1.74$, and $\kappa_{a,b} = w_b^2 / w_a^2$ for $\zeta < 1.74$.

Equation (31) highlights the relation between the position of the BOD within the SoC and its capacity to detect a fault injection at any location of the chip. A

[140] The attackers can perform tests on an external MCU of the same model. The attack must have a certain level of repeatability for this offline approach to work, as pointed in (Claudio BOZZATO et al. "Shaping the Glitch: Optimizing Voltage Fault Injection Attacks". In: *IACR Transactions on Cryptographic Hardware and Embedded Systems* 2019.2 [Feb. 2019], pp. 199–224).

straightforward approach to tackle glitch attacks with arbitrary waveforms would involve a distributed network of BODs positioned at different locations within the SoC. The number of BODs and their locations would be determined according to the power grid topology. This topology is obtained considering internal components, such as decoupling capacitors and key blocks' positions and load profiles. This methodology could be merged into the power integrity signoff of an SoC in a sort of security integrity analysis against glitch attacks.[141]

In the end, there is not a definitive solution to avoid MCU fault injections through glitch attacks. A security integrity analysis to set a network of BODs, would hypothetically reduce the success rate of glitch attacks (even those generated through neural networks or genetic algorithms), but eventually, an attacker may compromise the security. Instead of just shielding against attacks, a better alternative might be to understand the vulnerabilities of the system regarding fault injection, and in the process, learn how to counteract them.

The authors in[137] show a full characterization of faults injected through electromagnetic pulses on modern CPUs. With this characterization, they could identify the types of faults injected (e.g., instruction jump, data corruption), as well as the location of these faults (e.g., pipeline, registers, and memory).

## 6.4. Publications and Patents

### Journal Articles

**L. E. Rueda G.**, R. Vergel, E. Silva, E. Roa, "A-Connect: an ex-situ Training Methodology to Mitigate Stochasticity in Neural Network Analog Accelerators," in IEEE Transactions on Circuits and Systems–I: Regular Papers (TCAS-I), *In final production stage with IEEE Publishing Operations*, 2023, doi: 10.1109/TCSI.2023.3273188.

R. Torres, E. Roa and **L. E. Rueda G.**, "On the Design of a Reliable Current Reference for Systems-on-Chip," in Wiley International Journal of Circuit Theory and Applications, vol. 49, no. 7, pp. 2032–2046, April 2021, doi: 10.1002/cta.2955.

[141] The amount, the position, and the values of the decoupling capacitors may help not only the power integrity but also the security integrity of the SoC.

**Conference Proceedings**

C. Duran, M. Wachs, **L. E. Rueda G.** et al., ”An Energy-Efficient RISC-V RV32IMAC Microcontroller for Periodical-Driven Sensing Applications,” 2020 IEEE Custom Integrated Circuits Conference (CICC), Boston, MA, USA, 2020, pp. 1-4,
doi: 10.1109/CICC48029.2020.9075877.

R. Torres, **L. E. Rueda G.**, N. Cuevas and E. Roa, ”On the Design of Reliable and Accurate Current References,” 2020 IEEE 11th Latin American Symposium on Circuits & Systems (LASCAS), San Jose, Costa Rica, 2020, pp. 1-4,
doi: 10.1109/LASCAS45839.2020.9069041.

**L. E. Rueda G.**, J. S. Moya B. and E. Roa, ”A Compact Industrial-Grade Multi-Threshold Brown-Out Detector,” 2019 26th IEEE International Conference on Electronics, Circuits and Systems (ICECS), Genoa, Italy, 2019, pp. 923-926,
doi: 10.1109/ICECS46596.2019.8964634.

J. Santamaria, N. Cuevas, **L. E. Rueda G.**, J. Ardila and E. Roa, ”A Family of Compact Trim-Free CMOS Nano-Ampere Current References,” 2019 IEEE International Symposium on Circuits and Systems (ISCAS), Sapporo, Japan, 2019, pp. 1-4, doi: 10.1109/ISCAS.2019.8702294.

**L. E. Rueda G.**, N. Cuevas and E. Roa, ”An Ultra-Low Power Multi-Level Power-on Reset for Fine-Grained Power Management Strategies,” 2019 IEEE 10th Latin American Symposium on Circuits & Systems (LASCAS), Armenia, Colombia, 2019, pp. 185-188, doi: 10.1109/LASCAS.2019.8667574.

A. Amaya, **L. E. Rueda G.** and E. Roa, ”A Multi-Level Power-on Reset for Fine-Grained Power Management,” 2018 28th International Symposium on Power and Timing Modeling, Optimization and Simulation (PATMOS), Platja d’Aro, Spain, 2018, pp. 129-132, doi: 10.1109/PATMOS.2018.8464167.

C. Duran, ... **L. E. Rueda G.** et al., ”A system-on-chip platform for the internet of things featuring a 32-bit RISC-V based microcontroller,” 2017 IEEE 8th Latin American Symposium on Circuits and Systems (LASCAS), Bariloche, 2017, pp. 1-4, doi: 10.1109/LASCAS.2017.8126878.

C. Duran, ... **L. E. Rueda G.** et al., ”A 32-bit RISC-V AXI4-lite bus-based micro-

controller with 10-bit SAR ADC," 2016 IEEE 7th Latin American Symposium on Circuits and Systems (LASCAS), Florianopolis, 2016, pp. 315-318, doi: 10.1109/LASCAS.2016.7451073.

### Patents

**L. E. Rueda G.**, J. Moya and E. Roa. "Device and Process for Brown-out Detection," Patent nr. NC2018/0013057, Superintendencia de Industria y Comercio - Colombia, 2018.

**L. E. Rueda G.**, E. Roa., "Device and Process to Generate Voltage and Current References in the Digital Domain," Patent nr. NC2017/0012421, Superintendencia de Industria y Comercio - Colombia, 2017.

### Patent Pending

**L. E. Rueda G.**, G. Romero and E. Roa., "Device and Methodology for Voltage Regulation with Clock Auto-generation," Patent nr. NC2020/0012550, Superintendencia de Industria y Comercio - Colombia, 2020.

# BIBLIOGRAPHY


AHMED, Farid Uddin et al. “A Brief Overview of On-Chip Voltage Regulation in High-Performance and High-Density Integrated Circuits”. In: *IEEE Access* 9 (2021), pp. 813–826 (cit. on p. 136).

ALIBART, Fabien et al. “Pattern Classification by Memristive Crossbar Circuits Using ex situ and in situ Training”. In: *Nature Comm.* (June 2013) (cit. on p. 33).

AMAYA, A. et al. “A Multi-Level Power-on Reset for Fine-Grained Power Management”. In: *28th IEEE PATMOS*. July 2018, pp. 129–132 (cit. on pp. 89–90, 99).

AMBROGIO, S. et al. “Statistical Fluctuations in HfOx Resistive-Switching Memory: Part I - Set/Reset Variability”. In: *IEEE Trans. on Electron Devices* (2014) (cit. on p. 29).

ARDILA, Javier et al. “A Stable Physically Unclonable Function Based on a Standard CMOS NVR”. In: *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*. 2020, pp. 1–4 (cit. on p. 20).

ASHTON, Kevin et al. “That ‘internet of things’ thing”. In: *RFID journal* 22.7 (2009), pp. 97–114 (cit. on p. 15).

AY, S. U. “A Nanowatt Cascadable Delay Element for Compact Power-on-reset (POR) Circuits”. In: *2009 52nd IEEE International Midwest Symposium on Circuits and Systems*. Aug. 2009, pp. 62–65 (cit. on p. 99).

BACKUS, John. “Can Programming Be Liberated from the von Neumann Style? A Functional Style and Its Algebra of Programs”. In: *Commun. ACM* 21.8 (Aug. 1978), 613–641 (cit. on p. 27).

BANAGOZAR, A. et al. “Robust Neuromorphic Computing in the Presence of Process Variation”. In: *DATE*. 2017 (cit. on p. 33).

BANKMAN, Daniel et al. “An Always-On 3.8 $\mu$ J/86% CIFAR-10 Mixed-Signal Binary CNN Processor With All Memory on Chip in 28-nm CMOS”. In: *IEEE JSSC* 54.1 (2019), pp. 158–172 (cit. on p. 57).

BARENGHI, Alessandro et al. “Fault Injection Attacks on Cryptographic Devices: Theory, Practice, and Countermeasures”. In: *Proceedings of the IEEE* 100.11 (2012), pp. 3056–3076 (cit. on p. 108).

BAVANDPOUR, Mohammad et al. “aCortex: An Energy-Efficient Multipurpose Mixed-Signal Inference Accelerator”. In: *IEEE JESSCDC* 6.1 (2020), pp. 98–106 (cit. on p. 64).

BERINGUIER-BOHER, Noemie et al. “Voltage Glitch Attacks on Mixed-Signal Systems”. In: *2014 17th Euromicro Conference on Digital System Design*. 2014, pp. 379–386 (cit. on p. 108).

BOZZATO, Claudio et al. “Shaping the Glitch: Optimizing Voltage Fault Injection Attacks”. In: *IACR Transactions on Cryptographic Hardware and Embedded Systems* 2019.2 (Feb. 2019), pp. 199–224 (cit. on pp. 107–108, 114, 136, 138).

BREIMAN, Leo. “Bagging Predictors”. In: *Mach. Learn.* (Aug. 1996) (cit. on p. 31).

BRIAN BENCHOFF. *OPEN-V, THE FIRST OPEN SOURCE RISC-V MICROCONTROLLER*. Nov. 2016 (cit. on p. 18).

BÜCHEL, Julian et al. “Network Insensitivity to Parameter Noise via Parameter Attack During Training”. In: *International Conference on Learning Representations (ICLR)*. 2022 (cit. on pp. 32, 34–35, 39, 53–54).

CARTAGENA, Juan et al. “A fully-synthesized TRNG with lightweight cellular-automata based post-processing stage in 130nm CMOS”. In: *2016 IEEE Nordic Circuits and Systems Conference (NORCAS)*. 2016, pp. 1–5 (cit. on p. 20).

CHARAN, G. et al. “Accurate Inference With Inaccurate RRAM Devices: A Joint Algorithm-Design Solution”. In: *IEEE JXCDC* (2020) (cit. on p. 33).

CHEN, L. et al. “Accelerator-Friendly Neural-Network Training: Learning Variations and Defects in RRAM Crossbar”. In: *DATE*. 2017 (cit. on p. 33).

CHEN, Y. et al. “A 128-Channel Extreme Learning Machine-Based Neural Decoder for Brain Machine Interfaces”. In: *IEEE TBioCAS* (2016) (cit. on p. 32).

CHEN, Yi et al. “A 2.86-TOPS/W Current Mirror Cross-Bar-Based Machine-Learning and Physical Unclonable Function Engine For Internet-of-Things Applications”. In: *IEEE TCAS-I* 66.6 (2019), pp. 2240–2252 (cit. on p. 64).

CHIH, Yu-Der et al. “16.4 An 89TOPS/W and 16.3TOPS/mm2 All-Digital SRAM-Based Full-Precision Compute-In Memory Macro in 22nm for Machine-Learning Edge Applications”. In: *2021 IEEE International Solid- State Circuits Conference (ISSCC)*. Vol. 64. 2021, pp. 252–254 (cit. on p. 81).

CHOI, Shinhyun et al. “Data Clustering using Memristor Networks”. In: *Scientific Reports* 5.1 (May 2015) (cit. on p. 29).

DALGATY, Thomas et al. “In Situ Learning Using Intrinsic Memristor Variability via Markov Chain Monte Carlo Sampling”. In: *Nature Elect.* (Jan. 2021) (cit. on pp. 27, 29).

DE, Sourav et al. “Neuromorphic Computing with Fe-FinFETs in the Presence of Variation”. In: *2022 International Symposium on VLSI Technology, Systems and Applications (VLSI-TSA)*. 2022, pp. 1–2 (cit. on p. 30).

DE, Vivek. “Fine-grain power management in manycore processor and System-on-Chip (SoC) designs”. In: *2015 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. 2015, pp. 159–164 (cit. on p. 84).

DJELLID-OUAR, A. et al. “Supply voltage glitches effects on CMOS circuits”. In: *International Conference on Design and Test of Integrated Systems in Nanoscale Technology, 2006. DTIS 2006.* 2006, pp. 257–261 (cit. on pp. 106, 108, 110, 114, 116).

DURAN, C. et al. “A 32-bit RISC-V AXI4-lite bus-based microcontroller with 10-bit SAR ADC”. In: *2016 IEEE 7th Latin American Symposium on Circuits Systems (LASCAS)*. Feb. 2016, pp. 315–318 (cit. on p. 18).

DURAN, C. et al. “A 32-bit RISC-V AXI4-lite bus-based microcontroller with 10-bit SAR ADC”. In: *2016 IEEE 7th Latin American Symposium on Circuits Systems (LASCAS)*. 2016, pp. 315–318 (cit. on p. 123).

DURAN, C. et al. “A system-on-chip platform for the internet of things featuring a 32-bit RISC-V based microcontroller”. In: *2017 IEEE 8th Latin American Symposium on Circuits Systems (LASCAS)*. Feb. 2017, pp. 1–4 (cit. on p. 18).

DURAN, C. et al. “An Energy-Efficient RISC-V RV32IMAC Microcontroller for Periodical-Driven Sensing Applications”. In: *2020 IEEE Custom Integrated Circuits Conference (CICC)*. 2020, pp. 1–4 (cit. on pp. 18–19).

DURAN, Ckristian et al. “A 10pJ/bit 256b AES-SoC Exploiting Memory Access Acceleration”. In: *IEEE Transactions on Circuits and Systems II: Express Briefs* 69.3 (2022), pp. 1612–1616 (cit. on p. 20).

DURAN, Ckristian et al. “AES Sbox Acceleration Schemes for Low-Cost SoCs”. In: *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*. 2021, pp. 1–5 (cit. on p. 20).

DURAN, Ckristian et al. “An Energy-Efficient RISC-V RV32IMAC Microcontroller for Periodical-Driven Sensing Applications”. In: *2020 IEEE Custom Integrated Circuits Conference (CICC)*. 2020, pp. 1–4 (cit. on pp. 98, 101).

EENEWS EUROPE. *RISC-V MCU grown in Colombia*. Sept. 2016 (cit. on p. 18).

FALLAH, F. “Standby and Active Leakage Current Control and Minimization in CMOS VLSI Circuits”. In: *IEICE Transactions on Electronics* E88-C.4 (Apr. 2005), pp. 509–519 (cit. on p. 68).

GAO, Bin et al. “Ultra-Low-Energy Three-Dimensional Oxide-Based Electronic Synapses for Implementation of Robust High-Accuracy Neuromorphic Computation Systems”. In: *ACS Nano* (June 2014) (cit. on p. 29).

GLOROT, Xavier et al. “Understanding the Difficulty of Training Deep Feedforward Neural Networks”. In: *AISTATS*. 2010 (cit. on p. 39).

GOMEZ, Hector et al. “Low-cost TRNG IPs”. In: *IET Circuits, Devices & Systems* 14.7 (Oct. 2020), pp. 942–946 (cit. on p. 20).

GOMINA, Kamil et al. “Power supply glitch attacks: Design and evaluation of detection circuits”. In: *2014 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*. 2014, pp. 136–141 (cit. on p. 109).

GONZALES, David M. *Low Voltage CMOS Power-on Reset Circuit*. US Patent 9143137. 2015 (cit. on p. 85).

GUBBINS, D.P. *Brown-out Detector*. US Patent 6,894,544. May 2005 (cit. on p. 99).

HARA, Kazuyuki et al. “Analysis of Dropout Learning Regarded as Ensemble Learning”. In: *ICANN*. 2016 (cit. on p. 40).

HE, Kaiming et al. *Identity Mappings in Deep Residual Networks*. 2016. arXiv: 1603.05027 [cs.CV] (cit. on pp. 46–47).

HE, Z. et al. “Noise Injection Adaption: End-to-End ReRAM Crossbar Non-ideal Effect Adaption for Neural Network Mapping”. In: *ACM/IEEE DAC*. 2019 (cit. on pp. 32–33, 55).

HERREWEGEN, Jan Van DEN et al. “Fill your Boots: Enhanced Embedded Bootloader Exploits via Fault Injection and Binary Analysis”. In: *IACR Transactions on Cryptographic Hardware and Embedded Systems* (Dec. 2020), pp. 56–81 (cit. on pp. 108–109).

HU, M. et al. “Memristor Crossbar-Based Neuromorphic Computing System: A Case Study”. In: *IEEE TNNLS* (2014) (cit. on p. 32).

HUANG, Guang-Bin et al. “Extreme Learning Machine: Theory and Applications”. In: *Neurocomputing* 70.1-3 (2006), pp. 489–501 (cit. on p. 31).

HUO, Qiang et al. “Physics-Based Device-Circuit Cooptimization Scheme for 7-nm Technology Node SRAM Design and Beyond”. In: *IEEE Transactions on Electron Devices* 67.3 (2020), pp. 907–914 (cit. on p. 30).

IOFFE, Sergey et al. “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift”. In: *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*. Ed. by David BLEI et al. 2015 (cit. on p. 59).

ISLAM, S. M. R. et al. “The Internet of Things for Health Care: A Comprehensive Survey”. In: *IEEE Access* 3 (2015), pp. 678–708 (cit. on p. 15).

JACOB, Benoit et al. “Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2018 (cit. on p. 46).

JIANG, Zhewei et al. “C3SRAM: An In-Memory-Computing SRAM Macro Based on Robust Capacitive Coupling Computing Mechanism”. In: *IEEE JSSC* 55.7 (2020), pp. 1888–1897 (cit. on pp. 57, 81).

JOSHI, Vinay et al. “Accurate deep neural network inference using computational phase-change memory”. In: *Nature Communications* 11.1 (May 2020) (cit. on pp. 32–33, 55).

KANG, M. et al. “An Energy-Efficient VLSI Architecture for Pattern Recognition via Deep Embedding of Computation in SRAM”. In: *IEEE ICASSP*. May 2014 (cit. on p. 27).

KARIYAPPA, Sanjay et al. “Noise-Resilient DNN: Tolerating Noise in PCM-Based AI Accelerators via Noise-Aware Training”. In: *IEEE Trans. on Electron Devices* (2021) (cit. on pp. 32, 34, 39, 52–53, 55).

KHWA, W. et al. “A 65nm 4Kb Algorithm-Dependent Computing-in-Memory SRAM Unit-Macro with 2.3ns and 55.8TOPS/W Fully Parallel Product-Sum Operation for Binary DNN Edge Processors”. In: *IEEE ISSCC*. Feb. 2018 (cit. on p. 28).

KIM, Hyungjun et al. “Deep Neural Network Optimized to Resistive Memory with Non-linear Current-Voltage Characteristics”. In: *JETCS* 2 (July 2018) (cit. on p. 29).

KIM, W. et al. “Confined PCM-based Analog Synaptic Devices offering Low Resistance-drift and 1000 Programmable States for Deep Learning”. In: *2019 Symposium on VLSI Technology*. June 2019, T66–T67 (cit. on p. 27).

KRIZHEVSKY, A. “The CIFAR-10 and CIFAR-100 datasets”. In: *https://www.cs.toronto.edu/ kriz/-cifar.html/* (2009) (cit. on p. 46).

KRIZHEVSKY, Alex et al. “ImageNet Classification with Deep Convolutional Neural Networks”. In: *NIPS* (Jan. 2012) (cit. on p. 46).

KULALI, Emre et al. “Chip Power Model - A New Methodology for System Power Integrity Analysis and Design”. In: *2007 IEEE Electrical Performance of Electronic Packaging*. 2007, pp. 259–262 (cit. on p. 111).

LAPLANTE, P. A. et al. “The Internet of Things in Healthcare: Potential Applications and Challenges”. In: *IT Professional* 18.3 (May 2016), pp. 2–4 (cit. on p. 15).

LE, H. B. et al. “A Long Reset-Time Power-On Reset Circuit With Brown-Out Detection Capability”. In: *IEEE Transactions on Circuits and Systems II: Express Briefs* 58.11 (Nov. 2011), pp. 778–782 (cit. on pp. 85, 99).

LECUN, Y. et al. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* (1998) (cit. on p. 46).

LEE, Hyung-Min et al. “A Nonvolatile Flip-Flop-Enabled Cryptographic Wireless Authentication Tag With Per-Query Key Update and Power-Glitch Attack Countermeasures”. In: *IEEE Journal of Solid-State Circuits* 52.1 (2017), pp. 272–283 (cit. on p. 109).

LEE, I. et al. “A Subthreshold Voltage Reference With Scalable Output Voltage for Low-Power IoT Systems”. In: *IEEE Journal of Solid-State Circuits* 52.5 (May 2017), pp. 1443–1449 (cit. on pp. 88, 95).

LEE, Inhee et al. “Battery Voltage Supervisors for Miniature IoT Systems”. In: *IEEE Journal of Solid-State Circuits (JSSC)* (Nov. 2016) (cit. on pp. 99–100).

LEE, Jinseok et al. “Fully Row/Column-Parallel In-memory Computing SRAM Macro employing Capacitor-based Mixed-signal Computation with 5-b Inputs”. In: *2021 Symposium on VLSI Circuits*. 2021, pp. 1–2 (cit. on p. 81).

LI, H. et al. “Learning IoT in Edge: Deep Learning for the Internet of Things with Edge Computing”. In: *IEEE Network* 32.1 (Jan. 2018), pp. 96–101 (cit. on p. 16).

LIN, Ming-Guang et al. “D-NAT: Data-Driven Non-Ideality Aware Training Framework for Fabricated Computing-In-Memory Macros”. In: *IEEE Journal on Emerging and Selected Topics in Circuits and Systems* 12.2 (2022), pp. 381–392 (cit. on pp. 33–34).

LIU, B. et al. “Vortex: Variation-Aware Training for Memristor X-Bar”. In: *ACM/EDAC/IEEE DAC*. 2015, pp. 1–6 (cit. on pp. 33–34, 55).

LIU, Wenye et al. “Stealthy and Robust Glitch Injection Attack on Deep Learning Accelerator for Target With Variational Viewpoint”. In: *IEEE Transactions on Information Forensics and Security* 16 (2021), pp. 1928–1942 (cit. on p. 108).

LIU, Ying-Chuan et al. “A CMOS Current Mirror with Enhanced Input Dynamic Range”. In: *ICICIC*. 2008, pp. 571–571 (cit. on p. 65).

LONG, Y. et al. “Design of Reliable DNN Accelerator with Un-reliable ReRAM”. In: *DATE*. 2019 (cit. on pp. 32, 34, 39, 45, 52–53).

MA, C. et al. “Go Unary: A Novel Synapse Coding and Mapping Scheme for Reliable ReRAM-based Neuromorphic Computing”. In: *DATE*. 2020 (cit. on pp. 41, 55–56).

MARTÍN, Honorio et al. “Fault Attacks on STRNGs: Impact of Glitches, Temperature, and Underpowering on Randomness”. In: *IEEE Transactions on Information Forensics and Security* 10.2 (2015), pp. 266–277 (cit. on p. 108).

MENG, Ziqi et al. “Digital Offset for RRAM-based Neuromorphic Computing: A Novel Solution to Conquer Cycle-to-cycle Variation”. In: *DATE*. 2021 (cit. on pp. 33, 41, 55–56).

MIYASHITA, D. et al. “Time-Domain Neural Network: A 48.5 TSOp/s/W Neuromorphic Chip Optimized for Deep Learning and CMOS Technology”. In: *IEEE A-SSCC*. Nov. 2016 (cit. on p. 28).

MOHANTY, A. et al. “Random Sparse Adaptation for Accurate Inference with Inaccurate Multi-Level RRAM Arrays”. In: *IEEE IEDM*. 2017 (cit. on p. 32).

MURRAY, A.F. et al. “Enhanced MLP performance and fault tolerance resulting from synaptic weight noise during training”. In: *IEEE Transactions on Neural Networks* 5.5 (1994), pp. 792–802 (cit. on pp. 32, 34, 38–39).

NAVA WHITEFORD. *A COMPLETELY OPEN MICROCONTROLLER*. Oct. 2016 (cit. on p. 18).

NEFTCI, Emre O. et al. “Stochastic Synapses Enable Efficient Brain-Inspired Learning Machines”. In: *Frontiers in Neuroscience* 10 (June 2016) (cit. on p. 36).

O’FLYNN, Colin. “Fault Injection using Crowbars on Embedded Systems”. In: *IACR Cryptol. ePrint Arch.* 2016 (2016), p. 810 (cit. on p. 108).

O’FLYNN, Colin et al. “ChipWhisperer: An Open-Source Platform for Hardware Embedded Security Research”. In: vol. 8622. Apr. 2014 (cit. on pp. 105–106).

OPITZ, D. et al. “Popular Ensemble Methods: An Empirical Study”. In: *Journal of Artificial Intelligence Research* (Aug. 1999) (cit. on p. 31).

PAN, J. et al. “An Internet of Things Framework for Smart Energy in Buildings: Designs, Prototype, and Experiments”. In: *IEEE Internet of Things Journal* 2.6 (Dec. 2015), pp. 527–537 (cit. on p. 15).

PAPISTAS, I. A. et al. “A 22 nm, 1540 TOP/s/W, 12.1 TOP/s/mm2 in-Memory Analog Matrix-Vector-Multiplier for DNN Acceleration”. In: *2021 IEEE Custom Integrated Circuits Conference (CICC)*. 2021, pp. 1–2 (cit. on p. 81).

PRAKASH, R. “Zero Quiescent Current, Delay Adjustable, Power-on-reset Circuit”. In: *2014 IEEE Dallas Circuits and Systems Conference (DCAS)*. Oct. 2014, pp. 1–4 (cit. on p. 99).

PYLE, S. D. et al. “Leveraging Stochasticity for In Situ Learning in Binarized Deep Neural Networks”. In: *Computer* 52.5 (2019), pp. 30–39 (cit. on p. 33).

QUERLIOZ, D. et al. “Bioinspired Programming of Memory Devices for Implementing an Inference Engine”. In: *Proceedings of the IEEE* (2015) (cit. on p. 36).

RAJPUT, S.S. et al. “A Current Mirror for Low Voltage, High Performance Analog Circuits”. In: *AICPEF* 36.3 (2003), pp. 221–233 (cit. on p. 65).

RAMIREZ-ANGULO, J. “Low Voltage Current Mirrors for Built-in Current Sensors”. In: *IEEE ISCAS*. Vol. 5. 1994, 529–532 vol.5 (cit. on p. 65).

RICHTER, O. et al. “Device Mismatch in a Neuromorphic System Implements Random Features for Regression”. In: *IEEE BioCAS*. 2015 (cit. on p. 32).

RISC-V FOUNDATION. *Members at a Glance*. 2020 (cit. on p. 18).

RONCHI, N. et al. “A Comprehensive Variability Study of Doped $HfO_2$ FeFET for Memory Applications”. In: *2022 IEEE International Memory Workshop (IMW)*. 2022, pp. 1–4 (cit. on p. 30).

RUEDA G., Luis E. et al. “A Compact Industrial-Grade Multi-Threshold Brown-Out Detector”. In: *2019 26th IEEE International Conference on Electronics, Circuits and Systems (ICECS)*. 2019, pp. 923–926 (cit. on pp. 93, 99).

RUEDA G, Luis E. et al. *A-Connect for TensorFlow*. [Online] Available: https://github.com/onchipuis Connect. 2021 (cit. on pp. 31, 46).

RUEDA G., Luis E. et al. “An Ultra-Low Power Multi-Level Power-on Reset for Fine-Grained Power Management Strategies”. In: *2019 IEEE 10th Latin American Symposium on Circuits Systems (LASCAS)*. 2019, pp. 185–188 (cit. on pp. 87, 89, 91, 99).

SANTAMARIA, J. et al. “A Family of Compact Trim-Free CMOS Nano-Ampere Current References”. In: *2019 IEEE International Symposium on Circuits and Systems (ISCAS)*. May 2019, pp. 1–4 (cit. on p. 93).

SCHAPIRE, Robert E. “The Strength of Weak Learnability”. In: *Mach. Learn.* 5.2 (July 1990), 197–227 (cit. on p. 31).

SHANBHAG, Naresh R. et al. “Comprehending In-memory Computing Trends via Proper Benchmarking”. In: *2022 IEEE Custom Integrated Circuits Conference (CICC)*. 2022, pp. 01–07 (cit. on pp. 80–81).

SHE, X. et al. “Improving Robustness of ReRAM-based Spiking Neural Network Accelerator with Stochastic Spike-timing-dependent-plasticity”. In: *IJCNN*. 2019 (cit. on p. 36).

SI, X. et al. “A Twin-8T SRAM Computation-in-Memory Unit-Macro for Multibit CNN-Based AI Edge Processors”. In: *IEEE JSSC* (Jan. 2020) (cit. on p. 27).

SIMONYAN, Karen et al. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. Ed. by Yoshua BENGIO et al. 2015 (cit. on p. 46).

SINGH, Arvind et al. “Mitigating Power Supply Glitch based Fault Attacks with Fast All-Digital Clock Modulation Circuit”. In: *2019 Design, Automation Test in Europe Conference Exhibition (DATE)*. 2019, pp. 19–24 (cit. on p. 109).

SRIVASTAVA, Nitish et al. “Dropout: A Simple Way to Prevent Neural Networks from Overfitting”. In: *JMLR* (Jan. 2014) (cit. on pp. 35, 40).

SRIVASTAVA, P. et al. “PROMISE: An End-to-End Design of a Programmable Mixed-Signal Accelerator for Machine-Learning Algorithms”. In: *ACM/IEEE 45th Annual ISCA*. June 2018, pp. 43–56 (cit. on p. 27).

STANKOVIC, J. A. “Research Directions for the Internet of Things”. In: *IEEE Internet of Things Journal* 1.1 (Feb. 2014), pp. 3–9 (cit. on p. 16).

T, P. *Sensor monitoring device*. US Patent 3,842,208. Oct. 1974 (cit. on p. 15).

TANG, Adrian et al. “CLKSCREW: Exposing the Perils of Security-Oblivious Energy Management”. In: *26th USENIX Security Symposium (USENIX Security 17)*. Vancouver, BC: USENIX Association, Aug. 2017, pp. 1057–1074 (cit. on p. 106).

*The Internet of Thinks: An Overview*. Accessed: 2016-10-17. Oct. 2015 (cit. on pp. 15–16).

TIMMERS, Niek et al. “Controlling PC on ARM Using Fault Injection”. In: *2016 Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC)*. 2016, pp. 25–35 (cit. on pp. 108–109, 114, 127).

TRIPATHI, A. et al. “Analog Neuromorphic System Based on Multi Input Floating Gate MOS Neuron Model”. In: *IEEE ISCAS*. May 2019 (cit. on p. 32).

TROUCHKINE, Thomas et al. “Fault Injection Characterization on modern CPUs - From the ISA to the Micro-Architecture”. In: *13th IFIP International Conference on Information Security Theory and Practice (WISTP)*. Dec. 2019, pp. 123–138 (cit. on pp. 114, 139).

TSAI, Hsinyu et al. “Recent Progress in Analog Memory-Based Accelerators for Deep Learning”. In: *Journal of Physics D: Applied Physics* (June 2018) (cit. on pp. 27, 29).

VALAVI, H. et al. “A 64-Tile 2.4-Mb In-Memory-Computing CNN Accelerator Employing Charge-Domain Compute”. In: *IEEE JSSC* (2019) (cit. on p. 28).

— “A 64-Tile 2.4-Mb In-Memory-Computing CNN Accelerator Employing Charge-Domain Compute”. In: *IEEE JSSC* 54.6 (June 2019), pp. 1789–1799 (cit. on pp. 57, 81).

WADHWA, S. K. et al. “Zero Steady State Current Power-on-reset Circuit with Brown-out Detector”. In: *19th International Conference on VLSI Design held jointly with 5th International Conference on Embedded Systems Design (VLSID’06)*. Jan. 2006 (cit. on pp. 85, 99).

WAN, Li et al. “Regularization of Neural Networks Using Dropconnect”. In: *ICML*. 2013 (cit. on pp. 36, 40).

WANG, Z. et al. “Error Adaptive Classifier Boosting (EACB): Leveraging Data-Driven Training Towards Hardware Resilience for Signal Inference”. In: *IEEE TCAS-I* (2015) (cit. on p. 31).

WATERMAN, Andrew et al. *The RISC-V Instruction Set Manual, Volume I: User-Level ISA, Version 2.1*. Tech. rep. UCB/EECS-2016-118. EECS Department, University of California, Berkeley, May 2016 (cit. on p. 18).

WU, Dongxian et al. “Adversarial Weight Perturbation Helps Robust Generalization”. In: *NeurIPS*. 2020 (cit. on pp. 35, 53–54).

XIA, Qiangfei et al. “Memristive Crossbar Arrays for Brain-Inspired Computing”. In: *Nature Materials* 18.4 (Mar. 2019), pp. 309–323 (cit. on p. 29).

XIAO, Han et al. *Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms*. 2017 (cit. on p. 54).

XU, Jiangtao et al. “Low-leakage analog switches for low-speed sample-and-hold circuits”. In: *Microelectronics Journal* 76 (June 2018), pp. 22–27 (cit. on p. 68).

XUE, C. et al. “Embedded 1-Mb ReRAM-Based Computing-in- Memory Macro With Multibit Input and Weight for CNN-Based AI Edge Processors”. In: *IEEE JSSC* 55.1 (Jan. 2020), pp. 203–215 (cit. on p. 27).

YIN, S. et al. “Monolithically Integrated RRAM- and CMOS-Based In-Memory Computing Optimizations for Efficient Deep Learning”. In: *IEEE Micro* 39.6 (Nov. 2019), pp. 54–63 (cit. on p. 27).

YUCE, Bilgiday et al. “Fault attacks on secure embedded software: Threats, design, and evaluation”. In: *Journal of Hardware and Systems Security* 2.2 (2018), pp. 111–130 (cit. on pp. 105, 109, 114, 127).

YUE, Jinshan et al. “STICKER-IM: A 65 nm Computing-in-Memory NN Processor Using Block-Wise Sparsity Optimization and Inter/Intra-Macro Data Reuse”. In: *IEEE Journal of Solid-State Circuits* 57.8 (2022), pp. 2560–2573 (cit. on p. 82).

ZANELLA, A. et al. “Internet of Things for Smart Cities”. In: *IEEE Internet of Things Journal* 1.1 (Feb. 2014), pp. 22–32 (cit. on p. 15).

ZHANG, J. et al. “In-Memory Computation of a Machine-Learning Classifier in a Standard 6T SRAM Array”. In: *IEEE JSSC* (Apr. 2017) (cit. on pp. 31, 64).

ZHENG, N. et al. “Learning in Memristor Crossbar-Based Spiking Neural Networks Through Modulation of Weight-Dependent Spike-Timing-Dependent Plasticity”. In: *IEEE Transactions on Nanotechnology* (2018) (cit. on p. 36).

ZHENG, Yaowei et al. “Regularizing Neural Networks via Adversarial Model Perturbation”. In: *CVPR*. 2021 (cit. on pp. 35, 53–54).

ZHU, Y. et al. “Statistical Training for Neuromorphic Computing using Memristor-based Crossbars Considering Process Variations and Noise”. In: *DATE*. 2020 (cit. on pp. 32, 34–35, 54).

ZUSSA, Loïc et al. “Power supply glitch induced faults on FPGA: An in-depth analysis of the injection mechanism”. In: *International On-Line Testing Symposium (IOLTS)*. 2013, pp. 110–115 (cit. on pp. 106, 108, 110–111, 116–117).