Instruction-Based Acceleration for Post-Quantum Cryptography

Ivan Sarno[1], Stefano Di Matteo[1,2], Emanuele Valea[1] and Cyrille Chavet[3]
[1]Univ. Grenoble Alpes, CEA, List, F-38000 Grenoble, France,
[2]Univ. Grenoble Alpes, CEA, Leti, F-38000 Grenoble, France,
[3]Univ. Grenoble Alpes, CNRS, Grenoble INP, TIMA, 38000 Grenoble, France

***Abstract*—In this paper, we explore the potential of RISC-V platforms to accelerate Post-Quantum Cryptography (PQC) algorithms through dedicated instructions. We first discuss the benefits of Instruction-based Acceleration and categorize the primary implementation strategies: integrating a Custom Functional Unit within the CPU pipeline or leveraging dedicated Coprocessors. We then review notable works that propose custom instructions for PQC acceleration, analyzing their key design approaches. Finally, we emphasize the need for a standardized RISC-V extension to ensure efficient and portable PQC implementations across different hardware platforms.**

***Index Terms*—RISC-V, Post-Quantum, Cryptography, Hardware Acceleration, ML-KEM, ML-DSA, SLH-DSA.**

I. Introduction

The potential for large-scale quantum computers to be realized in the near future poses a significant threat to the security of digital communication systems, as they rely on cryptographic schemes that can be easily compromised using quantum algorithms. In response to this emerging threat, the National Institute of Standards and Technology (NIST) selected ML-KEM, ML-DSA and SLH-DSA as the next generation of public key cryptography schemes [4].

is a member of the Hash-based cryptography family, which relies on hash functions such as SHA-3, as the core security primitive. Lattice-based cryptographic schemes, such as ML-KEM and ML-DSA, derive their security from the hardness of lattice problems, which fundamentally rely on polynomial arithmetic. Additionally, they use hash function of the SHA-3 family to deterministically generate a random stream for polynomial sampling or for data hashing. Therefore, SHA-3 and polynomial arithmetic, in particular NTT-based multiplication, have been identified as the major bottlenecks of these new schemes [2].

RISC-V is an open and extensible Instruction Set Architecture (ISA). Companies and research groups can freely implement CPU targeting the ISA or modify the various open source cores available to better address their goals. RISC-V is a modular ISA, which means it consists of a mandatory core subset of instructions, along with various optional groups, known as Extensions, that are tailored for specific application domains and allow the ISA to be customized for different types of workloads. In addition to the standard extensions, RISC-V also provides the flexibility for users to define custom instructions, enabling them to optimize the ISA for their unique application requirements. This modularity makes RISC-V highly adaptable and scalable, allowing it to be used across a wide range of devices, from embedded systems to high-performance computing platforms. The possibility to define custom instructions opens to new acceleration techniques beyond traditional memory-mapped accelerators [1].

In this paper, we explore the landscape of custom instructions for Post-Quantum Cryptography acceleration, the main implementation strategies, and finally advocate for a standard extension to accelerate PQC schemes.

Corresponding author: mailto:ivan.sarno@cea.fr

II. Instruction-based Acceleration for PQC

The more common way to interface the processor with an accelerator is **Memory-mapped** interface in which the processor uses I/O operation both to exchange data with the accelerator and to control it. The accelerator is connected to the system bus and can be developed independently of the CPU. This kind of design ensures flexibility and high performance, but suffer from resources replication and data transfer latency. It also requires an ad-hoc software interface based on I/O primitives.

An alternative typical of RISC-V is **Intructions-based** interface, that uses a dedicate subset of instructions to control the accelerator. This approach fosters tighter integration with the CPU, allowing for more efficient sharing of resources and data. Additionally, once codified, the instructions provide a general software interface that is independent of the implementations and, with the right toolset support, can be portable and easy to use [2].

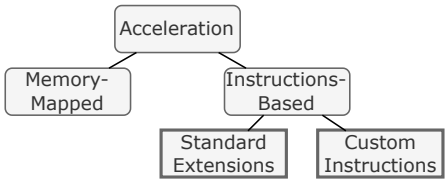


Fig. 1. Acceletation types classification

A dedicated instruction set in RISC-V can be categorized into two main types, as shown in figure 1:

- **Custom Instructions** – These are typically bound to a specific implementation and tailored to address particular needs. While they provide optimized performance, they are not standardized and may require custom toolchain modifications, limiting portability across different RISC-V cores.
- **Standard Extensions** – These follow the RISC-V ISA specification and must maintain coherence with the base architecture. They are designed for broader adoption, ensuring compatibility across multiple implementations. Standard extensions benefit from strong community support, established toolchains, and optimized compiler integration, making them more sustainable for long-term development.

When designing an instruction set, instructions can generally be categorized into two types based on how they process data: **Scalar Instructions** operate on individual operands that are equal to or smaller than a single word. They are well-suited for basic arithmetic and logic operations with data dependencies or irregular access patterns, such as modular arithmetic, NTT butterfly computations, and encoding/decoding at coefficient-level. **Vector Instructions** apply the same operation simultaneously to multiple elements of the same size, improving performance for data-parallel workloads. Vector instructions are ideal for processing entire polynomials efficiently [1] [4].

A. Implementation of Instruction-based Acceleration

To accelerate PQC schemes on RISC-V platforms, dedicated additional instructions can be implemented using two primary approaches, depending on performance needs and architectural constraints.

- **Custom Functional Units:** These are computational units integrated directly into the pipeline of the processor. They are well-suited for scalar operations, such as coefficient-wise arithmetic, as they can efficiently utilize existing CPU resources and operate directly on processor registers, minimizing communication overhead [2] [3].
- **Coprocessors:** These are independent computational units connected to the core. Unlike functional units within the pipeline, coprocessors are not constrained by its structure and can leverage additional hardware resources. This makes them ideal for executing complex operations and subroutines, such as polynomial computations or cryptographic hashing. Coprocessors are particularly well-suited for handling vector instructions, enabling efficient large-scale data processing [4] [5].

III. Custom Instructions for PQC: State of the Art

In this paragraph we present some custom instruction from the literature to highlight possible implementation approaches and figure out the performance gain and resource usage to expect from similar solutions.

In [3], the authors propose a set of custom instructions to accelerate coefficient-level operations, such as modular arithmetic and NTT butterfly computations. They provide a dedicated instruction set extension (ISE) for ML-KEM and one for ML-DSA. By integrating these new instructions directly into the processor's pipeline, the design enables faster NTT execution while efficiently reusing existing processor resources.

In [2], the authors introduce a more specialized ISE comprising 29 instructions targeting modular arithmetic, complex multiplications (such as multiply-accumulate), sampling, encoding, and hashing. The hashing functional unit executes a complete round of the SHA-3 hash function within the processor pipeline, leveraging floating-point registers to store the state. This approach enhances SHA-3 performance while minimizing memory overhead.

The work in [5] proposes a lightweight set of instructions—load, store, and round—for SHA-3 operations, implemented via a dedicated coprocessor. By utilizing its own resources and a specialized pipeline, the coprocessor achieves high performance while offloading hashing computations from the main processor. This design improves system scalability and overall efficiency in cryptographic workloads.

Finally, [4] presents a vector instruction set for polynomial arithmetic and SHA-3. The use of vector instruction executed in a highly parallel (32 butterfly units) powerful coprocessor enables high-performance polynomial arithmetic with a low instruction count, but comes at the cost of high resource usage.

IV. Conclusions: Towards a Standard RISC-V Extension for PQC

Custom instructions are often tailored to specific hardware accelerators, making them non-portable across different implementations. Despite these limitations, research efforts have identified critical bottlenecks in standard PQC schemes and explored various optimization strategies, demonstrating significant potential performance gains. These insights provide a strong foundation for designing a standardized RISC-V extension for PQC, similar to existing extensions for classical cryptography, general, flexible, and adaptable to various devices and cryptographic primitives. It must also accommodate future algorithmic changes in the evolving post-quantum cryptographic landscape

Acknowledgment

This work received funding from the France 2030 program, managed by the French National Research Agency under grant agreement No. ANR-22-PETQ-0008 PQ-TLS.

References

[1] Risc-v specification. https://riscv.org/specifications/ratified/.
[2] Fritzmann T. et al. Risq-v: Tightly coupled risc-v accelerators for post-quantum cryptography. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2020.
[3] Miteloudi K. et al. Pq. v. alu. e: Post-quantum risc-v custom alu extensions on dilithium and kyber. In *International Conference on Smart Card Research and Advanced Applications*, 2023.
[4] Zhao Y. et al. Enhancing risc-v vector extension for efficient application of post-quantum cryptography. 2023.
[5] Zhenjiang Wang, Shangshou Wang, Lei Wang, and Qiushi Yao. An Instruction Extension Based SHA-3 Algorithm Co-Processor Design Scheme. In *2023 3rd International Symposium on Computer Technology and Information Science (ISCTIS)*, 2023.

A dataflow-based design flow for heterogeneous embedded systems

Jacques Morin
INSA Rennes
IETR, UMR CNRS 6164
Rennes, France
jamorin@insa-rennes.fr

Hugo Miomandre
INSA Rennes
IETR, UMR CNRS 6164
Rennes, France
hmiomand@insa-rennes.fr

Mickaël Dardaillon
INSA Rennes
IETR, UMR CNRS 6164
Rennes, France
mdardail@insa-rennes.fr

Jean-François Nezan
INSA Rennes
IETR, UMR CNRS 6164
Rennes, France
jnezan@insa-rennes.fr

***Abstract*—Heterogeneous computing is widely used in HPC. It applies well to astronomical computing where large amount of data have to be processed in real-time with specialized signal-processing algorithms. However optimizing such systems manually is difficult due to their enormous design-space. DSE flows have been developed to help designers, but they have a hard time taking into account all processing elements' characteristics. To address these limitations we propose a new heterogeneous design and DSE flow using the PREESM prototyping tool, focused on FPGA and CGRA accelerators. It works by breaking the heterogeneous DSE into several, modular homogeneous DSEs to achieve optimized results in shorter time while facilitating the test of new scheduling-mapping techniques.**

***Index Terms*—DSE, heterogeneous systems, scheduling, mapping, dataflow, radio-astronomy, ska, hpc**

I. Introduction

The Square-Kilometer Array Observatory (SKAO) project aims at using hundreds of antennas to perform radio-frequency observations. The projected phase-1 data projection is 16 Tb/s, to be processed with limited power in remote observation sites. The processing will be under real-time constraints to respond quickly to stellar phenomenons. With an expected 135 FLOPS required under a 2 MW budget, the processing facilities won't be able to use only power-hungry GPUs. Instead they will use hardware accelerators such as Field-Programmable Gate Array (FPGA) and Coarse-Grain Reconfigurable Array (CGRA) [3]. However prototyping processing centres that use a large panel of processing architectures is difficult, and the Design-Space Exploration (DSE) can be challenging and time-consuming.

Much research has been made on heterogeneous DSE, however heterogeneous systems design tools usually either lack real-time features [7, 9] or mapping DSE [2, 8], or use simplistic communication models [1]. To fill this gap we propose a new heterogeneous design flow with automated scheduling and mapping DSE, focusing on FPGA and CGRA accelerators and based on PREESM.

II. PREESM and Dataflow

PREESM [6] is a fast prototyping tool that introduced and uses the PiSDF Model of Computation (MoC) to help design signal processing application on multicore and multinode systems, CPU-GPU systems and FPGA [4]. It performs mapping-scheduling, assesses an application's metrics such as deadlock-freeness, throughput and latency, optimizes the application's parallelism and memory usage and generates multi-threaded real-time C and HLS code.

Applications are described with the Parametrized & Interfaced Synchronous Dataflow (PiSDF) MoC as a graph of actors communicating via FIFO buffers. Each actor has production/consumption rates describing the amount of data tokens it consumes/produces from/to its input/output FIFO buffers each execution (firing). It fires only if enough input tokens are available and enough storage is available in output buffers. In PREESM an actor represents either code (its refinement) or another graph, in which case it is called a hierarchical actor. Actors and FIFOs can take moldable parameters [5] as inputs: numbers or lists of numbers that act as template parameters on which PREESM can perform a DSE.

Designers describe the processing system as a set of processing elements (PE) linked by shared memory and data or control links. The application is independent from the processing system, thus any application graph can be implemented on any architecture. During scheduling the graph is flattened into an SRDAG to expose data parallelism. It is then fed to an iterative scheduling-mapping algorithm: a heuristic schedules and maps the algorithm, and this schedule's cost is evaluated.

III. Heterogeneous design

Designing heterogeneous systems raises several challenges:

- Communications: latency and throughput can differ depending on input and output PE; there can be contentions and even unpredictability when using operating systems.
- Scheduling-Mapping: actors can execute on several PEs with different timing characteristics.
- Resources: one cannot simply map all actors on FPGA or CGRA due to resource constraints, e.g some schedulings are not mappable, and some mappings cause deadlocks.
- Dynamicity: a potential solution to the previous challenge would be to re-configure the accelerator for every actor; however this introduces new latencies to take into account and complicates both scheduling and mapping.

This research was funded, in whole or in part, by the Agence Nationale de la Recherche (ANR), grant ANR-23-CE46-0010-02.

The DSE cannot be exhaustive nor manual due to the design space's size. In order to reduce the complexity of the problem, our solution breaks the application down into simpler homogeneous sub-components that can be scheduled and mapped more easily. The underlying assumption is that these subcomponents will be locally well-scheduled using specialized schedulers, which should give a good global scheduling. Adding support or new features for an architecture would only require writing a new homogeneous scheduler, not the entirety of the scheduling process. Communications between homogeneous components are taken into account by their respective scheduler, and data manipulations/transfers such as copies or mergers can be modeled as actors in the application graph, thus constraints such as latency or contentions are taken into account by the global scheduler-mapper.

IV. Proposed design flow

The proposed design flow represented in Fig. 1 is built upon 3 functional blocks locally linked by DSE loops:

- Clustering: performs the mapping DSE. It takes as input the application graph, the architecture graph and a scenario. It clusters actors according to their available PE mapping into homogeneous sub-graph actors. It outputs a hierarchical graph and a list of possible PE mappings (CPU, FPGA...) for each actor and subgraph.
- Master Scheduler-Mapping: performs the global heterogeneous DSE by calling iteratively the slave scheduler-mappers with different constraints and PE mapping. If no result fits the requirements it calls the clustering step to redistribute actors.
- Slave Scheduler-Mappers: called by the master scheduler-mapper to perform homogeneous DSEs on sub-graphs with given objective functions. The DSE operates on moldable parameters that describe external and internal architectural settings such as parallelism, throughput, buffer sizes, behaviour and communications.

V. Conclusion & Future Works

We introduced a new design flow based on PREESM [6]. It aims to address the difficulties of automating heterogeneous system optimizations via DSE when dealing with communications delays, non-trivial mapping and a wide variety of hardware accelerators. This is done by applying an iterative DSE that clusters the application's heterogeneous components into homogeneous subcomponents. They are iteratively scheduled and mapped (local DSE) while ensuring their timing and mapping characteristics are satisfying for the global system (global DSE). If necessary a new clustering phase is performed to find a different clustering that fits the requirements better.

This design flow will be tested on astronomy data analysis algorithms that use several heterogeneous accelerators, such as DDFacet. We aim to compare its performance with tools such as MESSI [1] by implementing the same algorithms they were tested with and comparing outputs.

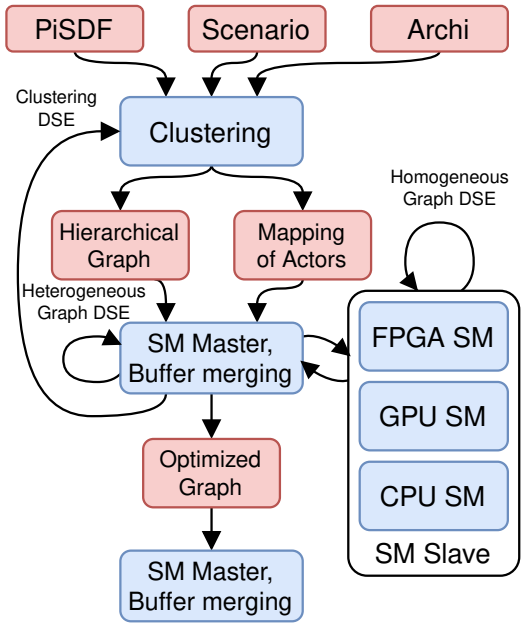


Figure 1. Illustration on the proposed heterogeneous DSE flow.

References

[1] S. Ahmadi-Pour et al. "MESSI: Task Mapping and Scheduling Strategy for FPGA-based Heterogeneous Real-Time Systems". In: *ACM Trans. Des. Autom. Electron. Syst.* (2025).

[2] A. Amarnath et al. "Heterogeneity-Aware Scheduling on SoCs for Autonomous Vehicles". In: *IEEE Computer Architecture Letters* (2021).

[3] S. Corda et al. "Reduced-Precision Acceleration of Radio-Astronomical Imaging on Reconfigurable Hardware". In: *IEEE Access* (2022).

[4] A. Honorat et al. "Automated Buffer Sizing of Dataflow Applications in a High-Level Synthesis Workflow". In: *ACM Trans. on Reconf. Tech. and Syst. (TRETS)* (2024).

[5] A. Honorat et al. "Influence of Dataflow Graph Moldable Parameters on Optimization Criteria". In: *Workshop on Design and Architectures for Signal and Image Processing*. 2022.

[6] M. Pelcat et al. "Preesm: A dataflow-based rapid prototyping framework for simplifying multicore DSP programming". In: *Education and Research Conference (EDERC)*. 2014.

[7] A. Vaishnav, K. Pham, and D. Koch. *Heterogeneous Resource-Elastic Scheduling for CPU+FPGA Architectures*. 2019.

[8] J. Xu, K. Li, and Y. Chen. "Real-time task scheduling for FPGA-based multicore systems with communication delay". In: *Microprocessors and Microsystems* (2022).

[9] Z. Zhu et al. "A Hardware and Software Task-Scheduling Framework Based on CPU+FPGA Heterogeneous Architecture in Edge Computing". In: *IEEE Access* (2019).

Benefits of On-wafer Calibration for RF Characterization of InP DHBT Technology Devices

Moussa Cissé[1], Nil Davy[2], Virginie Nodjiadjim[2], Bertrand Ardouin[2], Colin Mismer[2], Cristell Maneux[1], François Marc[1], Marina Deng[1]

[1] IMS Laboratory, University of Bordeaux, CNRS UMR 5218, Bordeaux INP, Talence, France.
[2] III-V Lab, joint lab between Nokia Bell Labs, Thales and CEA Leti, Palaiseau, France.
Corresponding author: moussa.cisse@ims-bordeaux.fr

***Abstract*—In this study, calibration methods are performed on passive structures up to 110 GHz using off-wafer standards and on-wafer standards. The open and short transistor interconnect measurements are analyzed through a comparison with the electromagnetic (EM) predictive simulation. The results clearly demonstrate the benefits of utilizing on-wafer calibration methods to improve measurement accuracy by significantly reducing the parasitic effects due to the transistor's interconnects.**

***Keywords*—RF on-wafer characterization, calibration, open-short de-embedding, EM simulation, bipolar transistors, indium phosphide (InP).**

I. Introduction

The electromagnetic radiation spectrum reveals a gap between the electronic and photonic domains (from 300 GHz to 3 THz in vacuum). Indium Phosphide (InP) Double Heterojunction Bipolar Transistors (DHBTs) have emerged as an attractive solution for this THz gap due to their superior frequency performance and high breakdown voltage, which enhance their power handling capabilities [1].

To fully exploit the potential of InP DHBTs, accurate on-wafer RF characterization is essential. In fact, it enables compact modelling by providing the necessary high-frequency measurements for the extraction of specific parameters, as well as the validation of the model at high frequency [2] to allow integrated circuit design at sub-millimeter wave frequencies. Previous studies have demonstrated the application of conventional on-wafer RF measurement methods, such as off-wafer Short-Open-Load-Thru (SOLT) calibration followed by Open-Short deembedding [3], with a precision of up to 110 GHz using pads optimized for industrial process fabrication [4]. However, limitations have been observed in extending these measurements up to 220 GHz. As it was foreseen by the electrical model simulation of the interconnects in [5], the reduction of the parasitics of the de-embedding test structures is crucial to improve the accuracy of the transistor's on-wafer RF measurements beyond 110 GHz. As a consequence, this study provides a comparison between off-wafer SOLT and on-wafer Thru-Reflect-Line (TRL) calibrations through measurements up to 110 GHz of identical test structures as in [4]. Section II shows the design of the RF test structures and the methodology for EM simulation; section III presents the results obtained by measurement and EM simulation up to 110 GHz of Open and Short de-embedding structures.

II. Methodology

A. RF Test Structure Design

Several samples on which passive devices were fabricated using the InP DHBT baseline technology were designed by III-V Lab. Different de-embedding structures, such as Open and Short, were available, together with transmission lines. Fig. 1 shows a picture of one of these test structures.

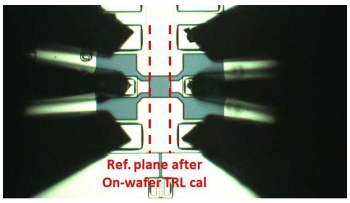


Fig. 1. Picture of a Pad-Open structure used for on-wafer TRL calibration

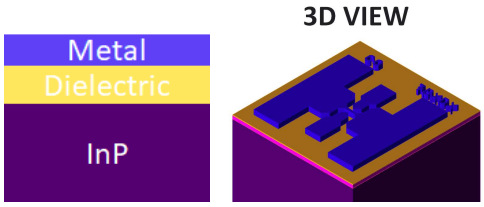
These passive devices are intended to be measured in order to benchmark calibration methods, and possibly propose any further improvements in the RF test structure design. The preliminary measurement results carried out up to 110 GHz will be shown in Section III.

B. EM Simulation

In order to verify the accuracy of the off-wafer SOLT calibration and the Open and Short measurements, the electromagnetic (EM) simulation was done by using the corresponding layout of these test structures. The model of the stack was simply based on a single metal layer on top of the dielectric deposited on the indium-phosphide layer, as depicted in Fig. 2. The EM simulation will be used as the physical reference for the validation of the high frequency on-wafer measurements.

C. Measurement Setup

RF measurements were performed using a probe station equipped with a Keysight E5270 Vector Network Analyzer (VNA), covering frequencies up to 67 GHz. To extend the measurement range, the N5260A mm-wave head controller, in conjunction with frequency extenders (N5260-60003), was used for frequencies beyond 67 GHz. 100-μm pitch Picoprobe RF probes, compatible with the RF pads of on-wafer standards, were utilized for measurements in the 1 to 110 GHz range. For calibration, a CS-5 calibration kit paired with the RF



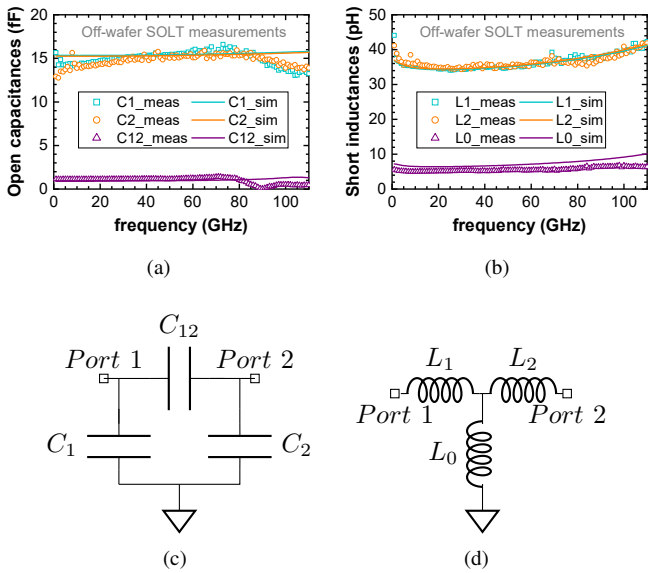
(a) (b)

Fig. 2. Material stack used for EM simulation of Open and Short structures: a) stack cross section, b) 3D view of the simulated Open structure

probes provider was employed for off-wafer SOLT calibration, while on-wafer TRL calibration was conducted using on-wafer standards, fabricated on the same wafer as the devices under test.

III. RESULTS AND DISCUSSION

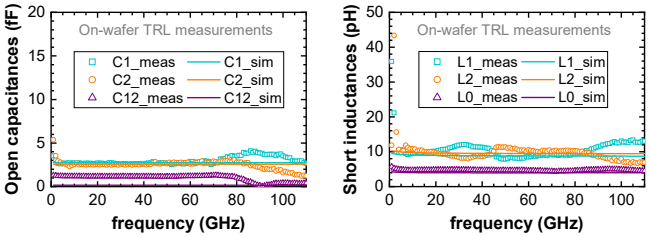
The available Open and Short structures intended for the de-embedding of the transistor's interconnects were measured from 1 to 110 GHz after performing the off-wafer SOLT calibration. As a result, an excellent agreement between off-wafer SOLT calibrated measurements and EM simulation of the Open and Short structures was achieved until 110 GHz, as it can be observed in Fig. 3. The extracted values of open capacitances and short inductances for off-wafer SOLT calibration are: $C_1 = C_2 = 15$ fF, and $C_{12} = 1$ fF, while L_1=L_2 = 36 pH and L_0 = 6 pH.



(a) (b) (c) (d)

Fig. 3. Comparison between off-wafer SOLT calibrated measurements and EM simulation of de-embedding test structures up to 110 GHz: a) Open structure, b) Short structure, c) Open equivalent circuit, d) Short equivalent circuit

The same comparison was made when the Thru-Reflect-Line calibration was performed using standards that were available on the wafer. Instead of placing the reference plane of the calibration at the probe tips, such as the off-wafer SOLT calibration, the on-wafer TRL calibration allows to push the reference plane after calibration along the Thru standard as indicated in Fig. 1. This way, the parasitic effects caused by the contact pads are removed by the calibration. The extracted values for on-wafer TRL calibration are: $C_1 = C_2 = 3$ fF and $C_{12} = 1$ fF, while $L_1 = L_2 = 10$ pH and $L_0 = 6$ pH. Consequently, the measurements of the Open and Short test structures show less parasitics (capacitive effects reduced by 80% and inductive effects by 72%), as shown in Fig. 4, which was confirmed again by EM simulation.



(a) (b)

Fig. 4. Comparison between on-wafer TRL calibrated measurements and EM simulation of de-embedding test structures up to 110 GHz: a) Open structure, b) Short structure

CONCLUSION

We demonstrated that on-wafer calibration techniques offer significant advantages over traditional off-wafer methods. The reduction in parasitic effects by calibration leads to more reliable measurements at high frequencies, particularly for sub-millimeter range applications. Future work will focus on extending this study about benchmarking off-wafer and on-wafer calibration techniques to higher frequencies up to 220 GHz and possibly explore other calibration techniques than SOLT and TRL.

ACKNOWLEDGMENT

This work is supported by the Chips Joint Undertaking and its members through Move2THz project, including the top-up funding by National Authorities under Grant Agreement n° 101139842.

REFERENCES

[1] A. Arabhavi et al., "InP/GaAsSb Double Heterojunction Bipolar Transistor Emitter-Fin Technology With f_{MAX} = 1.2 THz", in *2021 IEEE Transactions on Electron Devices Meeting (IEDM)*, Dec. 2021, pp. 11.4.1–11.4.4. doi: 10.1109/TED.2021.3138379.

[2] M. Deng et al., "InP DHBT Characterization up to 500 GHz and Compact Model Validation Towards THz Circuit Design", in *2021 IEEE BiCMOS and Compound Semiconductor Integrated Circuits and Technology Symposium (BCICTS)*, Dec. 2021, pp. 1–4. doi: 10.1109/BCICTS50416.2021.9682466.

[3] M.C.A.M. Koolen et al., "An improved de-embedding technique for on-wafer high-frequency characterization", in *Proceedings of the 1991 Bipolar Circuits and Technology Meeting*, Sep. 1991, pp. 188–191. doi: 10.1109/BIPOL.1991.160985.

[4] N. Davy et al., "InP DHBT test structure optimization towards 110 GHz characterization", in *ESSDERC 2022 - IEEE 52nd European Solid-State Device Research Conference (ESSDERC)*, Sep. 2022, pp. 320–323. doi: 10.1109/ESSDERC55479.2022.9947170.

[5] N. Davy et al., "InP DHBT On-Wafer RF Characterization and Small-Signal Modelling up to 220 GHz", in *2023 18th European Microwave Integrated Circuits Conference (EuMIC)*, Sep. 2023, pp. 101–104. doi: 10.23919/EuMIC58042.2023.10288849.

Ultra-Low Power Heart Rate Detection Module for Pacemakers dedicated to Small Animals

Quentin Vermot des Roches, Emilie Avignon-Meseldzija, Anthony Kolar, Caroline Lelandais-Perrault, Philippe Bénabès
Laboratoire de Génie Électrique et Électronique de Paris
Université Paris-Saclay, CentraleSupélec, CNRS, 91192 Gif-sur-Yvette, France
quentin.vermotdesroches@centralesupelec.fr, emilie.avignon@centralesupelec.fr, anthony.kolar@centralesupelec.fr
caroline.lelandais-perrault@centralesupelec.fr, philippe.benabes@centralesupelec.fr

I. Introduction

Pulmonary Arterial Hypertension is a progressive cardiovascular disease characterized by abnormally high blood pressure in the pulmonary arteries. It burdens patients' daily life and puts them at risk of heart failure and myocardial infarction. Researchers from the French National Institute of Health and Medical Research (INSERM) have come up with a hypothesis according to which lowering the heart rate has the potential to prevent the progression of the disease. For this theory to be tested, animal models are essential. Rats offer a high repeatability for moderate costs and a good insight into human health as their cardiac physiology is close to ours. There is therefore a need for an electrical device capable of modulating a rat's heart rate. A previous work has led to the realization of a stimulation circuit tailored for small rodents [1].

The stimulation provided by the aforementioned circuit still has to be monitored. The cardiac electrical activity can be recorded in form of an implantable electrocardiogram. A trade-off between performance and power consumption has to be taken into account when designing such a circuit as a great variety of methods exists in the state-of-the-art. Complex approaches such as the wavelet transform achieve great accuracy but rely on power-hungry computational units [3]. Their high power consumption (in the µW range) makes these methods unfit for implantable ECG sensors, battery replacements should indeed be avoided at all costs because they are synonymous with open-heart surgery, a costly and risky maneuver. This is especially true for rats as smaller pacemakers will only allow smaller batteries. Derivative-based methods have led to low-power low-area analog implementations consuming as low as 40nW [2] but may lack in accuracy compared to other filter banks approaches such as the Pan-Tompkins algorithm [4]. In this algorithm, the differentiation is followed by squaring, moving averaging, and adaptive thresholding circuits. It is often considered the best in the state-of-the-art to measure the heart rate of healthy patients with standard ECGs [5]. Both mixed [6] and analog [7] architectures exist, with analog implementations having a lower power consumption. Some achieve consumptions as low as 0.5nW (excluding the bias and reference voltages generating circuits [7]) but such a low consumption comes at the price of a lack of robustness.

II. Specifications and Proposed Architecture

A. Specifications

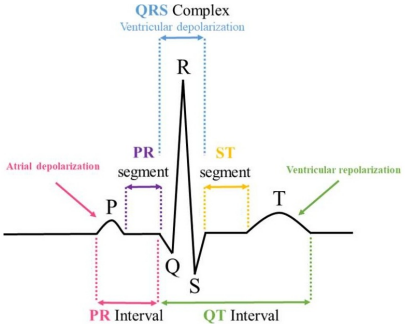


Fig. 1. Typical aspect of an ECG signal for mammals, figure from [1]

The designed circuit should be able to monitor the cardiac frequency of a rat. The best-defined feature of a mammal ECG is the R-peak (see Fig.1), the best strategy to monitor the cardiac frequency is therefore to identify the R-peaks and to compute the time interval between them. The frequency band of the QRS Complex ranges from 10 to 25 Hz. Our band-pass Front-End thus needs to amplify these frequencies while filtering out the two main sources of noise we have to deal with :

- The 50Hz Power Line Disturbance
- The ∼1 Hz Baseline Wander corresponding to the breathing frequency of the adult Sprague-Dawley rats [8].

B. System Architecture

The proposed ECG architecture, realized with xFAB xh018 180nm CMOS technology, readapts the Pan-Tompkins algorithm by performing the amplification, noise filtering and differentiation of the input signal in a unique block to limit the amount of relatively power-hungry Operational Transconductance Amplifiers (OTA), and by replacing the typically digital Moving Average Filter with a monostable Reshaping circuit to

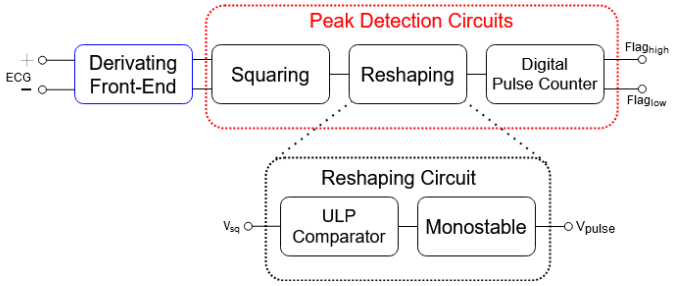


Fig. 2. Proposed adaptation of the Pan-Tompkins algorithm

avoid unnecessary complexity. The vast majority of transistors is operated in the subthreshold region to greatly reduce the power consumption.

Our fully differential band-pass Front-End consists in an RC high-pass filter followed with a g_m-RC low-pass filter. The designed OTA is a 1-Stage fully differential PMOS Folded Cascode OTA with a 34nS transconductance. The center frequency of the filter is 25Hz so that the R-peak is both amplified and derivated. The differentiation step highlights the steep slopes (positive then negative) of the R-peak. It is followed by a squaring circuit that relies on the quadratic V-I characteristic of 4 MOSFETs in a translinear loop to effectively square its input. The purpose of the squaring circuit is to amplify the successive positive and negative peaks corresponding to the derivated R-peak. An Ultra-Low-Power (ULP) Comparator then provides a proper rail-to-rail signal that is fed to a monostable flip-flop that makes sure that only one impulse appears for each heartbeat. These single rail-to-rail impulses are then suitable as inputs for a Digital Pulse Counter that counts all those impulses over a defined time period and raises a flag if the cardiac frequency is too high or too low compared to the targeted one.

III. Preliminary results

Waiting for murine ECGs for the circuit to be tested on, it was tested on the ECG database of the PhysioNet/-Computing in Cardiology Challenge 2020 [9] that was artificially noised to match murine cardiophysiology. The results are very satisfying as 100% of heartbeats on healthy ECGs were successfully identified in all of the process variation worst cases which means that the designed system is robust. Some heartbeats were missed on ectopic ECGs but this should not be a concern as the health of laboratory rats is very closely monitored.

The total power consumption of the circuit is $18.9nW$ which is lower than what we were able to find in papers taking the bias voltage sources into account [2], [5]. The used current and voltages sources were taken from [1] and account for $13,9nW$ which represents 73.5% of the total consumption.

The aspect of the signal at the output of each processing block is shown on Fig.3.

IV. Conclusion

This work proposes the description of an ultra-low power $18.9nW$ cardiac frequency detection module suited to rat-like cardiophysiology (heart rates up to 600bpm, baseline wandering up to 4Hz). Its efficiency and robustness have been demonstrated on a database of artificially noised human ECGs. The layouting of the circuit and measures on actual rats still have to be carried out.

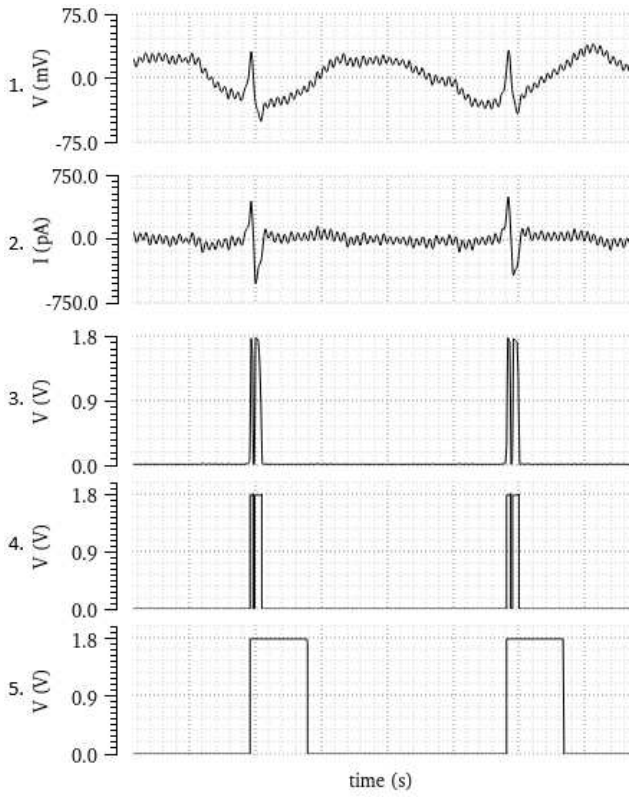


Fig. 3. Outputs of the successive processing blocks (1.:Raw input, 2.Derivating Front-End, 3. Squarer, 4. ULP Comparator, 5. Monostable flip-flop)

References

[1] Fanny Pan et al., "A low power frequency-programmable stimulation circuit for small rodent pacemaker". Analog Integrated Circuits and Signal Processing, Springer, 2024.

[2] Atiyeh Karimlou and Mohammad Yavari, "A First-Order Derivative-Based QRS-Detection Circuit in Time Domain for ECG Sensors", IEEE TCAS-II: Express Briefs, vol. 71, no. 1, 2024.

[3] Yao Zou et al., "An energy-efficient design for ECG recording and R-peak detection based on wavelet transform," IEEE TCAS-II Express Briefs, vol. 62, no. 2, 2015.

[4] Jiapu Pan and Willis Tompkins, "A real-time QRS detection algorithm," IEEE Trans. Biomed. Eng., vol. BME-32, no. 3, pp. 230–6, 1985.

[5] Michal Gala et al., "Robust QRS Complex Detector Algorithm Based on Modified Pan-Tompkins Method and Wavelet Transform", 43rd Int. Conf. on Telecommunications and Signal Processing, 2020.

[6] Md Niaz Imtiaz and Naimul Khan, "Pan-Tompkins++: A Robust Approach to Detect R-Peaks in ECG Signals", IEEE Int. Conf. on Bioinformatics and Biomedicine (BIBM), 2022.

[7] Güngör C. Berk and Hakan Töreyin. "A 0.5 nW Analog ECG Processor for Real Time R-Wave Detection Based on Pan-Tompkins Algorithm", IEEE Int. Conf. on Biomedical & Health Informatics (BHI), 2019.

[8] Julia K.L Walker et al., "Breath timing, volume and drive to breathe in conscious rats: comparative aspects", Elsevier Respiration Physiology 107, pp. 241-50, 1997.

[9] Matthew A. Reyna et al. "Classification of 12-Lead ECGs: The PhysioNet/- Computing in Cardiology Challenge 2020", 2020 Computing in Cardiology, 2020.

SoC-FPGA HW Trojan leaking data through EM Covert Channel

Marie-Aïnhoa Nicolas, Jordane Lorandel, Christophe Moy
Univ Rennes, CNRS, IETR UMR 6164, F 35000
Rennes, France
marie.nicolas@univ-rennes.fr, jordane.lorandel@univ-rennes.fr, christophe.moy@univ-rennes.fr

***Abstract*—This paper demonstrates an attack exploiting an Electromagnetic (EM) leak coming from the SoC-FPGA I/O. A covert channel is created by a dedicated Hardware Trojan controlling the EM emanations between the DDR3L SDRAM and the SoC-FPGA, exfiltrating sensitive data.**

***Index Terms*—SoC-FPGA, Electromagnetic leak, Pulse emanation, covert channel**

I. Introduction

System-on-Chip Field Programmable Gate Arrays (SoC-FPGA) have proven to be essential thanks to their programmability for specific application and execution speed. Its integrity and security has been studied thoroughly, from passive and active attacks to their associated countermeasures. Nowadays, with the expansion of Artificial Intelligence (AI), they have become a major actor for neural network inference. Their massive use in many application fields from edge-computing to cloud-based infrastructure highlights their attractiveness. New threats were revealed in the community, showing potential information leaks from this circuit. In [1], electromagnetic (EM) emanations from a SoC-FPGA with an open source neural network (NN) framework were studied. They successfully recovered the transmitted bitmap image from the EM emanations. Despite what was previously indicated, the leak was found to actually be coming from the SoC-FPGA input/outputs (I/O) towards the DDR3L SDRAM (Double Data Rate Synchronous Dynamic Random Access Memory) rather than the internal bus. This paper aims to demonstrate that the whole chip security could be compromised. More particularly, if an attacker is able to manipulate the communications entering or outgoing from the chip, an EM covert channel can be created to leak sensitive data and to recover them at distance. This paper focuses on the exchanges between the DDR memories and the SoC-FPGA.

II. Attack Scenario

The studied EM emanations occur in the form of pulses which are generated each time a transition from a state to another occurs [2]. In our case, the information leak is exacerbated by the state changes at the SoC-FPGA's I/O. Depending on the targeted I/O, an attacker could exploit the leak to potentially recover the initial data [1] or to create a covert-channel [3]. In this paper, communications between the SoC-FPGA and the DDR memory are considered. By writing or reading into the external memory, an attacker can control the corresponding EM pulses, leading to potential leaks of sensitive data. Similar work has been done on computer RAM [3] on which a malware was injected. Using this setup, the leaking RAM was able to generate a 1000 bits/s On-Off-Keying signal. In this work, a HT located inside the FPGA fabric is proposed to create a covert channel without software consideration [3].

A. DDR3L SDRAM Communication Protocol

DDR3L SDRAM is a high performance memory, widely used in many hardware platforms for data exchanges at very high throughput. At physical layer, multiple I/O are necessary, each one getting a specific role [4]. Data are sent on both positive and negative edges of the clock allowing a data transmission rate of twice the clock frequency. For example, the PYNQ-Z2 platform has a 525 MHz DDR3 clock frequency (T = 1.9 ns) resulting in 1.05 GHz data rate. The tracks for the read and write exchanges are called *DQ*. In our case, the DDR3 uses 16 *DQ* lines allowing 16 bits to be read or written every clock edge by burst of eight 16-bit data.

B. Hardware Trojan

The originality of this paper lies in the use of a malicious hardware IP allowing to shape the signal implemented into the logic fabric. The malicious IP implemented on the FPGA has a direct read and write access (AXI-Master interface) to the DDR through the High-Performance AXI port (HP AXI) of the SoC-FPGA. The attacker can then manipulate the data with the IP and control the shape of the resulting EM emanations accordingly. To perform an effective eavesdropping of the leaked information, the EM probe placement and polarity are decisive. According to the the SoC-FPGA package, the probe was placed on top of the *DQ* I/O to obtain the best quality of the signal.

III. Leak exploitation

The proposed attack is performed on a PYNQ-Z2 and depicted in Figure 1. The emanations are recovered with the RF Langer R-3-2 probe with a PA-203 amplifier and a Wavemaster 8620A oscilloscope to visualize the traces.

A. Preliminary experiments

The data request size must be adapted accordingly to the message that the attacker wants to transmit through the trojan. The write request is divided into packets, each one transporting 16 32-bit integers while the read request generates a single packet of the desired message length. For the attacker, it's

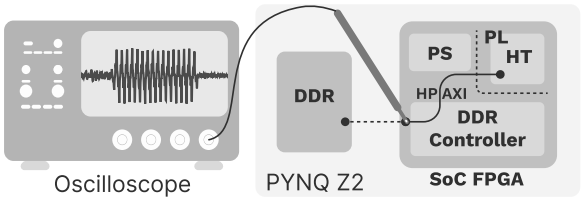


Fig. 1. Attack setup and scenario.

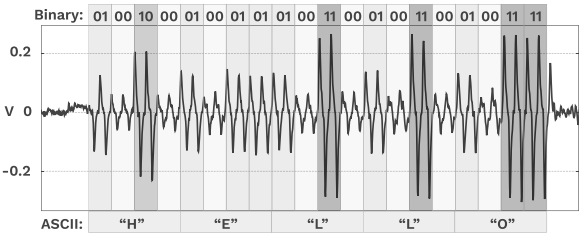


Fig. 3. Message successfully transmitted through the covert channel.

preferable to focus on the read request as the data is concatenated into a single transaction. By manipulating the data being exchanged, the attacker is able to create an emanation at the memory clock frequency and to modify its amplitude.

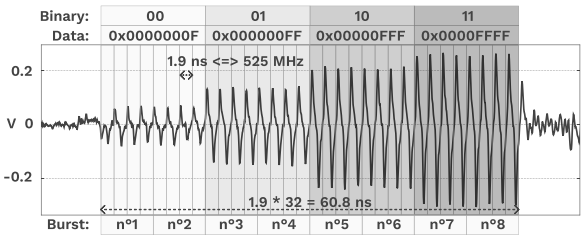


Fig. 2. Read packet emanation manipulation.

In our study, the DDR Clock frequency will act as carrier signal. For each state change, the generated pulse is shaped as a single peak. Depending on the change, positive (from 0 to 1) or negative (from 1 to 0), the peak direction is influenced (top or bottom). By continuously alternating between these two states, an emanation is created with the same period as the DDR clock. Otherwise, sending the same data does not create any pulse. The attacker has the capability to control the amplitude of the EM emanations by defining the number of switching bits in a transaction, which in our case, depends on the number of *DQ* tracks. For instance, if all the 16 tracks are switching simultaneously, the peak amplitude reaches its maximum with the value *0x0000FFFF*. Figure 2 depicts the corresponding EM emanations resulting from a read transaction. The amplitude of the emanation is associated to a binary symbol (e.g *00*, *01*, *10*, *11*). Every clock edge represents a 16-bits data exchange with the DDR, leading to a total transmission time of 60.8 ns. This creates a four-level amplitude modulation for which the carrier frequency corresponds to the DDR clock frequency.

B. EM Covert Channel

Figure 3 illustrates the resulting EM traces recorded on a PYNQ-Z2 board, in which the HT was configured to leak the message "Hello" through the covert channel with the DDR memory. Each binary symbol is identified from the signal amplitude on two clock periods. The resulting speed rate averages at 525 Mbits/s. For the word "Hello", the transmission is 76 ns long, since 40 integers are exchanged in total (five ASCII characters of 8 bits). The emations were successfully recovered and demodulated outside the chip.

C. Discussion

By using a HT, this paper brings a novel approach in comparison to [3]. Different modulations can be further investigated such as OOK, QAM, etc. Due to its low footprint, the HT can be hard to detect. Moreover, The HT could be used to leak critical information such as cryptographic keys directly from the FPGA fabric. Currently, the leaked data is recovered with a near-field EM probe but the attack range could be enhanced by using Software-Defined Radio (SDR) for which the major limitation is the sampling rate. For low end devices, the symbol identification time could be enlarged to match a lower sampling rate but this will decrease the transmission rate. One can consider that the HT would not be the only element accessing the DDR in real applications. The use of a preamble inserted to the sensitive data could also be considered as in [3] to make the covert transmission detection process easier.

IV. Conclusion

This paper shows that the SoC-FPGA I/O pins can be exploited as a way to create an AM covert channel. An attack was successfully performed by manipulating the data exchange with the DDR3L SDRAM to create a modulated signal carrying sensitive information. Here, only the I/O towards the DDR are studied, but it reveals that any other SoC-FPGA I/O pin could be used to perform this attack. Those leaks could be a critical security breach and must be investigated further.

V. Acknowledgments

This work is supported by the European Union through European Regional Development Fund (ERDF), Ministry of Higher Education and Research, CNRS, Brittany region, Conseils Départementaux d'Ille-et-Vilaine and Côtes d'Armor, Rennes Métropole, and Lannion Trégor Communauté, through the CPER Project CyMoCod.

References

[1] M. M. Thu, M. M. Réal, M. Pelcat, and P. Besnier, "Bus electrocardiogram: Vulnerability of soc-fpga internal axi bus to electromagnetic side-channel analysis," *2023 International Symposium on Electromagnetic Compatibility – EMC Europe*, 2023.

[2] M. G. Kuhn, "Compromising emanations: eavesdropping risks of computer displays," University of Cambridge, Computer Laboratory, Tech. Rep. UCAM-CL-TR-577, Dec. 2003.

[3] M. Guri, "Rambo: Leaking secrets from air-gap computers by spelling covert radio signals from computer ram," in *Secure IT Systems: 28th Nordic Conference, NordSec 2023, Oslo, Norway, November 16–17, 2023, Proceedings*. Berlin, Heidelberg: Springer-Verlag, 2023, p. 144–161.

[4] *8Gb: x4, x8, x16 DDR3L SDRAM*, Micron, 2015.

Parallel TIADC calibration for intermittent signal conversion

Grégoire LAPERT*†, Gerald CHARBONNIER*, Caroline LELANDAIS-PERRAULT†, Philippe BENABES†
*CEA, DAM DIF, Arpajon - France
Email: firstName.lastName@cea.fr
†Laboratoire de Génie Electrique et Electronique de Paris, CNRS, Université Paris-Saclay, CentraleSupélec, Gif-sur-Yvette - France
Sorbonne Université, CNRS, Laboratoire de Génie Electrique et Electronique de Paris, 75252, Paris, France
Email: firstName.lastName@centralesupelec.fr

***Abstract*—This article presents an innovative time-interleaved (TIADC) architecture capable of fully parallelizing TIADC foreground calibration and conversion. This architecture uses two identical TIADC split into groups. A specific input circuit stage, composed of bootstraps switches design in 28nm CMOS, alternately feeds the appropriate signal to each TIADC group. Effectively creating an hybrid background/foreground calibration solution for TIADC. A mixed-signal simulation of a 2.5-GS/s 11 bit 10x-interleaved analog to digital converter (ADC) achieves a SNDR/SFDR of 64.00/77.72 dB at 700MHz after skew mismatch calibration.**

***Index Terms*—Analog-digital calibration, analog-to-digital converter(ADC), CMOS, GHZ-sample rate, high resolution, time-interleaved(TI) ADC, mismatch calibration, foreground calibration, background calibration.**

I. Introduction

A time-interleaved architecture is selected to design an analog-to-digital converter with a sampling frequency of 2.5-GS/s with a resolution of 11 bits and a SNDR target of 60 dB. This architecture's downside is the reduction of conversion accuracy due to sub converter offset, gain and sampling time mismatches. In this paper, focus is brought on the continuous estimation of time skew mismatch when the ADC samples an intermittent signal. We assume gain and offset mismatches are previously corrected. We present a TIADC architecture that samples an intermittent signal and responds to the following constraints :

- The TIADC has to be calibrated at all times to offer the maximum SNDR whenever the signal starts
- The input signal has to be converted continuously
- The input signal is arbitrary. It has no known properties except that its band is in the range of 100 MHz to 700 MHz
- The input signal is intermittent

The literature offers various approaches to estimate and calibrate sub ADCs mismatches [1]. Yet, none of these methods provides a solution for the continuous calibration of a TIADC that samples an intermittent signal with unknown properties. In this paper, we propose an innovative TIADC architecture that fully parallelizes signal conversion and TIADC calibration.

II. TIADC, mismatches and calibration

A. TIADC mismatches

B. Mismatches calibration

The effects of mismatches led researchers to develop calibration methods that alleviate their impact on conversion quality. In 1987, K.Poulton [2] proposed a common sample and hold front stage achieving high speed sampling before the sub converters. This method completely removes the impact of time skew by guaranteeing uniform sampling of the input signal. But, as sampling frequencies get higher, the design of such high speed samplers becomes more complex, mainly due to non-linearities [3]

In 2011, B.Yu uses a foreground calibration method. A known sine wave is fed to the TIADC to estimate skew mismatches [4]. He uses the correlation of the difference between adjacent sub converters samples to detect sample time error relative to a reference channel.

The need for continuous conversion of intermittent signals prevents the use of a foreground calibration signal, thus rendering these methods unfit for the aforementioned constraints.

Background methods calibrate mismatches without interruption of the signal conversion. In 2014, N.Le Dortz proposes to use the correlation between a converter and its derivative to estimate the skew [5]. This method is effective but requires a wide sense stationary (WSS) input signal, and the use of an accurate derivative filter.

By its nature, an intermittent signal is not always present at the TIADC input, and thus prevents the use of these background calibration solutions.

The technique proposed in this paper parallelizes the mismatchs estimations and the signal conversion, thus allowing a continuous foreground calibration of sub-converters mismatches, even when the useful signal is unavailable.

III. Circuit design

A. TIADC architecture

A reduced architecture with four ADCs, representing the four groups is presented in Figure 1. The timing diagram in Figure 2 represents a full calibration cycle.

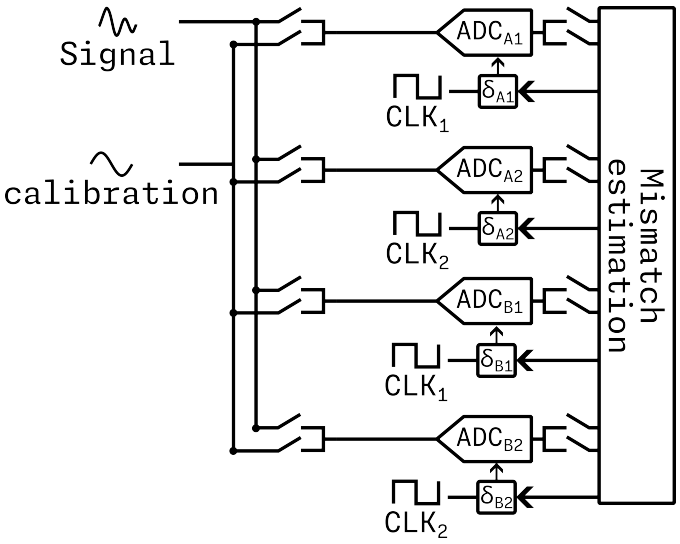


Fig. 1. Reduced architecture with four ADCs

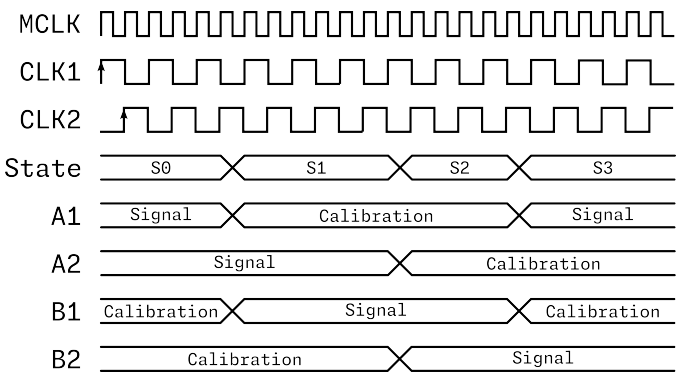


Fig. 2. Timing diagram of the reduced architecture

The architecture uses two TIADC (A and B) with common clocks. While TIADC A input is connected to the signal, TIADC B receives the calibration signal. After some time, an estimation and correction of TIADC B's mismatches is performed and TIADC A enter calibration while TIADC B convert the signal. To prevent any sample loss during the input signal switch, both TIADC are split into two groups.

To reduce the signal deformation due to switch non-linearities a bootstrap switch architecture is directly taken from [6].

IV. Simulation results

Tests are performed on a mixed signal simulation of the architecture seen in Figure 1. Two sine waves are used as input signal, one at $f_{in1} = 704.75MHz$ as the useful signal and another at $f_{in2} = 899.75MHz$ for the calibration signal. All simulations are performed using a 200 mV amplitude signal to reduce the impact of sample and hold non-linearities. Skew mismatch is then artificially introduced by delaying signals with a skew standard deviation of $\sigma_s = 300fs$. The power spectrum before and after calibration is presented in Figure 3.

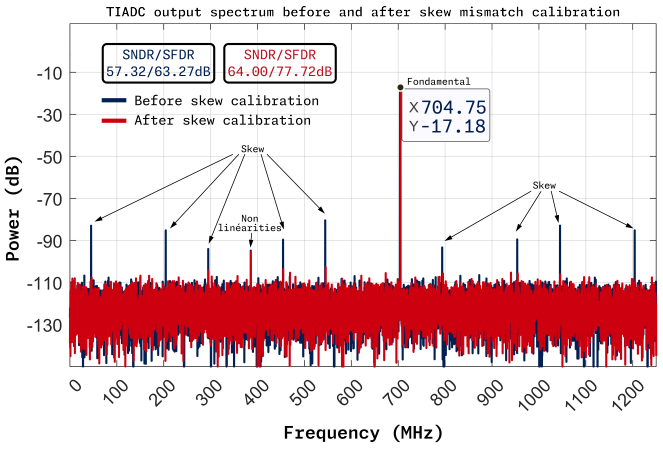


Fig. 3. Output spectrum with and without skew mismatch

The SNDR/SFDR is 57.32/63.27 dB before calibration and 64.00/77.72 dB after calibration. Therefore, we demonstrate that this parallelization architecture does not impact conversion quality.

V. Conclusion

This paper demonstrates the effectiveness of a new TIADC architecture that parallelizes foreground mismatch calibration and signal conversion. This architecture is especially useful to solve constraints such as the need for continuous conversion with optimum SNDR of an intermittent and unknown signal. A simulation of the aforementioned architecture in $28nm$ FDSOI proves its efficiency for a $2.5GHz$ 11 bits TIADC : it reaches a SNDR of 64.00 dB and a SFDR of 77.72 dB after calibration. This method's drawback is the need to double the number of ADCs which increases the energy consumption by as much.

References

[1] C. R. Parkey and W. B. Mikhael, "Time interleaved analog to digital converters: Tutorial 44," in IEEE Instrumentation Measurement Magazine, vol. 16, no. 6, pp. 42-51, December 2013, doi: 10.1109/MIM.2013.6704972.

[2] K. Poulton, J. J. Corcoran and T. Hornak, "A 1-GHz 6-bit ADC system," in IEEE Journal of Solid-State Circuits, vol. 22, no. 6, pp. 962-970, Dec. 1987, doi: 10.1109/JSSC.1987.1052844.

[3] J. Jang, Y. Chae and T. W. Kim, "A 1.5 V 2 GS/s 82.1 dB-SFDR Track and Hold Circuit Based on the Time-Divided Post-Distortion Cancelation Technique," in IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 69, no. 12, pp. 4719-4723, Dec. 2022, doi: 10.1109/TCSII.2022.3204031.

[4] B. Yu et al., "A 14-bit 200-MS/s time-interleaved ADC with sample-time error detection and cancelation," IEEE Asian Solid-State Circuits Conference 2011, Jeju, Korea (South), 2011, pp. 349-352, doi: 10.1109/ASSCC.2011.6123586.

[5] N. Le Dortz et al., "22.5 A 1.62GS/s time-interleaved SAR ADC with digital background mismatch calibration achieving interleaving spurs below 70dBFS," 2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC), San Francisco, CA, USA, 2014, pp. 386-388, doi: 10.1109/ISSCC.2014.6757481.

[6] C. -C. Liu, S. -J. Chang, G. -Y. Huang and Y. -Z. Lin, "A 10-bit 50-MS/s SAR ADC With a Monotonic Capacitor Switching Procedure," in IEEE Journal of Solid-State Circuits, vol. 45, no. 4, pp. 731-740, April 2010, doi: 10.1109/JSSC.2010.2042254. 2021.9451049.

Security of Dynamically Reconfigurable RISC-V Systems: I/O Attack Focus

Aya Jendoubi
INSA Rennes
CNRS, IETR, UMR 6164
Rennes, France

Jean-Christophe Prévotet
INSA Rennes
CNRS, IETR, UMR 6164
Rennes, France

Philippe Tanguy
Université Bretagne-Sud
Lab-STICC, UMR CNRS 6285
Lorient, France

Pascal Cotret
ENSTA
Lab-STICC, UMR CNRS 6285
Brest, France

***Abstract*—Dynamic Partial Reconfiguration (DPR) enhances flexibility in modern hardware but introduces security risks. This work demonstrates how a Malicious Hardware Accelerator (MHA) can exploit Direct Memory Access (DMA) to bypass Input Output Memory Management Unit (IOMMU) protections through device ID manipulation, enabling unauthorized memory access. This vulnerability exposes a fundamental security gap in the management of dynamically reconfigurable systems. By highlighting this issue and proposing mitigation strategies, we provide a conceptual framework to guide the development of security mechanisms for dynamically adaptable architectures.**

***Index Terms*—component, formatting, style, styling, insert.**

I. Introduction

Modern hardware architectures integrate accelerators and coprocessors to enhance efficiency by offloading tasks from the CPU. DPR further improves adaptability by enabling runtime hardware modifications without system downtime. However, increasing system complexity and interconnectivity introduce security risks, particularly in Input/Output (I/O) operations. DMA can be exploited by malicious devices to bypass security mechanisms, making the IOMMU essential for isolating device access.

Despite its protections, IOMMU faces new challenges in dynamically reconfigurable environments, where hardware modifications expand the attack surface. This study analyzes security risks in IOMMU-based systems, particularly in a cloud gateway deploying Field-Programmable Gate Array (FPGA) accelerators. We identify a novel attack where a malicious actor impersonates a legitimate hardware accelerator by stealing its Device ID, bypassing IOMMU protections to access restricted memory. The paper explores this threat, reviews existing research, and discusses potential mitigation strategies.

II. Background

A. IOMMU and Memory Protection

As modern applications demand greater processing power and efficiency, systems increasingly rely on specialized hardware accelerators to offload computational tasks.

In such systems, where I/O peripherals often require DMA access, the IOMMU plays a crucial role in securing memory by enforcing isolation between devices. It intercepts DMA requests from I/O peripherals (or, in our case, hardware accelerators), performs permission checks, and translates Input Output Virtual Address (IOVA) into physical addresses. Since these operations depend on the Device ID and virtual address, the IOMMU employs caches such as Device Directory Table Cache (DDTC), Process Directory Table Cache (PDTC), and Input Output Translation Look-aside Buffer (IOTLB) to optimize address translation efficiency.

The work presented in this paper was realized in the frame of the TrustGW project number ANR-21-CE39-0005, supported by a grant of the French National Research Agency (ANR).

B. Case Study: RISC-V IOMMU-based System

The studied system consists of a RISC-V CPU, IOMMU, hypervisor, and FPGA-based Dynamically Reconfigurable Area (DRA) where accelerators are dynamically reconfigured and interconnected via Advanced eXtensible Interface (AXI) bus. The IOMMU acts as the central security mechanism, regulating access to shared memory and enforcing isolation between accelerators.

In this architecture, as it is loaded, each hardware accelerator is assigned a new Device ID, which is used by the IOMMU to manage address translation requests. However, the study reveals a critical flaw: the IOMMU does not verify the authenticity of these Device IDs. This creates an attack vector where a MHA can impersonate a Legitimate Hardware Accelerator (LHA) by spoofing its Device ID. By doing so, the MHA can submit fraudulent DMA requests that the IOMMU processes as if they were from a trusted device, granting unauthorized access to protected memory regions.

This vulnerability is particularly concerning in dynamic environments where reconfigurable hardware changes frequently. Since new accelerators can be introduced at runtime, attackers have an opportunity to exploit this process by injecting malicious configurations that compromise system security.

C. Related Work

The security of IOMMU-based architectures has been a focus of multiple studies. Research has demonstrated that IOMMUs play a critical role in securing memory access in virtualized environments, yet their implementation may contain weaknesses that allow attackers to bypass protection mechanisms. Works such as [1] and [2] highlight how attackers can manipulate IOMMU configurations or exploit cache incon-

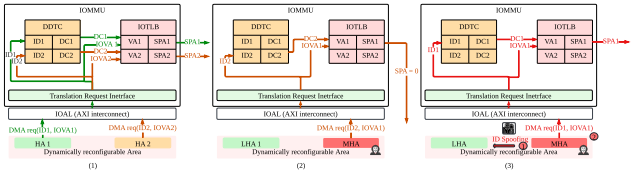


Fig. 1. Waveform analysis of Device ID Spoofing in IOMMU exploitation.

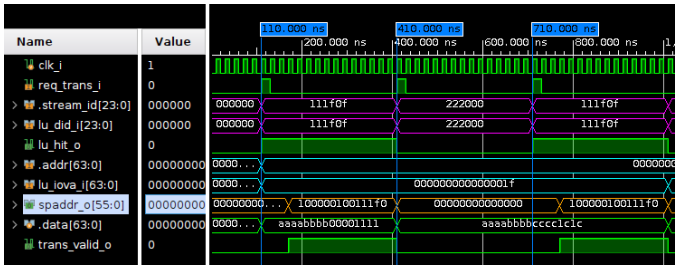


Fig. 2. Waveform analysis of Device ID Spoofing in IOMMU exploitation.

sistencies, such as stale IOTLB entries, to gain unauthorized access to protected memory regions.

Several studies such as [3] and [4] have also explored security vulnerabilities in dynamically reconfigurable hardware, particularly in FPGA-based architectures. Research on FPGA security has shown that multi-tenant environments introduce significant risks, as reconfigurable logic can be exploited for side-channel attacks, hardware trojans, and resource contention issues as shown in [5].

A particularly relevant approach to mitigating these threats is the use of FPGA Shells—predefined, secure regions within an FPGA that isolate dynamic reconfiguration from security-sensitive operations. Notable implementations include Coyote [6], an open-source framework enabling secure multiplexing of FPGA resources. Shells provide structured interfaces and restrict direct access to system peripherals, however, they are not IOMMU-based solutions as they focus rather on partitioning FPGA resources and isolating reconfigurable logic rather than managing DMA transactions or enforcing memory access control at the system level.

III. Threat Model and Attack Vector

Figure 1 describes an attack that exploits weak device ID verification in IOMMU-protected systems. In dynamically reconfigurable environments, a malicious hardware accelerator (MHA) can impersonate legitimate devices by spoofing their IDs, gaining unauthorized access to restricted memory. The attack begins with passive monitoring of unencrypted AXI transactions to capture active device IDs. The MHA then injects a spoofed DMA request using a stolen ID, exploiting the lack of strict authentication in the IOMMU. Since the IOMMU only checks if an ID exists in the DDTC without verifying its authenticity, it processes the request, granting illicit access. By strategically synchronizing spoofed requests with legitimate ones, the attacker can evade detection, manipulate arbitration delays, and systematically compromise memory integrity, exposing critical vulnerabilities in modern IOMMU implementations.

IV. Demonstration and Analysis

We simulated a RISC-V-based system using Xilinx Vivado to validate the attack. This demonstration is based on an implementation compliant with the RISC-V IOMMU specification version 1.0 [7]. As shown in Figure 2, a LHA successfully accessed protected memory using its valid Device ID (`lu_did_i` = `111f0f`), while a MHA was denied access when using its own ID. However, when the MHA spoofed the LHA's Device ID, the IOMMU incorrectly authenticated the request, allowing unauthorized access to protected memory (`spaddr_o` = `100000100111f0`). This highlights the need for stronger device ID authentication mechanisms.

V. Conclusion and Perspectives

The analysis highlights a critical security flaw in reconfigurable, multi-tenant architectures relying on an IOMMU for access control. While FPGA shells enforce isolation by restricting address spaces within statically pre-defined reconfigurable slots, they do not verify the integrity of the deployed IPs or authenticate accelerator IDs at runtime. This gap allows malicious devices to spoof legitimate IDs and gain unauthorized memory access due to the IOMMU's lack of built-in authentication. To address this, potential solutions include extending the IOMMU with ID verification or introducing an external authentication module. Approaches such as cryptographic or hardware-based ID verification, device ID locking, or an ID wrapper at the arbitration and translation stages could enhance security. However, these solutions introduce trade-offs in complexity, performance, and hardware overhead, emphasizing the need for standardized security mechanisms.

References

[1] B. Morgan, E. Alata, V. Nicomette, and M. Kaaniche, "Bypassing iommu protection against i/o attacks," in *2016 Seventh Latin-American Symposium on Dependable Computing (LADC)*, 2016, pp. 145–150.

[2] T. Tiemann, Z. Weissman, T. Eisenbarth, and B. Sunar, "Iotlb-sc: An accelerator-independent leakage source in modern cloud systems," in *Proceedings of the ACM Asia Conference on Computer and Communications Security*, ser. ASIA CCS '23. ACM, Jul. 2023. [Online]. Available: http://dx.doi.org/10.1145/3579856.3582838

[3] R. Elnaggar, R. Karri, and K. Chakrabarty, "Multi-tenant fpga-based reconfigurable systems: Attacks and defenses," in *2019 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2019, pp. 7–12.

[4] J. M. Mbongue, S. K. Saha, and C. Bobda, "A security architecture for domain isolation in multi-tenant cloud fpgas," in *2021 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, 2021, pp. 290–295.

[5] E. M. Benhani, C. Marchand, A. Aubert, and L. Bossuet, "On the security evaluation of the arm trustzone extension in a heterogeneous soc," in *2017 30th IEEE International System-on-Chip Conference (SOCC)*, 2017, pp. 108–113.

[6] D. Korolija, T. Roscoe, and G. Alonso, "Do OS abstractions make sense on FPGAs?" in *14th USENIX Symposium on Operating Systems Design and Implementation (OSDI 20)*. USENIX Association, Nov. 2020, pp. 991–1010. [Online]. Available: https://www.usenix.org/conference/osdi20/presentation/roscoe

[7] Zero Day Labs, "RISC-V IOMMU," 2025, accessed: 2025-03-12. [Online]. Available: https://github.com/zero-day-labs/riscv-iommu

Digital Designs comparative analysis on key parameters for 18, 28 and 40 nm technology nodes

Amelie Poullot * †, Sylvain Clerc*, Filipe Pouget *, Didier Gueze *, Lioua Labrak † and Ian O'Connor †
*STMicroelectronics, 850 Rue Jean Monnet, F-38926 Crolles Cedex, France
†INL - Institut des Nanotechnologies de Lyon, 3 Rue Enrico Fermi, 69100 Villeurbanne, Lyon, France
email: amelie.poullot@st.com

***Abstract*—This study aims to identify the root causes that limit the performance of digital designs in terms of power consumption, frequency, and area across different technology nodes (18 nm, 28 nm, and 40 nm). The approach involves building a comprehensive database of metrics derived from a diverse collection of designs, which will be analyzed based on the impact of Complementary Metal Oxyde Semiconductor (CMOS) technology scaling and design choices.**

***Index Terms*—Artificial Intelligence (IA), database management, technology nodes, Complementary Metal Oxyde Semiconductor (CMOS) scaling**

I. Introduction

The primary objective of this study is to identify the root causes that limit the performance on power consumption, frequency, or area of digital designs. To achieve this, we aim to build a comprehensive database of metrics derived from a diverse collection of designs, encompassing the wide range of possible choices in placement and routing [1], [2].

This database will include two main categories of metrics. The first category will focus on the invariants of technology nodes, enabling the identification of designs that deviate from expected norms. The second category will assess the impact of specific design choices on floorplan and placement , quality of clock tree, and signal integrity.

Once the database is established, an encoder-decoder system will be employed to compress the data, providing a more abstract representation of the design characteristics [3], [4]. Subsequently, a classifier will be trained by incorporating designs whose limitations have been evaluated by experts. This will enable the characterization of design limitations in comparison to previously evaluated designs, facilitating a deeper understanding of performance bottlenecks.

II. Experimental Dataset

This study utilizes a diverse set of designs to analyze key factors affecting digital design performance. The selection criteria include the presence of macros, track heights, technology nodes, operating frequencies, and design entry points into the placement and routing flow.

Designs with and without macros were chosen to study their impact on routing congestion, particularly at their edges where routing must be circumvented. Various track heights were selected to test the claim from [9] that larger cells provide higher drive capabilities but consume more area and power.

Different technology nodes were included to examine the effects of scaling and foreseen by models such as "Dennard scaling", "voltage scaling" and "lateral scaling" [5]–[8]. Designs with varying operating frequencies were selected to study their impact on power consumption. Additionally, designs entering the flow from RTL or gate-level netlists were included to assess the influence of entry points.

The designs are summarized in Table I.

III. Key metrics analysis

In this section, we present several key metrics to illustrate the complexity in identifying factors that affect overall performance.

A. Invariant Coupling Capacity Across Technologies

Coupling capacitance significantly impacts signal integrity. High coupling capacitance between nets allows the aggressor net to alter the victim net's signal, potentially degrading design functionality or requiring a reduction in the maximum operating frequency.

As technology advances, wires are placed closer together. However, by reducing the gate length of transistors and making necessary geometric adjustments, signal integrity can be maintained, preserving design functionality without impacting timing.

According to [7], scaling should not affect crosstalk, which is confirmed by Figure 1. The figure shows that coupling capacitance relative to ground capacitance remains consistent across different technology nodes.

This invariant helps identify designs that deviate from the norm. For instance, Design E deviates due to its much lower density (2.4%) compared to other designs (average 70%). Further analysis reveals Design E is limited by the I/O floorplan.

B. Invariant of Mean Net Resistance by Technology

Analyzing the distribution of point-to-point resistances across various designs reveals the impact of transistor scaling across different technology nodes. Table II presents the min, mean and max net resistance values normalized to the mean of design A in CMOS040 technology for the designs set.

The mean net resistance increases by 98% from CMOS040 to FDSOI28 and by 55% from FDSOI28 to CMOSP18. This trend illustrates the effect of transistor scaling: wire resistance

TABLE I
DESIGNS AND TECHNOLOGY NODES

Design	Techno	Tracks	Macros	Frequency	Entry	Description
Design A	P18; FDSOI28; CMOS40	9T; 8T; 9T	2 SRAM	300 MHz	RTL	Open source CPU, 1k flip-flops, 8x32 + 16x32 SRAM
Design B	P18	B1: 8T, B2: 9T	0	150 MHz	RTL	Proprietary CPU with 1k flip-flops, logic only
Design C	FDSOI28	C1: 8T; C2: 8T	0	500 MHz; 1 GHz	Gate	Direct Memory Access Control
Design D	FDSOI28	8T	10	500 MHz	Gate	Video decoder
Design E	P18	9T	10	350 MHz	Gate	Top level plus 144 instances I/O ring, 30k flip-flops

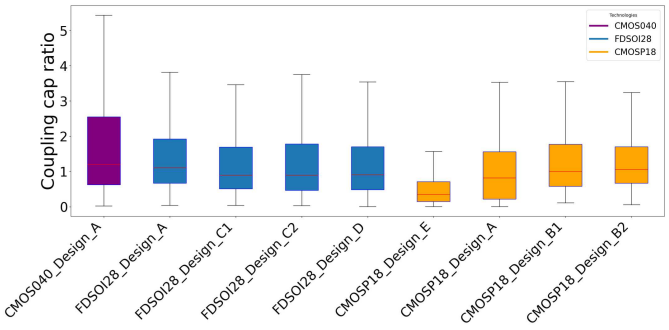


Fig. 1. Coupling Capacitance Relative to Ground Capacitance of Victim Nets

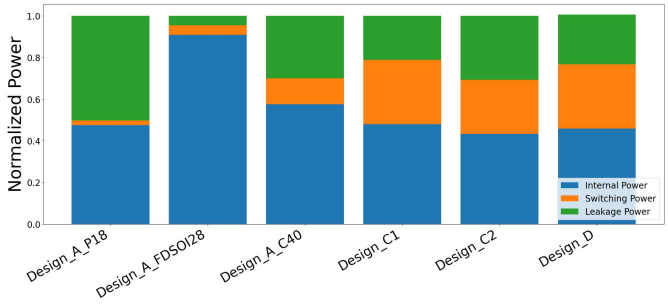


Fig. 2. Design's power proportions normalized to their total power

TABLE II
NORMALIZED POINT TO POINT RESISTANCE

Design	Techno	min	mean	max	Techno mean
Design_A	CMOS040	0,00	1,00	13,29	1,00
Design_A	FDSOI28	0,00	2,17	24,05	1,98
Design_C1		0,00	1,74	30,67	
Design_C2		0,00	1,74	34,27	
Design_D1		0,00	2,34	82,04	
Design_E	CMOSP18	0,00	3,06	68,55	3,08
Design_A		0,13	3,36	41,16	
Design_B1		0,13	3,01	22,23	
Design_B2		0,13	2,89	18,85	

increases as technology nodes advance. This increase is due to methods used to maintain stable coupling capacitance.

Furthermore, between designs B1 and B2, there is a variation in the number of tracks used for routing, yet no significant impact on the distribution of point-to-point resistance is observed. Similarly, designs C1 and C2, which operate at different frequencies, show no notable difference in the point-to-point resistance distribution.

C. Power Distribution Across Designs

This graph presents the normalized data for Internal Power, Switching Power, and Leakage Power, each expressed as a fraction of their respective Total Power.

At first glance, it is apparent in the P18 technology node, that reducing transistor Length (L) impacts the leakage.

When comparing designs C1 and C2, analysis shows the total power almost triples as the frequency doubles. Additionally, the proportion of leakage power also increases, which is a common occurrence with higher operating frequencies.

For the other designs, there are numerous parameters not accounted for in this analysis, making it challenging to draw definitive conclusions from the results.

IV. CONCLUSION

Leveraging design data from the Place and Route and SignOff phases enables precise diagnostics for evaluating implementation quality and identifying performance bottlenecks.

Key insights from the selected metrics include the importance of invariants like coupling capacitance and mean net resistance for maintaining signal integrity and minimizing power consumption. Power distribution metrics highlight the impact of operating frequency on total power, especially the increase in leakage power at higher frequencies.

These metrics will facilitate the encoding of design characteristics, enabling the use of machine learning in EDA. Training classifiers with these metrics will help develop predictive models to identify performance limitations and guide optimizations, enhancing our ability to improve digital designs efficiently and reliably.

REFERENCES

[1] Yu, Bei Machine Learning in EDA: When and How 2023

[2] Liu, Mingjie / Ene, Teodor-Dumitru / Kirby, Robert / Cheng, Chris / Pinckney, Nathaniel / Liang, Rongjian /others Chipnemo: Domain-adapted llms for chip design 2023

[3] Kazda, Michael / Monkowski, Michael / Antony, George APEX: Recommending Design Flow Parameters Using a Variational Autoencoder 2023

[4] Luo, Donger / Sun, Qi / Xu, Qi / Chen, Tinghuan / Geng, Hao Attention-Based EDA Tool Parameter Explorer: From Hybrid Parameters to Multi-QoR metrics 2024

[5] R. Dennard, et al., "Design of ion-implanted MOSFETs with very small physical dimensions," IEEE Journal of Solid-State Circuits, vol. SC-9, no. 5, pp. 256-268, Oct. 1974.

[6] M. Bohr, "A 30 Year Retrospective on Dennard's MOSFET Scaling Paper," IEEE SSCS Newsletter, Winter 2007.

[7] Weste, Neil / Harris, David CMOS VLSI Design: A Circuits and Systems Perspective 2010, 4th. edition

[8] Taur, Yuan / Ning, Tak H. Fundamentals of Modern VLSI Devices 2021, 3. edition

[9] Cao, Linan / Bale, Simon J. / Trefzer, Martin A. Multi-Objective Digital Design Optimization via Improved Drive Granularity Standard Cells 2021

Physical Unclonable Functions (PUFs): A Novel Approach for Generating Unique and Secure Signatures in Electronic Devices

Vasilii Kulagin
Univ. Grenoble Alpes, CNRS, Grenoble INP*, TIMA, 38000 Grenoble, France
vasilii.kulagin@univ-grenoble-alpes.fr

***Abstract*—Physical Unclonable Functions (PUFs) are integral for generating unique signatures, secret keys, and device identification, leveraging inherent manufacturing process variability. Mathematically defined as functions linking inputs (challenges) to outputs (responses), PUFs exhibit random properties. Key properties for high-quality PUFs include intra-device entropy (random distribution of responses within the same circuit), inter-device entropy (random distribution across different circuits for identical challenges), and reliability (response consistency for identical challenges and the same circuit). Inter-device entropy and reliability may be influenced by design discrepancies, systematic variability, noise, and aging.**

***Index Terms*—Physical Unclonable Functions, Ring Oscillator, Reliability, Entropy**

I. Introduction

Physical Unclonable Functions (PUFs) exploit the inherent manufacturing variability of electronic devices to provide unique cryptographic signatures [1]. In mathematical terms, a PUF is defined as a function with random properties linking a challenge to a response [2]. This principle is further exploited in various architectures, including delay-based PUFs [3], [4]—with the Ring Oscillator PUF (RO-PUF), shown in figure 1, being one of the most widely studied.

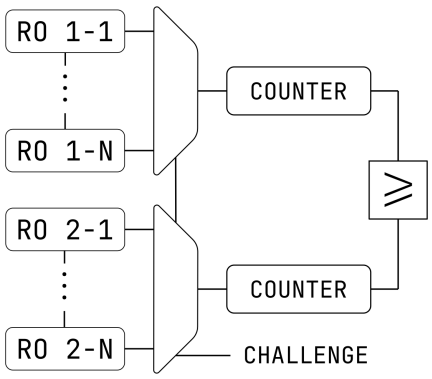


Fig. 1. Schematic representation of the Ring Oscillator based PUF

*Institut National Polytechnique Grenoble Alpes

Despite substantial progress in developing PUFs, significant challenges remain. The principal issues addressed in this work include the critical trade-off between intra-device reliability and inter-device entropy. While prior studies have successfully enhanced one metric through error correcting codes or filtering methods, these approaches often neglect the simultaneous optimization of both aspects. Moreover, many existing solutions rely heavily on simulation-based validations, leaving a gap in comprehensive experimental evaluation on industrial-grade platforms.

The urgency of these challenges is underscored by the increasing need for tamper-proof identification mechanisms in applications ranging from secure communications to critical infrastructure. Addressing these problems is essential not only to enhance device security but also to overcome the limitations of traditional approaches that compromise either reliability or randomness. This work introduces a novel methodology that dynamically refines filtering thresholds based on real-world experimental data, thereby ensuring both high reliability and sufficient entropy. By leveraging multidisciplinary techniques, including statistical analysis, simulation, and hardware prototyping on FPGA platforms, this research offers an integrated solution that bridges the gap between theoretical models and practical implementations.

II. State of the art

Current literature on PUFs emphasizes their pivotal role in hardware security by providing unique identifiers derived from manufacturing variability [5]. Two critical performance metrics for PUFs are reliability, which measures the consistency of responses under varying conditions, and entropy, which gauges the randomness and unpredictability of these responses. In order to achieve high-quality PUFs, the following key properties need to be ensured:

1) Random distribution of responses within the same circuit for different challenges (referred to as intra-device entropy, assessed by Uniformity Per Device).
2) Random distribution of responses across different circuits for identical challenges (known as inter-device entropy, evaluated through Uniformity Per Challenge).
3) The consistency of responses, meaning that for identical challenges and the same circuit, the response remains constant (evaluated by Reliability).

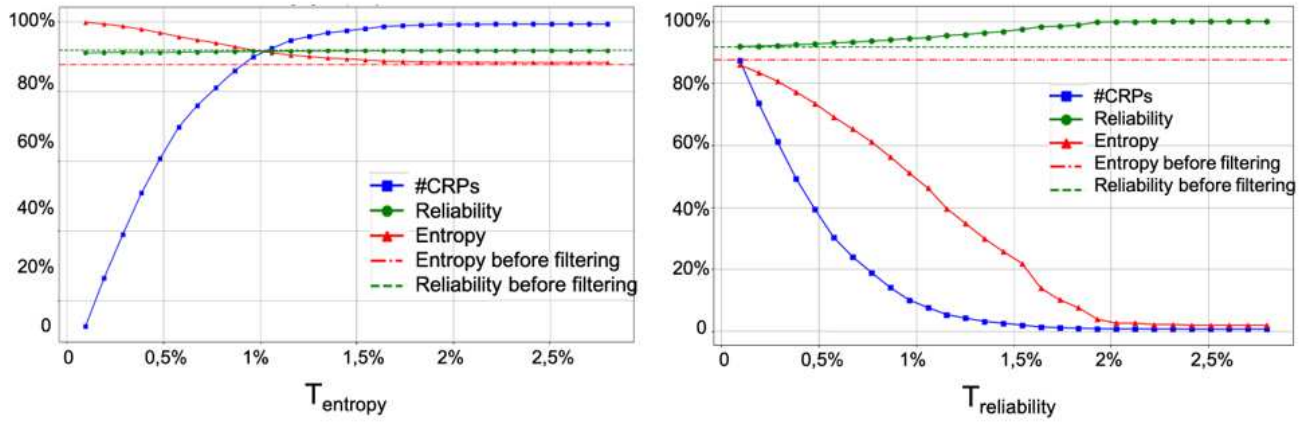


Fig. 2. Effect of filtering CRPs with: 1. high frequency difference values for entropy improvement 2. low frequency difference values for reliability improvement.

Several approaches have been explored to enhance PUF performance. Error Correcting Codes (ECC) introduce redundancy to correct unreliable responses, as demonstrated by Maes et al. [6]. While effective in improving reliability, ECC methods incur significant overhead in terms of area and power consumption and do not address entropy deficiencies. Alternatively, filtering techniques, which exclude PUF responses below a predefined reliability threshold, have been shown to enhance performance [7]. Schaub et al. [8] compared ECC and filtering approaches, concluding that filtering can be more efficient for reliability enhancement. However, these techniques demand extensive characterization under different environmental and aging conditions and often overlook the impact on entropy [9], [10].

In contrast, entropy improvement in PUFs has received comparatively less attention. A recent study specific to RO-PUFs highlighted that ring oscillators with very high frequency differences may exhibit low entropy due to systematic variability or design issues, such as proximity to power lines causing imbalanced frequency distributions [11]. As shown in Figure 2, adjusting filtering thresholds can significantly impact reliability and entropy in opposite ways: focusing on reliability typically reduces the pool of valid CRPs and hence lowers overall entropy, while emphasizing entropy can compromise reliability by allowing less stable CRPs. This interplay underscores the necessity of a balanced approach when designing robust PUF solutions.

The proposed work builds upon these findings by integrating a dynamic filtering strategy validated through industrial-grade datasets and extensive experiments on FPGA platforms. This approach aims to simultaneously optimize reliability and entropy, addressing the limitations observed in both ECC and traditional filtering techniques. In terms of expected impact, the research is poised to deliver significant advancements in hardware security. In the short term, the development of a versatile test platform and the refined methodology are expected to enhance industrial competitiveness by providing a secure, scalable solution for device identification. Over the long term, this innovative framework is anticipated to establish a foundation for a new line of scientific inquiry in secure electronic systems, influencing both academic research and practical applications in safety-critical environments.

REFERENCES

[1] G. E. Suh and S. Devadas, "Physical unclonable functions for device authentication and secret key generation," in *Proceedings of the 44th Annual Design Automation Conference*, ser. DAC '07, San Diego, California: Association for Computing Machinery, 2007, pp. 9–14, ISBN: 9781595936271. DOI: 10.1145/1278480.1278484. [Online]. Available: https://doi.org/10.1145/1278480.1278484.

[2] R. Pappu, B. Recht, J. Taylor, and N. Gershenfeld, "Physical one-way functions," *Science*, vol. 297, no. 5589, pp. 2026–2030, 2002.

[3] B. Gassend, D. Clarke, M. Van Dijk, and S. Devadas, "Delay-based circuit authentication and applications," in *Proceedings of the 2003 ACM symposium on Applied computing*, 2003, pp. 294–301.

[4] Z. Cherif, J.-L. Danger, F. Lozac'h, Y. Mathieu, and L. Bossuet, "Evaluation of delay pufs on cmos 65 nm technology: Asic vs fpga," in *Proceedings of the 2nd International Workshop on Hardware and Architectural Support for Security and Privacy*, 2013, pp. 1–8.

[5] A. Maiti, J. Casarona, L. McHale, and P. Schaumont, "A large scale characterization of ro-puf," in *2010 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, IEEE, 2010, pp. 94–99.

[6] R. Maes, A. Van Herrewege, and I. Verbauwhede, "Pufky: A fully functional puf-based cryptographic key generator," in *International Workshop on Cryptographic Hardware and Embedded Systems*, Springer, 2012, pp. 302–319.

[7] M. Bhargava and K. Mai, "An efficient reliable puf-based cryptographic key generator in 65nm cmos," in *2014 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, IEEE, 2014, pp. 1–6.

[8] A. Schaub, J.-L. Danger, O. Rioul, and S. Guilley, "The big picture of delay-puf dependability," in *2020 European Conference on Circuit Theory and Design (ECCTD)*, IEEE, 2020, pp. 1–4.

[9] A. Schaub, J.-L. Danger, S. Guilley, and O. Rioul, "An improved analysis of reliability and entropy for delay pufs," in *2018 21st Euromicro Conference on Digital System Design (DSD)*, 2018, pp. 553–560. DOI: 10.1109/DSD.2018.00096.

[10] H. Martin et al, "On the reliability of the ring oscillator physically unclonable functions," in *2019 IEEE 4th International Verification and Security Workshop (IVSW)*, IEEE, 2019, pp. 25–30.

[11] S. Vinagrero Gutierrez, G. Di Natale, and E.-I. Vatajelu, "On-line method to limit unreliability and bit-aliasing in ro-puf," in *2023 IEEE 29th International Symposium on On-Line Testing and Robust System Design (IOLTS)*, IEEE, 2023, pp. 1–6.

Méthodologie de synthèse de très haut niveau pour des architectures hétérogènes logicielles et matérielles

Gaëtan Lounes, Robin Gerzaguet, Matthieu Gautier
Univ Rennes, CNRS, IRISA
prenom.nom@irisa.fr

***Abstract*—L'utilisation croissante des circuits logiques programmables (Field Programmable Gate Arrays, ou FPGA) a permis l'accélération matérielle d'un large éventail d'applications. L'introduction de la synthèse de haut niveau (High Level Synthesis, ou HLS) a considérablement simplifié leur conception et permet l'exploration d'applications complexes. Cependant, la HLS demande encore une solide connaissance du matériel ciblé et repose sur des langages spécifiques. De nouveaux outils HLS s'appuie ainsi sur des environnements de compilation plus expressifs tels que la représentation intermédiaire à plusieurs niveaux (Multi-Level Intermediate Representation, ou MLIR). Dans ce contexte, nous visons à utiliser le langage de programmation Julia comme une interface flexible pour une HLS basée sur MLIR, en tirant parti de son système de types, de son infrastructure modulaire et de ses bibliothèques riches. Dans cet article, nous introduisons un nouveau compilateur qui exploite l'infrastructure de compilation de Julia afin de générer du code MLIR, facilitant ainsi l'accélération matérielle. L'intérêt de cette approche a été démontré pour du prototypage rapide et de l'exploration des espaces de types pour des algorithmes numériques.**

***Index Terms*—Synthèse de haut-niveau, Langage Julia, MLIR**

I. Introduction

Les Field Programmable Gate Arrays (FPGA) sont une classe de matériel configurable qui a démontré son efficacité pour des applications nécessitant une faible latence et un haut débit, telles que le traitement du signal ou les réseaux de neurones. Ils sont configurés à l'aide de langages de description matérielle (HDL) comme le VHDL. Toutefois, cette approche n'est pas idéale pour le développement rapide d'algorithmes ni pour l'exploration de solutions. La synthèse de haut niveau (High Level Synthesis, HLS), comme par exemple l'outil Vitis, comble partiellement cet écart en traduisant une description comportementale écrite dans un langage spécifique, proche des paradigmes de programmation classiques, vers une sémantique HDL.

Le langage de programmation Julia [1] comble un besoin similaire du côté logiciel en simplifiant la conception : Julia vise à produire directement des programmes très performants à partir d'un langage de haut niveau flexible. Pour atteindre cet objectif, Julia repose sur plusieurs principes : un système de types expressif, une typage progressif (on parle d'inférence), et la métaprogrammation. En particulier, Julia utilise une représentation intermédiaire (IR) qui est générée et transformée durant la compilation, avant d'être envoyée à la chaîne d'outils LLVM pour produire du code machine (CPU). Ces principes ont déjà été utilisés pour développer des back-ends vers d'autres cibles matérielles comme les GPU ou les TPU, mais les matériels spécialisés comme les FPGA n'ont pas encore été explorés.

Cet article présente un nouveau *back-end* Julia exploitant ces principes pour l'accélération matérielle, en utilisant une nouvelle forme de représentation intermédiaire issue de l'environnement de compilation Multi-Level Intermediate Representation (MLIR) [2]. MLIR est un framework qui généralise le concept d'IR en permettant la création de plusieurs sémantiques IR avec différents niveaux d'abstraction. Ce framework est notamment utilisé pour décrire des chaînes de compilation complexes, comme ScaleHLS [3], qui exploite l'expressivité et la modularité de MLIR pour créer un cadre HLS capable d'annoter automatiquement du matériel. Cet article propose donc une chaîne HLS complète allant du langage Julia jusqu'à une description HDL, en s'appuyant sur MLIR et ScaleHLS, permettant ainsi un prototypage rapide et des formes avancées d'exploration de l'espace de conception (Design Space Exploration, DSE).

La section 2 présente la chaîne de compilation de Julia vers MLIR et ScaleHLS. Ensuite, la section 3 présente un cas d'usage de ce nouveau compilateur. Enfin, la section 4 conclut cette étude.

II. Judias : introduction de MLIR et ScaleHLS dans le flot de compilation Julia

La chaîne de compilation Julia repose sur l'utilisation de LLVM [4] : l'intérêt de cette machine virtuelle est avant tout de permettre la génération de code machine. Ainsi, une partie de la sémantique du programme source est perdue, ce qui est préjudiciable car les annotations de Vitis ne peuvent pas être générées à partir du LLVM IR obtenu de Julia [5]. MLIR se distingue par sa capacité à représenter des programmes avec différents niveaux d'abstraction, ce qui le rend particulièrement intéressant pour modifier et optimiser le code à différentes échelles. Une des notions clés de MLIR est l'utilisation de "dialectes", qui permettent de définir des opérations spécifiques à un domaine ou à une architecture matérielle. Cet écosystème a démontré son utilité dans les

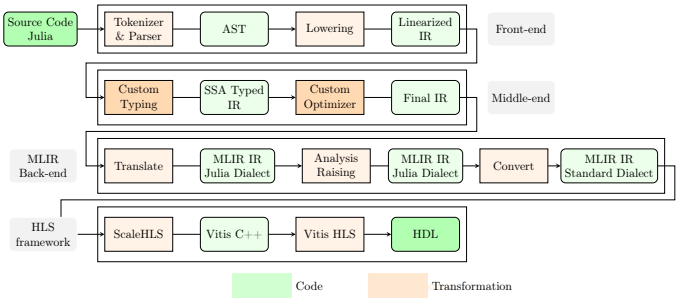


Fig. 1. Judias : Compilateur Julia modifié générant du MLIR IR.

compilateurs d'apprentissage machine ou dans le cadre de la HLS.

Par exemple, ScaleHLS [3] et son successeur HIDA sont des frameworks HLS qui utilisent MLIR pour définir le dialecte du matériel et permettre la génération de programmes HLS performants. Pour atteindre cet objectif, un pipeline de compilateur composé de plusieurs niveaux d'abstraction a été introduit, en commençant par les dialectes standard MLIR et en terminant par un dialecte Vitis personnalisé qui est traduit en Vitis HLS C++ synthétisable.

L'idée centrale de l'approche proposée dans cette étude, illustrée dans la Figure 1, est d'introduire la génération de MLIR juste avant l'étape de traduction vers LLVM, afin de maximiser les avantages offerts par le compilateur Julia. Cette stratégie permet de préserver le système *multiple dispatch* [4] et le système de types de Julia tout en exploitant des optimisations avancées. Seul un sous-ensemble du langage Julia est pris en charge : les programmes doivent être entièrement typés, sans appel de fonctions externes ni utilisation du runtime Julia, comme la réflexion. Ainsi, l'IR de Julia est parcourue et transformée en code MLIR, avec certaines constructions spécifiques, telles que les *phinodes*, converties en blocs fonctionnels MLIR.

Afin de faciliter l'intégration avec les dialectes MLIR standards, un dialecte MLIR personnalisé pour Julia est créé. L'objectif de ce dialecte est de simplifier la traduction, notamment pour la conversion de types et la passe d'élévation (*rising*). Ensuite, l'étape MLIR se déroule en deux phases :

- Une passe de *rising* : l'IR de Julia, étant non structuré, les boucles for et les structures if sont perdus durant la compilation. Ces structures sont extraites en tirant parti de la spécificité des itérateurs de Julia et de l'analyse de dominance.
- Plusieurs passes de *lowering* : les opérations et types Julia restants du dialecte personnalisé sont convertis en leurs équivalents MLIR, notamment dans les dialectes standards tels que `func`, `arith`, `index`, `memref` et Structured Control Flow (SCF). Cette étape est détaillée plus en profondeur dans [6].

III. Analyse comparative sur GEMM

Pour démontrer les capacités de ce compilateur à générer du code MLIR utilisable pour ScaleHLS, nous considérons l'exemple GEMM, une multiplication de matrices 3x3 en Float32. L'objectif est de démontrer qu'à partir de la même base de code flexible, nous pouvons générer des architectures matérielles. Nous proposons ici de comparer les résultats du MLIR généré avec le code C fourni par Polygeist [7]. Les résultats en ressources utilisées et nombres de cycles sont donnés dans la Table I.

TABLE I
Comparaison des métriques entre Polygeist et notre approche.

Métriques	Polygeist	De Julia
Cycles	31	35
DSP	34	34
FF	4195	4037
LUT	3114	3032

Les résultats montrent que les métriques sont très similaires. La solution générée par Polygeist présente un nombre légèrement inférieur de cycles d'horloge mais nécessite davantage de ressources matérielles. Le point clé ici est que la signature Julia est paramétrique, tandis que le code C fourni par Polygeist est typé statiquement. Cette flexibilité de Julia facilite une exploration plus polyvalente de l'espace de conception.

IV. Conclusions

Dans cet article, nous présentons un *back-end* Julia MLIR adapté à la synthèse de haut niveau, réalisé grâce à une pipeline de compilateur modulaire qui effectue une conversion statique d'IR vers IR sur des programmes Julia entièrement typés. Cette nouvelle approche est comparée à Polygeist à l'aide d'un benchmark de référence et permet une flexibilité dans l'exploration de l'espace de types.

References

[1] T. Besard, C. Foket, and B. De Sutter, "Effective Extensible Programming: Unleashing Julia on GPUs," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 4, pp. 827–841, Apr. 2019.

[2] C. Lattner, M. Amini, U. Bondhugula, A. Cohen, A. Davis, J. Pienaar, R. Riddle, T. Shpeisman, N. Vasilache, and O. Zinenko, "MLIR: A Compiler Infrastructure for the End of Moore's Law," Feb. 2020.

[3] H. Ye, C. Hao, J. Cheng, H. Jeong, J. Huang, S. Neuendorffer, and D. Chen, "ScaleHLS: A New Scalable High-Level Synthesis Framework on Multi-Level Intermediate Representation," in *2022 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, Seoul, Korea, Republic of, Apr. 2022, pp. 741–755.

[4] J. Bezanson, J. Chen, B. Chung, S. Karpinski, V. Shah, J. Vitek, and L. Zoubritzky, "Julia: Dynamism and performance reconciled by design," vol. 2, pp. 1–23.

[5] G. Lounes, R. Gerzaguet, and M. Gautier, "Julia meets the FPGA : Higher-level synthesis methodology for heterogeneous hardware and software architectures," Conference on the Julia programming language (JuliaCon), p. 1, Jul. 2024.

[6] ——, "Flexible front-end for high level synthesis leveraging heterogeneous compilation," in *Workshop on Rapid Simulation and Performance Evaluation for Design Optimization: Methods and Tools (RAPIDO)*, Barcelona (ES), Spain, Jan. 2025.

[7] W. S. Moses, L. Chelini, R. Zhao, and O. Zinenko, "Polygeist: Raising c to polyhedral mlir," in *Proc. 30th International Conference on Parallel Architectures and Compilation Techniques (PACT)*, 2021, pp. 45–59.

Automatic Characterization of Colorectal Cancer Markers

Garance LUCAS
Sorbonne University
CNRS, LIP6
Paris, FRANCE
garance.lucas@lip6.fr

Andrea PINNA
Sorbonne University
CNRS, LIP6
Paris, FRANCE
andrea.pinna@lip6.fr

Bertrand GRANADO
Sorbonne University
CNRS, LIP6
Paris, FRANCE
bertrand.granado@lip6.fr

***Abstract*—Colorectal cancer (CRC) represents a major public health issue worldwide, causing 930,000 deaths in 2020. Colonoscopy is the standard procedure for screening and treating colorectal diseases is colonoscopy. This procedure is invasive for the patient and requires extensive bowel preparation and general anesthesia. Systematic screening allow to detect at early stage a potential CRC and lowers mortality significantly. Manual colonoscopy examination is time-consuming and prone to human error. Automated methods have the potential to assist doctors in detecting and classifying polyps; however, existing approaches often fail to generalize effectively across these tasks, and lack interpretability, which hinders their integration and acceptance within clinical practice. In this article, we present our approach for developing an explainable and automated method for polyp classification. This work is part of a project aimed at designing an endoscopic capsule equipped with an embedded artificial intelligence (AI) model for the detection and diagnosis of polyps.**

I. Introduction

The colon is the distal part of the digestive tract. It is located in the abdomen, downstream from the small intestine, and is approximately 1.5 meters long. Colon cancer is common and easily curable at an early stage, which justifies systematic screening. In France, in 2018, CRC ranks second in cancer-related deaths (17,117 deaths), behind lung cancer (33,117 deaths) and ahead of breast cancer (12,146 deaths) [9]. In some European countries, CRC holds the top position for cancer-related deaths. The National Institutes of Health estimated 1.9 million new CRC cases and 930,000 deaths in 2020. Thus, it represents a public health issue both nationally and globally. The reference diagnostic examination for the colon is videocolonoscopy [2]. It allows (1) detection of polyps, (2) characterization of their risk of degeneration (based on their relief and the appearance of the colonic crypt openings, known as "pit pattern" [3]), and (3) resection of polyps when necessary (followed by an anatomopathological examination to support the diagnosis). This procedure relies on endoscopic classifications, the most well-known being the Kudo and Paris classifications. The former enables texture analysis, while the latter focuses on shape analysis. These classifications, along with others, have been integrated into a meta-classification called CONECCT (Colorectal Endoscopic Classification to Choose the Treatment) [8], that aims to aggregate all the medical knowledge used by the doctors for polyp classification and medical procedure decisions. This meta-classification analyzes and aggregates markers across all their known properties, enabling finer and more precise characterization [1].

II. State of the art

The application of AI to medical tasks has seen significant advancements in recent years. Within the domain of polyp classification, current state-of-the-art approaches predominantly rely on Convolutional Neural Networks (CNNs). In 2017, Zhang and al. [4] proposed a hybrid classification framework combining a CNN with a Support Vector Machine (SVM) to categorize polyp images into three distinct classes: non-polyp, hyperplastic, and adenomatous. The CNN component was first employed to determine the presence of polyps within an image. For images in which a polyp was detected, the SVM component subsequently performed a finer classification to differentiate between hyperplastic and adenomatous polyps. The model achieved 98.0% and 85.9% of accuracy respectively for the first and second-stage classifier, across two datasets, comprising both publicly available [10] and private data sources. In 2022, Chung-Ming and al. [5] introduced a classification approach that combined traditional image feature extraction techniques, such as the Gabor filter, with a CNN-based model for polyp classification. Their method was guided by the NBI International Colorectal Endoscopic (NICE) classification, excluding the third category (invasive cancer). The proposed system achieved an accuracy of 96.4% on a private dataset.

With the growing success of Transformer architectures in natural language processing, their application to computer vision tasks, particularly in the medical imaging domain, has garnered increasing attention. In this context, Krenzer and al. [6] presented a study in 2023 employing a Transformer-based framework for polyp classification following the Paris classification system. Their model considered four out of the seven categories defined in the classification and utilized a Vision Transformer (ViT) encoder coupled with a multilayer perceptron (MLP) classification head. The model was evaluated on a private dataset and a publicly available dataset [11] with private annotations, achieving classification accuracies of 87.42% and 89.35%, respectively. Similarly, Hossain and al. [7] adopted a ViT-based architecture combined with an MLP

head to classify polyps into two categories - adenomatous and hyperplastic - corresponding to two of the three classes defined in the NICE classification. Their experiments were conducted across four datasets [12]–[14] including one private dataset. The proposed method achieved an accuracy of 99%.

III. The challenges

The major challenges regarding polyp classification task can be summarized as follow: regarding the data we can note 1) the high cost of acquiring and labeling large amount of data does not enable to have access to consequent dataset for the learning process; 2) some types of polyp are extremely rare - such as invasive cancer polyps - which undue imbalance in the datasets; 3) there is a high inter-class similarity and intra-class variation - especially among small size and flat polyps - which makes harder to detect and distinguish them, even for human experts; 4) there can be a high background object similarity where the polyp does not always clearly standout from the background mucosa, or residues which can completely or partially hide polyps; 5) due to the image acquisition procedure, the image can often appear to be blurred or to contain light artefacts.
Regarding the methods, we can mention 6) the fact that the state-of-the-art methods are developed and trained over different datasets, that are often private, which result in complicated comparison of their performances; 7) the medical classification used in the labels of the datasets differ; 8) the performance metrics used vary across studies, with accuracy being the only consistently reported measure; however, accuracy alone may be insufficient, particularly in contexts such as class imbalance; 9) state-of-the-art deep learning models function as black boxes, raising ethical concerns and skepticism among clinicians due to their lack of interpretability. Moreover, they demand significant computational resources and often generalize poorly when trained on limited data.

IV. Our proposition

In this context, the objective of our work is to develop a method that achieves performances comparable to state-of-the-art approaches, while enhancing interpretability by aligning model predictions with clinically relevant classification criteria used by medical professionals. To this end, we adopt the CONECCT classification system, which differentiates five polyp types based on criteria that inform medical intervention strategies - specifically, whether and how a polyp should be resected. The CONECCT classification relies on four key visual features for polyp characterization: macroscopic appearance (size and morphology), color, vascular pattern, and pit pattern (surface texture). Our proposed approach involves designing a modular architecture in which each module is dedicated to extracting and analyzing one of these specific features. The combined output of these modules is then used to perform the final classification. This modular strategy not only aims to achieve high diagnostic accuracy, but also facilitates transparent and interpretable predictions that reflect the decision-making process of clinicians.

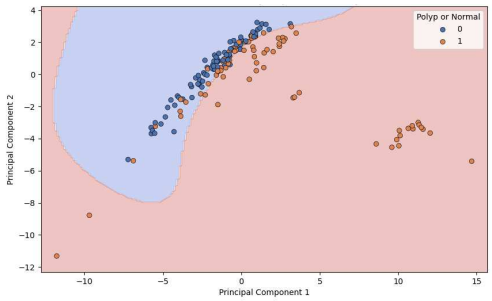


Fig. 1. SVM decision boundary over binary classification (normal vs polyp) projected in PCA space

This study is part of the Cyclope project, which aims to develop an AI-powered endoscopic capsule as a minimally invasive alternative to traditional colonoscopy. Funded by BpiFrance (i-Demo) and Région Normandie, the project seeks to embed an AI model within the capsule and its box to enable real-time detection and classification of polyps, improving diagnostic efficiency and patient comfort.

A. Texture analysis

As an initial step in our study, we focused on analyzing the texture characteristics of polyps in endoscopic images. Specifically, we aim to compare the performance of a conventional texture-based classification approach with that of a deep learning model, in order to assess whether the increased complexity and parameter count of deep models lead to significantly improved performance. To represent texture, we extracted a set of widely used statistical features derived from the gray-level co-occurrence matrix (GLCM) of the images. These features were then used to train a SVM classifier. The classification task was performed under two different settings: (1) binary classification to differentiate polyp from non-polyp images, and (2) multi-class classification to distinguish among three clinically relevant polyp types - adenoma, hyperplastic, and sessile serrated lesion.

B. Texture analysis

Preliminary results suggest that the extracted feature vectors effectively capture texture information relevant for polyp classification. These representations of images demonstrate the potential to support both binary and multi-class classification tasks. Figure 1 illustrates the SVM decision boundary obtained in one of our experiments, which aimed to evaluate the ability of the extracted feature vector to characterize texture by performing a binary classification between polyp-containing images and normal colon images. Future work will focus on enriching the feature vector with additional descriptors derived from the GLCM, with the aim of improving the expressiveness of the texture representation. In parallel, we plan to evaluate the performance of alternative classification algorithms beyond the SVM, in order to identify the most suitable model for this task. Once a robust and accurate model for texture-based classification is established, attention will shift toward the development of subsequent modules in the overall framework, each targeting a different characteristic defined in the CONECCT classification system.

REFERENCES

[1] C. Brule and al., "The COlorectal NEoplasia Endoscopic Classification to Choose the Treatment classification for identification of large laterally spreading lesions lacking submucosal carcinomas : A prospective study of 663 lesions" , United European Gastroenterology Journal, vol. 10, no 1, p. 80-92, janv. 2022, doi : 10.1002/ueg2.12194.

[2] I. N. du Cancer. La situation du cancer en france en 2010. Technical report, www.e-cancer.fr, 2010.

[3] S. F. d'Endoscopie Digestive. Classification des cryptes "pit pattern" dans le colon de k. eto. Technical report, SFED, 2009.

[4] R. Zhang and al., "Automatic Detection and Classification of Colorectal Polyps by Transferring Low-Level CNN Features From Nonmedical Domain" , IEEE Journal Of Biomedical And Health Informatics, vol. 21, no 1, p. 41-47, déc. 2016, doi : 10.1109/jbhi.2016.2635662.

[5] C.-M. Lo, Y.-H. Yeh, J.-H. Tang, C.-C. Chang, et H.-J. Yeh, "Rapid Polyp Classification in Colonoscopy Using Textural and Convolutional Features" , Healthcare, vol. 10, no 8, p. 1494, août 2022, doi : 10.3390/healthcare10081494.

[6] A. Krenzer and al., "Automated classification of polyps using deep learning architectures and few-shot learning" , BMC Medical Imaging, vol. 23, no 1, avr. 2023, doi : 10.1186/s12880-023-01007-4.

[7] M. S. Hossain and al., "DeepPoly : Deep Learning-Based Polyps Segmentation and Classification for Autonomous Colonoscopy Examination" , IEEE Access, vol. 11, p. 95889-95902, janv. 2023, doi : 10.1109/access.2023.3310541.

[8] C. Brule and al., "The COlorectal NEoplasia Endoscopic Classification to Choose the Treatment classification for identification of large laterally spreading lesions lacking submucosal carcinomas : A prospective study of 663 lesions" , United European Gastroenterology Journal, vol. 10, no 1, p. 80-92, janv. 2022, doi : 10.1002/ueg2.12194.

[9] Santé Publique France

[10] $https://www.depeca.uah.es/colonoscopy_dataset/$

[11] M. Misawa and al., "Development of a computer-aided detection system for colonoscopy and a publicly accessible large colonoscopy video database (with video)" , Gastrointestinal Endoscopy, vol. 93, no 4, p. 960-967.e3, juill. 2020, doi : 10.1016/j.gie.2020.07.060.

[12] K. Pogorelov and al., "KVASIR" , Association For Computing Machinery, New York, NY, United States, p. 164-169, juin 2017, doi : 10.1145/3083187.3083212.

[13] H. Borgli and al., "HyperKvasir, a comprehensive multi-class image and video dataset for gastrointestinal endoscopy" , Scientific Data, vol. 7, no 1, août 2020, doi : 10.1038/s41597-020-00622-y.

[14] A. Del Bimbo and al., Pattern Recognition. ICPR International Workshops and Challenges. 2021. doi : 10.1007/978-3-030-68793-9.

Adaptive Computing Architecture for Embedded Continual Learning

Abdelghani Bourenane
LIP6, Sorbonne Universite
abdelghani.bourenane@lip6.fr

Sebastien Pillement
IETR, Nantes Universite
sebastien.pillement@univ-nantes.fr

Andrea Pinna
LIP6, Sorbonne Universite
andrea.pinna@lip6.fr

***Abstract*—Retraining conventional deep learning (DL) models requires cloud-scale computing and risks catastrophic forgetting, making it ill-suited for edge learning. Continual Learning (CL) introduces efficient learning algorithms that balance stability and plasticity. However, their deployment in latency-critical scenarios has not yet been widely explored. This work aims to leverage System-on-Programmable-Chip architecture to host real-time and power-efficient inference with the capability of CL.**

***Index Terms*—CL, SoPC, latency, edge**

I. Introduction

DL has been widely applied to complex pattern recognition tasks like robotic vision. In industrial environments, robotic systems leverage edge computing on the device to process data streams in real time, minimizing latency and reducing the bandwidth demands associated with transmitting data through the Radio Access Network to the Core Network, as illustrated in Fig. 1. Typically, vision models are trained on static data distributions; however, in real-world scenarios, new data segments may emerge over time.

Fig. 1: Selected use case, industrial robotic vision in wireless network

At time t, a new sample j is observed, denoted as $\{D_t, t\}_{t\in T}$, captured as a distribution $p(X_j)$ that differs from the original training and testing distribution $p(X_i)$, on which the model f_θ was initially trained. This distributional shift leads to degraded performance, reflected in a drop in accuracy:

$$A_i = \frac{1}{i}\sum_{j=1}^{i} a_{i,j}$$

Retraining f_θ on the new sample j using standard DL approaches often results in catastrophic forgetting, where performance on previously seen samples degrades. The forgetting metric on sample set I is defined as:

$$F_i = \frac{1}{i-1}\sum_{j=1}^{i-1} f_{i,j}$$

where the forgetting at timestep j for sample i is given by:

$$f_{j,i} = \max_{l\in\{1,\dots,j-1\}}(a_{l,i}) - a_{j,i}, \quad \forall i < j$$

The ability of a model f_θ to retain previously acquired knowledge while simultaneously learning new patterns is known as the stability-plasticity trade-off, and it is a central challenge addressed by CL algorithms [1]. Many CL approaches have been proposed to manage this trade-off effectively. Among these, experience replay (ER) [2] has shown strong potential in maintaining a balance between learning new tasks and preserving generalization on past data. This is achieved by training the model on a combination of previous experiences M_{t-1} and the current data stream (x_t, y_t). The previous model f_{t-1} is updated to f_t, and the set of old experiences is updated from M_{t-1} to M_t. This update process can be formalized as:

$$\mathrm{A}_t : \{f_{t-1}, (x_t, y_t), M_{t-1}\} \rightarrow \{f_t, M_t\}$$

Although consistent efforts have been made to develop accurate CL methods, many of these approaches are not well-suited for embedded applications due to their high memory and computational demands. Moreover, the literature shows limited focus on deployment constraints, particularly in resource-constrained environments. While some studies have explored compression techniques, such as weight binarization and the use of feature replay instead of raw data [3], others have highlighted the trade-offs associated with deploying CL systems across computing platforms like GPUs and FPGAs [4]. However, to our knowledge, there has been little investigation into architectural solutions supporting CL methods on reconfigurable HW. Therefore, our approach focuses on studying CL techniques, mostly based on ER in terms of their computational and storage requirements and their alignment with on-device learning. This enables the creation of a holistic edge learning framework, addressing the question of which CL algorithms and design solutions best combine plasticity A_i and

This project is co-funded by the European Union's Horizon Europe research and innovation program Co-fund SOUND.AI under the Marie Sklodowska-Curie Grant Agreement No 101081674, and AdaptING project funded by the PEPR IA program - France 2030

stability F_i with energy and time efficiency for deployment on constrained HW platforms.

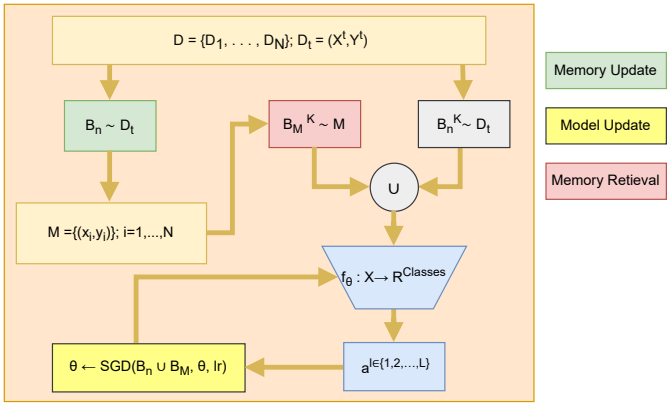


Fig. 2: Conceptual framework of ER for CL

II. Concept

A CL system based on ER consists of three main stages. As illustrated in Fig. 2, the first stage involves selecting a batch B_n from the current data stream D_t and combining it with a batch B_M retrieved from a memory buffer M. This combined batch is then fed into the model f_θ for a forward pass. The second stage performs a model update using Stochastic Gradient Descent (SGD) with a learning rate lr, updating the model parameters θ. The third stage involves a memory update process, where the buffer M is refreshed by incorporating newly observed samples. This may also involve discarding older samples to make room for future updates, ensuring the memory remains within its capacity constraints.

In this context, our work focuses on two objectives: first, mapping these functional blocks onto a heterogeneous computing architecture such as SoPC, taking into account the computational complexity of training and real-time processing requirements of inference , and second, developing an efficient HW architecture for the selected algorithms, enabling on-chip CL within resource-constrained environments.

III. Methodology

Customizable HW architectures, such as FPGA-based SoPC, have demonstrated enhanced performance for DL tasks due to their ability to parallel Multiply-Accumulate (MAC) operations, offering both decreased end-to-end latency and power consumption. These operations, central to weight-by-input computations, benefit significantly from FPGA's parallelism. However, their potential for CL systems has seen limited exploration. The reconfigurable nature of such HW architectures makes them well-suited for real-time adaptability. Specifically, CL system components can be dynamically mapped onto the SoPC platform based on task-specific embedding requirements. To this end, and shown in Fig. 3, the forward pass of the neural network is executed on the programmable logic (PL), utilizing quantized arithmetic to enable high-speed computation on a systolic array of MAC units. This systolic array is driven by weights stored in off-chip DRAM, which are fetched into local BRAM along with selected input data. The input consists of a mix of previous experiences and the current data stream, selected through a retrieval block interfaced with the memory system. The resulting output activations are passed through an activation function, stored in BRAM, and subsequently sent to the Processing System (PS). The PS, which includes a CPU, manages control over HW components and performs complex computations, such as SGD, using full-precision (FP32) data. These data are de-quantized from the quantized forward pass output using a quantization block. The computed gradients are used to update the model parameters in DRAM. They are also passed back to the retrieval block in the PL to guide the selection of previous experiences with the highest associated loss values. At the end of the training session, the knowledge memory M is updated by incorporating the most recent samples, preparing the system for future CL sessions.

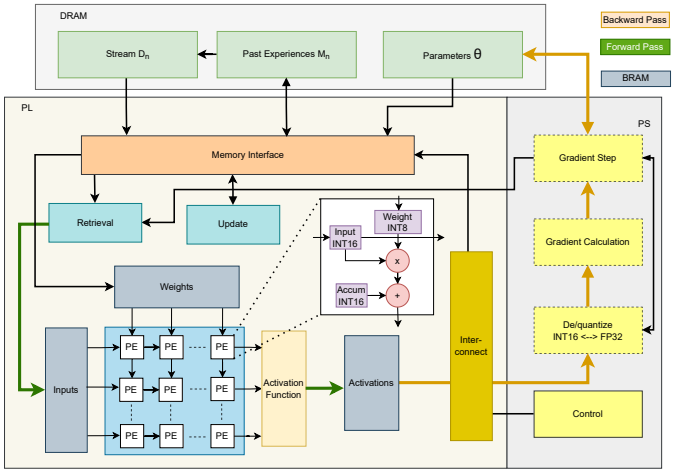


Fig. 3: Proposed CL system in SoPC

IV. Conclusion

The demand for edge learning systems continues to grow, with embedded CL agents offering a promising balance between accuracy and computational efficiency. This work demonstrates the potential of on-chip learning by integrating CL algorithms into a SoPC architecture. Future work will focus on deploying an ER model on the SoPC, with inference on the PL to achieve minimal latency and energy consumption.

References

[1] L. Wang, X. Zhang, H. Su, and J. Zhu, "A comprehensive survey of continual learning: theory, method and application," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[2] D. Rolnick, A. Ahuja, J. Schwarz, T. Lillicrap, and G. Wayne, "Experience replay for continual learning," *Advances in neural information processing systems*, vol. 32, 2019.

[3] L. Vorabbi, A. Carraggi, D. Maltoni, G. Borghi, and S. Santi, "Towards on-device continual learning with binary neural networks in industrial scenarios," *Image and Vision Computing*, p. 105524, 2025.

[4] S. Aggarwal, K. Binici, and T. Mitra, "Chameleon: Dual memory replay for online continual learning on edge devices," *IEEE Transactions on CAD of ICS*, vol. 43, no. 6, pp. 1663–1676, 2023.

Non-destructive characterization of Breakdown Voltage measurement and Application on a 55nm SiGe HBT featuring f_T/f_{MAX} of 400GHz/500GHz

L. Réveil[a,b,*,**], A. Sarafinof[b,**], F. Cacho[a,*], M. Pradeau [b,**], N. Derrier[a,*], M. De Matos[b,**], C. Mukherjee [b,**], C. Maneux[b,**]

[a]*STMicroelectronics France, 850 rue Jean Monnet, Crolles, 38926, France*
[b]*University of Bordeaux, 351 Cours de la Libération, Talence, 33400, France*

***Abstract*— This paper presents a non-destructive measurement setup allowing a complete Safe Operating Area (SOA) characterization. An automatic extraction methodology of the Breakdown Voltage is proposed. A preliminary ageing study of the SiGe HBT 55nm Breakdown Voltages (BVs): BV_{CEO} and BV_{CBO}, is presented.**

Keywords—HBT SiGe, 55nm, ageing, BVs, Impact-Ionisation, SOA

I. Introduction

In the context of increasing demand of RF-circuits and of their performances with the new 6G and satellite constellation network [1][2], STMicroelectronics is upgrading the 55nm SiGe BiCMOS technology called B55. The enhanced technology named B55X features a SiGe Heterojunction Bipolar Transistor (HBT) reaching f_T/f_{MAX} respectively of 400GHz/500GHz [3]. The increasing RF-performance goes with a reduction of the BV_{CEO} and BV_{CBO} below 1.5V and 5V for High-Speed (HS) devices [2]. Therefore, the Safe Operating Area (SOA) of the devices is reduced leading the RF and mmW circuits to operate closer to the SOA edges. To drive the transistor in these operating conditions over the circuit lifetime, circuit designers need an accurate ageing model based on an accurate SOA definition and a comprehensive ageing study.

This work presents a non-destructive BV_{CEO} and BV_{CBO} extraction methodology from measurement and HiCuM model, as a prerequisite for ageing study and a preliminary study of BVs evolution after ageing tests is presented.

II. Measurement Setup and Extraction Methodology for SOA and BVs Study

The SOA is defined as the bias conditions for which no critical failure occurs [4]. As shown in Fig 1., the SOA is limited by the BV of the Base-Collector junction with emitter grounded (BV_{CBO}), materialized by the current abrupt increase at $BV_{CBO} + V_{BE}$, according to two limiting charge transport mechanisms: the snapback and the pinch-in.

A. BV_{CEO} and BV_{CBO} description

The weak and strong avalanches consist in an increase of the I_B and I_C currents due to Impact-Ionization (II) mechanism [5] and leading to critical failure. The II mechanism is triggered with a specific voltage known as the BV_{CEO}. The BV_{CEO} occurs at low V_{BE} in the weak avalanche regime and corresponds to the voltage where I_B becomes negative due to the inversion of the majority carrier type in the Base. Beyond this voltage, the device suffers greater degradation [6].

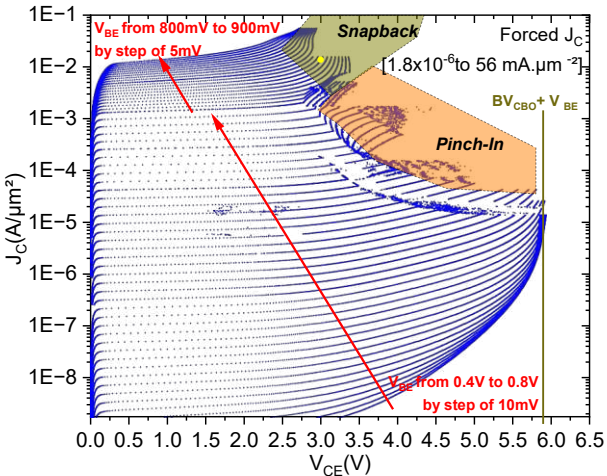


Fig 1. J_C versus V_{CE} for different V_{BE} showing the SOA of a B55X NPN HS device featuring $A_E = 0.2 \times 5\mu m^2$.

$$M = \frac{1}{1-\left(\frac{V_{CBi}}{BVCBO}\right)^{n_{AVL}}} \quad (1)$$

$$I_C = M \times I_{C0} \quad (2)$$

Strong avalanche effect occurs at high V_{BC} and the BV_{CBO} corresponds to the voltage for which I_C tends to infnity and leads to catastrophic failure. The semi-empirical equations (1) and (2) describe I_C current values in avalanche conditions [4] with M the multiplication factor calculated with V_{CBi} the intrinsic Collector-Base voltage and n_{AVL} a fitting parameter close to 5.

B. Measurement setup

To measure the BVs and the SOA, the transistor is driven close to electro-thermal runaway and in weak and strong avalanche regimes. Consequently, the proximity of the electro-thermal destruction requires dedicated forced- J_C setup with the V_{BE} is swept [4]. V_{CE} and J_B are measured to obtain an J_C versus V_{CE} graph shown Fig 1. By forcing the current, the controlled current. To avoid a catastrophic failure of the device, the J_C is controlled while avalanche regime is controlled so that the unstable effect such as snapback can be measured (Fig 1). The measurements are performed using GSG RF-pad structures to ensure the measurement stability and to avoid measurement oscillations as well. Fig 2. presents the results of the forced V_{CE} (destructive) and the forced J_C setups BVs extraction methodology simulated through HiCuM L2 with model cards extracted through the typical extraction flow of HiCuM [6] using forced voltage setups.

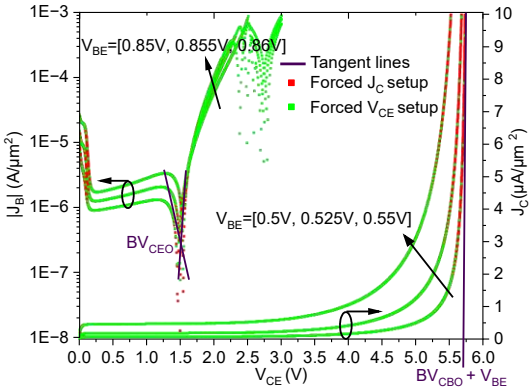


Fig 2. J_C and for Forced J_C and V_{CE} setups from HiCuM L2 showing BVs extraction methodology through tangent line tracing on B55X $0.2 \times 5\mu m^2$ device.

The results show no difference between the two setups, validating J_C forced setup for BVs extraction.

C. *Extraction methodology*

To extract the BVs of the B55X $0.2 \times 5\mu m^2$ HBT under test, the measurement setup presented in *Section II.B* is used. The BV_{CEO} values are observed when the absolute value of I_B drops to 0. To extract this value, J_C is controlled from $1.8mA.\mu m^{-2}$ to $8.8mA.\mu m^{-2}$ and a V_{BE} is set according to the HBT dimension to drive the transistor in the same operating condition i.e. same Base current density. The three current densities are $J_{B1} = 655nA.\mu m^{-2}$, $J_{B2} = 872nA.\mu m^{-2}$, $J_{B3} = 1\mu A.\mu m^{-2}$. BV_{CBO} measurements are conducted under the same conditions as the BV_{CEO} ones. However, the measurements are done below the pinch-in regime at low V_{BE}. So, the J_C is controlled from $1.8nA.\mu m^{-2}$ to $17.5\mu A.\mu m^{-2}$ with V_{BE} set at 0.5V, 0.525V and 0.55V.

BV_{CBO}=V_{CE_t}-V_{BE} with V_{CE_t}extracted when slope crosses the x-axis (6).

The BV values are automatically extracted from the measurement data using a Python script. Fig 2. shows how the BVs are extracted using the tangent method. For BV_{CEO}, two tangent lines are traced along the $|J_B|$ drops, and the BV is extracted where the tangents lines intersect. As for the BV_{CBO}, one tangent line is traced where J_C tends to infinite, the BV is taken where the tangent line crosses the x-axis and calculated through equations (6). Therefore, the points to trace the tangent line are chosen at the middle of the drop or the middle of the infinite increase to fit perfectly the curve slope.

III. Application to Breakdown Voltage Ageing

A DC Forward stress at 25°C applying a V_{BE}=0.85V, V_{CE}=2.75V (Yellow point in Fig 1.), choose beyond BV_{CEO} to assess the DUT reliability in a reasonable time range (30min), is performed on a HS B55X device featuring A_E=$0.2 \times 5\mu m$. Fig 3. presents the result on a Gummel plot before and after stress. Due to Hot-Carrier-Degradation (HCD), I_B shows a positive drift, higher for V_{BE} below 0.8V [7,8]. The BVs extraction methodology is applied before and after the stress and results are shown on Fig 4. One observes a positive drift of BV_{CEO} around 0.01V, and a slight negative drift of the BV_{CBO} around 0.1V. The BV_{CEO} drifts are mainly due to the increase of I_B inducing of higher level of II to reach BV_{CEO}. Regarding the BV_{CBO} drift, it might result from a degradation of the Emitter series resistances or an evolution of the electric field due to traps creation. The Gummel plot is a restrictive figure which does not show all the degradations of the DUT.

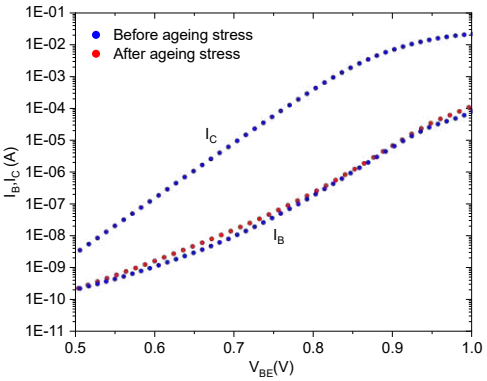


Fig 3. Gummel plot before and after a DC Forward stress of a B55X NPN HS device featuring $0.2 \times 5\mu m$.

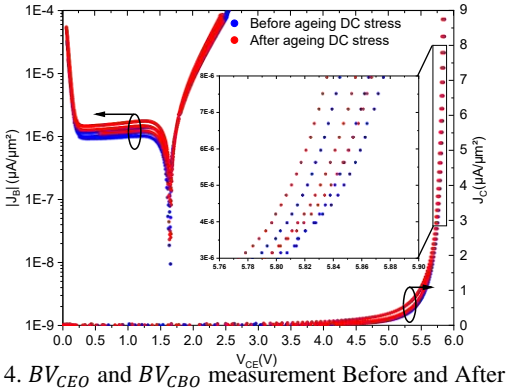


Fig 4. BV_{CEO} and BV_{CBO} measurement Before and After DC Forward stress performed on a B55X NPN HS device featuring $A_E = 0.2 \times 5\mu m^2$.

IV. Conclusion

In conclusion, the measurement setup and extraction methodology to study the SOA and the BVs are validated against the HiCuM simulation and the forced V_{CE} method for the BVs extraction. A study of the BVs after HCD showing slight degradations have been performed. The non-destructive methodology BVs extraction shows precise results, opening BVs ageing study through other stress levels/types or form factors.

References

[1] E. Calvanese Strinati et al. The Hardware Foundation of 6G: The NEW-6G Approach. 2022 EuCNC/6G Summit, Grenoble, France, 2022.

[2] M. Schröter et al. SiGe HBT Technology: Future Trends and TCAD-Based Roadmap. Proceedings of the IEEE, vol. 105, no. 6, pp. 1068-1086, 2017.

[3] P. Chevalier et al. A Versatile 55-nm SiGe BiCMOS Technology for Wired, Wireless, and Satcom Applications. BCICTS, Fort Lauderdale, USA, 2024.

[4] M. Jaoul. Study of HBT operation beyond breakdown voltage : Definition of a Safe Operating Area in this operation regime including the aging laws. Thesis, University of Bordeaux, Talence, France, 2020.

[5] M. S. Peter et al. Impact ionization and neutral base recombination in SiGe HBTs. Proceedings of the Bipolar/BiCMOS Circuits and Technology Meeting, 1999.

[6] D. Berger et al. HICUM parameter extraction methodology for a single transistor geometry. Proceedings of the Bipolar/BiCMOS Circuits and Technology Meeting, 2002.

[7] N. Zagni et al. Characterization and TCAD Modeling of Mixed-Mode Stress Induced by Impact Ionization in Scaled SiGe HBTs. IEEE TED, 2020.

[8] Uppili Raghunathan et al. Overview of Aging Mechanisms in SiGe HBTs. ECS Trans, 2022.M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.

Characterization of the Bacteria E.coli Impedance

Yuxuan XI
LIP6, CNRS, UMR7606
Sorbonne University
Paris France
yuxuan.xi@lip6.fr

Nathalie ROLHION
CRSA, INSERM, UMR938
Sorbonne University
Paris France
nathalie.rolhion@inserm.fr

Sylvain FERUGLIO
LIP6, CNRS UMR7606
Sorbonne University
Paris France
sylvain.feruglio@lip6.fr

Andrea PINNA
LIP6, CNRS UMR7606
Sorbonne University
Paris France
andrea.pinna@lip6.fr

***Abstract*—This study explores the feasibility of using Electrical Impedance Tomography (EIT) for gastrointestinal microbial detection. Escherichia coli (E. coli) is used as a model to collect conductivity data for different concentrations of bacterial liquids under different current and frequency conditions, and EIDORS toolbox is utilized for the image reconstruction.**

I. Introduction

Electrical impedance tomography (EIT) is a non-invasive technique that is particularly interesting for the characterization of objects under test, whether for civil engineering, chemistry, or the biological environment, for example. Notably, it allows for visualization of conductivity or impedance changes in the local area in different configurations.

In the biological domain, bacteria have different conductive characteristics. This preliminary study is to test whether EIT can allow reconstruction of the conductivity of a chosen bacterial species during in-vitro experiments. An experimental protocol is therefore proposed, and the first tests are presented.

This work corresponds to the first six months of a PhD project that began in November 2024.

II. Data Acquisition and Image Reconstruction

EIT is an imaging technique that reconstructs images of the target area based on the distribution of electrical conductivity. EIT systems usually place a network of electrodes around the target area, inject a small current between the electrodes, and measure the voltage response between each pair of electrodes to infer the conductivity distribution inside the target area, thereby achieving image reconstruction. The EIT system consists of two main components: an electrode array and a data acquisition system, which includes control hardware and dedicated software for data collection. Data acquisition requires controlling the current injection mode and sequence, as well as synchronously collecting voltage signals. The configuration and data storage are managed through the software interface.

In the experiment, Sciospec's EIT lab instrument is selected, an EIT device associated with an acquisition system with 32 electrodes and specific software (Figure 1). An array of electrodes consisting of 8, 16, or 32 electrodes is used, which are evenly distributed on the inner wall of the cylindrical container (tank) to ensure uniformity of signal acquisition. A PCB board with two identical tanks is also selected (Figure 2). The array of 32 electrodes can be divided by 2, typically in function of the application. In our case, two independent tanks of 16 electrodes are used to perform parallel and independent measurements, allowing comparison between different samples or experimental conditions.

This work is funding by ANR LabCom project ICI-lab in collaboration with BodyCAP.

Fig. 1. Sciospec's EIT lab instrument.

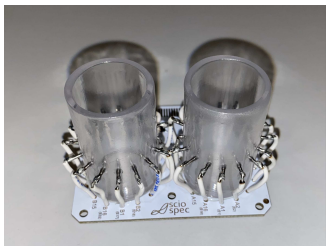


Fig. 2. Two tanks on the PCB board, each with 16 electrodes.

Depending on the imaging reconstruction purpose, EIT can be applied in two main ways:

- Absolute EIT (abEIT) [1]: Based on one complete measurement data, the conductivity distribution is reconstructed under a certain mode setting, without relying on the reference state.
- Differential EIT (dEIT) : Two dEIT are possible [2], [3]. In time-difference EIT (tdEIT), voltage data can be collected at different times, the difference is calculated by the two measurement results of the reference data and the target data. The reconstruction result is the distribution image of the conductivity change in the imaging area. In frequency-difference EIT (fdEIT), voltage data can be collected at different frequencies, and the difference is calculated by comparing the measurement results of the reference data and the target data. The reconstruction result is the distribution image of the conductivity change by the frequency in the imaging area.

EIT image reconstruction can be performed using EIDORS [4], based on MATLAB. EIDORS is an open-source software toolbox and features a variety of forward modeling tools and image reconstruction algorithms, as well as visualization capabilities, to support the entire process from data processing to image reconstruction.

III. Experiments

Escherichia coli (E. coli) is first selected as the preliminary experimental object. This bacteria is inoculated in a Luria-Bertani (LB) liquid medium, a transparent yellow liquid that is widely used for bacterial culture broth. The main ingredients of LB medium are peptone, yeast extract, and sodium chloride, which together provide enough nutrients and a favorable environment to sustain E. coli's stable physiological state and metabolic activity.

To ensure the optimal growth of E. coli and maintain its stable physiological state, it is inoculated in LB liquid medium under aerobic conditions. The culture is cultured at 37°C with a constant shaking speed of 180 rpm to promote oxygen exchange (Figure 3). After 6–8 hours of culture, E. coli enters the logarithmic growth phase, which is suitable for subsequent EIT measurements. Then, the pure bacterial culture is prepared into four samples of different concentrations: 1/1000, 1/100, 1/10, and pure culture. Under the same mode setting, EIT data can be collected for these samples.

These different samples are measured by various current amplitudes and frequencies to compare and analyze the conductivity changes under various parameter settings. Specifically, five current levels are applied: 0.1 mA, 0.5 mA, 1 mA, 5 mA, and 10 mA. For each current setting, frequency is swept from 100 Hz to 1 MHz in 20 linearly spaced steps. LB liquid is added to both tanks as a reference standard. Subsequently, the same concentration of bacterial solution is added to the two tanks, and EIT measurements are performed.

Fig. 3. Laboratory microbiological shaker used for culturing E. coli, with a constant temperature set at 37°C.

IV. Preparation

Before performing bacterial EIT measurements, the system needs to be calibrated and its accuracy needs to be verified to ensure the reliability of the measurement data and the applicability of the image reconstruction algorithm. abEIT and tdEIT are used here.

First, when the equipment system is in a steady state, abEIT measurements are performed on the empty tank. Since the conductivity of air is close to $0mS/cm$. Therefore, the results of the non-zero conductivity in the reconstruction are associated to noise. This is due to the system hardware noise associated with the fact that the abEIT model may introduce numerical instability when dealing with extremely low conductivity conditions.

Then, the tank is filled with pure water and reconstructed using the abEIT algorithm. By comparing the difference between the pure water reconstruction results and the air reconstruction results, the conductivity of pure water reconstructed by abEIT can be approximated. Comparing this value with the standard reference conductivity can be used to verify the imaging accuracy of the abEIT model under current hardware conditions.

Next, the reconstruction results of the air are used as reference data, and the reconstruction results of the pure water are used as target data. The tdEIT algorithm can be used for reconstruction to verify the accuracy of the tdEIT algorithm (Figure 4).

After completing the calibration of air and pure water, we further use the EIT system to perform conductivity imaging and analysis on E. coli solutions in different configurations.

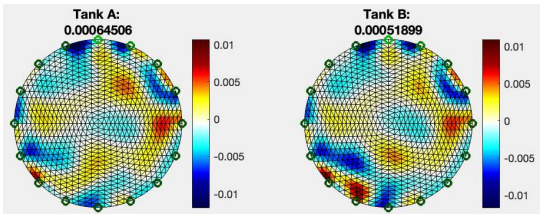


Fig. 4. Example of an EIT reconstruction image obtained using tdEIT between air and pure water. Tank A and Tank B are two separate compartments on the same PCB. The displayed values represent the mean conductivity of all pixels within the region of interest. In the color scale, red indicates an increase in conductivity relative to the reference state (air), while blue indicates a decrease or negligible change.

V. Conclusion

This study aims to identify the conductive properties of bacterial solutions with different concentrations using EIT. To evaluate the feasibility of this method for bacterial detection, E. coli is used as an example to study the changes in conductivity with bacterial concentration. If the experimental results confirm that EIT can reliably identify E. coli, the method can be extended to other bacteria related to gastrointestinal pathology. In addition, EIT can dynamically record the activity and life cycle of bacteria, thus representing an innovative and non-invasive method for in-situ analysis of the gastrointestinal microbiota.

References

[1] D. Isaacson, J. Mueller, J. Newell, and S. Siltanen, "Reconstructions of chest phantoms by the d-bar method for electrical impedance tomography," *IEEE transactions on medical imaging*, vol. 23, pp. 821–8, 08 2004.

[2] A. Adler and R. Guardo, "Electrical impedance tomography: regularized imaging and contrast detection," *IEEE Transactions on Medical Imaging*, vol. 15, no. 2, p. 170–179, Apr. 1996.

[3] S. Ahn, S. C. Jun, J. K. Seo, J. Lee, E. J. Woo, and D. Holder, "Frequency-difference electrical impedance tomography: Phantom imaging experiments," *Journal of Physics: Conference Series*, vol. 224, p. 012152, Apr. 2010.

[4] A. Adler and W. R. B. Lionheart, "Uses and abuses of eidors: an extensible software base for eit," *Physiological Measurement*, vol. 27, no. 5, p. S25–S42, Apr. 2006.

Exploring custom instruction support on CGRAs through fined-grained, eFPGA-based elements

Léo Pajot[1,2], Simon Rokicki[1], Bertrand Le Gal[1], Jérémie Crenne[2]
[1] University of Rennes, INRIA, IRISA, Rennes, France
[2] KEYSOM, Pessac, France

***Abstract*—The performance-per-watt efficiency ratio is a critical constraint for embedded systems. Among the architectural solutions designed for computation-intensive applications, CGRAs have emerged as a promising approach. CGRAs depend on a compilation infrastructure to efficiently manage and control the available PEs in executing data-intensive kernels. During compilation, instructions are mapped onto PEs based on the arithmetic and logic operations supported by their FUs. This one-time mapping simplifies the CGRA's control logic, reducing both its cost and energy consumption — similarly to VLIW architectures. However, like CPUs, FUs are limited in their capabilities, often requiring long instruction sequences to implement application-specific operations (e.g., popcount). An approach to address this limitation at design time is to extend the ISA of the PEs with domain-specific instructions. Yet, this method increases hardware complexity and energy consumption, while offering limited adaptability to emerging application domains. In this work, we propose an alternative solution: the integration of eFPGA-based extensions within selected CGRA PEs. These reconfigurable regions provide the flexibility to adapt hardware-supported instructions dynamically.**

***Index Terms*—Hardware acceleration, CGRA, eFPGA**

I. Introduction

Over the past decades, numerous hardware accelerator architectures have been proposed, each offering different trade-offs between raw computing performance, energy consumption, and implementation cost. While dedicated ASICs and FPGAs deliver the highest performance and energy efficiency, they inherently lack flexibility and programmability. Conversely, CPUs and GPUs offer high programmability but suffer from poor energy efficiency.

A more balanced and flexible alternative lies in the design and use of Coarse-Grained Reconfigurable Arrays (CGRAs) [1]. From a hardware perspective, a CGRA consists of an array of interconnected Processing Elements (PEs), often complemented by additional functional blocks such as memory units and bus interfaces. These architectures, which may resemble manycore processors, trade runtime flexibility for reduced hardware complexity and improved energy efficiency. Like GPUs, CGRAs are typically employed to accelerate specific portions of a program, offloaded from a host CPU.

Numerous comparative studies have demonstrated that CGRAs can achieve higher energy efficiency than GPUs and FPGAs for comparable performance levels, while requiring only a fraction of the hardware complexity [1], [2].

The Processing Elements (PEs) in CGRA architectures typically include only Functional Units (FUs), local registers, and interconnection links to neighboring PEs. Both the individual PEs and the CGRA as a whole generally lack any traditional form of control unit. Unlike Out-of-Order (OoO) processors, which rely on hardware schedulers, CGRAs leverage the compiler's global view of the application to identify and exploit parallelism. This approach offloads the complexity of scheduling to the compilation process, making it significantly more intricate. CGRAs are programmed by statically scheduling the execution of kernel operations at compile time. This process, known as mapping, encompasses both spatial scheduling (deciding which PE should execute each operation) and temporal scheduling (determining when each operation should occur). As a result, synchronization and data exchange among PEs are statically defined and fixed at runtime.

The role of the CGRA controller is thus primarily limited to issuing and coordinating the control signals for the PEs [3].

II. Custom instructions and fine-grain reconfigurability

Despite their efficiency, CGRAs may face performance limitations due to the restricted set of operations supported by their FUs. In cases where required operations are not natively supported, designers may choose to emulate them using sequences of atomic instructions which induces significant performance overhead. Alternatively, strategies similar to those employed in RISC-V CPUs can be considered, such as extending the instruction set architecture (ISA) with domain-specific features [4]. For instance, operations like popcount, which benefit from dedicated hardware support, may prompt designers to enhance or specialize the capabilities of certain FUs in a subset or all of the PEs. However, extending FUs with specialized instructions tailored to specific applications introduces additional hardware cost and reduces flexibility, particularly as future applications may require different sets of operations. Initiatives exist in order enable the use of custom instructions. EGRA [5] uses complex arithmetic units made of successive, interconnected, heterogenous ALUs to implement identified custom instructions. KAHRISMA [6] mixes fine-grained and coarse-grained processing elements so different ISAs can be executed on the same computing fabric. In our work we focus on providing greater adaptability at the application level, this work explores the potential of integrating embedded FPGAs (eFPGAs) into the FUs.

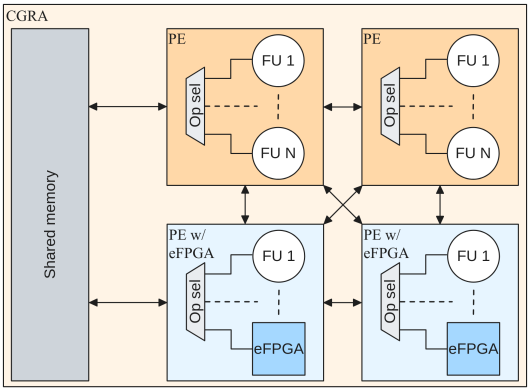


Fig. 1. Overview of a 2x2 CGRA-Flow-style architecture including FGEs

III. Current results and future work

In order to address these two issues, we investigate introducing Fine-Grained Elements (FGEs) inside the CGRA as shown in Fig. 1. Those FGEs are small eFPGA fabrics (100 to 1k LUTs), inserted inside some or all of the PEs, and interconnected with the array the same way any standard or custom FU would be. A custom compiler toolchain can consequently perform custom instruction detection beforehand, then implement the detected custom instruction in the available FGEs. The compiling process then needs to be amended, the Data Flow Graph (DFG) edited to account for the new custom instruction. Mapping can then be performed as usual: several existing CGRA mapper projects have already shown ability to target heterogeneous CGRAs. To this end, steps should be taken to ensure: (1) the potential beneficial effects in reducing execution time of providing the compiler with opportunities to fuse operations and to map those fused operations on dedicated hardware rather than complex ALUs like in [6]; and (2) the feasibility of introducing such fine-grain reconfigurable hardware inside an already reconfigurable device, in particular in terms of area and energy overheads.

We first studied many open-source CGRA frameworks, and chose OpenCGRA [7] (now called CGRA-Flow). The goal was to evaluate which one could best serve as a basis for our work, regarding their respective functionalities. A key feature was its ability to generate out-of-the-box complete CGRA architectures, so that we could collect synthesized area measurements. We repeated the same process of framework selection and evaluation for open-source FPGA fabric generators, and selected the FABulous framework [8].

That tool selection was also the occasion to check the orders of magnitude related to the costs of those architectures. Synopsys Design Compiler was used in combination with the open-source Skywater 130nm PDK to obtain hardware cost estimations. Table I summarizes various cost estimations and comparisons between an eFPGA Configurable Logic Block (CLB), two types of CGRA PEs (a default one and one without SHIFT, MUL and MAC FUs), a complete CGRA, and two projections of varying sizes of eFPGA. We consider those preliminary results promising, as the envisioned eFPGA size would fit reasonably well in a CGRA, if replicated in select PEs. That cost would also be dramatically lower if synthesized with a PDK containing memory and multiplexer cells.

TABLE I
Synthesized costs

Architecture	Area (kGe)	Relative area
FABulous default CLB	8.43	1.00
CGRA-Flow default PE	22.94	2.72
CGRA-Flow "light" PE	5.46	0.65
CGRA-Flow default CGRA	487	1.00
Projected 100 LUT eFPGA	105	0.22
Projected 1 kLUT eFPGA	1054	2.16

Next steps to dive into are amending the CGRA mapping process. We expect challenges regarding taking into account variable propagation delays inside the FGEs, and then in integrating the eFPGA exploration and synthesis steps into the mapping process in order to perform automatic constrained design-space exploration.

IV. Conclusion

Considering the research around spatial dataflow accelerators and custom instructions exploitation in computing systems, we demonstrated that hardware designers explore diverse optimums between their designs' power, performance, and area. The overhead incurred by the reconfigurability of a given system, while often not negligible, is yet another trade-off in favor of the adaptability and versatility it enables. We showed that such a trade-off remains to be explored regarding embedding eFPGA zones in CGRAs, and laid out future work to achieve a complete CGRA design and exploitation flow.

References

[1] L. Liu, J. Zhu, Z. Li, Y. Lu, Y. Deng, J. Han, S. Yin, and S. Wei, "A Survey of Coarse-Grained Reconfigurable Architecture and Design: Taxonomy, Challenges, and Applications," *ACM Computing Surveys*, vol. 52, no. 6, Nov. 2020.

[2] A. Podobas, K. Sano, and S. Matsuoka, "A Survey on Coarse-Grained Reconfigurable Architectures From a Performance Perspective," *IEEE Access*, vol. 8, 2020.

[3] Bingfeng Mei, S. Vernalde, D. Verkest, H. De Man, and R. Lauwereins, "Exploiting loop-level parallelism on coarse-grained reconfigurable architectures using modulo scheduling," in *Proc. of 2003 Design, Automation and Test in Europe Conference and Exhibition*. IEEE Comput. Soc, 2003.

[4] E. Cui, T. Li, and Q. Wei, "RISC-V Instruction Set Architecture Extensions: A Survey," *IEEE Access*, vol. 11, 2023.

[5] G. Ansaloni, P. Bonzini, and L. Pozzi, "EGRA: A Coarse Grained Reconfigurable Architectural Template," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 19, no. 6, 2011.

[6] R. Koenig, L. Bauer, T. Stripf, M. Shafique, W. Ahmed, J. Becker, and J. Henkel, "KAHRISMA: A Novel Hypermorphic Reconfigurable-Instruction-Set Multi-grained-Array Architecture," in *Proc. of Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, Mar. 2010.

[7] C. Tan, C. Xie, A. Li, K. J. Barker, and A. Tumeo, "OpenCGRA: An Open-Source Unified Framework for Modeling, Testing, and Evaluating CGRAs," in *Proc. of 2020 IEEE 38th International Conference on Computer Design (ICCD)*. Hartford, CT, USA: IEEE, Oct. 2020.

[8] D. Koch, N. Dao, B. Healy, J. Yu, and A. Attwood, "FABulous: An Embedded FPGA Framework," in *Proc. of the 2021 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*. ACM, Feb. 2021.

Deep Visual-Geolocalization in Maritime Coastal Environment

Alexandre Foucher, Cédric Seguin, Dominique Heller, Johann Laurent
Lab-STICC, CNRS UMR 6285, Université Bretagne-Sud
Lorient, France

***Résumé*—Dans les environnements marins, la géolocalisation est cruciale pour l'autonomie des véhicules de surface sans pilote (USV). Aujourd'hui, elle repose principalement sur un système de positionnement par satellites (GNSS). Toutefois, ce système peut être vulnérable à des attaques tel que le jamming ou le spoofing et doit être compensé par d'autres méthodes. La géolocalisation visuelle (VG) est une proposition intéressante et nécessaire lorsque le contexte ne permet pas l'utilisation de capteurs actifs tel que le lidar, le radar ou le sonar. Cette étude explore une méthode de localisation visuelle en temps réel, basée sur du deep-learning et conçue pour la navigation côtière à l'aide d'une caméra à champ de vision limité. Habituellement, les méthodes de localisation visuelle montrent une diminution de la précision lorsque le champ de vision est restreint. Pour y remédier, nous proposons une approche fondée sur la corrélation de la profondeur de l'horizon via l'extraction de caractéristiques visuelles par des réseaux de neurones convolutifs (CNN). Nos résultats mettent en évidence la faisabilité de la navigation sans GNSS pour les USV, ouvrant de nouvelles possibilités pour des opérations maritimes en autonomie.**

***Mots clés*—Visual-Geolocalization (VG), Unmanned Surface Vehicle (USV), K Nearest Neighbor (KNN), Deep Learning.**

I. Introduction

Les capacités croissantes de l'Intelligence Artificielle (IA) et des systèmes embarqués ont permis aux véhicules de surface sans pilote (USV) d'intégrer des chaînes d'automatisation complètes en temps réel. Cependant, malgré ces avancées, les USVs dépendent toujours de superviseurs distants pour surveiller les opérations et intervenir en cas de dysfonctionnement. Dans cette qupête d'autonomie, afin de réduire la dépendance humaine, il est nécessaire d'accroitre les capacités de résilience des systèmes.

Un aspect critique de l'autonomie est la géolocalisation, qui permet la planification de la navigation. Pour cela, le positionnement par satellites (GNSS) constitue généralement une méthode fiable et précise. Cependant, sa précision peut être compromise en raison d'une panne de capteur ou d'attaques malveillantes telles que le spoofing et le jamming. Ces types d'interférences ne sont pas limitées uniquements aux zones de guerre, par example, la Corée du Nord émet un fort brouillage depuis ses côtes, perturbant les avions et les navires voisins [1].

De plus dans des contextes d'application spécifiques, tels que la défense, la furtivité des USVs peut constituer un élément critique de la réussite d'une mission. Cet aspet restreint les capteurs pouvant être utilisés pour les processus de localisation, éliminant les capteurs actifs tels que le sonar, la lidar ou le radar en raison de leur détectabilité.

II. Méthode proposée

Pour obtenir une localisation précise à partir d'une image malgré la limitation du champ de vision tout en respectant les contraintes temps réel, nous proposons une chaîne de traitement illustrée sur la figure 2. Cette méthode est basée sur l'extraction de caractéristiques utilisant des CNN et une recherche des plus proches voisins pour extraire les hypothèses les plus probables et les visualiser sur une carte de chaleur.

Le principe consiste à générer un jeu de données suffisamment exhaustif de rendus d'horizons numériques à partir des données topographiques (DEM) visible sur la figure 1. Ensuite, après avoir entraîné un modèle ResNet-18 [2] pour reconnaître les similarités et les différences dans les horizons, une base de données vectorielle est créée pour stocker toutes les embeddings du jeu de données d'origine. Il convient de noter que tous les processus coûteux en termes de calcul sont effectués à l'avance sur le serveur de calcul. Finalement, seul la base de données des embeddings et le modèle entraîné sont stockés à bord du USV, limitant l'utilisation de la mémoire. Pour la localisation, le drone devra alors effectuer uniquement une inférence et une recherche des horizons les plus similaires. Comme mentionné précédemment, nous distinguons le jeu de données, qui représente les ensembles des images pour l'entrainement du deep-learning, et la base de données, qui elle représente le stockage des embeddings.

Figure 1. Example de rendu de simulation de la forme et de la profondeur du terrain depuis Unity3D.

Cette étude est réalisée avec le soutien financier de l'Agence de l'Innovation de Défense - Ministère des Armées - France.

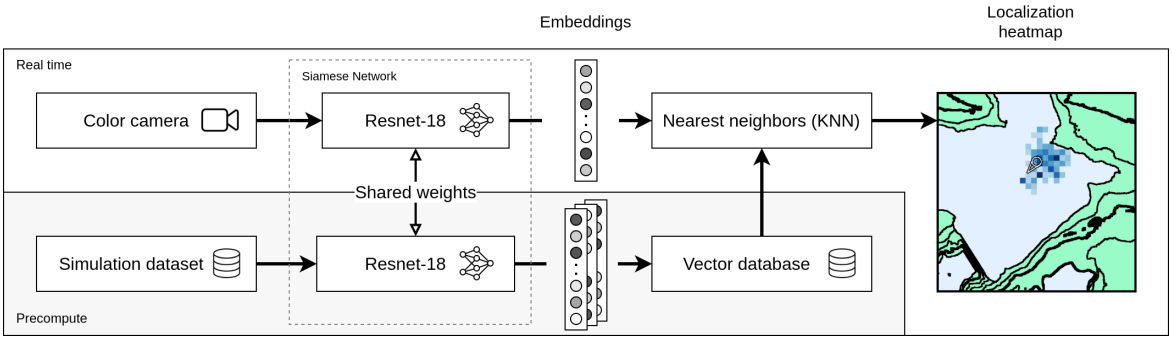


FIGURE 2. Pipeline de génération des cartes de probabilités de localisations.

Notre approche repose sur des réseaux siamois pour entraîner notre modèle à extraire les informations pertinentes depuis les images d'horizon. Cette méthode d'entrainement est appropriée dans notre cas, où nous devons former un modèle à trouver les similarités et les disparités entre des horizons perçus et simulés. Ces deux réseaux identiques partagent les mêmes poids et travaillent en parallèle sur des entrées différentes. Les deux vecteurs de sortie, appelés embeddings, sont ensuite comparés en calculant la distance euclidienne entre eux. L'objectif de cet entrainement est de garantir que la distance entre deux embeddings est inversement proportionnelle à la similitude des images. Cela signifie que plus la distance est faible entre deux images, plus les images devraient être similaires.

III. Resultats experimentaux

La méthode de localisation est évaluée par un taux de réussite, calculé en fonction de la précision maximale pour un nombre donné de voisins sur un jeu d'évalutation. Par exemple, le taux de réussite pour une précision de 1m parmi 10 voisins correspond au pourcentage de prédictions qui, parmi leurs 10 voisins les plus proches, ont un candidat inférieur à 1m de la position réelle de l'observateur. A noter que les résultats suivants sont obtenus en évaluant le modèle sur 238 images uniques dans une zone de 1,5km^2.

Sur la figure 4, nous pouvons observer le taux de réussite selon le nombre de voisins. Le plus intéressant est d'observer le cas le plus difficile, à savoir le taux de réussite pour un unique voisin. Cette courbe nous permet de dire que notre méthode, sans optimisation particulière est capable dans 70% des cas de fournir une localisation à moins de 50m dans une zone de 1,5km^2 parmis 13600 localisations possibles.

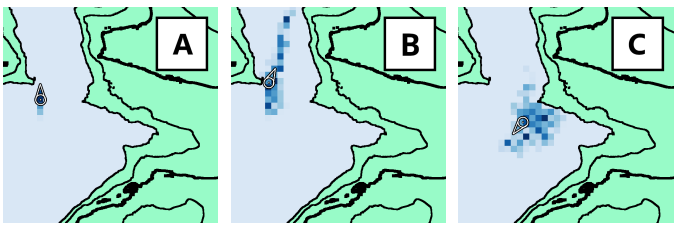


FIGURE 3. Cartes de probabilité générées sur une zone de 1.5km^2 avec les 200 plus proches voisins affichés. Le pointeur blanc représente la position GNSS réelle avec l'indicateur d'orientation. Autrement, l'intensité de la couleur est proportionnelle à la probabilité de localisation.

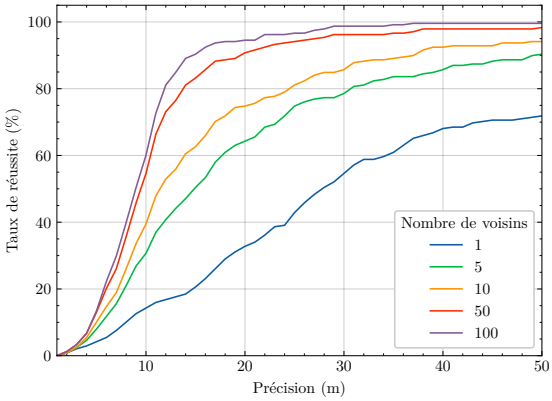


FIGURE 4. Taux de réussite en fonction de la précision et du nombre de voisins.

IV. Conclusion

Nos recherches constituent une preuve de concept d'une méthode de localisation visuelle en temps réel avec un champ de vision limité, basée sur l'extraction de caractéristiques par intelligence artificielle et recherche des plus proches voisins.

Cette preuve est un élément nécessaire dans le développement d'une méthode de localisation visuelle dans un environnement côtier sans GNSS par imagerie thermique.

Références

[1] J. Han, Y. Cho et J. Kim, « Coastal SLAM with marine radar for USV operation in GPS-restricted situations, » *IEEE Journal of Oceanic Engineering*, t. 44, n° 2, p. 300-309, 2019.

[2] K. He, X. Zhang, S. Ren et J. Sun, « Deep residual learning for image recognition, » in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, p. 770-778.

Adaptive layer compression and storage with QoS-aware loading for LLMs serving

Meriem Bouzouad
Lab-STICC, CNRS UMR 6285 , ENSTA
Institut Polytechnique de Paris
Brest,FRANCE
meriem.bouzouad@ensta.fr

Yuan-Hao Chang
Institute of Information Science
Academia Sinica
Tapei, TAIWAN
johnson@iis.sinica.edu.tw

Jalil Boukhobza
Lab-STICC, CNRS UMR 6285 , ENSTA
Institut Polytechnique de Paris
Brest,FRANCE
jalil.boukhobza@ensta.fr

***Abstract*—Today, large language models have demonstrated their strengths in various tasks ranging from reasoning, coding generation, complex problem solving. However, This advancement comes with high computational cost, and requires considerable memory to store the model parameters and request context, making it challenging to deploy these models on the edge devices to ensure real-time responses and data privacy.**

The rise of edge accelerators has significantly boosted on-device processing capabilities; however, memory is still a major bottleneck, a problem that is especially pronounced in large language models with high memory requirements, when it is unclear whether using all model's layers is crucial for maintaining generation quality. In addition to that, the varying workloads on edge devices exhibit an adaptive solution for the efficient utilization of resources. In this paper, we propose a flexible layer-wise compression approach that produces multiple model variants. Our solution leverages a smart storage mechanism to ensure efficient storage and rapid loading of the most appropriate variant tailored to the quality of service (QoS) requirements and the dynamic workload of the system.

***Index Terms*—LLM, memory optimization, embedded systems, edge intelligence.**

I. Introduction

The enhancement of reasoning capabilities in large language models (LLMs) is mainly associated with the growth in their number of parameters. Thus, enabling those models to capture complex relationships and dependencies within the data [1]. However, such scale-up necessitates substantial computational power and significant memory capacity, often requiring deployment on resource-rich cloud platforms for inference requests, specially to ensure scalability and assure quality of services for the users. This shift raises various security concerns, as well as a communication overhead, particularly due to internet connection and availability issues in certain regions.

With the advent of edge intelligence and real-time processing of data, deploying LLM models on the edge has gained much interest in research areas, especially with the advancement in hardware accelerators in edge devices, like integrated GPU, NPU, TPU and powerful CPUs. However, the memory bandwidth does not scale up with the compute speed, which hinders the overall performance of systems, especially for LLM models that require a significant amount of memory to store the weight matrices and context requests.

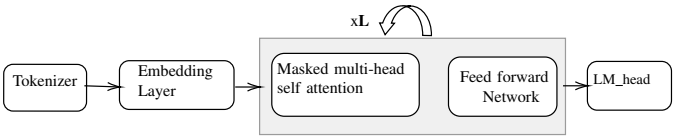


Fig. 1. Large language model architecture

Another challenge with LLMs is to serve multiple requests concurrently, where each request may have varying memory demands and priority levels, leading to dynamic workload with different compute and memory requirements that need to meet the desired quality of service, creating a more complex topic on resource allocation and efficient scheduling.

The decoder-only transformer architecture is used in LMMs [2], described in Fig. 1. The natural language prompt is converted to tokens, which are mapped to high-dimensional vectors called embeddings to represent the meaning of the tokens, those embeddings are then passed through L decoding layers, each containing a self-attention layer and feedforward network (FNN), to capture the contextual relationship between the different embeddings to predict the next token by the LM_head module. The FNN layers contribute the most to the memory footprint of LLM models, many studies focus on reducing their size with techniques like quantization, sparsification, parameter sharing,..etc, while some proposed methods to predict active neurons to reduce memory consumption [3].

An intriguing phenomenon emerges within the LLM layers: the outputs of two consecutive layers tend to be very similar [4]. This observation raises an important question about the necessity of passing through each layer for accurate prediction [5], opening the door to strategies such as early exit mechanisms and parameter sharing.

Our solution is designed on top of three core ideas: (1) **Offline layer wise compression module,** where we attribute a score to each layer and compress it accordingly. (2) **Offline multi-version model compression module** consists of producing and storing multiple model variants, each corresponding to a different level of compression determined by an aggressiveness factor α_j with an efficient storage solution not to overload the embedded storage media. (3) **Online adaptive loading module** that loads layer-wise compressions using

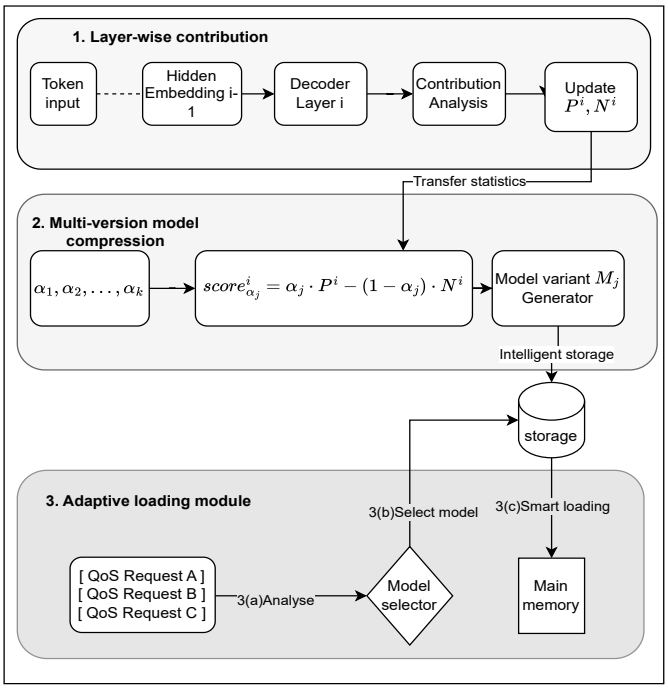


Fig. 2. Solution Overview

different models' versions according to the requested QoS metrics.

II. Contribution

This contribution is based on three modules, encompassing both offline and online phases described in Fig. 2, in offline phase, we use the layerwise contribution module to calculate the contribution of each layer and the multi-version model compression module to have different variants from different aggressive compression factors. On the online phase, we use the online adaptive loading module that chooses the best-fit model to load in order to respond to the QoS requirements and dynamic workload changes.

A. Layer-wise contribution module

The core idea behind this module is to calculate a score per layer depending on the contribution of the latter, as in LLMs, the embeddings are progressively transformed as they pass through the model's layers. Some layers introduce substantial changes to the hidden embeddings and are more likely to play a critical role in the model's output.In order to calculate the contribution score of each layer, we compute the cosine similarity between the embeddings of each two consecutive layers to calculate the contribution of the latter. We collect those statistics from multiple prompts from different tasks, like reasoning, Question-answering, coding, etc., to avoid task-specific bias and allow better generalization.

We propose a Reward-Penalty approach that uses layer-wise statistics in order to estimate the layer's contribution. Each layer i is evaluated by comparing its output to the previous layer $i-1$; if the distance exceeds a fixed threshold γ, we increase its reward P^i, reflecting an information gain, otherwise a penalty is applied by increasing N^i.

B. Multi-version model compression module

In order to have different models with different sizes that correspond to the memory footprint and quality of service, we use a compression aggressiveness factor α_j, to regularize the frequency of P^i and N^i, for instance, if we want to penalize our model, we decrease the value of α and hence decrease the contribution of the layer i using the equation:

$$score^i_{\alpha_j} = \alpha_j \cdot P^i - (1-\alpha_j) \cdot N^i \quad (1)$$

The score $score^i_{\alpha_j}$ will serve to calculate the compression factor. The layers scores take values in R, and then mapped to a probability distribution that sums to 1, in order to fairly distribute the contribution of each layer, by applying a softmax function. The result will be a probability vector that will measure the contribution of each layer i. A layer with a smaller probability is more susceptible to be quantized / pruned, therefore, those probabilities would be mapped to an interval of permitted quantization/sparsity values, in order to create a compressed model that we note M_{α_j}. An intelligent storage mechanism is integrated into this module to efficiently manage and store shared weights across different model variants.

C. Online adaptive loading module

During the online phase, inference is carried out using selectively loaded compressed model variant to maintain the desired QoS while reducing memory footprint. This is enabled by intelligent storage management that avoids loading all model layers simultaneously. The module dynamically selects and loads the most suitable model variant based on current system workload and QoS related to request priority, energy consumption..etc, by smart storage access module.

III. Conclusion

In conclusion, the proposed solution has as goal to enable LLM inference on resource-constrained edge devices with a dynamicity to determine the compression rate of each layer based on its contribution. We believe that our contribution will make a trade-off between efficiency and generative quality of the LLM models while ensuring a quality of service that can fit the dynamic nature of workloads on edge devices.

References

[1] J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, D. Yogatama, M. Bosma, D. Zhou, D. Metzler *et al.*, "Emergent abilities of large language models," *arXiv preprint arXiv:2206.07682*, 2022.

[2] Y. Chen, G. Ou, M. Liu, Y. Wang, and Z. Zheng, "Are decoder-only large language models the silver bullet for code search?" *arXiv preprint arXiv:2410.22240*, 2024.

[3] Z. Liu, J. Wang, T. Dao, T. Zhou, B. Yuan, Z. Song, A. Shrivastava, C. Zhang, Y. Tian, C. Re *et al.*, "Deja vu: Contextual sparsity for efficient llms at inference time," in *International Conference on Machine Learning*. PMLR, 2023, pp. 22 137–22 176.

[4] R. Miao, Y. Yan, X. Yao, and T. Yang, "An efficient inference framework for early-exit large language models," *arXiv preprint arXiv:2407.20272*, 2024.

[5] Y. Chen, X. Pan, Y. Li, B. Ding, and J. Zhou, "Ee-llm: Large-scale training and inference of early-exit large language models with 3d parallelism," *arXiv preprint arXiv:2312.04916*, 2023.

Hardware telemetry for the security of embedded systems

Adam Henault
Université Bretagne Sud
UMR 6285, Lab-STICC
Lorient, France
adam.henault@univ-ubs.fr

Camille Monière
Université Bretagne Sud
UMR 6285, Lab-STICC
Lorient, France
camille.moniere@univ-ubs.fr

Philippe Tanguy
Université Bretagne Sud
UMR 6285, Lab-STICC
Lorient, France
philippe.tanguy@univ-ubs.fr

Vianney Lapôtre
Université Bretagne Sud
UMR 6285, Lab-STICC
Lorient, France
vianney.lapotre@univ-ubs.fr

Format de la présentation: Poster
Langue de la présentation : Français/Anglais

I. Introduction

Dans un monde de plus en plus interconnecté, les systèmes embarqués sont devenus des cibles privilégiées pour un large éventail de menaces de sécurité. Les systèmes sur puce (SoC) embarqués doivent aujourd'hui faire face à des attaques toujours plus sophistiquées : malwares embarqués, fuites d'information via canaux auxiliaires (side-channels), injections de fautes, etc. Cette diversité des vecteurs d'attaque rend la protection des SoC particulièrement complexe, d'autant plus qu'ils évoluent dans des environnements contraints où les ressources (énergie, mémoire, calcul) sont limitées. Dans ce contexte, le monitoring du comportement à l'exécution apparaît comme une stratégie de défense efficace. Être capable de détecter en temps réel des anomalies comportementales, des déviations par rapport à un fonctionnement attendu ou des signes avant-coureurs d'une compromission, permet non seulement de renforcer la résilience du système, mais aussi d'activer des contre-mesures dynamiques adaptées. Pour nos travaux, nous nous sommes intéressés à deux approches de monitoring des processeurs qui par la suite pourront être étendu à l'ensemble des SoC.

II. Etat de l'art

Les Hardwares Performance Counters (HPC) sont des compteurs matériels intégrés aux processeurs, permettant de mesurer des événements micro-architecturaux (instructions exécutées, cache misses, branchements, etc.). Initialement conçus pour le profilage et l'optimisation, ils ont été largement détournés en outils de sécurité ces dernières années. En cybersécurité embarquée, les HPC servent principalement à détecter des anomalies d'exécution (indiquant par exemple la présence de malwares ou d'exploits) et à détecter des attaques par canal auxiliaire (notamment des attaques micro-architecturales sur les caches). De nombreux travaux [1], [2], [3] exploitent ainsi les modèles de données fournis par les HPC pour repérer un comportement malveillant à l'exécution, par exemple en comparant des profils de performance à un profil de référence sain. Cette popularité s'explique par plusieurs facteurs : les HPC offrent une vision à grain fin sur le comportement du processeur en temps réel et sont disponibles sur des plateformes embarquées courantes (ARM, RISC-V, etc.).
Malgré leur intérêt, les HPC présentent des limitations lorsqu'on les utilise à des fins de sécurité :

- Bruit et non-déterminisme : Les valeurs de compteurs peuvent varier d'une exécution à l'autre de façon imprévisible (interruptions, interférences d'autres processus), introduisant du bruit dans les mesures. Même de faibles fluctuations peuvent tromper un classifieur ou un seuil de détection sensible. Il est donc délicat d'établir des signatures fiables sans méthode de filtrage ou d'agrégation.
- Couverture partielle : Un jeu limité de HPC ne peut capturer qu'une vue partielle du comportement du système. Seuls certains événements micro-architecturaux sont comptabilisés, et il n'est pas possible de tout surveiller simultanément (nombre de compteurs hardware restreint par rapport aux événements disponibles). Par conséquent, une attaque peut passer inaperçue si elle n'affecte pas les événements monitorés, ou si elle exploite un aspect non mesuré par les HPC présents.
- Manque de contexte : Les données brutes fournies par les HPC manquent de haut niveau sémantique. Un même profil de compteurs peut provenir d'un comportement bénin ou malveillant, ce qui rend l'interprétation directe difficile. Il faut souvent recourir à des analyses sophistiquées (modèles d'IA, corrélation avec le flux d'exécution, etc.) pour inférer la cause d'une anomalie observée dans les compteurs.

Les sondes matérielles permettent d'extraire des signaux microarchitecturaux révélateurs du comportement d'un système. Ces signaux ont été exploités pour la classification de comportements logiciels et la détection d'attaques. Les mécanismes matériels standard (HPC, interfaces de debug comme Arm CoreSight) n'ont pas été conçus à l'origine pour la cybersécurité : ils offrent un choix limité de signaux observables et un débit restreint d'informations. Cela réduit la granularité des observations et peut nuire à la capacité de détection d'attaques. Pour dépasser ces limites, des recherches récentes [4], [5] explorent l'ajout de sondes dédiées au sein

même de la microarchitecture, afin de capturer des signaux internes plus riches et de les analyser en continu à des fins de sécurité. Malgré leur potentiel, les solutions de sécurité basées sur des sondes matérielles présentent plusieurs limitations et défis. D'abord, la complexité d'intégration est non négligeable, instrumenter un processeur avec des sondes personnalisées ou des modules reconfigurables requiert une connaissance approfondie de la microarchitecture et des attaques à contrer, ainsi que de nombreux ajustements expérimentaux pour identifier les signaux pertinents et calibrer la détection. Ceci complique l'adoption de ces techniques sur des processeurs commerciaux fermés, d'autant que chaque architecture peut nécessiter un modèle de menace et un paramétrage spécifiques. De plus, le coût matériel n'est pas nul. Bien que des travaux comme MATANA [4] montrent un overhead faible, l'ajout de logique de surveillance consomme des ressources silicium et de l'énergie, ce qui peut être critique dans les systèmes embarqués contraints.

III. Approche envisagée

Nous proposons d'étendre les approches existantes décrites dans l'état de l'art en les généralisant à l'ensemble du système sur puce (SoC). Cette extension débute par le processeur, en ciblant les limitations actuelles des compteurs de performance matériel (HPC). Une des pistes envisagées consiste à introduire de nouveaux HPC spécifiquement dédiés à la sécurité, capables de mesurer des indicateurs pertinents dans ce domaine. Afin de limiter les interférences et le bruit dans la collecte de ces données, un mécanisme de filtrage devra également être intégré. Comme illustré à la Fig. 1, chaque cœur du processeur serait doté de HPC personnalisés pour la sécurité, associés à un système de filtrage local.

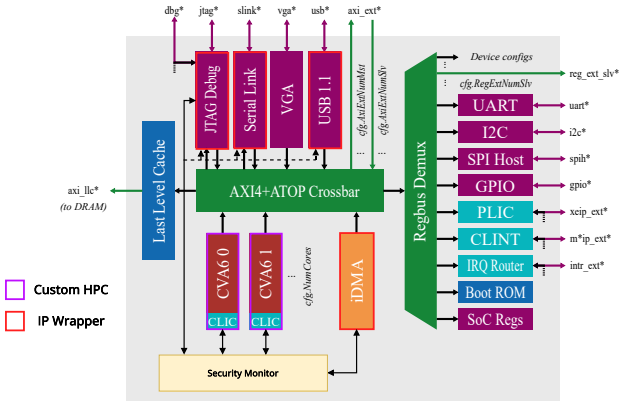


Fig. 1: Représentation simplifiée du SoC Cheshire [6] avec des moniteurs de sécurité.

Ces compteurs spécialisés pourront ensuite être adaptés à d'autres composants matériels du SoC, tels que les bus de communication, les mémoires cache, etc. Par ailleurs, nous nous intéressons au défi que représente la supervision de modules matériels considérés comme des boîtes noires. Étant donné l'impossibilité d'instrumenter directement ces blocs IP propriétaires, nous proposons le développement de wrappers permettant d'observer leur comportement de manière indirecte. Il est alors nécessaire d'identifier des métriques génériques, pertinentes pour ces modules, afin de faire émerger des indicateurs représentatifs de leur fonctionnement. Ces métriques pourront elles aussi être mises en évidence à l'aide de HPC intégrés dans les wrappers.

IV. Conclusion

En conclusion, nos travaux visent à pallier les limitations actuelles du monitoring des systèmes sur puce en contexte de sécurité. Pour ce faire, nous proposons la définition de métriques dédiées à la sécurité, construites à partir d'un modèle de menace défini. Ces métriques pourront être déployées de manière généralisée sur l'ensemble du SoC, permettant ainsi une observation fine et cohérente du comportement du système.

References

[1] S. Das, J. Werner, M. Antonakakis, M. Polychronakis, and F. Monrose, "Sok: The challenges, pitfalls, and perils of using hardware performance counters for security," in *2019 IEEE Symposium on Security and Privacy (SP)*, 2019, pp. 20–38.

[2] A. P. Kuruvila, A. Mahapatra, R. Karri, and K. Basu, "Hardware performance counters: Ready-made vs tailor-made," *ACM Trans. Embed. Comput. Syst.*, vol. 20, no. 5s, Sep. 2021. [Online]. Available: https://doi.org/10.1145/3476996

[3] P. Bhade, J. Paturel, O. Sentieys, and S. Sinha, "Lightweight hardware-based cache side-channel attack detection for edge devices (edge-cascade)," *ACM Trans. Embed. Comput. Syst.*, vol. 23, no. 4, Jun. 2024. [Online]. Available: https://doi.org/10.1145/3663673

[4] Y. Mao, V. Migliore, and V. Nicomette, "Matana: A reconfigurable framework for runtime attack detection based on the analysis of microarchitectural signals," *Applied Sciences*, vol. 12, no. 3, 2022. [Online]. Available: https://www.mdpi.com/2076-3417/12/3/1452

[5] T. Langehaug and S. Graham, "Cumonitor: Continuous monitoring of microarchitecture for software task identification and classification," *Digital Threats*, vol. 5, no. 3, Sep. 2024. [Online]. Available: https://doi.org/10.1145/3652861

[6] A. Ottaviano, T. Benz, P. Scheffler, and L. Benini, "Cheshire: A lightweight, linux-capable risc-v host platform for domain-specific accelerator plug-in," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 70, no. 10, pp. 3777–3781, 2023.

Using I/Os for Temperature Regulation for GMM Learning

Lina GANA
Lab-STICC, CNRS UMR 6285 , ENSTA, Institut Polytechnique de Paris, 29806, Brest, France
École Nationale Supérieure D'Informatique (ESI)
Algiers, Algeria
kl_gana@esi.dz

Yanis MOHAMMEDI
Lab-STICC, CNRS UMR 6285 , ENSTA, Institut Polytechnique de Paris, 29806, Brest, France
École Nationale Supérieure D'Informatique (ESI)
Algiers, Algeria
ky_mohammedi@esi.dz

Meriem Bouzouad
Lab-STICC, CNRS UMR 6285 , ENSTA, Institut Polytechnique de Paris, 29806 Brest, France
meriem.bouzouad@ensta.fr

Jalil Boukhobza
Lab-STICC, CNRS UMR 6285 , ENSTA, Institut Polytechnique de Paris, 29806 Brest, France
jalil.boukhobza@ensta.fr

***Abstract*—Nowadays embedded systems are widely used for data processing and machine learning tasks due to their low cost and energy efficiency. However, training machine learning models on edge devices without any active cooling may result in a spike in temperature, which can impact performance or even cause system failure due to overheating. This leads us to a critical question: How do we enable learning on resource constrained devices while limiting temperature increase? In this paper, we investigate this problem in a case study using Gaussian Mixture Models (GMMs). We introduce an innovative solution that leverages I/O operations as a passive cooling mechanism to manage overheating and avoid thermal throttling during training.**

***Index Terms*—Machine learning algorithms, embedded systems, Gaussian Mixture Model, Clustering, Temperature management, ARM, I/O optimisation.**

I. Introduction

Machine learning algorithms have been designed to run on high-performance machines with abundant computational resources and large memory. However, the need to process data locally, to reduce latency and enhance security, has led to the rise of Edge Intelligence as an interesting alternative. This shift introduces new challenges to run these algorithms such as energy management and thermal management, an often neglected yet crucial aspect [6].

Training ML models on embedded devices can rapidly increase processor temperature, which could lead to overheating and trigger thermal throttling [2] thereby degrading performance. Several thermal management techniques have been proposed. Some adjust frequency using power models or machine learning; others use core mapping. ComBoost [1] dynamically tunes voltage and frequency based on instruction complexity. EdgeCoolingMode [2] employs a smart agent with linear regression heuristics to monitor and control temperature. The scheduling method in [5] migrate tasks from hot to cooler cores. In [3], a PMC-based model enables precise power estimation and dynamic frequency scaling. State-of-the-art solutions manage temperature by lowering frequency or by scheduling, which affects performance as both techniques slow down computation.

Our study focuses on the GMM clustering algorithm known for its high computational cost. Authors in [4] introduce PIGMMaLIOn, an optmized version of GMM designed for embedded systems with memory constraints. PIGMMaLIOn [4] divides the dataset into chunks (increments), each being loaded into memory and processed independently to generate a partial GMM model. These partial models are then merged to form a global GMM. PIGMMaLIOn [4], however, does not take thermal constraints into account, which may lead to major issues such as increased latency or material damage; in our study, we focus on the thermal aspect. In PIGMMaLIOn [4], the execution is alternated between the loading and processing phases. During the Input/Output (I/O) phases, the cores remain inactive, which promotes a drop in temperature. Our approach leverages this behavior and proposes an objective function, based on online learning models, that sets the largest optimal size of each increment to take advantage of the I/O phase while keeping the temperature below the threshold, thus avoiding any loss of performance.

II. Empirical Analysis

In PIGMMaLIOn [4], each increment undergoes a loading phase where data is loaded from storage system to the main memory, and a processing phase where the EM algorithm is run to form a partial GMM. In this section, we observe the impact of I/O on temperature reduction via experiments.

Execution analysis, as shown in Figure 1, confirms that I/O phases introduce temporary periods of inactivity for the CPU and therefore, causes the processor to cool down when it is at high temperatures. A second observation is that, for an increment N, the temperature curve ($T_{I/O}$) evolves linearly during the I/O phase (t_L), whereas the processing temperature

(T_{proc}) follows a logarithmic trend during the processing phase (t_P).

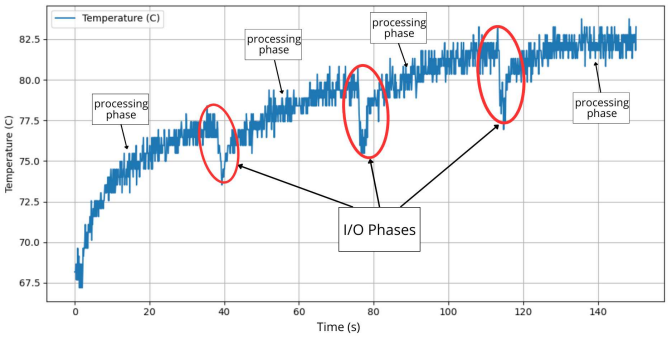


Fig. 1: I/O and processing phases in PiGMMaLiOn algorithm

III. Proposed solution

In PiGMMaLIOn [4], processing is done in fixed-size increments where data loading and processing phases are clearly separated. The goal is to dynamically adapt the increment size (s) based on the available thermal budget. In other words, we want to process the largest possible size (s) while remaining below a defined temperature threshold (T_{max}).

Our proactive approach is based on two online models; during the execution of increment $N-1$, we collect information to learn the execution behavior. Linear regression is used to model the variation in execution time based on size during the I/O phase (t_L) and the processing phase (t_P). This helps identify the relation between data size and execution time.

Moreover, and still based on the increment $N-1$, a second model captures the evolution of temperature over time. A linear regression is used to model the temperature drop ($T_{I/O_{N-1}}$) during the I/O phase (t_L), while a logarithmic regression is applied to the temperature evolution ($T_{proc_{N-1}}$) during the processing phase (t_P). These models are used to estimate the temperature for any given increment size and determine the largest size that stays below T_{max}.

A. Time prediction model based on size

The objective is to model the I/O and processing phases (t_L and t_P) for each size (s) of increment N.

a) I/O Phase: The I/O time (t_L) is estimated based on size s using linear regression, with the throughput (Tp) and an offset b:

$$t_L = \frac{s}{Tp} + b \tag{1}$$

b) Processing Phase: Similarly, the processing time (t_P) is modeled using linear regression:

$$t_P = a_0 \times s + b_0 \tag{2}$$

B. Temperature prediction based on time

The objective is to model the decrease of temperature $T_{\text{I/O}_N}$ over time t_L during the I/O phase, as well as the increase of temperature T_{proc_N} over time t_P during the processing phase.

a) I/O Phase: In the data loading process, the temperature generally evolves linearly over time, as shown in figure 1. A linear regression estimates the temperature variation on t_L, adjusted by substituting the offset (originally the initial temperature of increment $N-1$) with the initial temperature of increment N (T_{init_N}).

$$T_{\text{I/O}_N} = a_2 \times t_L + T_{\text{init}_N} \tag{3}$$

b) Processing Phase: The temperature evolution T_{proc_N} follows a logarithmic trend (figure 1). So, a logarithmic regression models the temperature increase during processing, also adjusted by the temperature after loading increment N ($T_{\text{I/O}}$):

$$T_{\text{proc}_N} = a_3 \times log(t_P) + T_{\text{I/O}_N} \tag{4}$$

C. Objective Optimization Function

By replacing t_L and t_P with their expressions in the temperature equations, and replacing $T_{\text{I/O}_N}$ by its expression we obtain:

$$T_{\text{proc}_N}(s) = a_3 \times \log(a_0 \times s + b_0) + (a_2 \times (\frac{s}{Tp} + b) + T_{\text{init}_N}) \tag{5}$$

Combining everything, the final equation allows predicting the processing temperature based on size s, and the optimization problem becomes:

$$\max_{S} \quad \text{subject to} \quad T_{proc}(s) \leq T_{max} \tag{6}$$

IV. Conclusion

We proposed a solution that uses I/O phases as a cooling strategy, dynamically adjusting I/O size in PIGMMaLIOn. By combining predictive models, our proactive approach manages the workload while ensuring thermal limits are not exceeded.

References

[1] S. H. Choi, J. Kong, et S. W. Chung, « ComBoost: An Instruction Complexity Aware DTM Technique for Edge Devices », in Proceedings of the 29th ACM/IEEE International Symposium on Low Power Electronics and Design, in ISLPED '24. New York, NY, USA: Association for Computing Machinery, sept. 2024, p. 1-6.

[2] S. Dey, E. Z. Guajardo, K. R. Basireddy, X. Wang, A. K. Singh, et K. McDonald-Maier, « EdgeCoolingMode: An Agent Based Thermal Management Mechanism for DVFS Enabled Heterogeneous MPSoCs », in 2019 32nd International Conference on VLSI Design and 2019 18th International Conference on Embedded Systems (VLSID), janv. 2019, p. 19-24.

[3] N. Che et al., « OS-Level PMC-Based Runtime Thermal Control for ARM Mobile CPUs », IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 43, no 7, p. 2023-2036, juill. 2024

[4] M. Bouzouad, Y. Benhamadi, C. Slimani, et J. Boukhobza, « PIGMMaLIOn: a Partial Incremental Gaussian Mixture Model with a Low I/O Design », in Proceedings of the 39th ACM/SIGAPP Symposium on Applied Computing, in SAC '24. New York, NY, USA: Association for Computing Machinery, mai 2024, p. 428-435.

[5] H. Khan, Q. Bashir, et M. U. Hashmi, « Scheduling based energy optimization technique in multiprocessor embedded systems », in 2018 International Conference on Engineering and Emerging Technologies (ICEET), févr. 2018, p. 1-8.

[6] T. Adegbija et A. Gordon-Ross, «Temperature-aware Dynamic Optimization of Embedded Systems », 14 février 2016

Flexible Framework for Implementing Spiking Neural Networks on FPGA

Loïc THOMAS
LAAS-CNRS, Université de Toulouse, CNRS, INSA Toulouse.
Toulouse, France
loic.thomas@laas.fr
0009-0001-0375-3185

Gaël LOUBET
LAAS-CNRS, Université de Toulouse, CNRS, INSA Toulouse.
Toulouse, France
gael.loubet@laas.fr
0000-0003-3347-0036

Daniela DRAGOMIRESCU
LAAS-CNRS, Université de Toulouse,CNRS, INSA Toulouse.
Toulouse, France
daniela.dragomirescu@laas.fr
0000-0001-8589-6093

***Abstract*— In this work, a simple and flexible 3-stages framework is proposed for implementing feedforward Spiking Neural Networks on FPGA. Our goal is to emulate the behavior of a neural circuit for a variety of edge AI applications. The Spiking Neural Network is designed and trained with surrogate gradient. Then, the weights are quantized into integers. Finally, the model is implemented on a Basys 3 Artix 7 FPGA board.**

Keywords—Spiking Neural Network (SNN), Neuromorphic Architecture, Edge AI, Field-Programmable Gate Arrays (FPGA).

I. INTRODUCTION

In current Wireless Sensor Networks (WSN), Artificial Intelligence (AI) tasks are more often delocated and processed in the cloud. While solutions aim to optimize data transmission between sensor nodes and remote servers, an alternative –named Edge AI– consists in bringing AI computation near the sensors. This would significantly reduce decision latency, network congestion and required resources.

However, integrating AI at the edge introduces several constraints, primarily due to the limited energy, computational and memory resources inherent to WSN. Traditional Von Neumann architectures –characterized by the separation of memory and computation units– are not well suited for such scenarios. Their high energy consumption, driven by intensive memory access, becomes a major bottleneck for low-power embedded devices [1], [2].

Neuromorphic architectures –inspired by the structure and function of the human brain– offer a better alternative. By doing event-driven processing and in-memory computation, they enable highly parallel, low-power execution of neural tasks [3]. Within these architectures, Spiking Neural Networks (SNN) seems to be the best fit for biological plausibility and computational efficiency.

There already exists some notable neuromorphic architectures, such as Intel's Loihi [4] or IBM's TrueNorth [5]. FPGA-based ones, as RANC [6], exist too. These have already demonstrated high performances with a low power consumption. However, they rely on complex Network-on-Chip (NoC) architectures, which takes them away from the brain behavior. Analog neuromorphic circuits which replicates brain behavior more closely *via* their asynchronous operations could be even better suited for edge applications [7], [8]. However, their high cost and complexity might make them less scalable.

II. NETWORK INITIALISATION

A. Spiking Neural Networks

SNN mimic brain-like computation by processing discrete events called spikes. Among various neuron models, the Leaky Integrate-and-Fire (LIF) one is a good balance between biological plausibility and complexity [3]. LIF neurons weight incoming spikes in the synapse and integrate their values to the neuron membrane potential. If the neuron potential reaches a threshold, a spike is fired**.** When there are no incoming spikes in the synapses the potential leaks and is reduced by a factor β.

SNN can adopt a variety of architectures, ranging from conventional topologies, like feedforward networks with fully connected or convolutional layers [9], to more biologically inspired structures featuring recurrent connections, such as Liquid State Machines (LSM) [10].

Even if spike activity is not differentiable, SNN can still be trained using the backpropagation algorithm by using surrogate gradient technique [11]. SNN can also be trained with unsupervised ways as the Synaptic Temporal Dependent Plasticity (STDP) where the weights are modified depending on the spiking activity allowing the network to learn spiking patterns [12].

B. Software simulation of the SNN

Our SNN models are implemented with the Snntorch Python library. They are designed to process spike trains directly as input, iterating through each time step before producing an output. The predicted class corresponds to the output neuron that accumulates the highest number of spikes over time.

Each layer in the network is made of a fully connected linear layer followed by several LIF neurons. Training is performed using the surrogate gradient method, where only the weights of the fully connected layers are updated. The leaky factor β is fixed at 0.5 to match the hardware implementation, which uses a single right shift to compute membrane potential leak between iterations.

III. HARDWARE IMPLEMENTATION

A. Quantization

To facilitate hardware implementation, a post-training quantization is applied to the network weights. The quantization process rescales the floating-point weights into integers. The process is simple, the maximum weight of the layer is scaled to a given integer value. After that, each weight is scaled accordingly.

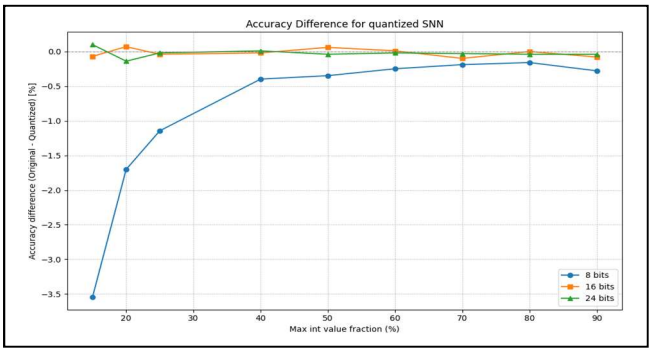


Fig. 1. Accuracy precision loss after quantization

For example, with an 8-bit representation, weights are stored as signed integers in the range [−255; 255], while neuron potentials use unsigned values in [0; 512[. As shown in Fig. 1, this quantization has minimal impact on accuracy. The reduced precision can be seen as noise on the weights, something SNN are resilient to [13]. With enough discrete weight levels, the behavior remains close to a float32 SNN. In Fig. 1, the x-axis shows the fraction of the maximum weight. For 8 bits, a 0.2 fraction corresponds to a maximum weight of 51. Even with only 102 weight values, the accuracy drops by just 1.7 %.

B. VHDL design

Our VHDL design is based on a synchronous implementation of the LIF neuron model. Upon receiving a spike train, the neuron integrates incoming spikes iteratively, processing one spike per clock cycle. After the entire spike train has been integrated, the membrane potential undergoes a right-shift operation to simulate leakage, then, the accumulated variations are applied.

Spikes are represented with a high-level state during one clock cycle. Since, spikes can arrive faster than the neuronal process, we need to add a buffer before the layers.

Post-simulation results shown a critical path delay of less than 10 ns (for the whole network). Since neurons are processed in parallel, the total processing time scales with the number of synapses per neuron and the spike train length. For MNIST (784 synapses, 10 spikes), this results in a processing time of 0.08 ms per input. As shown in Table I, FPGA utilization is mainly affected by the number of neurons, due to parallel processing. In contrast, increasing synapses impacts latency but not resource usage. Our architecture would suit small SNN better. A single-layer network with 10 neurons achieves 91% accuracy on MNIST, using only 3,820 LUTs and 825 slice registers.

TABLE I. FULLY CONNECTED LAYER UTILIZATION ON BASYS 3 ARTIX 7 BOARD

Input synapses	**Output neurons**				
	5	10	20	50	100
10	LUT: 577 Reg: 291	LUT: 1,095 Reg: 569	LUT: 2,154 Reg: 1,092	LUT: 5,004 Reg: 2,710	LUT: 10,103 Reg: 5,446
20	LUT: 596 Reg: 290	LUT: 1,171 Reg: 566	LUT: 2,070 Reg: 1,113	LUT: 5,072 Reg: 2,768	LUT: 10,237 Reg: 5,523
50	LUT: 642 Reg: 295	LUT: 1,253 Reg: 575	LUT: 2,465 Reg: 1,136	LUT: 6,817 Reg: 2,833	LUT: 12,302 Reg: 5,624
100	LUT: 804 Reg: 337	LUT: 1,683 Reg: 660	LUT: 3,025 Reg: 1,299	LUT: 8,293 Reg: 3,234	LUT: 15,861 Reg: 6,437

Fig. 2. LUT and slice registers utilisation of a fully connected layer on Basys 3 Artix 7 FPGA board.

IV. CONCLUSION

We implemented a trained SNN directly into a FPGA. Although the current framework supports only small-scale networks, it is designed to be easily extended and adapted. While it has not yet been applied to WSN applications, the architecture is general enough to support any dataset. The simplicity and modularity of our implementation make it a strong foundation for future developments, including the integration of more complex network structures such as convolutional layers or LSM. Our quantization algorithm shown a remarkable robustness as it induced a negligible overhead in accuracy.

REFERENCES

[1] C. Loyez, K. Carpentier, I. Sourikopoulos, et F. Danneville, « Subthreshold neuromorphic devices for Spiking Neural Networks applied to embedded A.I », in *2021 19th IEEE International New Circuits and Systems Conference (NEWCAS)*, juin 2021, p. 1-4. doi: 10.1109/NEWCAS50681.2021.9462779.

[2] D. V. Christensen *et al.*, « 2022 roadmap on neuromorphic computing and engineering », *Neuromorphic Comput. Eng.*, vol. 2, nº 2, p. 022501, juin 2022, doi: 10.1088/2634-4386/ac4a83.

[3] C. D. Schuman *et al.*, « A Survey of Neuromorphic Computing and Neural Networks in Hardware », 19 mai 2017, *arXiv*: arXiv:1705.06963. doi: 10.48550/arXiv.1705.06963.

[4] M. Davies *et al.*, « Loihi: A Neuromorphic Manycore Processor with On-Chip Learning », *IEEE Micro*, vol. 38, nº 1, p. 82-99, janv. 2018, doi: 10.1109/MM.2018.112130359.

[5] S. Moran, B. Gaonkar, L. Macyszyn, W. Whitehead, A. Wolk, et S. S. Iyer, « Deep learning for medical image segmentation – using the IBM TrueNorth neurosynaptic system », in *Medical Imaging 2018: Imaging Informatics for Healthcare, Research, and Applications*, J. Zhang et P.-H. Chen, Éd., Houston, United States: SPIE, mars 2018, p. 39. doi: 10.1117/12.2286419.

[6] V. C. Nguyen, L. T. Nguyen, H. P. Dam, D. M. Nguyen, et H. H. Nguyen, « RANC-Based Hardware Implementation of Spiking Neural Network for Sleeping Posture Classification », in *Computer Vision and Robotics*, P. K. Shukla, H. Mittal, et A. Engelbrecht, Éd., Singapore: Springer Nature, 2023, p. 259-271. doi: 10.1007/978-981-99-4577-1_21.

[7] Z. Jouni, T. Soupizet, S. Wang, A. Benlarbi-Delai, et P. M. Ferreira, « 1.2 nW Neuromorphic Enhanced Wake-Up Radio », in *2022 35th SBC/SBMicro/IEEE/ACM Symposium on Integrated Circuits and Systems Design (SBCCI)*, août 2022, p. 1-6. doi: 10.1109/SBCCI55532.2022.9893247.

[8] E. Dalmas, F. Danneville, M. Bocquet, et C. Loyez, « Neuromorphic Coincidence Detector for Interaural Time Difference Encoding and Sound DOA Estimation », *IEEE Trans. Instrum. Meas.*, vol. 73, p. 1-11, 2024, doi: 10.1109/TIM.2024.3460950.

[9] M. Dong, X. Huang, et B. Xu, « Unsupervised speech recognition through spike-timing-dependent plasticity in a convolutional spiking neural network », *PLOS ONE*, vol. 13, nº 11, p. e0204596, nov. 2018, doi: 10.1371/journal.pone.0204596.

[10] Y. Zhang, L. Mo, X. He, et X. Meng, « Unsupervised spiking neural network based on liquid state machine and self-organizing map », *Neurocomputing*, vol. 620, p. 129120, mars 2025, doi: 10.1016/j.neucom.2024.129120.

[11] J. K. Eshraghian *et al.*, « Training Spiking Neural Networks Using Lessons From Deep Learning », *Proc. IEEE*, vol. 111, nº 9, p. 1016-1054, sept. 2023, doi: 10.1109/JPROC.2023.3308088.

[12] T. Masquelier, R. Guyonneau, et S. J. Thorpe, « Spike Timing Dependent Plasticity Finds the Start of Repeating Patterns in Continuous Spike Trains », *PLOS ONE*, vol. 3, nº 1, p. e1377, janv. 2008, doi: 10.1371/journal.pone.0001377.

[13] W. Maass, « Networks of spiking neurons: The third generation of neural network models », *Neural Netw.*, vol. 10, nº 9, p. 1659-1671, déc. 1997, doi: 10.1016/S0893-6080(97)00011-7.

Implementation and Training of Convolutional Neural Networks using Adder Graphs

Rémi Garcia, Léo Pradels, Silviu-Ioan Filip, Olivier Sentieys
Université de Rennes, IRISA, Inria
Rennes, France
{remi.garcia1, leo.pradels, silviu.filip, olivier.sentieys}@inria.fr

I. Introduction

Convolutional Neural Network (CNN) models are used for many computer vision tasks such as super-resolution or image classification. Larger and larger networks are used to obtain better accuracy [1]. However, this trend is not easily compatible with CNN implementation on embedded systems. Considering resource constraints led to new techniques aiming at compressing large networks without degrading the accuracy too much. In particular, two common compression approaches are quantization [2] and pruning [3]. In our work, we consider the quantization of CNN whose goal is to reduce the weight format, from standard Single-Precision Floating-Point (FP32) to less costly Fixed-Point (FxP) values with small bit widths. Using such a method is basically a necessary step for dedicated resource-constrained hardware implementations [4], [5].

Quantization can be performed at two different stages, either during the training process or after the training. Quantization-Aware Training (QAT) optimizes quantization during the training to keep an accuracy close to the FP32 reference, even for small FxP bit widths such as 4 bits [2]. However, access to the training or to the dataset is often out of reach, and there is a need to quantize pretrained networks instead. Applying a Post-Training Quantization (PTQ) method [6] is a straightforward way to still compress CNNs.

Quantization can be either symmetric or asymmetric [7] and involves a scaling factor. In our work, we only consider symmetric quantization and power-of-2 scaling factors. This way, we avoid common implementation issues, which are often ignored, and certainly lead to what has been called the "Hardware Gap" [8]. Thus, to quantize FP32 weights W to b-bit FxP quantized weights W_{FxP}, we apply

$$W_{\text{FxP}} = \text{clip}\left(\text{round}\left(\frac{W}{S}\right), -2^{b-1}, 2^{b-1}-1\right), \quad (1)$$

where

$$S = \frac{2^{\lceil \log_2(\max|W|)\rceil}}{2^b}. \quad (2)$$

It is also possible, as in DoReFa-Net [9], to introduce a transformation to W before applying the quantization.

The rounding operator differs between quantization approaches. For example, the most straightforward approach, linear quantization, applies a simple rounding to the nearest. Other quantization approaches choose different rounding to maintain a better accuracy, as AdaRound [10], or to round towards precomputed integer subsets which can be implemented at a lower cost. For example, rounding towards power of 2 as proven efficient to reduce implementation cost [11].

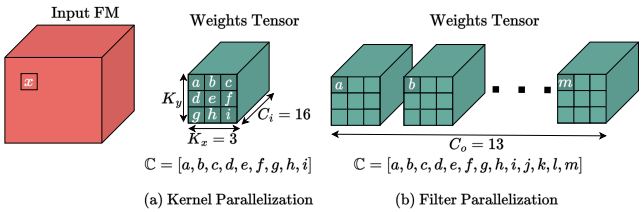


Fig. 1: Example of possible parallizations in CNNs.

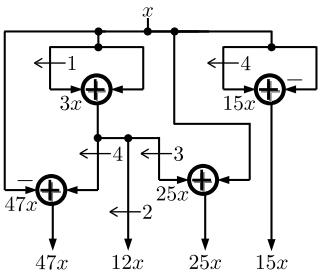
In our work, we propose different techniques to improve quantization, taking the implementation cost into account. This implementation can be performed in various manners. But, in any case, weight-input multiplications occur, and this is the part of the circuit that we focus on. These multiplications by constants (the weights) can be performed on parallel as illustrated in Fig. 1. Replacing generic multipliers with adder graphs is an efficient way to implement parallel multiple constant multiplications [12]–[14].

II. Adder Graph-Aware Quantizations

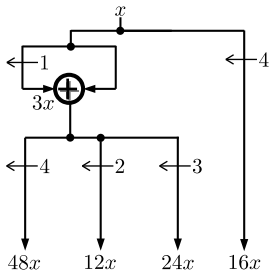
A. Post-Quantization Training

Implementing a pretrained model on embedded systems usually comes with a two step process: first, quantize the network; then, perform the hardware implementation. The second step can be performed advantageously by solving Multiple Constant Multiplication (MCM) problems to obtain an adder graph-based implementation. With our work, we propose to combine both step so we can guide the rounding operator of the PTQ with the implementation cost: we propose an Implementation-Aware PTQ.

Weights are robust to a slight change of value [15]. We use this property by rounding towards weight values less costly to implement. This could be done straightforwardly, considering weights independently. Our key contribution is that we propose a way of doing so that takes the parallelism into account. This permits us to round towards values that are costly independently but cheap within a larger adder graph: this way, we round to the nearest as often as it does not increase the overall implementation cost.



(a) Adder graph implementation of $x \times \{47, 12, 25, 15\}$.



(b) Adder graph implementation of $x \times \{48, 12, 24, 16\}$.

Fig. 2: Examples of adder graphs for the implementation of base weight multiplications and their approximation.

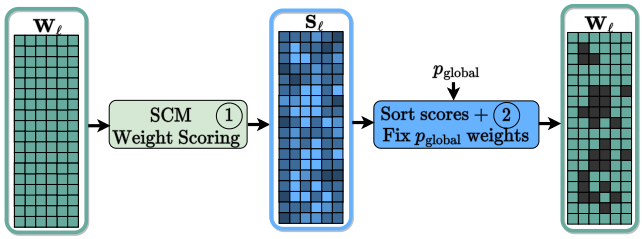


Fig. 3: Progressive weight fixation during network QAT.

Lets consider the weights $W = \{47.1, 12.2, 25.3, 15.1\}$. The adder graph to compute $\lfloor W \rceil x$, represented in Fig. 2a, requires four adders. However, by toggling a few rounding directions permits the reduction of the implementation cost to a single adder, as illustrated in Fig. 2b. The new rounded weights would be $\widehat{W} = \{48, 12, 24, 16\}$.

We defined a new PTQ approach that requires solving an optimization problem, MCM-Approx, and we proposed a solving method for it. Our solving approach relies on a mathematical model, defined using the Mixed-Integer Linear Programming (MILP) paradigm.

B. Quantization-Aware Training

When quantization can be incorporated during the training step, accuracy is more easily preserved. In our work, we integrate the notion of adder graphs within the QAT flow. The core idea is to fix the weights incrementally, based on a shift-and-add complexity score. Our goal is to be able to take into account already fixed weights when computing a score. This way, weights with the smallest implementation cost are fixed for the rest of the training. Then, the other weights are fine-tuned, and the process is repeated until the end of the training, when all weights have been fixed. The process is illustrated in Fig. 3. The score computation is based on the MCM problem, thus we proposed a solving approach based on an MILP model. At each fixing step, we solve our new MILP model, for each weight.

III. Results and Discussion

With two different approaches, we show ways of incorporating adder graph implementation knowledge within the quantization. This permits to obtain significant hardware cost reductions w. r. t. state-of-the-art QAT and PTQ. In the case of PTQ, using our MCM-Approx method, we reduce the LUT consumption by more than 10%. For QAT, more room for improvement is possible due to flexibility in the training process. Hence, we reduce the hardware utilization by around 25%. In both cases, using adder graphs instead of a more common implementation using simply $*$ operators and letting the synthesis tool do the hardware optimization lead to significant gains. Indeed, this simple difference leads to a cost reduction by almost 50% and our gains build on it.

References

[1] R. Desislavov, F. Martínez-Plumed, and J. Hernández-Orallo, "Trends in AI inference energy consumption: Beyond the performance-vs-parameter laws of deep learning," *Sustainable Computing: Informatics and Systems*, vol. 38, p. 100857, Apr. 2023.

[2] M. Nagel, M. Fournarakis, R. A. Amjad, Y. Bondarenko, M. van Baalen, and T. Blankevoort, "A White Paper on Neural Network Quantization," 2021, arXiv:2106.08295.

[3] H. Wang, C. Qin, Y. Bai, and Y. Fu, "Why is the State of Neural Network Pruning so Confusing? On the Fairness, Comparison Setup, and Trainability in Network Pruning," 2023, arXiv:2301.05219.

[4] S. Han and W. J. Dally, "Bandwidth-efficient deep learning," in *Proceedings of the 55th Annual Design Automation Conference*, Jun. 2018, pp. 1–6.

[5] W. J. Dally, C. T. Gray, J. Poulton, B. Khailany, J. Wilson, and L. Dennison, "Hardware-Enabled Artificial Intelligence," in *2018 IEEE Symposium on VLSI Circuits*. IEEE, Jun. 2018, pp. 3–6.

[6] P. Kluska and M. Zięba, *Post-training Quantization Methods for Deep Learning Models*. Springer International Publishing, 2020, pp. 467–479.

[7] A. Gholami, S. Kim, Z. Dong, Z. Yao, M. W. Mahoney, and K. Keutzer, "A Survey of Quantization Methods for Efficient Neural Network Inference," 2021, arXiv:2103.13630.

[8] Y. Li, M. Shen, J. Ma, Y. Ren, M. Zhao, Q. Zhang, R. Gong, F. Yu, and J. Yan, "MQBench: Towards Reproducible and Deployable Model Quantization Benchmark," in *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021.

[9] S. Zhou, Y. Wu, Z. Ni, X. Zhou, H. Wen, and Y. Zou, "DoReFa-Net: Training Low Bitwidth Convolutional Neural Networks with Low Bitwidth Gradients," 2016, arXiv:1606.06160.

[10] M. Nagel, R. A. Amjad, M. van Baalen, C. Louizos, and T. Blankevoort, "Up or Down? Adaptive Rounding for Post-Training Quantization," in *International Conference on Machine Learning*. PMLR, 2020, pp. 7197–7206.

[11] J. Li, M. Yanagisawa, and Y. Shi, "An Area-Power-Efficient Multiplier-less Processing Element Design for CNN Accelerators," in *2023 IEEE 15th International Conference on ASIC (ASICON)*. IEEE, Oct. 2023, pp. 1–4.

[12] R. Bernstein, "Multiplication by integer constants," *Software: Practice and Experience*, vol. 16, no. 7, pp. 641–652, Jul. 1986.

[13] Y. Voronenko and M. Püschel, "Multiplierless multiple constant multiplication," *ACM Transactions on Algorithms*, vol. 3, no. 2, p. 11, May 2007.

[14] R. Garcia and A. Volkova, "Toward the Multiple Constant Multiplication at Minimal Hardware Cost," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 70, no. 5, pp. 1976–1988, 2023.

[15] D. T. Nguyen, N. H. Hung, H. Kim, and H.-J. Lee, "An Approximate Memory Architecture for Energy Saving in Deep Learning Applications," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 67, no. 5, pp. 1588–1601, May 2020.

Development of a VHDL code generator for fixed point activation functions

Aurélien Delmotte
LIP6, Sorbonne Université
Paris, France
adelmotte@lip6.fr

Andrea Pinna
LIP6, Sorbonne Université
Paris, France
andrea.pinna@lip6.fr

Thibault Hilaire
LIP6, Sorbonne Université
Paris, France
thibault.hilaire@lip6.fr

***Abstract*—This paper presents an automated framework for generating FPGA-optimized fixed-point implementations of neural activation functions. Addressing precision-resource tradeoffs through numerical analysis and hardware-aware optimization, we demonstrate a working exponential approximation as foundation for an ongoing softmax implementation. The tool features error propagation tracking, visual dataflow representation, and architecture-specific pipelining to explore accuracy-resource compromises in embedded ML applications.**

***Index Terms*—FPGA, Machine Learning, Fixed-Point arithmetic, activation functions, automated code generation**

I. Introduction

Field-Programmable Gate Arrays (FPGAs) have become pivotal for energy-efficient machine learning inference in edge devices, where their reconfigurability enables precision-optimized implementations. Activation functions like softmax present unique implementation challenges due to their nonlinear characteristics and numerical sensitivity. While floating-point arithmetic dominates research prototypes, fixed-point representations offer superior resource efficiency for FPGA deployments since it can be fine-tuned. However, manual optimization of bit-widths and operator pipelines remains error-prone and time-consuming, particularly when balancing numerical accuracy with timing constraints.

Existing tools like FloPoCo [1] already tackle a lot of these issues [2]. What motivates the development for an alternative however is missing support for key activation functions such as Softmax, as well as tackling resource optimization and more relaxed error bounds. Still a work in progress, the project offers a python-based alternative producing readable VHDL code and a purpose-built ressource-counscious fixed-point exponential approximation implementation.

A. Current Implementation Status

The developed framework, currently provides:

- Elementary operators with error modeling;
- Tabulation-based function approximation;
- Delay models for Xilinx Zynq-7000 FPGAs;
- Automated pipeline insertion guided by delay models;
- Visual dataflow representation via Graphviz.

While full softmax implementation remains ongoing, we demonstrate our methodology through a complete exponential approximation algorithm.

B. Fixed Point Representation

We denote fixed-point formats as $Q_{m.l}$ where m is the integer bits (including sign) and l the fractional bits. Negative values imply the fractional dot lies outside the number.

II. Methodology

The framework transforms mathematical functions into optimized FPGA implementations through a layered approach combining numerical analysis and hardware constraints. At its core, operator graphs constructed from parameterized arithmetic blocks (adders, multipliers) and approximation units (LUTs, piecewise polynomials) automatically propagate bit-width requirements while tracking error bounds via interval arithmetic. Device-specific timing models for Xilinx Zynq-7000 FPGAs drive pipeline insertion, balancing clock frequency targets with resource utilization.

A three-stage workflow first decomposes target functions into dataflow graphs, then allocates precision using error propagation rules, and finally generates synthesizable VHDL with architecture-aware register placement. The tool provides visual dataflow verification through a Graphviz dot file.

A. Precision

For the most part the precision target is a maximum absolute error below 1 ulp (unit in last place), meaning that for an aproximation $\widetilde{f(x)}$ of $f(x)$ in $Q_{m.l}$ the target error is:

$$\forall x, \quad |\widetilde{f(x)} - f(x)| < 2^{-l} \tag{1}$$

This is the same approach FloPoCo uses, although in the future different approaches will be considered as, for the purposes of machine learning using the maximum absolute error might be too conservative.

III. Operators

For now the implementation of all elementary operators is left to the synthesizer. This is currently a work in progress, as for some operators fast implementations have been devised (see [1], outperforming Vivado on some metrics).

Currently Implemented operators include:

- Addition (`ADD`);
- Multiplication (`MUL`);
- Tabulation (`LU` and `TABLE`);
- Slicing (`CUT`).

There also exists composite operators like EXP (exponential) and certain rounding operators, which automatically generate optimized operator trees using core operators. These may adapt their structure based on user-specified precision and implementation constraints.

A. Error Analysis

Currently we track maximum absolute errors using multiprecision interval arithmetic. Future enhancements plan to propagate full error distributions to allow for more relaxed error constraints.

B. Rounding

By default when rounding is implied, it is done through truncation, introducing a biased error. Other rounding methods are also made available, such as a Round Half Up which neutralizes truncation bias by adding $\frac{1}{2}$ ulp before truncation at the cost of an addition.

C. Delay estimation

Our latency heuristics address three key challenges: Architecture-specific implementation details, unpredictable routing/placement effects and vendor tool optimizations (black-box implementations). Current methodology focuses on logic path delay and includes safety margin for routing delays. All delay models are simplified approximations but sufficient for pipeline scheduling decisions.

IV. Pipelining

The (simplified) pipelining algorithm is as follows:

1) **Stage Construction:**
   - Start with input operators as first stage
   - For current stage, add downstream operations if:
     - All input dependencies are ready
     - Total estimated delay $\leq$ clock period
2) **Edge Handling:**
   - Add edges with ready inputs to new stage
   - Insert buffers to lagging inputs of each ready edge
3) **Iterate:** Repeat for new stages until no unprocessed operations remain

This method was inspired by [3] and does not guarantee optimal pipelining but does guarantee the pipeline is synchronized.

V. Exponential approximation

Our fixed-point exponential implementation combines domain decomposition with rigorous error propagation analysis, building on [4]–[6].

A. Approximation Method

- **Input decomposition**: For x in $Q_{m.l}$ format:

$$x = \sum_{i=0}^{k} x_i, \quad x_i = x[i{\cdot}n : (i{+}1){\cdot}n] \tag{2}$$

TABLE I
Comparaison of different exponential implementations

Implementation[a]	Latency	Frequency (Target)	LUTs	DSPs
Ours (16bit)	4 cycles	217MHz (200MHz)	115	1
FloPoCo[b] (16bit)	4 cycles	140MHz (200MHz)	496	0
Ours (32bit)	7 cycles	147MHz (200MHz)	1667	8
FloPoCo[b] (32bit)	4 cycles	100MHz (200MHz)	3664	1

[a]Done targeting a Zynq7000 and includes I/O Registers
[b]With piece-wise polynomial approximation of degree 2

- **Hybrid computation**:

$$\exp(x) \approx \prod_{i=0}^{k} \widetilde{\exp}(x_i) \tag{3}$$

  where $\widetilde{\exp}(x_i)$ uses (depending on precision needed) either direct tabulation for MSB segments or degree-1 Taylor expansion $(1 + x_i)$ for LSBs

B. Error Requirements Propagation

We can compute a strict bound on the number of accurate bits p (Integer satifying $\max(|\text{error}|) < 2^{m-p}$) needed for a and b for a given precision requirement p_{ab} on their product:

$$p = p_{ab} - m_{ab} + m_a + m_b + 1 \tag{4}$$

C. Optimization

Exact multiplication leads to rapidly expanding bit-widths in computation trees. Our optimization strategy employs:

Iterative width adjustment:

- Start with arbitrary large bit-width initialization;
- Widen widths until either local p or global p_{out} is met;
- Subsequently tighten widths while maintaining p_{out}.

While this approach produces practical results, we are developing more rigorous optimization methods, as currently solution quality greatly depends on iteration order and starting points.

D. Tests and comparisons

Using FloPoCo generated testbenches we verified ulp accuracy. We generally obtain higher throughputs and lower LUT utilization at the cost of more DSP blocks (see Table I).

VI. Planned development

Key ongoing and planned developments:

- Complete softmax implementation with division;
- Core operator optimization;
- Space-aware optimisation;
- Other activation Functions;
- Generic approximation methods;
- Flexible error requirements;
- Rigorous bit-width optimization methods.

Acknowledgment

This work has been funded funded through the PEPR IA AdapTING project as part of an M2 Internship at Sorbonne Université and LIP6.

REFERENCES

[1] F. de Dinechin and M. Kumm, *Application-Specific Arithmetic*. Springer, 2024. [Online]. Available: https://link.springer.com/book/10.1007/978-3-031-42808-1

[2] T. Hubrecht, O. Desrentes, and F. de Dinechin, "Activations in Low Precision with High Accuracy," Nov. 2024, working paper or preprint. [Online]. Available: https://inria.hal.science/hal-04776745

[3] M. Istoan and F. de Dinechin, "Automating the pipeline of arithmetic datapaths," Mar. 2017.

[4] M. Chandra, "On the implementation of fixed-point exponential function for machine learning and signal- processing accelerators," *IEEE Design & Test*, vol. 39, no. 4, pp. 64–70, 2022.

[5] J. Partzsch, S. Höppner, M. Eberlein, R. Schüffny, C. Mayr, and D. R. Lester, "A fixed point exponential function accelerator for a neuromorphic many-core system," in *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, pp. 28–31.

[6] J.-M. Muller, *Elementary Functions*. Boston, MA, USA: Birkhäuser. [Online]. Available: https://link.springer.com/book/10.1007/978-1-4899-7983-4

Two Case Studies in Low-Bits Quantization-Aware Training

Romain Bouarah
INSA Lyon - Inria, CITI (UR3720)
69621 Villeurbanne, France
romain.bouarah@insa-lyon.fr

Bastien Barbe
INSA Lyon - Inria, CITI (UR3720)
69621 Villeurbanne, France
bastien.barbe@insa-lyon.fr

***Abstract*—This poster studies two families of number formats and their corresponding hardware accelerators for neural network inference. The first is the logarithmic number system (LNS); the second is non-uniform integer quantization, where levels are selected to better match weight and activation distributions. To preserve accuracy, we use quantization-aware training (QAT). We then present two custom accelerators: the LNS Neuron and the Reconfigurable Single Constant Multiplier (RSCM) circuit.**

I. Background

A. Quantization Aware Training

Quantization-Aware Training (QAT) is a technique in which both the forward and backward passes are performed on the quantized model. After each gradient update, the model parameters are quantized, allowing the network to adapt to quantization effects during training.

B. Logarithmic Number System

Logarithmic number system (LNS) is a number format used to represent real numbers. More formally, for a base $b > 1$ and integers $m, \ell \in \mathbb{Z}$ s.t. $m > \ell$, we define LNS as:

$$\text{LNS}(b, m, \ell) = \left\{(-1)^s \cdot b^{-L_X} | s \in \{0, 1\}, L_X \in \text{ufix}(m, \ell)\right\} \tag{1}$$

where $\text{ufix}(m, \ell)$ denotes the set of unsigned fixed-point numbers with msb at position m and lsb at position l. Fig. 1 shows that LNS values align well with normally distributed weights, making them a suitable choice for neural networks.

This work was supported by the PEPR IA HOLIGRAIL project of the Agence Nationale de la Recherche, ANR-23-PEIA-0010.

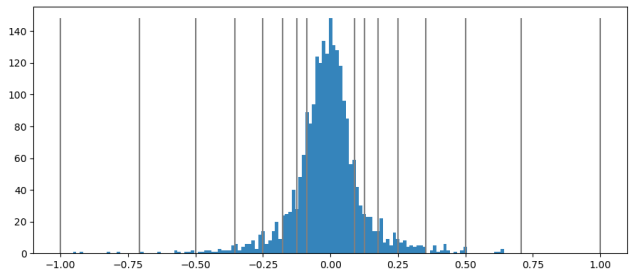


Fig. 1: Weight histogram of ResNet-56's first convolutional layer with $\text{LNS}(2, 1, -1)$ values overlaid.

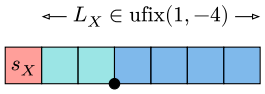


Fig. 2: Hardware representation of $X \in \text{LNS}(b, 1, -4)$.

LNS is implemented in hardware by storing only the sign bit and the unsigned fixed-point exponent. Fig. 2 shows the hardware representation of a LNS value.

The main advantage of LNS appears when computing products. For example, if $b^{-L_X}, b^{-L_Y} \in \text{LNS}(b, m, \ell)$, then

$$b^{-L_X} \times b^{-L_Y} = b^{-(L_X + L_Y)} \tag{2}$$

This simplicity in multiplication, however, comes at the cost of a more complex addition.

In the LNS Neuron [1] depicted in Figure 3 this problem is avoided by converting LNS to linear before accumulation. A single lookup table performs both activation and conversion back to LNS.

We extend this work by selecting number format parameters optimized for FPGA targets. We show that m can be fixed to 1 via an equivalence relation, and that certain logarithmic bases allow the b^{-x} table to emulate zero. This poster also presents experiments with QAT in this context.

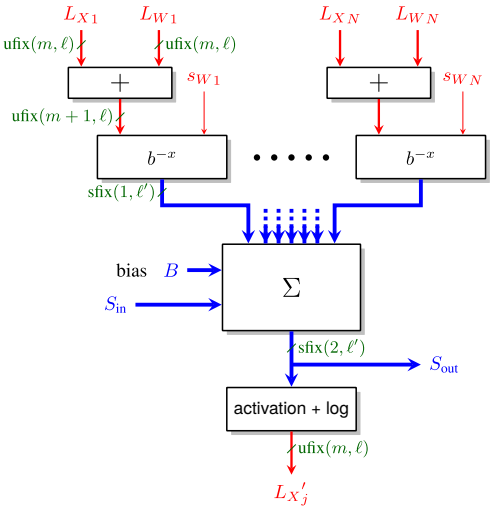


Fig. 3: LNS Neuron (red: LNS, blue: linear)

C. Reconfigurable Single Constant Multiplier

A Single Constant Multiplication (SCM) operator is typically a combination of constant shifts and additions that is simpler than a generic multiplier [2, ch. 12]. For example, in Fig. 4a, the multiplication of a binary integer input X by the constant 17 can be expressed with only a shift and an adder: $17X = 16X + X = (X \ll 4) + X$. The shift operation $X \ll 4$ consists of wires and therefore comes for free.

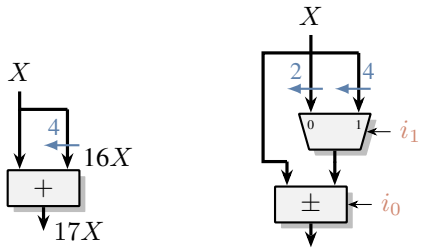


Fig. 4: (a) SCM for 17. (b) RSCM for $T = \{-15, -3, 5, 17\}$.

In this poster, we address its reconfigurable variant: the *reconfigurable* single constant multiplier (RSCM) [4], also known as time-multiplexed constant multiplier [3, 5].

As illustrated in Fig. 4b, an RSCM multiplies its input by a constant T_i, chosen at runtime from a set of possible constants $T = \{T_i | i \in [\![0, ...n]\!]\}$ known at design time. Here the index i is another input to the multiplier, used to select the constant to multiply by. It is decomposed into its bits (i_1 and i_0 in Fig. 4b), where each of them can be used either as the selection bit of a two-input multiplexer, or as the $\pm$ bit to an adder/subtractor.

D. RSCM Neuron

Fig. 5 shows a RSCM for a set of constants ranging between -20 and 19. If a network is trained by QAT to have all its weights in this set, then these weights may be encoded in 4 bits, but still provide a dynamic range that is almost two bits more.

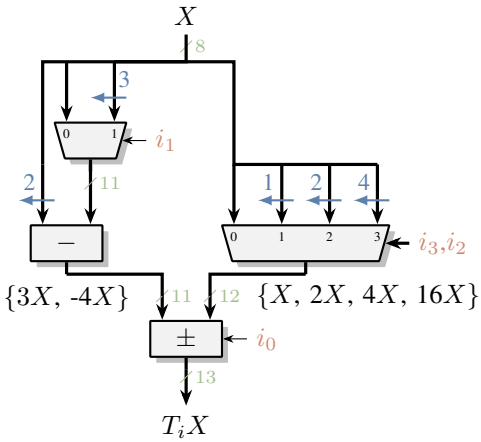
Synthesis results obtained with Vivado 2024.1 for Kintex7, part xc7k70tfbv484-3 are given in Table I. When compared to generic 4x8-bit and 6x8-bit multipliers, and its naive implementation, the Fig. 5 offers a favorable trade-off.

TABLE I: Synthesis of RSCM on FPGA

	latency	area
RSCM from Fig. 5	2.611ns	31 LUT
mult 4x8	3.133ns	29 LUT
mult 6x8	3.182ns	48 LUT
naive RSCM	3.451ns	54 LUT

II. QAT Results

Table II demonstrates that LNS and RSCM have potential to be used to quantize weights in a neural network without loss of accuracy compared to FP32. To obtain this table, the methodology was to download a state-of-the-art pre-trained ResNet56 on CIFAR100 from https://github.com/osmr/pytorchcv, then apply QAT with LNS and RSCM using PyTorch.



$i_3 i_2 i_1 i_0$	T_i
0000	4
0001	2
0010	-3
0011	-5
0100	5
0101	1
0110	-2
0111	-6
1000	7
1001	-1
1010	0
1011	-8
1100	19
1101	-13
1110	12
1111	-20

$T_i \in$ {-20, -13, -8, -6, -5, -3, -2, -1, 0, 1, 2, 4, 5, 7, 12, 19}

Fig. 5: Example 2-adder RSCM for an 8-bit input, with a target set with 6-bit dynamic encoded in 4 bits.

TABLE II: Average accuracy of ResNet-56 on CIFAR-100 with different quantizers. Average best accuracy, 10 runs.

Weight quantizer	Top-1 Accuracy (%)	Top-5 Accuracy (%)
FP32 baseline	75.09	93.05
INT4	74.87 ± 0.07	93.53 ± 0.07
RSCM from Fig. 5	75.19 ± 0.03	93.42 ± 0.04
LNS$(3.46, 1, -4)$	75.16	93.82

References

[1] M. Christ, F. de Dinechin, and F. Pétrot. "Low-Precision Logarithmic Arithmetic for Neural Network Accelerators". In: *33rd IEEE International Conference on Application-specific Systems, Architectures and Processors (ASAP)*. 2022.

[2] F. de Dinechin and M. Kumm. *Application-Specific Arithmetic*. Springer, 2024.

[3] C. Eleftheriadis and G. Karakonstantis. "Optimal Adder-Multiplexer Co-Optimization for Time-Multiplexed Multiplierless Architectures". In: *IEEE Transactions on Circuits and Systems I: Regular Papers* (2023).

[4] K. Möller, M. Kumm, M. Kleinlein, and P. Zipf. "Reconfigurable constant multiplication for FPGAs". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 36.6 (2016), pp. 927–937.

[5] P. Tummeltshammer, J. C. Hoe, and M. Puschel. "Time-multiplexed multiple-constant multiplication". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 26.9 (2007), pp. 1551–1563.

Fast Vulnerability Assessment For Selective Fault-Tolerance

Pegdwende Romaric Nikiema, Marcello Traiola, Angeliki Kritikakou
Univ Rennes, CNRS, Inria, IRISA - UMR 6074, F-35000 and IUF
{firstname.lastname@inria.fr}

***Abstract*—Modern technology trends such as high level complexity of integration, lower operating voltages, and higher switching frequencies exacerbate the failure rates in microprocessors. Devices running in constrained environments face serious threats caused by radiation, which impact the transistor, causing Single Event Upsets (`SEUs`), which, in turn, cause failures. The faults can be benign or not, and in critical safety systems, where failures must be avoided because of the catastrophic consequences, it becomes important to perform a comprehensive vulnerability analysis of the system. Existing analysis faces complexity and increased time as they often time go through rounds of lengthy fault injection. In this work, we propose a framework that uses machine learning models to reduce the required amount and time of fault injection and quickly provide insight into what inputs or instructions are the most vulnerable. Designers can benefit from this framework as a tool for fast assessment and fast iterative design changes to increase reliability.**

***Index Terms*—Reliability, Real-time, Transient faults, Fault-tolerance, RISC-V, Machine Learning**

I. Introduction

In safety-critical systems, a reliable execution is required to prevent catastrophic problems. With the computing demands rising across systems, performant processors are becoming common. The drawback of these processors, which use smaller transistor nodes and have very high switching frequencies, is that the transistors become more susceptible to SEU, which causes them to flip their state. These bit-flips can be masked, aka no error is found at the end of the execution; cause symptoms on the systems that can be simply detected and or corrected by symptom-based methods –checking illegal memory accesses, or be silent and cause an erroneous result at the end of the execution –Silent Data Corruption(`SDC`).

In order to offer a reliable system, multiple protection methods exist and are based on redundancy, where the modules or parts of the software are replicated. A complete hardware duplication can be costly in many situations, and other methods can be used to achieve some level of resilience. To apply partial protection, one needs to analyze the systems' vulnerability to detect the most vulnerable parts. Studies in [1], [2] analyze the vulnerability of the SDC-causing instructions in order to apply some selective protection.

Most vulnerability analyses use lengthy fault injection campaigns for the model training and primarily target SDC. To alleviate injection time, studies in [3], [4] apply software-level fault injection with some statistical analysis to selectively inject in some parts of the application. Although they achieve a significant level of injection time reduction, we note that the injection is at the software level, which doesn't account for the microarchitecture of the system. Additionally, most of the studies do not consider input diversity, which is found in [5], [6] to be insufficient as the SDC coverage becomes low when the inputs used are different from the ones from the testing. The study in [7] offered an overview of compiler optimizations on the reliability of the system, and the results show that an actual vulnerability analysis is mandatory to assure safe execution for each use case.

In this work, we propose machine learning methods to accelerate the vulnerability assessment, with a focus on functional and timing errors. Our method allows us to use a fraction of the faults injection to train and predict the outcome from an application profile alone. This application profiling is as easy as running the application once to collect the data and then use it to predict various outcomes. The model also allows us to identify the input sets and or instructions responsible for increased vulnerability. This opens the door to selective protection either on the software level using LLVM to duplicate vulnerable instructions, as shown in the works cited above, or hardware level duplication.

II. Proposed vulnerability analysis

To use machine learning models to predict the vulnerability and avoid lengthy fault injection campaigns, we characterize the application using four (4) different profiling methods. `P1`, is based on a coarse-grained version of the instruction percentage on the entire application execution cycles. These features describe the memory accesses, the arithmetic operations, and branches. `P2`, we consider the same features but in a more fine-grained way where we have individual instructions like `LD, ADD....` `P3` and `P4` are based on the sequence of instructions of the application. For P3, we generate sequence splits with instructions changing the control flow. For example, `ADD_LD_ST_BGE_ADD_...` will be split into `ADD_LD_ST` and `BGE_ADD_....` In P4, we automatically explore a set of instructions common to several application inputs: `LD_ADD`, `LUI_ADD`... This set of instructions is generated using n-gram, where n represents the number of consecutive instructions. Those four profiles allow us to generate the input points for the models.

In the optic of choosing a best-suited model, we use tree-based and ensemble regressors: Random Forest (`RF`), XGBoost (`XGB`), Support Vector Machine (`SVM`) and Classification And

TABLE I
EXPLORATION PARAMETERS

Parameter	Variants
Benchmarks	Bitonic, Binary Search, Bubble Sort, Count Negative, Factorial, Insert Sort, Matrix Multiplication, Quick Sort
ML. model	RF, XGB, SVM, CART
Optim. flags	-O0, -O3
App. profile	P1, P2, P3, P4
TT split	10%, 20%

Regression Trees (CART). We have regressors dedicated to the different types of errors we classify in the applications: Execution Cycles Mismatch (ECM), Application Output Mismatch (AOM), HANG, and CRASH. We combine benchmarks, profiling, and learning models, along with the value of train/test split, which defines the number of inputs from the fault injection result used for train and test datasets. These combinations (in Table I) are used for a Design Space Exploration (DSE) to find the best configurations. The reference data are obtained from fault injection consisting of 8 kernel benchmarks where we randomly generated the input sets and compiled them with several compiler optimizations such as `-O1, -O3`.

III. EXPERIMENTAL SETUP AND RESULTS

The reference data consists of 385 fault injections on each of the 650 inputs per benchmark. This makes up to 2′002′000 in microarchitectural fault injections. We then applied several optimization levels and different inputs to achieve higher confidence. The DUT is an RV32I RISC-V core that consists of 5 pipeline stages [8]. The table I shows the combination used for the DSE. Early experiment results are shown below.

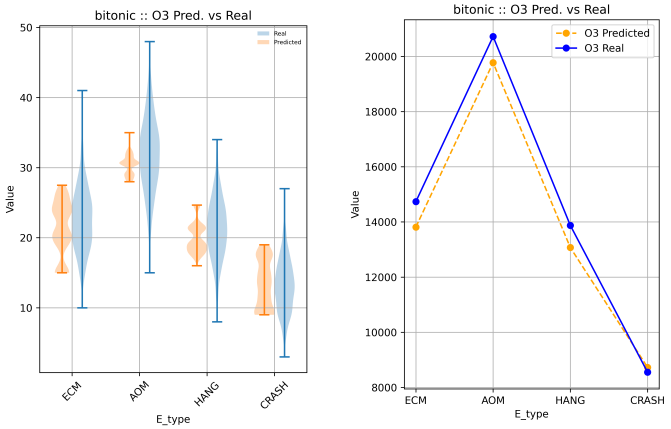


Fig. 1. Prediction result Bitonic -03

Figure 1 shows the variation of the vulnerability metrics from the application execution. The real values (in blue) are recorded with the fault injection and serve as a reference, and the predicted values (orange) are the values that the given model predicts.

We use the Kolmogorov-Smirnov `statistic`, which gives the highest absolute distance between the real and predicted distribution, to rank the configurations leading to a distribution (orange curve in Figure 1) that resembles the reference values (blue curve in Figure 1) best. The Table II shows the best configurations we recorded. With only a fraction of the inputs used for the training, the models paired with the profiling methods provide information about the application's vulnerability, as seen in Figure 1. This effectively provides insight into the system's reliability while taking a significantly shorter time, `10%` or `20%`. For reference, on average, it takes $\approx$ `16 hours` per benchmark for a complete fault injection with the aforementioned injection details.

TABLE II
BEST CONFIG RESULT FOR ECM REGRESSOR WITH -O3
C(CART), X(XGB), R(RF), S(SVM), MODEL X% KS_STATISTIC

Benchmark	P1	P2	P3	P4
Bitonic	X 20% .18	X 20% .18	X 20% .18	X 20% .18
Bsearch	C 20% .25	C 20% .25	R 10% .23	C 20% .25
Bsort	C 20% .07	C 10% .06	X 10% .08	C 20% .08
Cntnegative	X 20% .12	C 20% .08	X 20% .09	C 20% .10
Fac	X 20% .28	X 20% .28	X 20% .28	X 20% .28
Insertsort	C 10% .13	C 10% .11	C 10% .14	X 10% .11
Matmult	C 20% .07	C 20% .05	C 20% .04	C 20% .06
Qsort	C 20% .07	C 20% .05	C 10% .08	C 10% .08

REFERENCES

[1] L. Liu, L. Ci, W. Liu, and H. Y. and, "Identifying sdc-causing instructions based on random forests algorithm," *KSII Transactions on Internet and Information Systems*, vol. 13, no. 3, pp. 1566–1582, March 2019.

[2] J. Gu, W. Zheng, Y. Zhuang, and Q. Zhang, "Vulnerability analysis of instructions for sdc-causing error detection," *IEEE Access*, vol. 7, pp. 168 885–168 898, 2019.

[3] J. Li and Q. Tan, "Smartinjector: Exploiting intelligent fault injection for sdc rate analysis," in *2013 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFTS)*, 2013, pp. 236–242.

[4] K. Pattabiraman, N. M. Nakka, Z. T. Kalbarczyk, and R. K. Iyer, "Symplfied: Symbolic program-level fault injection and error detection framework," *IEEE Transactions on Computers*, vol. 62, no. 11, pp. 2292–2307, 2013.

[5] Y. Huang, S. Guo, S. Di, G. Li, and F. Cappello, "Mitigating silent data corruptions in hpc applications across multiple program inputs," in *SC22: International Conference for High Performance Computing, Networking, Storage and Analysis*, 2022, pp. 1–14.

[6] M. H. Rahman, S. Di, S. Guo, X. Lu, G. Li, and F. Cappello, "Druto: Upper-bounding silent data corruption vulnerability in gpu applications," in *2024 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, 2024, pp. 582–594.

[7] P. R. Nikiema, M. Traiola, and A. Kritikakou, "Special session: Impact of compiler optimizations on the reliability of a risc-v-based core," in *2024 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)*, 2024, pp. 1–6.

[8] S. Rokicki, D. Pala, J. Paturel, and O. Sentieys, "What You Simulate Is What You Synthesize: Designing a Processor Core from C++ Specifications," in *ICCAD 2019 - 38th IEEE/ACM International Conference on Computer-Aided Design*. Westminster, CO, United States: IEEE, Nov. 2019, pp. 1–8. [Online]. Available: https://hal.archives-ouvertes.fr/hal-02303453

Accelerating Kolmogorov-Arnold Networks on Systolic Arrays

Sohaib Errabii, Olivier Sentieys, Marcello Traiola
University of Rennes, Inria, Rennes, France

I. Introduction

Kolmogorov-Arnold Networks (KANs) [1] have attracted much attention for their promise of better parameter efficiency and interpretability than Multi-Layer Perceptrons (MLPs). Their key innovation lays in the use of learnable non-linear activation functions, instead of learnable linear weights and fixed non-linear activation functions. Such activation functions are parametrized as splines (cf. Fig. 1).

From the computational perspective, evaluating spline functions poses a significant challenge. In particular, their evaluation in turn relies on the evaluation of Basis functions (B-splines), which is not efficiently parallelizable on GPUs due to their recursive definition.

Systolic array (SA) based architectures have shown great promise as Deep Neural Network (DNN) accelerators thanks to their energy efficiency and low latency. Thus, in this work, we explore the use of systolic array architecture to accelerate the KAN inference. We first show that, while SAs can be used to execute part of the KAN inference, their utilization is limited to **20-50**% depending on typical values of the KAN hyperparameters. Hence, by leveraging interesting regularity and sparsity properties of B-splines, we propose a novel SA-based accelerator to enable efficient execution of KANs. We include an efficient LUT-based implementation of B-spline evaluation, which relies on the B- spline properties of being symmetric and translation-invariant on a uniform grid. We also leverage the local support of B-spline functions, leading to a contiguous *N:M* sparsity pattern, which significantly improves PE utilization and the array's latency.

The RTL synthesis with yosys on the SkyWater SKY130 PDK shows that our implementation achieves **6**× the throughput with **100%** vs **50%** PE utilisation for the tradeoff of 27% increase in area.

II. Kolmogorov-Arnold Networks

The KAN layer replaces the scalar multiplication by the weights at each connection with a learnable spline evaluation. As illustrated in 1, these learnable splines are parameterized using B- spline functions which form a basis for the spline function space. The KAN layer can thus be viewed as a two layer network where we first the activate the inputs using all the basis functions and then perform the usual MLP linear combination with the coefficients.

As such, to map the KAN layer on a systolic array-based accelerator, it is only required to implement the additional B-spline unit as shown in Fig. 5.

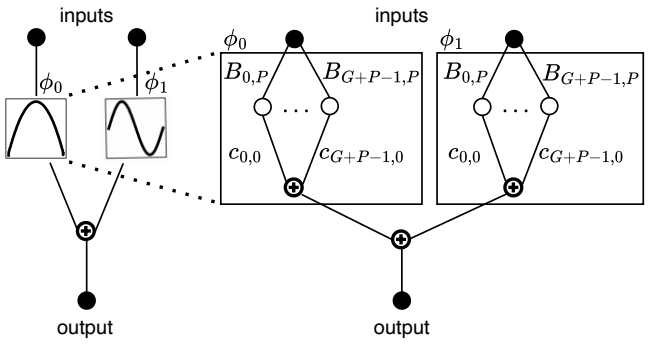


Fig. 1. A KAN layer with learnable spline activations instead of weights on the connections. The splines are parameterized in a function basis with the coefficients as the model's learnable parameters.

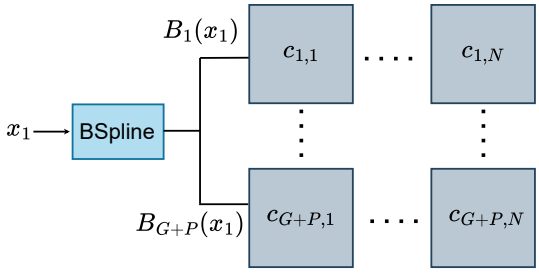


Fig. 2. A WS systolic array capable of executing the KAN layer.

III. LUT Implementation of B-spline Unit

Direct implementation of B-spline functions (piecewise polynomials) is quite expensive and would required several floating-point multipliers. The basis is quite simple on a uniform grid (cf. Fig. 3.) It would only require the tabulation of half a single B-spline to infer the rest through translation and symmetry.

The proposed implementation is illustrated in Fig. 4. This unit directly outputs the $P+1$ non-zero B-spline activations, $B_{k-P}(x), \ldots, B_k(x)$ for an input $x \in [t_k, t_{k+1}]$.

It is also important to note from the Cox–de Boor formula [2] that $B_{[0,\Delta,\ldots]}(x\Delta) = B_{[0,\hat{\Delta},\ldots]}(x\hat{\Delta})\ \forall x$. Therefore, for tabulation, choosing sample points proportional to the grid step, lead to unique values that can be store in a read-only memory, increasing the efficiency.

The quantization levels $q = 2^n$ is chosen as a power of two, due to symmetry we only need to store $2^{n-1}(P+1)$ values tabulating the B-spline in local support that spans $P+1$ intervals.

As the grid is uniform we perform a uniform integer quantization between mapping $[t_0, t_{G+2*P}]$, this enables us to directly use n lower bits of the quantized inputs (from $log_2(q)$), as an address fo the LUT. The figure shows the extra logic required to handle symmetric using index inversion and reverse packing.

The higher bits of the quantized inputs on the other hand, encodes k such as $x \in [t_k, t_{k+1}]$. This enable us to select the right coefficients of the non zero B-splines among the $G+P$ coefficients.

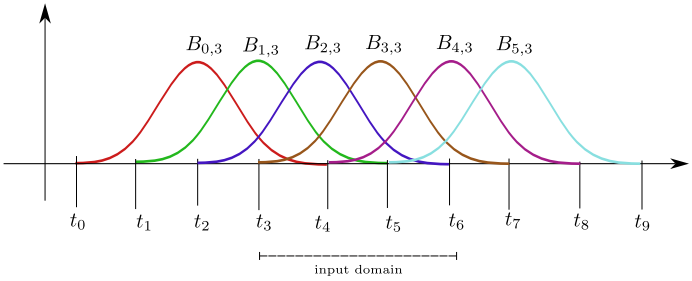


Fig. 3. B-spline functions for a grid size $G = 3$ and $P = 3$

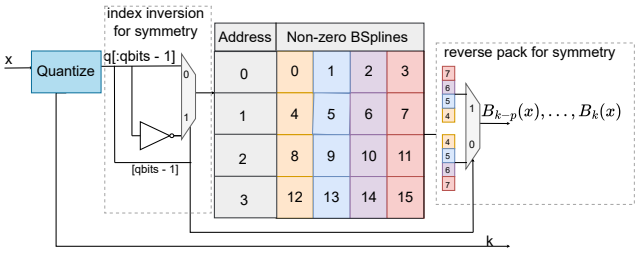


Fig. 4. Efficient LUT implementation of B-spline functions leveraging their symmetry and translation equivalence on a uniform grid.

IV. Sparsity Aware PE

The processing element of the systolic array must take as input both the non zero $P+1$ activations and the index k indicating the interval. As illustrated in Fig. 5, we use a WS dataflow to preload the $G+P$ coefficients in each PE. And based on the index k we select the corresponding contiguous $P+1$ coefficients, the bounds are handled by ensuring that extra activations (in the boundaries of the grid, cf. Fig 3) are zero. Therefore, making the retrieved extra coefficients irrelevant. Finally, the mac unit, a $P+1$-SIMD, performs the linear combination.

V. Results and Conclusion

The RTL implementation was developed using amaranth [3], a hardware description language in python, enabling us to develop an RTL generator for the proposed architecture in order to explore the design space.

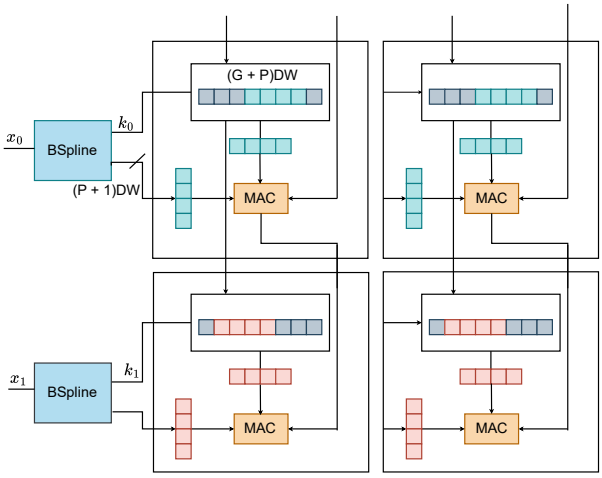


Fig. 5. A WS systolic array for KAN that avoids the sparsity introduced by the local support property of B-spline functions.

TABLE I
Area obtained with yosys RTL synthesis for the SkyWater SKY130 PDK.

	PE($i8\_32$)	PE($f8\_32$)	KANPE($i8\_32$)	KANPE($f8\_32$)
area (μm^2)	2994.1216	7895.072	16854.9152	34882.2048

Table I shows the synthesis results of the PE for the conventional WS with scalar MAC, and KANPE for the proposed sparsity aware PE. $i8\_32$ stands for 8bit integer multiplication and 32bit accumulation, f for floating-point.

Table II shows the synthesis results for the systolic array along with B-spline units. The conventional 16×16, $G = 5$, $P = 3$ systolic array can process two inputs every cycle at 50% utilisation. The tradeoff is an increase of 27% in area.

Finally, If we consider a specific workload such as the first 7×7 convolution of ResNet50 where the filters are using splines instead of weights. Then comparing arrays with the same number of PEs, 32×32, then the equivalent GEMM workload of $M = 50176, N = 64, K = 1911$ results in **6M** cycles for the conventional SA versus **500K** cycles for KAN adapted array.

References

[1] Z. Liu, Y. Wang, S. Vaidya, F. Ruehle, J. Halverson, M. Soljačić, T. Y. Hou, and M. Tegmark, "Kan: Kolmogorov-arnold networks," 2024. [Online]. Available: https://arxiv.org/abs/2404.19756

TABLE II
Area of the systolic array for with the B-spline unit for $i8\_16$, $G = 5$, $P = 3$.

	Baseline SA 16×16	KAN SA WS (12×12)
area (mm^2)	0.9324	1.1885

[2] C. de Boor, “Subroutine package for calculating with b-splines,” 1971. [Online]. Available: https://www.osti.gov/biblio/4740859
[3] “A modern hardware definition language and toolchain based on python,” https://github.com/amaranth-lang/amaranth.

Emerging methodologies for system-circuit-technology co-optimisation for near-memory computing

Benamara Hichem
Univ. Grenoble Alpes, CEA, List,
F-38000
Grenoble, France

Pronsato Rosario
INL, CNRS, Ecole Centrale de Lyon, UMR5270
F-69130 Ecully, France

Ciampolini Lorenzo
Univ. Grenoble Alpes, CEA, List,
F-38000
Grenoble, France

O'Connor Ian
INL, CNRS, Ecole Centrale de Lyon, UMR5270
F-69130 Ecully, France

Bricout Thaddee
Univ. Grenoble Alpes, CEA, List,
F-38000
Grenoble, France

Abstract— Digital circuit design is constrained by three main factors: power, performance and area. In the traditional Von Neumann computing paradigm, however, some additional power might arise from the way in which ever-increasing amounts of data are transmitted to the computing core. To address this challenge, memory-centric compute paradigms have been proposed like Near-Memory Computing (NMC) and In-Memory Computing (IMC). These approaches minimize data transfer by bringing computation closer to or inside memory units. These new paradigms, however, open up a vast design space, whether in terms of technology (emerging non-volatile technologies) or architecture, and new tools are being introduced for design space exploration. We propose a multi-level exploration environment that establishes a link between device-level memory technology and system-level architecture by combining a descriptive memory model and SystemC-TLM simulation with emphasis on co-optimization of technology, design and system (SDTCO).

Keywords—In/Near-Memory Computing, System/Design-Technology Co-Optimzation, SYSTEMC, TLM, Design Space Exploration, non-volatile memory

I. INTRODUCTION

With the growing demand for artificial intelligence applications, systems based on the von Neumann paradigm face a major bottleneck, related to the huge amount of data movement between the computing core and memories and the associated energy and latency costs[1]. The emergence of advanced integrated circuit manufacturing techniques (3D stacking, Through-Silicon Via, … etc.) [2] offers a unique opportunity to bring the computing unit closer to the memory, enabling Near-Memory Computing (NMC) to be implemented [3]. At the same time, the emergence of new non-volatile memories such as ReRAM, PCRAM or FeRAM, opens the way to In-Memory Computing (IMC) approaches [4], which promise a significant increase in processing speed and energy efficiency. Such systems can be conceptualized through a hierarchical stack of abstraction levels, starting from the storage device technology up to the system-level. Each level encapsulates specific functionalities and increases the dimension of the parameter space where optimization should be carried out. Thus, for a specific application, an NMC/IMC architecture based on a given technology may be more appropriate. Previous work on the evaluation of systems [5] [6] [7] shows that they can be characterized by:
The types of memory technologies considered; whether or not NMC/IMC is taken into account; the levels of abstraction considered; the type of evaluators (analytical or simulators); the target application; KPI to optimize.
In this work, we propose to carry out SDTCO (System-Design-Technology Co-optimization) during the early development cycle of an NMC/IMC computing system, developing a solution for exploring the design space linking the Design-Technology and System-Technology Co-Optimization (DTCO/STCO) that relies on both a memory descriptive model and system-level simulator. A Python-based cockpit will be used for iterative architectural and technological refinement to identify optimal solutions.

II. OUR APPROACH

The SDTCO methodology is based on a Design Space Exploration (DSE) process that is carried out at the technological and architectural levels. It consists of solving a Multi-Objective Optimization Problem (MOOP). The KPIs we are seeking to optimize correspond to these objectives (performance, energy consumption, area, etc.), which are very often in conflict. This leads to a front of non-dominated solutions, known as the Pareto front. A multi-objective optimization problem is solved in two steps:

- Solutions evaluation: for a given set of technological and architectural parameters, evaluation can be performed either using a real prototype, analytical tool or a simulation-based tool, at different accuracy, faithfulness and evaluation's speeds.
- Solutions' search: The optimal solutions (Pareto front) are determined using an optimization algorithm. There are two main categories of optimization algorithms: exact and heuristic, at different convergence speed and solution's guarantee.

Our proposed approach, as illustrated in figure 1, starts by defining the specifications, which take the form of an initial configuration phase. Following this setup phase, a hierarchical and descriptive modelling of the memory, based on a specific technology and configuration, is elaborated.
The memory can incorporate computation logic either intrinsically (IMC) or extrinsically (NMC). Finally, using a SystemC-based simulator developed in our lab, the Artificial Intelligence ACCelerator or AiAx, a virtual model of the system is created, including all NMC/IMC computation blocks. The exploration cockpit shall then orchestrate the flow of exploration and optimization of the design space from high-level AIAX results.

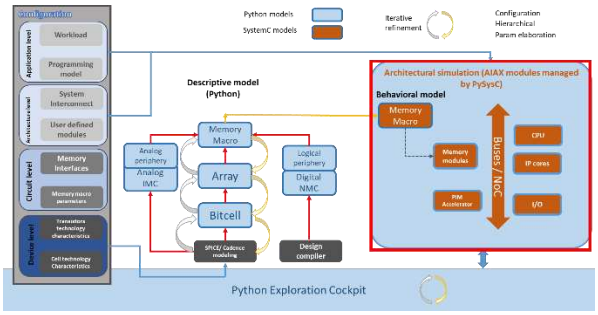


Figure 1Multi-level exploration plateform

III. Methodology and Results

At the current stage of the work, architectural evaluations have been carried out, including a preliminary exploration of the NMC row size. The NMC block, integrated into the storAIge chip developed in our laboratory [3], consists of an SRAM combined with compute logic at the periphery to enable local vector processing, called C-SRAM (figure 2).

A. Architecture case study: StorAIge circuit:

AIAX includes as Instruction-Set Simulator (ISS) the opensource RISCV-TLM [8] for the CPU, an interconnect bus, a multi-segment memory model and other custom peripherals and finally the NMC calculation block, as shown in Figure 2 [9]. The modeled "StorAIge" circuit is a low-power circuit incorporating next-generation storage elements ready for artificial intelligence (AI) in microcontrollers (MCUs).

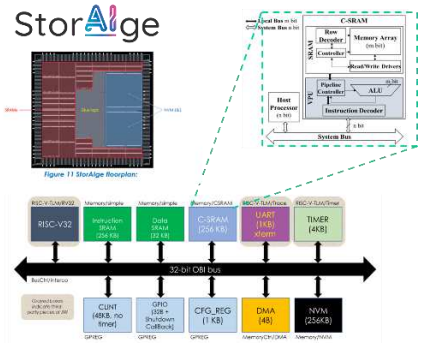


Figure 2 StorAIge circuit architecture simulated in AiAx

B. Convolution programme on the StorAige's AIAX model

A convolution operator written in .C and cross-compiled for the RISC-V [10] is executed by the AIAX model of the storAIge circuit to estimate the instruction count on various configurations of CSRAM (SRAM's row size).

C. Initial Architectural Exploration result

The current design space exploration of the CSRAM block lets the SRAM memory row size vary and allows analyzing its impact on instruction count across multiple convolution datasets. The results, illustrated in Figure 3, show a clear trend: increasing the memory line size leads to a reduction in the number of executed instructions, with a saturation effect at 128 bytes, and thus a reduction in data transfers from memory.

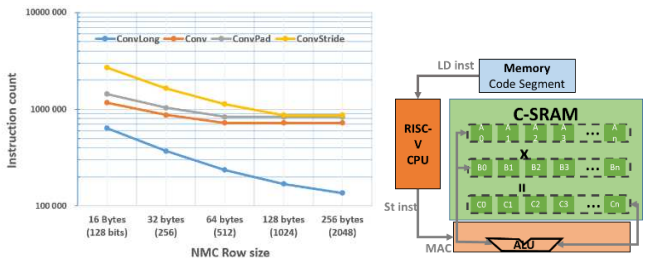


Figure 3 The number of instructions (log scale) resulting from the execution of the convolution application with different data sets on AIAX and different row size

Beyond 128 bytes, the available space is already sufficient to hold the data of the simple examples, making further increases in row size ineffective.

IV. Conclusion

In this work, we introduced our SDTCO methodology that relies on multi-level exploration plateformc. Our purpose is to enable the impact evaluation of low-level circuit performance on high level system requirements for NMC/IMC solutions at early stage of conception. At the current stage of the work, we are already able to carry out some basic, preliminar architectural exploration.

V. Acknowledgements

The circuit presented herein was developed in our laboratory as part of a contribution to (a) the European StorAIge project and has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 101007321 and (b) France 2030 PEPR Electronique CHOOSE project and has received funding from ANR under ANR-22-PEEL-0013.

References

[1] X. Zou, S. Xu, X. Chen, L. Yan, and Y. Han, "Breaking the von Neumann bottleneck: architecture-level processing-in-memory technology," *Sci. China Inf. Sci.*, vol. 64, no. 6, p. 160404, Apr. 2021,.

[2] "Near-memory computing: Past, present, and future," *Microprocess. Microsyst.*, vol. 71, p. 102868, Nov. 2019,.

[3] J.-P. Noel *et al.*, "A 35.6 TOPS/W/mm^2 3-Stage Pipelined Computational SRAM With Adjustable Form Factor for Highly Data-Centric Applications," *IEEE Solid-State Circuits Lett.*, vol. 3, pp. 286–289, 2020,

[4] K. C. Akyel *et al.*, "DRC2: Dynamically Reconfigurable Computing Circuit based on memory architecture," in *2016 IEEE International Conference on Rebooting Computing (ICRC)*, Oct. 2016, pp. 1–8.

[5] X. Dong, C. Xu, Y. Xie, and N. P. Jouppi, "NVSim: A Circuit-Level Performance, Energy, and Area Model for Emerging Nonvolatile Memory," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 31, no. 7, pp. 994–1007, Jul. 2012,

[6] L. Pentecost, A. Hankin, M. Donato, M. Hempstead, G.-Y. Wei, and D. Brooks, "NVMExplorer: A Framework for Cross-Stack Comparisons of Embedded Non-Volatile Memories," in *2022 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, Apr. 2022, pp. 938–956.

[7] Z. Zhu *et al.*, "MNSIM 2.0: A Behavior-Level Modeling Tool for Processing-In-Memory Architectures," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 42, no. 11, pp. 4112–4125, Nov. 2023,

[8] M. Montón, "A RISC-V SystemC-TLM simulator," Oct. 20, 2020, *arXiv*: arXiv:2010.10119. doi: 10.48550/arXiv.2010.10119.

[9] R. Gauchi *et al.*, "Memory Sizing of a Scalable SRAM In-Memory Computing Tile Based Architecture," in *2019 IFIP/IEEE 27th International Conference on Very Large Scale Integration (VLSI-SoC)*, Oct. 2019, pp. 166–171.

[10] T. Bricout, M. Kooli, Valea Emanuele, H.-P. Charles, M. Ramirez Corrales, and J.-P. Noel, "Flexible Integration of a Neural Network Library inTensorFlow Lite Framework for Efficient Programmable Near-Memory Computing Architectures."

Large SRAM Cache Architecture Design Challenges

Enzo Rafinesque, David Novo, and Pacal Nouet

LIRMM, Univ. Montpellier, CNRS, Montpellier, France

***Abstract*—The growing complexity of modern systems such as AI accelerators, GPUs and TPUs has driven an increasing demand for large on-chip SRAM memories. However, designing large SRAM blocks introduces significant challenges, such as signal propagation across interconnects. Reducing simulation time becomes critical, as full-transistor-level simulations of large memory arrays are computationally intensive. To address this, we adopt compact modeling and efficient initialization techniques to accelerate the simulation process.**

***Index Terms*—SRAM Macros, Simulation, Compact Modeling, Script**

I. Introduction

The increasing complexity of modern integrated circuits such as AI accelerators, GPUs, and TPUs has intensified the demand for large on-chip SRAM memories. These memories are crucial in order to achieve high bandwidth, low latency, and energy efficiency required by those applications. For instance, AI accelerators necessitate substantial SRAM capacity to store intermediate computations and model parameters, while GPUs rely on large SRAMs for efficient graphics rendering and parallel processing.

However, designing large SRAM macros presents significant challenges, particularly in simulation and verification. As SRAM array size grows, accurately modeling their behavior becomes increasingly complex, often resulting in prohibitively long simulation times. This issue is especially pronounced when full transistor-level accuracy is required, such as in leakage or timing analysis.

To mitigate these challenges, we adopt a compact modeling strategy that abstracts the fully detailed macros into simplified representations that retain essential characteristics. Our methodology is inspired by prior work [1], [2], where compact models have been used effectively to reduce simulation overhead without compromising accuracy.

Specifically, we define three levels of abstraction for SRAM array modeling. The first model employs a fully detailed array composed entirely of standard 6T bitcells, used when full electrical accuracy is required for example, in leakage power analysis. The second model reduces the array to a single row or column, effectively isolating the critical path for timing evaluation. The third model abstracts the memory array to a single representative cell, with the wordline/bitline network modeled using an equivalent RC circuit. This tiered modeling approach allows designers to select the appropriate level of abstraction depending on the simulation objective retaining full accuracy when needed while significantly accelerating simulations for functionality or timing analysis.

Additionally, to further improve efficiency, we implement a programmable initialization script that preloads initial conditions to selected nodes. Without this step, initializing the entire memory array would require simulating twice the sequences as during the first one, all 6T cells will not be fully stable.

II. Compact modeling

Simulation time is directly related to the number of memory cells simulated. One effective way to reduce the simulation is to work with different memory array models depending on the specific metrics being analyzed. We propose three different models, illustrated in Figure 1, each involving a different number of memory cells, as summarized in Table I:

- **Full Array (a):** This model includes the entire memory array, which in our case contains $128 \times 256 = 32{,}768$ cells. It serves as the reference model, with no reduction applied.
- **Reduced Array (b):** This model represents only the critical path to the observed cell, consisting of one row and one column. It contains $128 + 255 = 383$ cells (excluding the shared intersection).
- **Fully Reduced Array (c):** In this model, all the cells in the row and column except the observed one are replaced with an equivalent RC model.

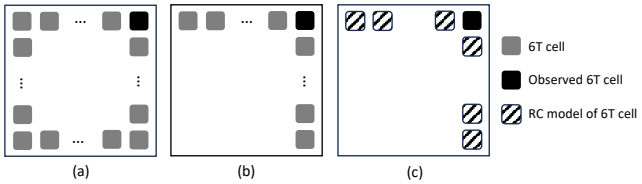


Fig. 1. SRAM array models: (a) full memory array, (b) reduced model, and (c) fully reduced model.

TABLE I
Number of 6T cells in each model

Model	6T cells
Full	32,768
Reduced	383
Fully reduced	1

Results of transient simulations of the different models are presented in Table II. These results are obtained using a common sequence: W1, W0, W1, R1, W0, R0, where W and R stand for Write and Read, respectively. We observe that the reduced model enables simulations that are 20 times

TABLE II
DELAY AND SIMULATION TIME OF DIFFERENT MODELS

	Full	Reduced	Fully reduced
Write0 delay	858.1ps	851.7ps	848.9ps
Write1 delay	858.3ps	848.8ps	848.9ps
Read0 delay	559.7ps	560.1ps	560.1ps
Read1 delay	572.2ps	573.5ps	573.3ps
Simulation time	10.41ks	591.4s	291.61s

faster, while maintaining high accuracy in read and write delay metrics. The fully reduced model also shows good accuracy and further reduces simulation time by a factor of two. However, using simple RC models instead of 6T cells may not provide sufficient accuracy when evaluating metrics other than delay.

III. INITIALIZATION SCRIPT

The 6T cells in SRAM macros are composed of cross-coupled inverters. If not properly initialized, these inverters may start at undefined voltages between ground and VDD, eventually settling to a logical `0` or `1` only after some activity. This initialization phase introduces unpredictable behavior, making it unreliable to measure key metrics during this period. Consequently, the test sequence often needs to be replicated multiple times to ensure that the cells reach a stable state before measurement. As an alternative, manually setting the initial conditions of all 6T cells in the memory array avoids this warm-up phase, eliminating the need to duplicate test sequences and significantly reducing simulation time.

A. Nodeset and Initial Conditions

An effective approach to initializing the design state is to provide assistance to the simulator, which in our case is Cadence Spectre. There are two alternatives that allow net initialization:

- Nodeset: Nodeset helps find the DC or initial transient solution. Nodeset specifies voltages to be tried on various nodes in the circuit for the first few iterations and then released The system continues to iterate from this solution to the final answer.
- Initial Condition (IC): The Initial Condition statement lets you specify values for the starting point of transient analysis. The idea behing ICs is to avoid DC convergence solution and to set a value after it, at transient start.

Both Nodeset and Initial conditions solutions will be considered for our script.

B. Script

Our simulated design can be represented as a hierarchical database, with a top-level instance containing sub-instances, which in turn may contain further sub-instances, and so on. Our script traverses all instances of the design recursively. When it encounters a cell named `6TCell`, it records the full hierarchical path (e.g., `I7/I3/I2/I8`). For each of these `6TCell` instances, the script generates two function calls (one for the nodeset and another one for initial condition case) to set initial values for the `Q` and `Qb` nodes. The final output looks like this:

```
ic I7/I3/I2/I8/Q = 1.0
ic I7/I3/I2/I8/Qb = 0.0
...
```

The script generates a file containing `nbCells` $\times$ 2 lines, which is read at the start of the simulation.

C. Results

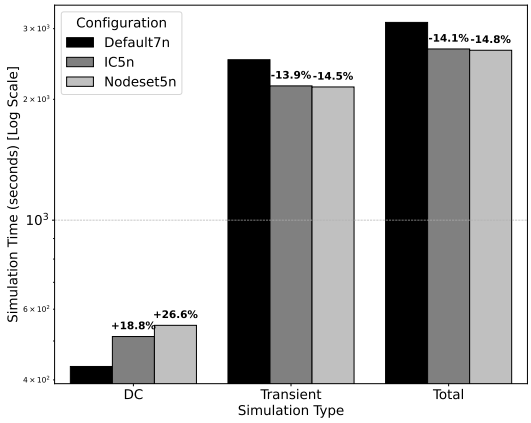


Fig. 2. Simulation time results with script used

Figure 2 presents the results obtained using our initialization script. Although the inclusion of nodeset and initial condition statements increases the DC convergence time due to additional solver overhead, the overall simulation time is reduced by approximately 14%. This improvement is primarily due to the elimination of the initial phase of the simulation sequence, which was previously required to mitigate metastability in the 6T SRAM cells at startup.

CONCLUSION AND FUTURE WORKS

In this work, we demonstrate that compact modeling and memory cell initialization techniques can significantly reduce simulation time for large SRAM macros while maintaining acceptable accuracy. Although the initialization script does not drastically reduce simulation time on its own, it enables faster convergence and makes the simulation results usable from the start. Moreover, flexible memory preloading will be useful in future studies to explore scenarios like full `0`/`1` initialization or custom patterns for realistic workloads.

As future work, we plan to address signal integrity issues in large arrays by modeling interconnect effects and exploring mitigation techniques such as buffering and differential signaling.

REFERENCES

[1] Z. Zhou and G. Zhang, "The fast simulation model of sram," in *2006 8th International Conference on Solid-State and Integrated Circuit Technology Proceedings*, 2006, pp. 1333–1335.

[2] N. Muralimanohar, R. Balasubramonian, and N. Jouppi, "Optimizing nuca organizations and wiring alternatives for large caches with cacti 6.0," in *40th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO 2007)*, 2007, pp. 3–14.

A 22nm Ferroelectric nvSRAM for Critical Systems

Lucas RHETAT[1], Jean-Philippe NOEL[1], Bastien GIRAUD[1],
Laurent GRENOUILLET[2], Julie LAGUERRE[2], Cédric MARCHAND[3], Ian O'CONNOR[3]
[1]Univ. Grenoble Alpes, CEA LIST [2]Univ. Grenoble Alpes, CEA LETI
[3]Univ. Lyon, Ecole Centrale de Lyon, INL, CNRS

***Abstract*—Static Random Access Memories (SRAM) are fast and efficient circuits used as the main working memory of processing units. However, these memories are volatile and associating SRAMs with external non-volatile memories leads to energy consumption increases and security issues. Ferroelectric nvSRAMs are one of the most promising ways of combining the high efficiency of SRAMs with non-volatile operations to tackle these challenges. In this work, an nvSRAM bitcell array is studied : several design parameters are optimized to ensure error-less data transfer between 6 transistors (6T) SRAM and 4 ferroelectric capacitors (4C). The presented 6T4C bitcell presents STORE and RECALL energies of ranging from 3 fJ/bit to 200 fJ/bit, and a RECALL time of 80ns. A high reliability is achieved from -40°C to +85°C for SS, TT and FF fabrication corners. We quantify the area overhead as well as the time and energy increases in SRAM operations resulting from the integration of FeCAPs into the bitcell. A previously developed fast-erase system has also been integrated for countering cold-boot attacks. Combining design optimizations and fast-erase system ensures cold-boot attack immunity of the memory and enables WRITE after STORE operations with very few errors, leading to new use-cases of nvSRAM circuits.**

***Index Terms*—SRAM, nvSRAM, FeRAM, ferroelectrics, security**

I. Introduction

SRAM are the fastest and most energy-efficient memories. However, they are volatile, which means that data must be transferred to a non-volatile (NV) memory before power-off. nvSRAMs overcome theses problems by finely co-integrating non-volatile storage elements and SRAM bitcell. In this work, FeRAM technology (based on Ferroelectric CAPacitors, FeCAPs) that presents the best combination of write speed and energy as well as endurance among others has been selected[1]. The resulting bitcell can then perform 4 different operations: reading from SRAM (op. READ), writing to SRAM (op. WRITE), copying data from SRAM to FeRAM (op. STORE) and moving data from FeRAM to SRAM (op. RECALL). Using these four operations permits to obtain a fast, efficient and non-volatile memory.

In a security point of view, SRAM are prone to cold boot attacks, enabling attackers to recover data previously stored in the memory just after it has been shut down. To counter this, a fast erasing system has recently been developed [2].

In this work, we show that thanks to optimized FeCAPs sizing and the fast-erase system we can obtain a reliable nvSRAM circuit with WRITE ops. between STORE and RECALL, which leads to new applications of these circuits, e.g. using them as Physical Unclonable Functions (PUF) [3].

II. Bitcell and testbench

A. The 6T4C nvSRAM bitcell

Figure 1 shows the 6T4C nvSRAM bitcell. That is composed of a classical 6T SRAM bitcell in which we add 4 FeCAPs, two on each internal node BLTI/BLFI, connected to two different plate-lines (PL). By correctly driving PL we can perform STORE and RECALL operations.

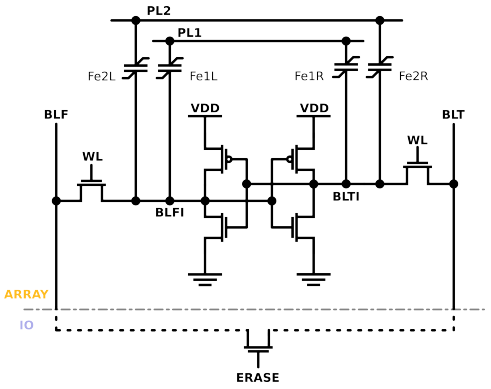


Fig. 1. 6T4C bitcell with two different Plate-Lines

When switched off, SRAM memory retains its data for some time due to limited leakage current from the internal nodes, up to several minutes if the circuit is cooled. Firstly, this raises a security problem and secondly this makes it difficult to RECALL the data stored from the FeCAPs in case their data are different from the last written data in SRAM. To tackle these issues, we are incorporating a previously-developed fast erase system into our circuitry [2].

B. Simulation environment and testbench

The circuit is a 32 bitcell array designed in GF 22nm FDSOI technology.

Fig. 2 shows the overall chronogram. We can see that PL1 and PL2 behave the same during the STORE, READ and WRITE operations but differ during the RECALL one (see [4]). The main variation of this work regarding the existing one is the blue part of the chronogram : SRAM operations after STORE and the addition of the ERASE signal that equalizes the bitcell's internal nodes.

III. Simulation results

A. Fast erase principle

Figure 3 shows the principle of the ERASE operation. This operation presents two objectives. Firstly, it permits to very

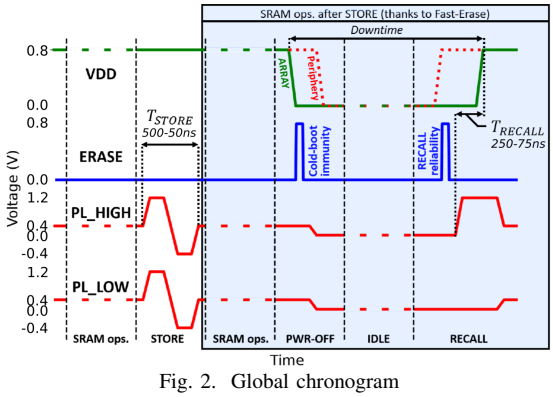


Fig. 2. Global chronogram

quikly erase the data in SRAM, making cold-boot attacks far more tricky [2]. In addition fast-erase takes part in the RECALL operation. Indeed, if the system is restarted very quickly after a shutdown, it may be difficult to perform a RECALL while data stored in the FeCAPs is different than the last data written in SRAM, that are still present.

In the example given Figure 3, the stored data in the FeCAPs is '1' (BLTI=1, BLFI=0) (Fig. 3(a)). In Fig. 3(b) we see that absence of ERASE prevents from performing correct RECALL of the stored data due to the additional potential difference created by residual charges in the internal nodes. Using one ERASE pulse after the PL falling edge (Fig. 3(c)) permits to cancel the above-mentioned bias and greatly improve the reliability of RECALL. However, in some corners it can be necessary to use two ERASE pulses, one before and one after the PL falling edge (Fig. 3(d)). The duration of the ERASE pulses doesn't matter as long as it's greater than the erasing time of the internal nodes (in the nanosecond time scale, see [2]).

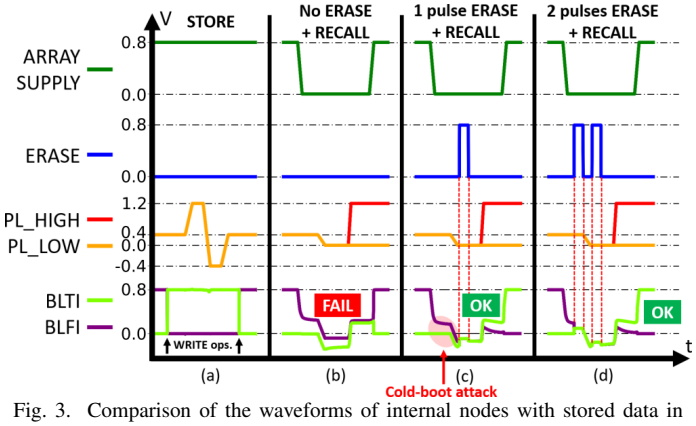


Fig. 3. Comparison of the waveforms of internal nodes with stored data in (a) and RECALL without erase (b), with one erase pulse (c), with 2 erase pulses (d)

B. FeCAPs sizing optimization

We define A_1 and A_2 as the surface of FeCAP's connected to PL1 and PL2, respectively. We define as well: $A_{tot} = A_1 + A_2$ and $R = \frac{A_1}{A_1+A_2}$.

Figure 4 shows the number of errors with respect to the R ratio, for 100 Monte-Carlo. A low error rate can be achieved with $R \in [0.1; 0.4]$ with an optimum for $R \approx 0.1$.

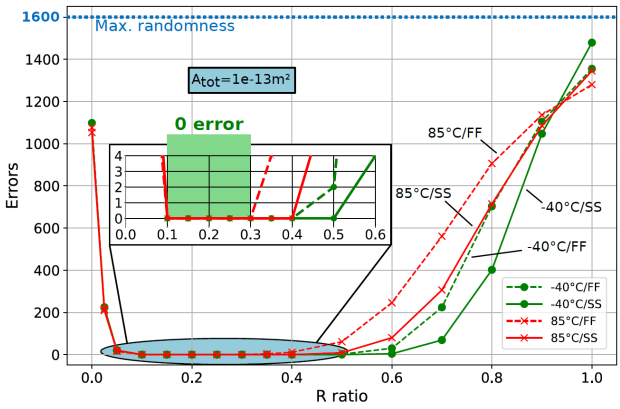


Fig. 4. Number of errors w.r.t. the R ratio

IV. Conclusion

The presented design techniques demonstrate a good scalability in simulation down to the 22nm node at least, with a few design adjustments. The proposed ferroelectric-based 6T4C NVSRAM circuits demonstrate the ability to both STORE data in non-volatile part of memory, for an energy of about 20 fJ/bit, and RECALL it to the SRAM memory,for an energy of about 3 fJ/bit, with a high reliability. In return, we observe worst-case inverting WRITE energy and timing overheads of a factor 5x at the array level, that can be acceptable in the targeted applications and in sub-GHz context. In this work, the combination of FeCAPs sizing optimization with the previously developed Fast-Erase technique demonstrates the ability to continue using the SRAM after storing data in the FeCAPs and recalling data different from that previously written in the internal nodes of SRAM, without having to wait for a significant amount of time. The integration of the Fast- Erase technique also enhances robustness against cold-boot attacks. This design combination leads to new, previously unaddressed use cases of ferroelectric-based NVSRAMs, e.g. in critical systems, including fast context switching or safe-state checkpointing.

References

[1] T. Miwa, J. Yamada, H. Koike, *et al.*, "Nv-sram: A nonvolatile sram with back-up ferroelectric capacitors," in *Proceedings of the IEEE 2000 CICC*, 2000.

[2] J.-P. Noel, M. Pezzin, J.-F. Christmann, *et al.*, "A near-instantaneous and non-invasive erasure design technique to protect sensitive data stored in secure srams," in *ESSCIRC 2021*, 2021.

[3] D. E. Holcomb, W. P. Burleson, and K. Fu, "Power-up sram state as an identifying fingerprint and source of true random numbers," *IEEE Transactions on Computers*, 2009.

[4] K. Takeuchi, M. Kobayashi, and T. Hiramoto, "A feasibility study on ferroelectric shadow srams based on variability-aware design optimization," *IEEE JEDS*, 2019.

Evaluating GEMM Execution on Systolic Arrays

Mohammadali Zoroufchian and David Novo

LIRMM, Univ. Montpellier, CNRS, Montpellier, France

***Abstract*—The growing adoption of artificial intelligence has led to increasingly large and compute-intensive deep neural networks (DNNs), necessitating high-performance hardware solutions. Systolic arrays have emerged as a promising architecture to meet these demands. Efficient development on such hardware requires accurate and fast simulation tools for algorithm evaluation and deployment. In this work, we investigate two simulation platforms by analyzing the execution of a single GEMM operation. Our study reveals that optimized data placement can yield a 1.14× speedup and that computation accounts for less than 19% of the total execution latency, underscoring the importance of memory hierarchy and data layout in overall system performance.**

***Index Terms*—Systolic array, Simulation, Artificial Intelligence**

I. Introduction

In recent years, artificial intelligence (AI) has gained increasing attention, with deep neural networks (DNNs) emerging as a particularly prominent area of focus. DNNs have exhibited remarkable performance across a range of domains, including computer vision, machine translation, and speech recognition. In recent years, the computational requirements for training deep neural network (DNN) models across various applications have increased exponentially [1]. This rapid growth in both model complexity and deployment has driven the development and adoption of specialized hardware accelerators, such as Tensor Processing Units (TPUs) [2].

Deep neural networks (DNNs) primarily rely on matrix multiplication as their core computational operation. Matrix multiplication is inherently parallelizable. Additionally, it exhibits significant data reuse: each element of the input matrices is involved in the computation of multiple output elements. The systolic array architecture is well-suited to exploit both the parallelism and data reuse characteristics of matrix multiplication [3], making it a promising choice for accelerating DNN workloads. The rapid advancement of AI applications has spurred interest in highly parameterized hardware accelerator generators, such as Gemmini [4], and high-level simulators like ONNXim [5]. These tools allow researchers to more efficiently design, prototype, and evaluate hardware-specific algorithms. In this work, we evaluate both Gemmini and ONNXim with the aim of:

- Comparing the performance of a systolic array-based accelerator to that of a general-purpose processor.
- Investigating how data movement overhead impacts overall computational throughput.
- Identifying the limitations and challenges associated with simulation tools.

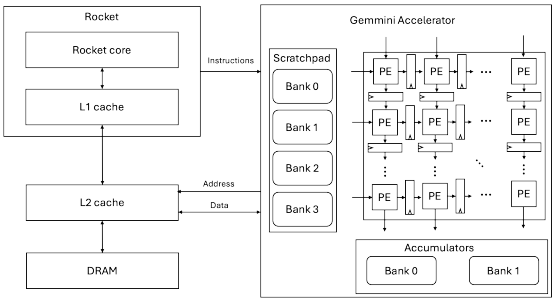


Fig. 1. Generated hardware simplified architectural overview

II. Methodology

We evaluate the performance of a single General Matrix Multiplication (GEMM) operation. To simplify the setup and avoid matrix tiling, all input and output matrices are configured to match the dimensions of the systolic array. Our evaluation employs two simulation platforms: Verilator, an RTL-level simulator, simulating Gemmini, and ONNXim, a high-level, cycle-accurate software simulator.

A. RTL simulation

We employ Gemmini, a systolic-array accelerator generator from the Chipyard framework [6], to perform RTL simulations. The accelerator is configured as detailed in Table I, generating a hardware consisting of a single Rocket chip [7] and a Gemmini coprocessor. A simplified architectural overview is provided in Figure 1.

TABLE I
Gemmini configuration

Configuration	Value
Systolic array dimensions	16×16
Scratchpad capacity	256 KB
Number of scratchpad banks	4
Accumulators capacity	64 KB
Number of accumulators banks	2

To implement the General Matrix-Matrix Multiplication (GEMM) operation on Gemmini, we developed a C program. This program executes on the Rocket core, utilizing Gemmini as a hardware accelerator. For benchmarking purposes, the same operation was also executed solely on the Rocket core. While the generated hardware supports both weight-stationary and output-stationary dataflows, our experiments were con-

ducted exclusively in weight-stationary mode.
A GEMM operation is defined as follows:

$$Out = Input \times Weight + Bias. \quad (1)$$

We developed two variants of the above program, differing solely in the placement of data within Gemmini's scratchpad memory. In the first implementation, all input matrices—namely $Input$, $Weight$, and $Bias$—are allocated within the same scratchpad bank. In contrast, the second implementation assigns each matrix to a separate scratchpad bank to enable parallel access and reduce memory contention.

B. Cycle level simulator

We utilized ONNXim for cycle-level software simulation. ONNXim operates on computational graphs defined in the ONNX format; therefore, we constructed a simple ONNX graph representing a single GEMM operation on 16×16 matrices.

The simulation was configured to model a single-core Neural Processing Unit (NPU) equipped with a 16×16 systolic array. For memory subsystem simulation, ONNXim is integrated with Ramulator [8], we employed the provided DDR4 configuration file to model DRAM behavior accurately.

III. Results

The results from both RTL-level and cycle-level simulations are summarized in Table II. For the RTL simulations, the reported latencies represent the number of cycles from the perspective of the Rocket processor—that is, the number of Rocket clock cycles elapsed from the start to the completion of the GEMM operation.

TABLE II
RTL & Cycle level simulations results

Sim. type	Condition	Latency	Sim. time
RTL	Rocket processor	53,374	160 s
RTL	Gemmini, same bank	398	120 s
RTL	Gemmini, different banks	350	115 s
Cycle level	ONNXim	244	3 ms

Simulation times are provided for comparative purposes only, as they are influenced by the specifications of the host system on which the simulators were executed.

IV. Discussion

As demonstrated, the use of the hardware accelerator yields a performance improvement of over 150× compared to the Rocket processor. This significant speedup highlights the substantial benefits of employing dedicated hardware accelerators in AI applications.

Theoretical computation latency for a GEMM operation involving 16 × 16 matrices executed on a 16 × 16 systolic array is 47 cycles (factoring weight preloading) [5]. When the theoretical computation latency is compared to the total observed GEMM latency, it reveals that the actual computation accounts for only 13% of the overall latency in Gemmini and 19% in ONNXim, highlighting the dominant impact of memory access and data movement on total execution time.

This work also highlights the critical role of data layout in memory performance. For a single GEMM operation, a speedup of 1.14× is observed when input matrices are distributed across different scratchpad memory banks, as opposed to placing all matrices in a single bank. This improvement is due to the simultaneous access to $Input$ and $Bias$ matrices.

Finally, a discrepancy of approximately 30% is observed between the results obtained from ONNXim and the Gemmini. This difference can be attributed to several key architectural and modeling variations:

- DRAM Model: ONNXim employs Ramulator for DRAM simulation, whereas the Gemmini simulation via Verilator assumes a single-cycle SRAM with fixed latency.
- Memory Hierarchy: ONNXim models a two-level memory hierarchy consisting of DRAM and the NPU's local memory, in contrast to Gemmini's three-level hierarchy comprising DRAM, a shared L2 cache, and local memories.
- NPU Model: ONNXim utilizes a deterministic delay model based on input tile size and systolic array dimensions, while Varilator relies on a cycle-accurate RTL model of the hardware.

While RTL simulation offers higher fidelity, it is significantly slower than software-based cycle-accurate simulation, rendering it impractical for large workloads.

V. Conclusion and future work

In this work, we evaluated the execution of GEMM operations on a systolic array using two distinct simulation platforms, each with its own limitations in modeling certain aspects of the system. Additionally, we highlighted the critical impact of data layout on computational performance. These findings point to valuable directions for future research, including the development of advanced memory management techniques and the design of improved simulation tools that enable fast and accurate modeling of NPUs within heterogeneous computing environments.

References

[1] "AI and compute,," https://openai.com/index/ai-and-compute, accessed: Apr 12, 2025.

[2] N. P. Jouppi *et al.*, "In-datacenter performance analysis of a tensor processing unit," in *Proceedings of the annual international symposium on computer architecture*, 2017, pp. 1–12.

[3] H.-T. Kung, *Why systolic architecture?* Design Research Center, Carnegie-Mellon University, 1982.

[4] H. Genc *et al.*, "Gemmini: Enabling systematic deep-learning architecture evaluation via full-stack integration," in *Proceedings of the Design Automation Conference (DAC)*, 2021.

[5] H. Ham *et al.*, "Onnxim: A fast, cycle-level multi-core npu simulator," *IEEE Computer Architecture Letters*, 2024.

[6] A. Amid *et al.*, "Chipyard: Integrated design, simulation, and implementation framework for custom socs," *IEEE Micro*, vol. 40, no. 4, pp. 10–21, 2020.

[7] K. Asanović *et al.*, "The rocket chip generator," Tech. Rep. UCB/EECS-2016-17, Apr 2016. [Online]. Available: http://www2.eecs.berkeley.edu/Pubs/TechRpts/2016/EECS-2016-17.html

[8] Y. Kim *et al.*, "Ramulator: A fast and extensible DRAM simulator," *IEEE Computer architecture letters*, vol. 15, no. 1, pp. 45–49, 2015.

On the Possibility of Relying Solely on FeMFET Variability for PUF Implementations

Miqueas Filsinger, Antoine Cauquil, Damien Deleruyelle, David Navarro, Ian O'Connor, Cédric Marchand
Univ Lyon, Ecole Centrale de Lyon, CNRS,
INSA Lyon, Université Claude Bernard Lyon 1,
CPE Lyon,
INL, UMR5270, 69130 Ecully, France
miqueas.filsinger@ec-lyon.fr

***Abstract*—The promising features introduced by the integration of ferroelectricity into conventional transistor processes have led to extensive studies on device reliability, system-level applicability, and commercial viability. Its low-power characteristics make it particularly suitable for Internet of Things (IoT) applications, where a reliable power source cannot always be guaranteed. Additionally, its intrinsic memory properties make it a strong candidate for non-volatile memory implementations. Since these applications are data-intensive, the need to ensure secure data storage and transmission has motivated the adaptation of classic hardware security strategies to this emerging paradigm, demostrating high effectiveness with ferroelectric designs. However, a variability analysis from a design perspective remains unexplored.**

In this work, we focus on the variations expected in a commercial 28nm process and its compatibility with memory cell design, in view of an implementation of a Physical Unclonable Function in ferroelectric technology.

I. Introduction

In recent years, ferroelectric technology has gained attention as a possible solution to address many applications involving the use of memories in general. From Edge Computing, to the Internet of Things (IoT), improved memory capabilities represent the next step in pushing system performance boundaries. In general, the low power operation and reduced area of Ferroelectric-Metal Field Effect Transistors (FeMFETs) [1] can be attractive for the design of certain system-level applications. Consequently, the importance of implementing security primitives arises in ferroelectric memory implementations as an important tool for data intensive applications.

Plenty of work has been presented in the recent literature about Physical Unclonable Functions (PUFs) as an effective security solution in ferroelectric technology [2], [3], [4]. The idea behind these devices is to leverage random variations in the fabrication stage, in order to give every instance (chip) a unique identity. Each implementation includes the same security primitive, which is later tested by the design house in order to store a record of the output to every possible input, called the challenge-response pair (CRP) library. At the moment of use, the authenticator will interrogate the device in order to compare its CRPs with the ones stored in the cloud, if they coincide, access is granted to the user.

A valid implementation requires proof that the design satisfies certain properties [5]. Such characteristics imply an abstract relationship with the entropy source, that is, the source of random variations. This disconnection between output metrics and design requirements makes achieving a successful PUF a delicate task throughout the entire design process. For instance, while a cell can be engineered to enhance variability, numerous other factors may still influence the cell's stochastic behavior, potentially introducing bias that only becomes evident after fabrication.

At the same time, designers must be given a certain degree of flexibility to ensure that a fair trade-off between performance, reliability, and randomness can always be achieved.

In this work, we tackle the entropy source: how random variations in FeMFETs can enable the design of physical unclonable functions using a 28nm commercial design kit. We aim to investigate how much stochasticity can be effectively harnessed from the technology, and its cell compatibility with a memory application.

II. Methodology and Results

A commercial 28nm design kit was selected to enable a realistic evaluation of ferroelectric integration in advanced CMOS nodes. Knowing the ferroelectric layer characteristics, a sufficiently low capacitance ratio $CR = \frac{C_{fe}}{C_{mos}}$ was selected by tuning the area ratio A_r. That is, affecting the ratio between the area of the transistor and the area of the Fecap. In this way, we ensure that most of the input voltage is dropped in the ferroelectric capacitor, thereby respecting transistor oxide integrity while still allowing the proper programming conditions in the inter-layer. At the same time the ferroelectric capacitor needs to be big enough to allow a gate current to polarize the FET. It was observed that a satisfactory behavior was achieved using thin oxide transistors and a $CR \approx 5\%$.

For the design of each FeMFET, ferroelectric capacitors were chosen to be as small as possible. To the best of our knowledge, the smallest diameters used in fabrication range between 60 nm $\leq \varnothing \leq$ 100nm. For the transistor sizing, a design exploration was performed in order to capture the best I_{HVT}/I_{LVT} relation. As a result, a design case is stated for both ferroelectric sizes as follows in Table I:

	$\varnothing = 60$nm	$\varnothing = 100$ nm
W	750 nm	
L	250 nm	
V_p	3V	2.4V
V_r	150mV	120mV

Table I: Design parameters for both FeCap sizes.

To explore variability impact, Monte-Carlo simulations were conducted, taking into account transistor and Fecap area variability. In order to study different intra-chip scenarios, we emulate the local variations via simulating 500 FeMFETs for each Monte-Carlo run. The results, shown in Figure 1, talk about an important dispersion in the LVT state current, which can be useful for circuit fingerprinting.

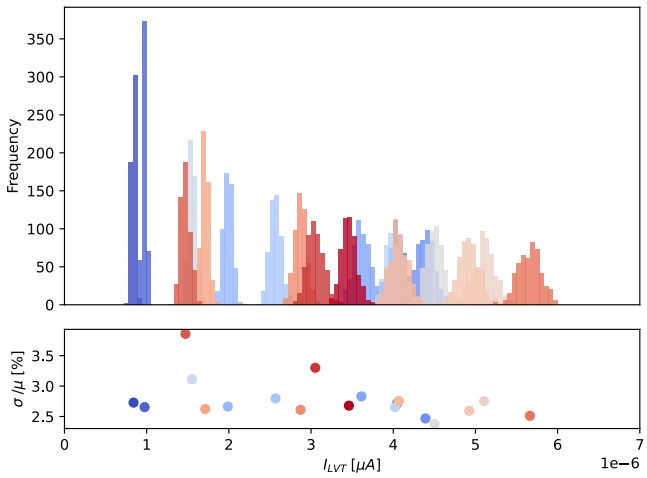


Figure 1: Histogram of LVT current for 500 FeMFETs for various Monte Carlo runs. Below, I_{LVT} relative deviation.

On the other hand, FeCap variations do not introduce important variations in the output, following a linear relation $\sigma/\mu(I_{LVT}) = 1.78\ \sigma_{\text{Area}}$. This aligns with the known stability of the fabrication process, with a reported relative area variability of approximately $\Delta A/A_0 = 0.8\%$ [6].

Observing the global variations, a significant I_{LVT} dispersion becomes evident. This can be interpreted as a potential problem for memory array applications, where fixed current ratios are expected. Simulations indicate a worst-case current ratio of $I_{LVT}/I_{HVT} \approx 100$, and a minimum $I_{LVT} \approx 50$ nA, still suggesting some room for state differentiation with this transistor size.

To asses the feasibility of a cell-to-cell detection, a simple latch circuit was implemented in [2], to amplify the mismatch between both cells LVT state current. The circuit successfully detects and amplifies the simulated unbalance in the 500 FeMFETs pairs, suggesting the latch as a viable intra-chip variability detector. To broadly cover more cases, the same idea is applied in Figure 2. A Monte-Carlo simulation is ran over a thousand times on only one latch circuit to see how different wafer conditions can affect the bistability of the detector.

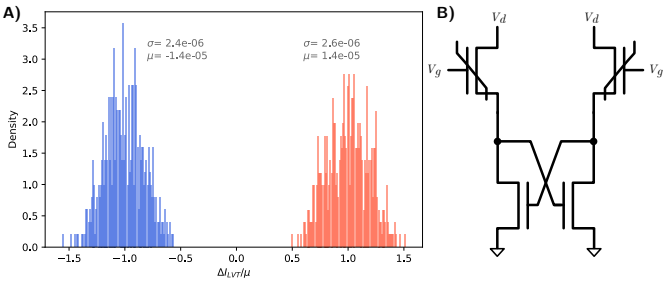


Figure 2: A) Relative current difference between left and right FeMFET B) Latch circuit to enhance identification of variability

III. Conclusion and Future Works

By using a Preisach model fitted with experimentally obtained parameters we could observe the behavior of a single cell, and the repercussion of FeCap and transistor variability in the read-write scheme. Although a proof of concept was made, we could observe that the positive feedback of latch circuits would be sufficient for identifying FeMFET mismatch. Via the emulation of intra-chip Monte Carlo simulations, current relative variations were observed, varying in the range of 2%-4%, which sets a reference for the design of any variability detection scheme. It would be interesting in the future to explore the dual operation of a memory array and a PUF, and its feasibility taking into account layout parasitics and memory perturbations.

Acknowledgment

References

[1] J. F. Scott, *Ferroelectric Memories*. Springer Berlin Heidelberg, 2000. [Online]. Available: http://dx.doi.org/10.1007/978-3-662-04307-3

[2] S. Lim, J. Hwang *et al.*, "Design of physically unclonable function using ferroelectric fet with auto write-back technique for resource-limited iot security," *IEEE Internet of Things Journal*, vol. 11, no. 16, p. 27676–27686, Aug. 2024. [Online]. Available: http://dx.doi.org/10.1109/JIOT.2024.3399482

[3] H. Shao, Y. Zhou *et al.*, "A novel fefet array-based puf: Co-optimization of entropy source and crp generation for enhanced robustness in iot security," in *2023 International Electron Devices Meeting (IEDM)*. IEEE, Dec. 2023. [Online]. Available: http://dx.doi.org/10.1109/IEDM45741.2023.10413787

[4] T. Li, X. Guo *et al.*, "Demonstration of high-reconfigurability and low-power strong physical unclonable function empowered by fefet cycle-to-cycle variation and charge-domain computing," *Nature Communications*, vol. 16, no. 1, Jan. 2025. [Online]. Available: http://dx.doi.org/10.1038/s41467-024-55380-x

[5] P. Jimenez, R. Cardoso *et al.*, "Complexity assessment of analog and digital security primitives signals using the disentropy of autocorrelation," 2024. [Online]. Available: https://arxiv.org/abs/2402.17488

[6] J. Laguerre, "Mémoires ferroélectriques ultra-basse consommation à base de hfo ferroélectrique: vers des matrices intégrables au noeud 28nm et en-deçà," Ph.D. dissertation, Aix-Marseille, 2024, thèse de doctorat dirigée par Bocquet, Marc Sciences pour l'ingénieur : spécialité Micro et Nanoélectronique. [Online]. Available: 2024AIXM0639

Analyse de la sensibilité des réseaux de neurones convolutifs quantifiés

Ahmed Al Kaf, Mounir Benabdenbi, Régis Leveugle
Univ. Grenoble Alpes, CNRS, Grenoble INP*, TIMA, 38000 Grenoble, France

***Résumé*—Les réseaux de neurones convolutifs (CNN) sont de plus en plus utilisés dans divers domaines. Pour réduire leur consommation et leur taille, la quantification est utilisée. La quantification peut aussi améliorer la robustesse face aux fautes mais les réseaux restent vulnérables et leur fiabilité est une préoccupation, surtout pour les applications critiques. Cette étude analyse l'impact des inversions de bits sur un réseau LeNet quantifié en entiers signés 8 bits. Les résultats des campagnes d'injection de fautes révèlent que les données intermédiaires et les poids présentent une sensibilité proche pour les deux directions d'inversion (mais un peu plus petite pour des inversions de 0 vers 1), tandis que pour les biais, les inversions de 0 vers 1 sont très peu critiques. Ces résultats contrastent avec de nombreuses conclusions de l'état de l'art, qui présentent les inversions de 0 vers 1 comme très nettement plus critiques que les inversions en direction opposée. Ils démontrent que les mécanismes de protection proposés dans la littérature, basés sur la plus grande criticité d'une direction, ne sont pas efficaces dans tous les cas et doivent être reconsidérés en fonction de l'implémentation du CNN ciblé.**

***Mots-clés*—Réseaux de neurones, CNN, quantification, robustesse, injection de fautes**

I. INTRODUCTION

Les réseaux de neurones convolutifs (CNN), dont l'utilisation et la complexité croissent rapidement, voient leur consommation énergétique augmenter fortement, rendant cruciale l'adoption de techniques comme la quantification [1]. Dans des contextes critiques (systèmes autonomes ou médicaux par exemple), la vulnérabilité des CNNs aux perturbations pouvant créer des inversions de bits nécessite une évaluation de leur robustesse. Bien que les CNNs quantifiés montrent une meilleure résilience que leurs homologues utilisant des calculs en virgule flottante [2], ils restent vulnérables. Notre étude analyse, pour un exemple typique de CNN quantifié, l'effet de certaines perturbations via des campagnes d'injection de fautes logicielles, définies afin d'être indépendantes de l'implantation du réseau (logiciel sur CPU ou GPU, accélérateur matériel ou circuit dédié …). Le premier résultat est l'identification, pour chaque couche du réseau, des fautes dites critiques, c'est à dire entraînant une baisse de précision de la classification. Mais le principal objectif est de déterminer la direction la plus critique pour les inversions de bits et de comparer ce résultat avec les conclusions présentées dans la littérature pour le même modèle de réseau de neurones, implanté sans quantification. Les résultats montrés dans cet article se concentrent donc sur la sensibilité du CNN en fonction de la direction et en fonction du type de données perturbé (poids, biais ou données intermédiaires entre couches) et de la couche ciblée. La suite de l'article s'organise ainsi : la section II positionne plus précisément l'étude au sein de l'état de l'art ; la section III présente les spécifications de l'étude de cas ; la section IV résume et discute les résultats issus de nos campagnes d'injection de fautes ; enfin, la conclusion ouvre des perspectives, notamment par rapport aux méthodes de protection présentées dans la littérature.

II. POSITIONNEMENT PAR RAPPORT A L'ETAT DE L'ART

Il n'est pas possible dans ces deux pages de résumer précisément les contributions des très nombreux articles ayant abordé la robustesse des CNNs face à différents types de fautes. L'objectif ici est donc seulement de donner quelques exemples de travaux ayant motivé l'étude présentée.

Dans de nombreuses publications, les auteurs observent que les inversions de bits de 0 vers 1 sont les plus critiques. Dans [3], cette conclusion est établie de manière très détaillée sur le réseau LeNet utilisant des paramètres codés en virgule flottante avec plusieurs formats de nombres à virgule flottante. Une solution de tolérance aux fautes est proposée sur cette base et évaluée sur des images du jeu de test MNIST. Dans [4], la même conclusion est présentée, cette fois pour des CNNs quantifiés utilisant des entiers signés 8 bits et pour des fautes dans les neurones d'un accélérateur matériel. Le réseau LeNet faisait partie des exemples pris en compte. Dans [5], nous avions conduit un petit exemple de campagne d'injection de fautes qui montrait au contraire sur un réseau LeNet quantifié utilisant également des entiers signés 8 bits que les inversions critiques observées étaient de 1 vers 0. Cependant, cette tendance avait été obtenue avec seulement 26 images de test issues de MNIST.

Afin de pouvoir conclure avec plus de certitude sur les liens entre criticité et direction des inversions de bit, une campagne d'injection de fautes plus significative était nécessaire et les résultats principaux sont résumés dans cet article.

III. DEFINITION DE L'ETUDE DE CAS

Cette étude utilise le modèle LeNet-5 [6] pour la classification des images MNIST (chiffres 0-9). LeNet est composé de deux couches de convolution (C1, C3), deux couches de regroupement (S2, S4) et trois couches denses (F5-F7). Le chiffre dans la notation des couches indique l'ordre d'apparition de la couche dans le réseau. Le réseau se termine par une fonction d'activation SoftMax. Nous utilisons une implémentation C de LeNet [7] dont les paramètres sont transférés depuis un modèle quantifié réalisé avec TensorFlow. Avec les paramètres transférés, les inférences de la version C sont réalisées avec une précision de 98,05% sur les 10K images de test de MNIST.

Les fautes injectées correspondent à l'inversion d'un seul bit au cours d'une inférence. Les inversions sont appliquées aux paramètres (poids et biais) de LeNet avant le début de l'inférence et aux données intermédiaires après leur calcul

* Institute of Engineering Univ. Grenoble Alpes

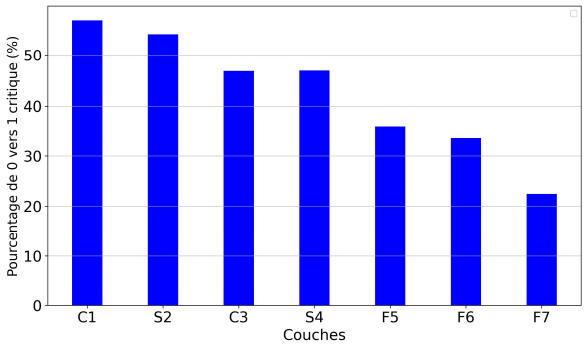


Fig 1 Pourcentage d'injections critiques de direction 0 vers 1 dans les données intermédiaires

dans une couche, avant leur première utilisation dans la couche suivante. Cela représente un sous-ensemble significatif de fautes possibles, permanentes ou transitoires et pouvant avoir des origines diverses, tout en restant indépendant d'une implémentation particulière du réseau.

Le temps de simulation prohibitif d'une campagne exhaustive sur tous les bits impose une approche statistique [8]. Les fautes à injecter sont échantillonnées indépendamment pour chaque type de données associé à chaque couche afin d'obtenir une marge d'erreur de 1% et un niveau de confiance de 99% pour chaque sous-ensemble. Chaque faute est ensuite injectée lors des inférences successives avec chaque image de test de MNIST.

Nous considérons dans la suite qu'une faute est critique si le résultat de la classification ne correspond pas au label MNIST, alors qu'il était correct lors d'une inférence sans faute. Les fautes critiques sont donc celles qui font baisser la précision de la classification.

IV. Resultats experimentaux

Nous présentons ici plus particulièrement deux résultats obtenus lors de nos campagnes concernant la direction des inversions de bit critiques.

Tout d'abord, la figure 1 montre le pourcentage d'inversions de bits de 0 vers 1 parmi les fautes critiques dans les données intermédiaires en sortie de chaque couche (les données intermédiaires de sortie de la couche F7 étant modifiées avant la fonction SoftMax). Ce pourcentage varie selon la couche ciblée, en diminuant lors de la progression dans le réseau. À l'inverse, le pourcentage augmente pour les injections dans les poids. Concernant les biais, aucune tendance significative n'a été observée. Ces résultats divergent de ce qui est observé dans [3] où les inversions de 0 vers 1 ont une criticité nettement dominante pour toutes les couches sauf pour F7 lorsque les données intermédiaires sont ciblées.

Le Tableau 1 synthétise les résultats obtenus globalement sur l'ensemble des couches en fonction du type de données perturbé. Pour chaque cible, nous reportons le nombre total de fautes critiques et parmi celles-ci les fautes critiques de 0 vers 1. Les résultats révèlent des différences marquées : les inversions de 0 vers 1 représentent 44,21% et 48,22% des fautes critiques pour les données intermédiaires et pour les poids respectivement, mais seulement 11,55% pour les biais. De plus, ces observations contrastent avec [3] qui, pour un codage en virgule flottante, indique des taux variant de 90,25% à 99,67%. Par ailleurs nos résultats diffèrent aussi de ceux présentés dans [4].

Tableau I. Resume des injections sur les cibles

Cible	# total fautes critiques	# total fautes critiques de 0 vers 1	% fautes critiques de 0 vers 1
Données intermédiaires	535015	236514	44,21%
Biais	109194	12608	11,55%
Poids	91257	44006	48,22%

V. Conclusion

Les résultats des campagnes d'injection de fautes menées sur une version quantifiée du réseau LeNet montrent que les erreurs de classification dues à des inversions de bits peuvent provenir d'inversions dans les deux directions. Toutefois la contribution de chaque direction dépend de la couche et du type de données perturbé. Les deux directions d'inversion ont pour notre cas d'étude un impact assez proche sur les données intermédiaires et les poids, bien que la direction de 0 vers 1 apparaisse comme la moins critique. Pour les biais, les inversions de 0 vers 1 s'avèrent réellement beaucoup moins critiques. Ces résultats contrastent fortement avec de nombreuses conclusions de l'état de l'art.

La conséquence directe de cette étude est qu'il n'est pas judicieux de partir du principe qu'une direction donnée est très nettement plus critique que l'autre. La comparaison entre les différentes versions de LeNet citées dans cet article illustre que pour un même modèle de réseau des différences d'implémentation peuvent conduire à des sensibilités très différentes. Ceci implique également qu'un mécanisme de protection basé sur des constats de sensibilité d'une implémentation ne sera pas forcément efficace pour une implémentation différente. Il est donc nécessaire d'adapter les mécanismes de protection en fonction des caractéristiques exactes de l'implémentation visée. L'impact par exemple du choix des fonctions d'activation est un sujet à considérer, au même titre que le format de nombres employé.

References

[1] S. Gholami, S. Kim, Z. Dong, Z. Yao, M. W. Mahoney, and K. Keutzer, "A survey of quantization methods for efficient neural network inference," in Low-Power Computer Vision. Chapman and Hall/CRC, 2022, pp. 291–326.

[2] H.-B. Wang, Y.-S. Wang, J.-H. Xiao, S.-L. Wang, and T.-J. Liang, "Impact of single-event upsets on convolutional neural networks in Xilinx Zynq FPGAs," IEEE Trans. Nucl. Sci., vol. 68, no. 4, Apr. 2021, pp. 394–401.

[3] W. Guillemé, A. Kritikakou, Y. Helen, C. Killian, and D. Chillet, "HTAG-eNN: Hardening technique with AND gates for embedded neural networks," 61st ACM/IEEE Design Autom. Conf. (DAC '24), New York, NY, USA, 2024

[4] M. H. Ahmadilivani, M. Taheri, J. Raik, M. Daneshtalab and M. Jenihhin, "Enhancing Fault Resilience of QNNs by Selective Neuron Splitting," IEEE 5th International Conference on Artificial Intelligence Circuits and Systems (AICAS), Hangzhou, China, 2023

[5] R. Leveugle, M. Benabdenbi, A. Al Kaf and L. Noizette, "Combining Acceleration and Approximation in Dependable Edge AI: Optimization Methodology and Tools Applied to a Case Study," *IEEE International Conference on Design, Test and Technology of Integrated Systems (DTTIS)*, Aix-En-Provence, France, 2024.

[6] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," Proceedings of the IEEE, vol. 86, no. 11, Nov. 1998, pp. 2278–2324.

[7] https://github.com/fpetrot/lenet

[8] R. Leveugle, A. Calvez, P. Maistri and P. Vanhauwaert, "Statistical fault injection: Quantified error and confidence," Design, Automation & Test in Europe Conference & Exhibition (DATE), Nice, France, 2009, pp. 502-506.

Détection de Hardware Trojan basée sur les Anomalies du Comportement Transitoire

Anonymous Authors

***Abstract*—Les chevaux de Troie matériels (HT), menace persistante pour les circuits intégrés, compromettent leur intégrité via des modifications discrètes pouvant causer dysfonctionnements ou fuites de données. Nous proposons une méthode de détection exploitant le comportement transitoire des circuits combinatoires. Notre approche génère automatiquement des "patterns" de test spécifiques révélant les anomalies comportementales créées par les HT lors de la stimulation des "designs". La comparaison du comportement transitoire d'un circuit suspect avec un modèle de référence permet ensuite la détection d'éventuelles altérations. L'évaluation expérimentale sur diverses applications synthétiques et réelles montre que la méthode détecte 98,3% des altérations en moyenne (91-99,99%). Fait marquant, les "patterns" élaborés avec des modèles de délais simplifiés conservent en grande partie leur efficacité avec des modèles plus réalistes, ouvrant des perspectives d'adaptation pour la détection de HT sur des circuits.**

I. Introduction

Au cours des dernières décennies, les architectures des circuits intégrés (CI), les technologies associées et la chaîne d'approvisionnement ont connu une évolution considérable. Aujourd'hui, leur complexité est plus grande que jamais, ce qui exige des efforts sans précédent pour garantir la conception et la fabrication de circuits intégrés fiables et sécurisés.

Les chevaux de Troie matériels (Hardware Trojans - HTs) sont des altérations furtives et malveillantes pouvant être réalisées des premières phases de conception jusqu'aux étapes de fabrication des circuits intégrés [1]. Ces altérations visent à provoquer des fuites de données ou à induire des dysfonctionnements (partiels ou totaux, temporaires ou permanents). Les HTs peuvent être classés en fonction de leurs conditions de déclenchement, de leur impact sur le circuit ou de leur architecture [2][3]. Ces attaques peuvent se produire dans divers scénarios caractérisés par : l'entité à l'origine de l'attaque, l'étape de conception à laquelle l'attaque se produit, et l'architecture du cheval de Troie.

Dans nos travaux, nous nous concentrons sur les HTs *à charge utile explicite* (en référence à la classification de Jin et Makris [4]). Ces HTs se caractérisent par leur effet sur la fonction logique du circuit qui modifie son comportement standard. Afin de détecter ces HTs dans les couches combinatoires, nous proposons une approche exploitant le comportement transitoire des valeurs des signaux, induit par leur propagation à travers les portes logiques. Le comportement transitoire d'un bloc combinatoire fait référence aux transitions (c.à.d. aux valeurs intermédiaires) résultant d'une séquence de valeurs appliquées à ses entrées. Il est défini comme l'opposé sémantique de l'*état stabilisé*. L'analyse du comportement transitoire exploite les caractéristiques physiques du circuit, des propriétés intrinsèques servant de signature unique de la structure du circuit. Dans la méthode proposée, la détection s'effectue par analyse dynamique de modèles pré-silicium et par leur stimulation au moyen de patterns de test spécifiques.

II. Contexte et Travaux Connexes

Pour couvrir toute la surface d'attaque offerte par les flots de conception complexes actuels, plusieurs méthodes de détection complémentaires doivent être combinées. Certaines approches, relevant généralement du domaine de la vérification formelle [5], se concentrent sur la vérification de la validité fonctionnelle du circuit. D'autres approches s'appuient sur les paramètres physiques du circuit (par exemple, la consommation d'énergie, le rayonnement électromagnétique, etc.) généralement classée comme des analyses par canaux auxiliaires. Il existe également des méthodes destructives telles que l'imagerie couche par couche des circuits intégrés [6].

Dans nos travaux, nous travaillons sur une méthode de détection basée sur l'analyse du comportement transitoire. À notre connaissance, il existe deux familles d'analyse du comportement transitoire : la mesure du délai des chemins pendant le fonctionnement et le surcadencement ("overclocking").

La mesure du délai des chemins pendant le fonctionnement [7][8][9] consistent à mesurer le délai présenté par les multiples chemins du circuit en cours de fonctionnement. L'objectif est de détecter toute violation des références temporelles induite par l'insertion d'un HT. Les méthodes du surcadencement [10][11][12] consistent à piloter les bascules avec un signal d'horloge dont la période est inférieure au délai du chemin critique du circuit. Les approches basées sur le surcadencement provoquent un fonctionnement anormal des circuits intégrés et utilisent des méthodes statistiques (qui manquent souvent d'interprétabilité).

Dans nos travaux, nous utilisons une méthode de détection des HTs qui exploite la variation du comportement transitoire

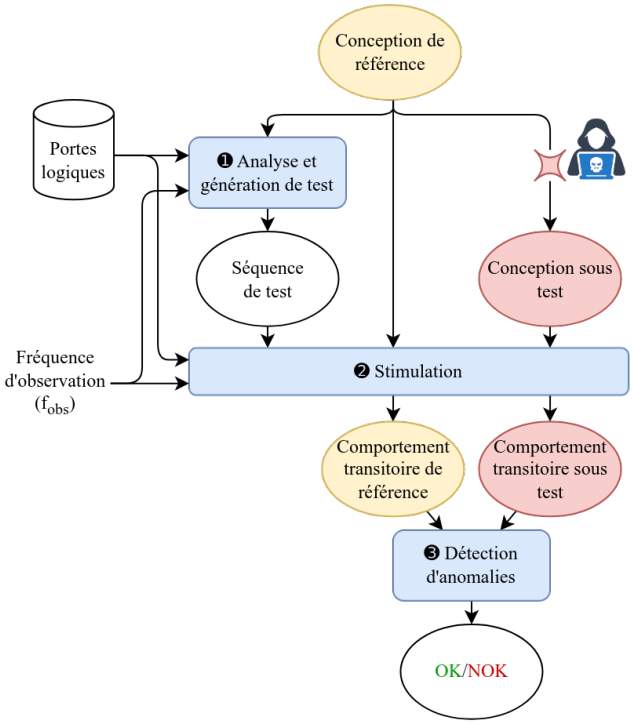


Fig. 1: Flot de la méthode proposée

plutôt que la variation des retards sur le chemin. Le comportement transitoire est observé en utilisant une approche basée sur une technique de *balayage de cycle* plutôt que sur le surcadencement. Cette technique d'observation simple pourrait être adaptée à long terme aux circuits intégrés.

III. Méthodologie

Notre méthode s'appuie sur une formalisation du comportement transitoire via des Equations Booléennes Temporalisées (EBTs). Ces EBTs représentent la valeur d'un signal pour tout instant t. Ainsi, l'EBT $s(t)$ modélise la valeur d'un signal s en fonction des valeurs aux entrées du circuit en prenant en compte les délais des portes. Grâce à ces EBTs, nous formalisons ensuite une Anomalie du Comportement Transitoire (ACT) telle qu'une différence dans le comportement transitoire de deux circuits dont la structure est supposée identique. Plus formellement, soit C un circuit initial et C' un circuit altéré par l'insertion d'une porte (un HT). Alors, il existe une sortie o et son équivalent o' (dans C'), pour laquelle $o(t_\alpha) \neq o'(t_\alpha)$ avec t_α l'instant d'une anomalie. Cette formalisation permet la construction d'un système d'équation résolvable automatiquement dont la solution correspond à une Paire de Vecteurs de Test (PVT).

La méthode que nous proposons exploite les ACT et leur formalisation pour la détection des HTs. Elle repose sur trois étapes principales, comme le montre la figure 1. La première étape *Analyse et génération de tests*, analyse le circuit à l'aide d'une bibliothèque de portes logiques afin de générer une séquence de test contenant des PVTs pour chaque signal à risque identifié. Chaque PVT provoque une ACT lorsque la structure du circuit sous vérification a été modifiée par l'insertion d'une porte sur le signal à risque associé. Chaque PVT est obtenue par la résolution d'un système de contraintes construit sur les EBTs évoquées plus tôt, obtenue pendant l'analyse du circuit. La deuxième étape *Stimulation* consiste à produire et à observer des comportements transitoires en appliquant les PVTs générées à une conception de référence et à une conception en cours de vérification. L'observation s'effectue par une méthode de *parcours de cycle* dans laquelle les valeurs transitoires aux sorties des circuits sont échantillonnées à intervalles de temps réguliers, correspondant à une résolution r. Plus la valeur de r est petite, plus la probabilité d'échantillonner les ACTs est grande. La dernière étape *Comparaison*, compare les deux comportements transitoires observés pour révéler les ACTs provoqués.

IV. Evaluation Expérimentale

La méthode est évaluée expérimentalement sur divers circuits (synthétiques et réels), divers HTs (3 modèles différents) et deux modèles de temps de propagation. Chaque circuit est analysé afin de générer une séquence de test constituée de PVTs. Chaque circuit est altéré par chaque HT, créant ainsi 3 versions altérées. La version de référence et les 3 versions altérées sont stimulées avec les PVTs en simulation et les comportements transitoires sont observés par la méthode de parcours de cycle avec plusieurs résolutions. En comparant les comportements transitoires des versions altérés avec celui de la version de référence, chaque différence est une ACT qui trahie alors l'altération structurelle. Les expériences réalisées montrent que :

- la méthode proposée permet de générer des PVTs pour 99.99% des signaux à risques identifiés dans des circuits à applications réelles ;
- tous les PVTs générés provoquent une anomalie lorsque le circuit a été altéré sur le signal à risque associé ;
- la méthode de capture des ACTs proposée, compatible avec une implémentation matérielle, permet d'en observer 92.5% pour les résolutions les plus faibles (max. 100%) ;
- 94% des PVTs générés avec un modèle de délai, restent efficaces sur un modèle de délai plus réaliste.

V. Conclusion

Nous avons proposé une méthode qui tire parti des anomalies du comportement transitoire (ACT) pour détecter l'insertion de HT. Ce travail démontre que le comportement transitoire contient des informations exploitables pour la détection de HTs. Les premiers résultats présentés suggèrent également que notre méthode reste pertinente pour détecter les HTs avec des modèles de délai plus détaillés. Les travaux futurs proposeront de prendre en compte des modèles de délai encore plus précis, des HTs *à charge utile implicite*, des architectures séquentielles et des circuits intégrés réels pour les expériences. En outre, la méthode sera optimisée pour réduire le temps de génération des PVT et le temps de stimulation des conceptions.

REFERENCES

[1] B. Shakya, T. He, H. Salmani, D. Forte, S. Bhunia, and M. Tehranipoor, "Benchmarking of Hardware Trojans and Maliciously Affected Circuits," *Journal of Hardware and Systems Security*, vol. 1, no. 1, pp. 85–102, Mar. 2017.

[2] M. Tehranipoor and F. Koushanfar, "A Survey of Hardware Trojan Taxonomy and Detection," *IEEE Design & Test of Computers*, vol. 27, no. 1, pp. 10–25, Jan. 2010.

[3] K. Xiao, D. Forte, Y. Jin, R. Karri, S. Bhunia, and M. Tehranipoor, "Hardware Trojans: Lessons Learned after One Decade of Research," *ACM Transactions on Design Automation of Electronic Systems*, vol. 22, no. 1, pp. 6:1–6:23, May 2016.

[4] Yier Jin and Y. Makris, "Hardware Trojan detection using path delay fingerprint," in *2008 IEEE International Workshop on Hardware-Oriented Security and Trust*. Anaheim, CA, USA: IEEE, Jun. 2008, pp. 51–57.

[5] X. Guo, R. G. Dutta, Y. Jin, F. Farahmandi, and P. Mishra, "Pre-silicon security verification and validation: A formal perspective," in *Proceedings of the 52nd Annual Design Automation Conference*, ser. DAC '15. New York, NY, USA: Association for Computing Machinery, Jun. 2015, pp. 1–6.

[6] N. Vashistha, H. Lu, Q. Shi, M. T. Rahman, H. Shen, D. L. Woodard, N. Asadizanjani, and M. Tehranipoor, "Trojan Scanner: Detecting Hardware Trojans with Rapid SEM Imaging Combined with Image Processing and Machine Learning," in *ISTFA 2018*, Phoenix, Arizona, USA, Nov. 2018, pp. 256–265.

[7] Jie Li and J. Lach, "At-speed delay characterization for IC authentication and Trojan Horse detection," in *2008 IEEE International Workshop on Hardware-Oriented Security and Trust*. Anaheim, CA, USA: IEEE, Jun. 2008, pp. 8–14.

[8] Y. Lyu and P. Mishra, "Automated Test Generation for Trojan Detection using Delay-based Side Channel Analysis," in *2020 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. Grenoble, France: IEEE, Mar. 2020, pp. 1031–1036.

[9] D. Ismari, J. Plusquellic, C. Lamech, S. Bhunia, and F. Saqib, "On detecting delay anomalies introduced by hardware trojans," in *Proceedings of the 35th International Conference on Computer-Aided Design*. Austin Texas: ACM, Nov. 2016, pp. 1–7.

[10] V. R. Surabhi, P. Krishnamurthy, H. Amrouch, K. Basu, J. Henkel, R. Karri, and F. Khorrami, "Hardware Trojan Detection Using Controlled Circuit Aging," *IEEE Access*, vol. 8, pp. 77 415–77 434, 2020.

[11] V. R. Surabhi, P. Krishnamurthy, H. Amrouch, J. Henkel, R. Karri, and F. Khorrami, "Exposing Hardware Trojans in Embedded Platforms via Short-Term Aging," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 11, pp. 3519–3530, Nov. 2020.

[12] X. Meng, R. Hassan, S. M. P. Dinakrrao, and K. Basu, "Can Overclocking Detect Hardware Trojans?" in *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2021, pp. 1–5.

Compression Schemes for Edge AI

Manar Gani, Alberto Bosio, David Novo
Ecole Centrale de Lyon, CNRS, CPE Lyon, INL,
UMR5270, 69130 Ecully, France

***Abstract*—Deep Neural Networks (DNNs) offer state-of-the-art performance across a variety of AI tasks, yet their large size and computational demands hinder deployment on resource-constrained edge devices. This work presents an early-stage study aimed at reproducing and evaluating existing compression techniques—including quantization, pruning and huffman coding as described in seminal works like Deep Compression—within the context of low-power microcontrollers. Our goal is to build a reproducible baseline and identify the practical limitations and opportunities for future innovation in extreme compression schemes for Edge AI.**

***Index Terms*—Edge AI, Deep Neural Networks, Compression, Quantization, Pruning, Autoencoders, Memory Optimization, Low-Power,Inference**

I. Introduction

Deep Neural Networks (DNNs) have become foundational tools in machine learning, delivering high accuracy across diverse domains such as image recognition, natural language processing, healthcare, and autonomous driving systems. However, their practical deployment on edge devices—where computational power, memory, and energy resources are severely limited—remains a major challenge. Traditional DNN architectures, often trained and executed in data centers using high-end GPUs, require significant memory bandwidth and storage, making them unsuitable for resource-constrained environments.

To address this issue, the research community has explored various *model compression techniques* aimed at reducing the storage and computational footprint of DNNs. Among the most impactful of these are *quantization*, which replaces 32-bit floating-point weights with lower-bit representations (e.g., 8-bit or binary), and *weight sharing*, where similar weights are grouped and indexed to reduce redundancy. The seminal work by Han et al. on Deep Compression [1] demonstrated that combining pruning, quantization [2], and Huffman coding can achieve up to 49×compression with minimal accuracy loss. In the early stages of this Ph.D. project, our focus is on reproducing and rigorously evaluating such established methods in the specific context of *low-power edge devices*. The primary goal of this phase is to understand the real-world limitations and behavior of these compression techniques under such extreme constraints. By building a reproducible and configurable evaluation framework, we aim to identify which strategies are most effective, what trade-offs exist between compression ratio and accuracy, and where current methods fall short. These insights will inform future work on novel, hybrid, and generative compression [3] strategies designed specifically for the next generation of edge AI systems.

II. Methodology

The goal of this study is to reproduce and rigorously evaluate existing compression techniques—namely quantization, pruning, and Huffman coding. We replicate the methodology described in Han et al.'s Deep Compression [**?**], using a selection of standard neural networks to build a strong reproducibility baseline for future exploration.

A. Model Architectures

Three well-known architectures were selected for evaluation due to their diverse complexity and relevance in both classical and modern contexts:

- **LeNet-300-100**: A fully-connected feedforward network with two hidden layers (300 and 100 units) commonly used on the MNIST dataset.
- **VGG-16**: A deep convolutional neural network with 134 million parameters, widely used in image classification tasks. Experiments were performed on CIFAR-10
- **AlexNet**: A deep convolutional neural network with approximately 60 million parameters, In this work, we evaluate AlexNet using a downsampled version of the ImageNet.

These models provide a range from low-complexity (LeNet) to high-complexity (VGG-16), allowing the study of compression scalability and feasibility in edge contexts.

B. Compression Techniques Applied on LeNet-300-100

The compression pipeline consists of three sequential stages:

We replicate the three-stage pipeline proposed by Han et al. [1], which combines pruning, quantization with weight sharing, and Huffman coding for efficient model compression.

1) Pruning: The first step involves removing redundant connections by setting small-magnitude weights to zero. This creates a sparse network structure, which significantly reduces the number of parameters. The pruning is followed by retraining to recover any loss in accuracy.

2) Quantization with Weight Sharing: Rather than representing each weight individually, this step clusters the non-zero weights into a fixed number of shared values (centroids) using k-means clustering.

3) Huffman Encoding: To further reduce storage, Huffman coding is applied to both the weight indices and the sparse matrix structure .At this stage, VGG-16 and AlexNet results are limited to pruning and retraining future work will focus on the quantization and Huffman Coding part.

III. Results

In this section, we report the effects of applying pruning, quantization and Huffman coding . Our analysis is based on the compression ratio, top-1 accuracy supported by qualitative results such as weight distributions.

A. Pruning Effects

Pruning was applied to remove unimportant connections based on magnitude thresholds, aiming to reduce the number of non-zero weights without significantly impacting model performance. This was especially effective for LeNet-300-100 which exhibited a high degree of redundancy as well as on VGG-16 and AlexNet. Figure 1 shows the weight distributions after pruning for the VGG-16 weights.

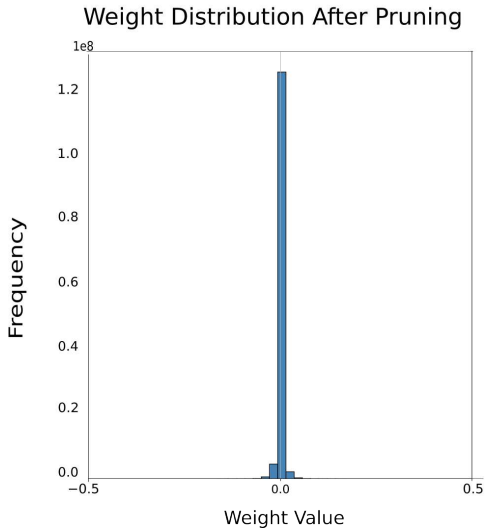


Fig. 1. Weight distribution after pruning for VGG-16

The compression ratios we aimed for gave the Top-1 accuracies presented in the table,the deep compression paper do not mention accuracies after pruning only.

TABLE I
Compression Ratio and Top-1 Accuracy after pruning for VGG-16, LeNet-300-100 and AlexNet

Model	Compression Ratio After Pruning	Top-1 Accuracy After Pruning
VGG-16	13 .99x	89.82%
LeNet-300-100	12 .5x	98.16%
AlexNet	12.5x	71.47%

B. Weight Sharing Outcomes

Weight sharing was implemented using k-means clustering. This technique provides high compression with relatively low impact on model accuracy, particularly in layers with redundant or smoothly distributed weights. In these experiments with LeNet-300-100 applying weight sharing with retraining preserved accuracy while achieving a substantial reduction in parameter storage.

Figure 2 shows sample weight sharing visualizations.

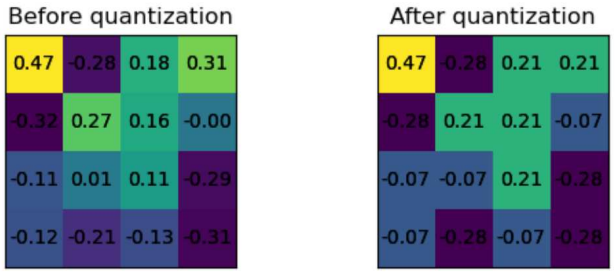


Fig. 2. Visualization of a 4×4 weight matrix before and after applying linear quantization on a LeNet-300-100 Fully Connected Layer.

C. Combined Compression Results

Applying Huffman encoding on top of pruning and quantization in this order [4] yielded additional size reductions the combined pipeline showed a high compression ratio of 48x for LeNet-300-100, future work will be applying the quantization and the Huffman Encoding on VGG-16 and AlexNet.

IV. Conclusion and Future Work

In this work, we presented a study of state-of-the-art compression techniques for deep neural networks on resource-constrained edge devices. We focused on replicating the Deep Compression framework, applying its full pipeline—pruning, weight sharing via k-means clustering, and Huffman encoding—to LeNet-300-100. For VGG-16, we implemented the pruning stage to reduce redundancy while having a negligible loss in accuracy, laying the groundwork for further compression. These early results demonstrate that significant memory savings can be achieved with minimal accuracy degradation, confirming the practicality of such methods in edge contexts. Future work will extend the full compression pipeline to deeper networks like VGG-16 and explore more compression techniques including generative models and hybrid compression schemes. The ultimate goal is to build scalable, deployment-ready solutions that make efficient edge AI a reality.

Acknowledgment

This work has been funded by Project AdaptING: Adaptive architectures for embedded artificial intelligence ANR-23-PEIA-0009

References

[1] S. Han, H. Mao, and W. J. Dally, “Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding,” 2016. [Online]. Available: https://arxiv.org/abs/1510.00149

[2] Y. Shen, M. Sun, J. Lin, J. Zhao, and A. Zou, “Order of compression: A systematic and optimal sequence to combinationally compress cnn,” 2024. [Online]. Available: https://arxiv.org/abs/2403.17447

[3] T. Jůza and L. Sekanina, “Gpam: Genetic programming with associative memory,” in *26th European Conference on Genetic Programming (EuroGP) Held as Part of EvoStar*, ser. LNCS, vol. 13986, no. 3. Cham: Springer Nature Switzerland AG, 2023, pp. 68–83.

[4] Y. Shen, M. Sun, J. Zhao, and A. Zou, “Order of compression: A systematic and optimal sequence to combinationally compress cnn,” 2024. [Online]. Available: https://api.semanticscholar.org/CorpusID:268692052