Multi-Partner Project: Securing Future Edge-AI Processors in Practice (CONVOLVE)

Sven Argo[3], Henk Corporaal[1], Alejandro Garza[4], Marc Geilen[1], Manil Dev Gomony[1], Tim Güneysu[3,6], Adrian Marotzke[4,7], Fouwad Mir[2,5], Christian Larmann[2,5], Jan Richter-Brockmann[3], Jeffrey Smith[1], Mottaqiallah Taouil[2,5], and Said Hamdioui [2,5]

[1]Eindhoven University of Technology, [2]Delft University of Technology
[3]Ruhr University Bochum, [4]NXP Semiconductors Germany GmbH, [5]CognitiveIC, Delft, The Netherlands
[6]German Research Center for Articial Intelligence, Bremen, [7]Hamburg University of Technology

Abstract—**Artificial Intelligence (AI) has had a profound impact on our contemporary society, and it is indisputable that it will continue to play a significant role in the future. To further enhance AI experience and performance, a transition from large-scale server applications towards AI-powered edge devices is inevitable. In fact, current projections indicate that the market for Smart Edge Processors (SEPs) will grow beyond 70 Billion USD by 2026 [1]. Such a shift comes with major challenges, as these devices have limited computing and energy resources yet need to be highly performant. Additionally, security mechanisms need to be implemented to protect against diverse attack vectors as attackers now have physical access to the device. Besides cryptographic keys, Intellectual Property (IP), including neural network weights, may also be potential targets.**

The CONVOLVE [2] project (currently in its intermediate stage) follows a holistic approach to address these challenges and establish the EU in a leading position in embedded, ultra-low-power and secure processors for edge computing. It encompasses novel hardware technologies, end-to-end integrated workflows, and a *security-by-design* approach. This paper highlights the security aspects of future edge-AI processors by illustrating challenges encountered in CONVOLVE, the solutions we pursue including some early results, and directions for future research.

Index Terms—**PQC, TEE, composability, real-time, CIM**

I. Introduction

Bringing Artificial Intelligence (AI) applications to edge devices is the next big step for AI researchers as well as the entire AI industry. It promises better performance by cutting out the time needed to send and retrieve data via networks – a common bottleneck in practice. Local computations also benefit from the privacy of the processed data, which may become a strict requirement for widespread adoption and a critical selling point for commercial applications. The accompanying challenges, however, are numerous and complex. Edge-devices have significantly restricted computing and energy resources. Models and Intellectual Property (IP) need to be deployed, updated and stored on devices in the field without being compromised. Moreover, AI applications need to be reliable with respect to timing constraints to support advanced use-cases where they no longer run in fully controlled environments.

The European Union (EU) is in a prime position to overcome these difficulties and take a leading position in this emerging market. The CONVOLVE project was initiated with the goal of enhancing the development and deployment of edge-AI applications. CONVOLVE's core objectives comprise 1) a 100x improvement of energy efficiency, 2) a ten-fold reduction of design time, 3) a modular, long-term, and compositional hardware security framework, and 4) smarter edge applications.

To address these ambitious goals, CONVOLVE considers the entire design stack. Beginning at the hardware, we define a modular, Ultra-low Power (ULP) System-on-Chip (SoC) architecture with a RISC-V host and customized accelerators for AI and cryptographic computations. By using these dedicated hardware accelerators, we aim to improve performance and thus reduce energy consumption compared to pure software solutions. In addition, novel Compute-in-memory (CIM) technologies like RRAM- and SRAM-based CIM are employed to delegate Multiply-Accumulate (MAC) operations to memory and avoid performance bottlenecks. To integrate these diverse accelerators into software applications, CONVOLVE develops an end-to-end compilation flow. In this case, end-to-end starts at the AI model itself enabling the integration of Design Space Exploration (DSE) tools like ZigZag [3] or Stream [4] to find hardware mappings. Ultimately, CONVOLVE improves the AI models themselves, as the quantization of weights, pruning and early-exiting strategies further improve performance without affecting accuracy negatively.

The project also features four diverse use-cases: speech quality enhancement, acoustic scene analysis, traffic supervision, and computer vision tasks for satellite imagery. Each use-case serves as guidance to the actual needs of the industry and contributes new facets of AI applications, e.g., Artificial Neural Networks (ANNs) vs. Convolutional Neural Networks (CNNs), online vs. offline learning, or dynamic vs. static Neural Networks (NNs). Likewise, distinct applications require different security features. For instance, chips deployed to space are not susceptible to side-channel based IP theft, but have a strong need for long-term secure communication channels with a remote controller. For optimal energy efficiency, end-users must be able to adapt the security framework to their individual use-case and requirements and shed any unnecessary overhead.

To this end, we start by introducing our adversary model and the major security challenges of CONVOLVE in Section II. We

This work has received funding from the European Union's Horizon research and innovation programme under grant agreement No 101070374.

provide insights into our solutions in Section III and conclude with an outlook on remaining challenges for CONVOLVE and the scientific community in general in Section IV.

II. Security Challenges in CONVOLVE

In this section, we highlight the main security related challenges of CONVOLVE. For that, we first present our philosophy in general and discuss our attacker model.

A. CONVOLVE's Philosophy

CONVOLVE adheres to a strict *security-by-design* philosophy. Considering security in the entire design process is a essential step towards achieving a ten-fold decrease in design time. It allows end-users to pick and combine security features only when required. However, the design of such a modular security architecture poses interesting and complex challenges.

B. Attacker Model

To address these challenges, we consider a powerful and versatile adversary model. Hence, we assume that the attacker has access to a large-scale quantum computer. This is vital for ensuring long-term security and longevity of the project's results. Moreover, for edge-devices, it is natural to assume the attacker has physical access and can obtain side-channel information like execution time [5], power consumption [6] or electromagnetic radiation [7]. Attackers with the ability to physically manipulate the execution, e.g., via fault injections [8], are out of scope. However, a potential adversary can run arbitrary software on the same system, possibly exploiting software bugs, interfere in scheduling, or attempt to block peripherals. This attacker model is a worst-case assumption for all current and future use-cases and serves as a basis for our *security framework*. Nevertheless, end-users can then derive a concrete *security architecture* for their application, with weaker adversary models if needed.

C. Challenges

Concretely, we identify the following core challenges specific to CONVOLVE:

- **Edge processors** are naturally restricted in computing and energy resources. This is in strong contrast to Post-Quantum Cryptography (PQC) which has significantly larger resource requirements than classic asymmetric schemes. Protections against side-channels increase these requirements even further. Energy-efficient cryptography has also not been a major focus of the research community, yet.
- **Novel technologies** like RRAM- and SRAM-based CIM introduce new possibilities to optimize computing in various aspects. At the same time, they open new attack vectors, in particular if sensitive data is processed. Consequently, attacks and countermeasures against side-channels need to be adapted to and re-evaluated for these new technologies. In some cases, new defenses need to be developed as conventional techniques do not suffice.
- Combining **real-time constraints** and **Trusted Execution Environments (TEEs)** is non-trivial. There are existing solutions like FreeRTOS, CompSOC, ARM TrustZone, and Keystone, but they are either real-time systems or TEEs. While both offer isolation mechanisms and share common traits, real-time systems and TEEs cannot be trivially combined. Nesting a TEE inside a real-time system, breaks the security guarantees of the TEE. Conversely, nesting a real-time system inside a TEE breaks any real-time guarantees, as the TEE may (unintentionally) inhibit the scheduling. A customized solution is therefore required.
- Existing **TEE** solutions do not feature post-quantum secure cryptographic primitives. Switching is non-trivial as PQC schemes are significantly larger, slower and less deterministic, i.e., use rejection-sampling internally.

III. Security Solutions in CONVOLVE

To tackle the aforementioned challenges, we are pursuing a multitude of interdependent solutions. They will eventually be integrated into suitable demonstrators.

A. HADES

1) Overview: With HADES [9], we have developed a tool to perform automated DSEs of hardware implementations of cryptographic schemes. Even more, an arbitrary masking order can be specified to only consider side-channel secure implementations for use-cases with physical attackers if desired. The idea is to avoid "intuitive, but arbitrary" choices made by implementers based on experience or intuition. Instead, we systematically traverse thousands (and even millions) of different designs and rank them based on the specified optimization target, e.g., minimal area or latency. We achieve this by leaving design choices for subcomponents generic in so-called *templates* (more details are provided in the next paragraph). The templates are written in SpinalHDL and are provided as a library together with HADES. In case new templates are required to realize an implementation, these templates need to be provided by the user. However, HADES significantly decreases the design time of masked hardware implementations. It also simplifies side-channel protection via masking, as any arbitrary design can automatically be masked at any masking order without additional implementation effort.

2) Templates and DSE: The templates abstractly describe the cryptographic primitives or subroutines thereof with placeholders for nested components such as adders or masked gadgets. Templates can be nested as needed and a user need only be concerned with the interface of a template. The internal implementation is used to generate the final design.

To perform a DSE, HADES considers several performance metrics such as cycle count, latency, area, or, in the case of masked implementations, randomness requirements. For trade-offs, HADES also considers common combinations such as the area-latency-product. In future work, this could even be extended to power consumption, given that the relevant data sets are available.

More precisely, each template must provide a customized performance prediction which may depend on the performance of sub-templates. The prediction is done based on one or more

TABLE I
RUNTIME OF EXHAUSTIVE DSE FOR DIFFERENT NUMBERS OF EXPLORED CONFIGURATIONS IN DIFFERENT ALGORITHMS.

Algorithm	# Configurations	Time
Keccak	14	0.5 s
AdderModQ	42	0.7 s
Sparse Polynomial Multiplication	372	1.2 s
ChaCha20	1080	3.2 s
AES	1440	5.4 s
Polynomial Multiplication	1302	7.9 s
Kyber-CPA	40 362	196.5 s
Kyber-CCA	1 148 364	36 h

performance metrics. The tool then proceeds to build up an internal tree structure with all possible instantiations of all templates. Since the number of combinations grows rapidly and the optimization target is not necessarily attainable via a greedy search, HADES offers two options. The naive approach traverses the design space exhaustively and obtains provably optimal results. The smarter approach employs a heuristic strategy called *local search*. Initially, a performance baseline is established for a random, probably non-optimal instantiation. Then all parameters are varied individually instead of jointly to identify local performance optima.

In any way, the individual performance predictions in the tree can be folded bottom-up with the optimal instantiation(s) for each template. Eventually, this yields a small set of implementations optimized towards one or more optimization goals.

3) Performance: HADES is a powerful, efficient, and practically relevant tool. We obtain the first arbitrary-order masked implementation of CRYSTALs-Kyber [10]. As Table I shows, several thousand possible configurations can be explored in a few seconds or minutes. For a Chosen Ciphertext Attack (CCA)-secure implementation of Kyber more than 1.1 million designs can be explored exhaustively in 36 h. The heuristic strategy finds an optimized Kyber in less than 200 s. This outperforms any manual analysis and optimization efforts and decimates the design time.

Our evaluations also show that we can produce smaller and faster designs than other tools. For instance, HADES produces adders which outperform those generated with AGEMA [11], which applies straight-forward post-processing to synthesized netlists. Further, the accuracy of our heuristic approach depends on how many starting points we choose. In practice, we obtain perfect results for Kyber-CCA for as few as 50 random performance base-lines.

For CONVOLVE, we are specifically interested in AES-256 as the algorithm for payload encryption. Table II shows the performance metrics for different designs and optimization goals.

In CONVOLVE, we also realize Keccak in hardware as it is an important subroutine of BIKE [12], CRYSTALs-Dilithium and can be used by the TEE for signing as well. The corresponding case study can be found in the original HADES paper [9]. We refrain from implementing full PQC schemes in hardware since they easily exceed the entire SoC in size. This also gives more flexibility for future adjustment, as the

TABLE II
PERFORMANCE METRICS FOR DIFFERENT AES-256 DESIGNS BY OPTIMIZATION GOALS (**L**ATENCY, **A**REA, **R**ANDOMNESS, **P**RODUCT) AND MASKING ORDER d.

d	Opt.	Area [kGE]	Rand. [bits]	Lat. [cc]
0	L/ALP	41.4	0	19
	A	12.9	0	1378
1	L	1205.3	16 200	71
	A	29.9	144	2948
	R/ALRP	32.2	68	4514
	ALP	142.8	1224	75
2	L	2321.1	48 588	71
	A	49.1	408	2946
	R/ALRP	58.2	204	4514
	ALP	252.7	3660	75

standardization is still on-going and PQC may unexpectedly be broken.

Contribution to CONVOLVE: In summary, HADES is a powerful tool that contributes significantly to a ten-fold decrease in design time. We also obtain side-channel protected hardware implementations for AES-256, which enables post-quantum (i.e. long-term) encryptions of sensitive data. Eventually, HADES flexible optimization goals nicely caters to the restricted resources for edge-devices.

B. Post-Quantum TEE and Keystone

Trusted Execution Environments (TEE) aim at providing additional security by isolating high-risk software inside so called enclaves from untrusted code. An example for such high-risk software could be the cryptographic components that ensure the confidentiality and integrity of an ML model. A typical operating system (OS) in this example would be untrusted, as OS are complex, often involving millions of lines of code which are almost certain to contain bugs. Thanks to the hardware enforced isolation, a TEE would ensure that even the OS could not access the secrets inside an enclave.

In order to experiment with a PQC-enabled TEE, we selected the Keystone Framework [13], as its open nature enables customization and extensions. Open-source frameworks such as Keystone aim at providing secure, reliable, and yet still flexible TEE implementations for platforms such as RISC-V edge processors. The TEE can provide proof that the environment is secure and has not been tampered by delivering cryptographical proof to a third party, a so-called remote attestation. This process relies also on a per-device unique secret, e.g. stored in a root-of-trust. The TEE can also provide data sealing, where data is encrypted in such a way so that only a specific enclave, identified by its hash value, running on a specific device and a specific Keystone implementation can decrypt the data. This can be used to ensure that only a genuine, uncompromised devices get access to sensitive data such as model weights or other sensitive data, and even then the data is restricted to an enclave.

Keystone achieves the hardware isolation via the RISC-V physical memory protection (PMP) registers [13], [14]. These registers allow the security monitor (SM), running in M mode, to block access to memory regions from other software, including the OS. This allows the SM to set up a TEE to run enclaves in a secure way, which are isolated both from the OS as well as from each other.

As a hardware evaluation target, we selected a RISC-V SoC built with the Chipyard framework [15], running on a Xilinx VCU118 FPGA. There, we configured an SoC using four Rocket cores [16] (with the PMP registers enabled), an L2 cache and 2GB of external DRAM. To support Keystone, we modified the SoC bootrom to perform a measurement of the the SM located in DRAM, sign the measurement hash with a unique device key currently stored in the bootrom, and derive key material for the SM to use for its own signing operations, such as attestation reports.

By default, Keystone use SHA3 for the measurements, and Ed25519 as a signature algorithm. To enable post-quantum security, we additionally add ML-DSA [17], also known as CRYSTALs-Dilithium [18], as a signature scheme. This hybrid approach ensures that the security is always at least as that of Ed25519, while also ensuring long-term security from quantum attackers. This requires modifying several parts of Keystone:

- The signing of the SM measurement hash during boot using both Ed25519 and ML-DSA.
- The derivation of a SM Ed25519 and ML-DSA key pair from the unique device Ed25519 and ML-DSA key.
- The signing of attestation reports of enclaves uses both signature schemes.
- The data sealing keys are derived from both the Ed25519 and ML-DSA SM secret keys. The derived key is then also signed with both signature schemes.

We use the PQClean implementation of ML-DSA [19]. We encounter several practical problems to achieve a functional implementation: The additional code size of ML-DSA, together with the large size of the ML-DSA private key, considerably increase the size of the bootrom. This is undesirable, as the on-chip memory is generally comparatively expensive. We mitigate this by storing the ML-DSA key as 32-byte seed, and deterministically regenerate the key during boot.

A second issue is that the SM by default only reserves 8KB of stack memory per core for operations such the attestation. However, the implementation of ML-DSA requires more memory during signing, corrupting the stack. One solution would be to optimize the RAM usage of ML-DSA, such as in [20], to stay below the 8KB bound. As a stopgap, we increase the stack size per core to 128KB.

With these changes, we are able to successfully boot our device, observe that the SM is running and has set the PMP registers to enforce the isolation, as well as run a demonstrator enclave that succeeds in generating a signed attestation report. We are then able to verify this attestation report using the Ed25519 and ML-DSA public device key. Our next steps is to perform additional benchmarking, as well as to make use of the hardware accelerators in Section III-A.

Contribution to CONVOLVE: A TEE is a fundamental yet versatile component of CONVOLVE's security architecture. It can protect sensitive data at rest, isolate potentially malicious software and, by means of remote attestation, guarantee that devices have not been tampered with. By integrating PQC in Keystone, the system becomes long-term secure and hardware accelerated cryptographic subroutines contribute towards lowering the energy consumption by 100x.

TABLE III
A COMPARISON OF THE KEYSTONE DEFAULTS WITH OUR PQ-ENABLED MODIFICATIONS.

Component	Keystone default	PQ-enabled Keystone
Bootrom size	50.7 KB	60.2 KB
Signature algorithms	Ed25519	Ed25519 & ML-DSA-44
Attestation report size	1320 Byte	7472 Byte
SM stack size per core	8 KB	128 KB

C. CIM Security

Training neural networks (NNs) is time-consuming and resource-intensive. Moreover, trained models represent valuable intellectual property (IP) that can be compromised through power side-channel attacks. These attacks exploit hardware vulnerabilities to extract sensitive information, such as NN weights and other parameters [21]. CIM architectures are gaining popularity as NN accelerators due to their increased power efficiency and high parallelism [22]. And just like conventional processors, these CIM architectures are also vulnerable to side-channel attacks [23].

For CONVOLVE, we proposed a machine learning (ML) based novel attack method to extract weights from digital CIM accelerator, and further validate the attack method successfully using gate level implementation of CIM macro at 40nm technology [23]. A typical digital CIM macro is considered for side-channel evaluation where bit-wise multiplication is performed between inputs and weights, and the result is provided to an adder-tree which subsequently accumulates the products of all inputs and weights in a multiply-and-accumulate (MAC) accumulator. Through careful inspection of the MAC operation, it is observed that the switching activity of the accumulator can be confined to the desired level through input manipulation. This allows us to calculate the Hamming Weight (HW) of the accumulator register by monitoring power consumption of the operation.

1) Attack Method: Our method involves selective inclusion or exclusion of 4-bit weights in the accumulation process by providing binary input values as masks. This enables the selective addressing of 4-bit weights stored in the SRAM, thereby controlling their participation in the addition process. Our strategy is divided into two phases, in the initial phase, weights are categorized into 5 clusters using k-means clustering. The criteria for clustering is the HW of the weight, which correlates with the switching activity and power consumption of the MAC operation. By manipulating input values and employing k-means, we accurately determine the HW of each weight, resulting in clusters representing HWs 0 to 4.

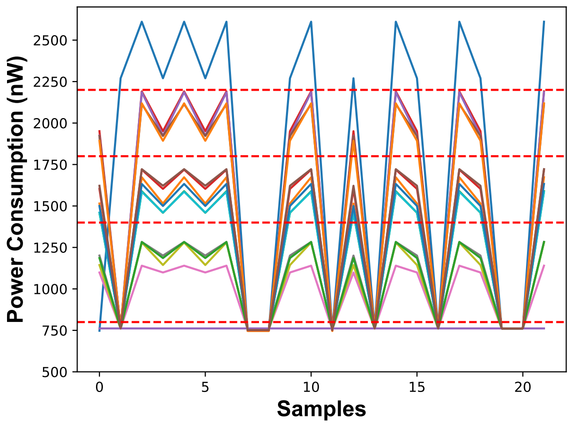

Fig. 1. First Phase: Clustering Results

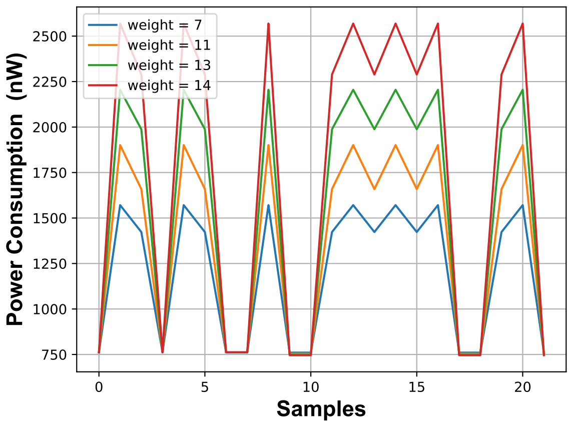

Fig. 2. Second Phase: HW=3 Results

The second phase utilizes weights with HW 4 (value 15) to deduce unknown weight values. By combining known weights with unknown weights of the same HW category, unique HW values are obtained. This iterative process, optimized through exhaustive search, minimizes additions and enables the determination of all unknown weights.

2) Experimental Results: To evaluate the security of the SRAM-based CIM design against power side-channel attacks, we utilized a comprehensive tool-chain. Cadence Genus was employed for logic synthesis, Questasim for functional simulation, and Synopsys Spyglass for power analysis. The adder tree was synthesized in TSMC 40 nm technology for 4-bit weight values. The attacks were implemented in Python, leveraging the capabilities of scikit-learn.

In the first phase of the attack, we examined the power traces of each weight individually by activating them one at a time. As shown in Figure 1, a clear correlation between the HW of a weight and its power consumption pattern during adder tree operations emerges. The k-means clustering algorithm successfully grouped these power traces into distinct clusters, demonstrating significant differences between weight categories based on their HW. Once we determine the Hamming weight (HW) of each weight in Phase 1, the next step is to identify their exact values. We begin by analyzing weights with extreme HWs (0 and 4), whose values are known. For weights with HW 1, we simultaneously activate an unknown weight and a known weight with a value of 15, generating unique power patterns for each possible value of the unknown weight. This approach is applied to other HW values as well. As shown in Figure 2, the power consumption of the adder tree for unknown weights with HW 3 (values 7, 11, 13, and 14) is distinct when activated with and without a known weight of value 1. This clearly demonstrates the vulnerability of these power patterns to attacks, even in noise-free environments.

Contribution to CONVOLVE: CIM architectures are a vital step towards CONVOLVE energy efficiency goals. At the same time they introduce new attack targets, i.e. the NN model parameters and their inputs. To protect this IP, side-channel attacks and counter-measures must be meticulously analyzed and integrated to enable adoption in industry.

D. *FreeRTOS + TEE*

System-on-Chip (SoC) designs are fundamental across various technological applications, including automotive systems, mobile devices, and data centers. These advanced SoCs integrate multiple cores processor, a hierarchical cache system, and numerous subsystems that share memory and resources, which enhances functionality but also introduces potential security vulnerabilities. Understanding these risks, researchers have prioritized enhancing the security on SOC platforms. A notable example is FreeRTOS, a popular operating system for embedded systems, frequently used in SoC platform. Given the widespread use of FreeRTOS, several efforts have been made to enhance its security. One such effort is FreeRTOS's own release of a more secure version with Memory Protection Unit (MPU) support [24]. An MPU is a hardware unit within the microprocessor that can be configured to protect memory regions from being accessed. This addresses the security weakness of the flat memory model. The FreeRTOS changelog [25] , the first official MPU version of FreeRTOS, was actually already released back in 2009 for the ARM Cortex M3 family. However, for the most part, there were only minor updates and not many platforms were supported. Moreover, until 2019 the implementation contained some serious flaws that made it easy to circumvent the security features [26]. Since 2020, however, significant enhancements have been made, with a stronger focus on improving the MPU version's robustness—though official MPU support remains exclusive to Arm platforms [24]. For non-Arm platforms, SiFive attempted to create a secure FreeRTOS version for RISC-V, leveraging RISC-V's Physical Memory Protection (PMP) feature, which is functionally similar to an MPU. However, this implementation was minimal, only protecting the task stack and placing task code in an unprivileged area without inter-task protection. Despite these efforts, the SiFive repository was archived and is no longer maintained, lacking updates that reflect post-2020 security advancements.

We implemented an improved version of the FreeRTOS operating system that integrates the PMP functionality as shown in the Figure 3. The modified FreeRTOS evaluated on a System-on-a-Chip (SoC) that is built upon the RISC-V architecture. Di-

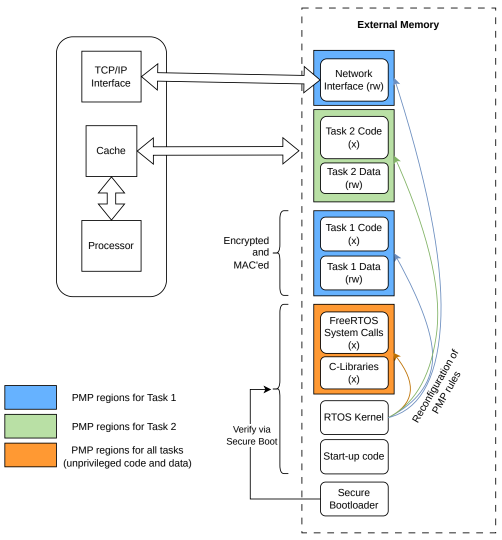

Fig. 3. Enhancing FreeRTOS Security on RISC-V Architecture with Physical Memory Protection (PMP)

verse attack scenarios utilized to evaluate the system's capacity to endure and recuperate from these attacks.

Contribution to CONVOLVE: Enhancing FreeRTOS with TEE-like capabilities extends the security-framework to be applicable for use-cases with real-time constraints. At the same time, it offers IP protection and isolation mechanisms to confine software bugs and increase the longevity of CONVOLVE results beyond currently known vulnerabilities and exploits.

E. Composability

Edge processing applications that CONVOLVE aims to facilitate often share limited hardware resources. The scheduling of these shared resources must guarantee responsiveness for real time applications while also accommodating other applications of various time criticality. Real Time Operating Systems (RTOS), such as FreeRTOS, may provide predictable execution of time critical applications, however to share hardware resources for real and mixed time critical applications a composable framework is required. This composable framework protects applications from interference from other applications on the shared resources providing execution time guarantees.

Research into composable execution designs for the CONVOLVE project focuses on the State-of-the-Art composable platform CompSOC [27]. The CompSOC platform already provides a proven composable and predictable execution platform. A key concept of the CompSOC platform is the Virtual Execution Environment (VEP) that creates a predefined subset of hardware that isolates a user application from all other applications on the shared hardware. The VEP design inherently provides security in a similar way to a TEE as all resources are protected from interference, however formal security analysis is yet to be conducted.

Composable implementations simplify verification, as applications can be verified independently, as opposed to being verified together. Since tasks executed in composable architectures are protected from interference with other applications through isolation within VEPs, verification for each application can be done in isolation. This solves a significant problem for SoC implementation of applications with mixed criticality. These applications would otherwise need to be verified as a whole, preventing run-time composition of applications.

The CONVOLVE project implements PULP [28] — an open source, ultra-low power RISC-V platform. As yet, there is no variant of PULP that implements composable design concepts. The Carfield platform, based on the Cheshire variant of PULP [29] does offer a predictable platform aimed at automotive applications that require time-predictable execution. The Carfield platform is being investigated for the possibility of integration with composable execution that extends the predictable and safe execution model provided by Carfield.

Research tasks for CONVOLVE include an investigation of security features required for CompSOC to to provide a composable platform that is secure. In addition to the isolation of applications on VEPs, a root of trust must be established and security features for signing and encryption implemented at the user and system level. These security features are required for use cases where applications need to transmit information between the composable VEPs and a third party or for software updates at the application or system level. Investigation is ongoing for how Carfield might be extended for composable execution designs.

A drawback of composable execution the additional processing overhead. To showcase composable execution, a separate demonstrator is planned for research outcomes that will show optional features or design concepts that can be included with the CONVOLVE project where composable execution is required. A key area of research is to find efficient methods for implementing composable concepts in the context of CONVOLVE and the provided use cases.

Contribution to CONVOLVE: Our investigations regarding CompSOC and Carfield have similar contributions to CONVOLVE as extending FreeRTOS with PMP: IP protection, software isolation, real-time guarantees and a modular, customizable security framework.

IV. Summary

The CONVOLVE project aims at boosting edge-AI processors made in Europe by decimating the design time and energy consumption of the SoC. The built-in security architecture is modular and composable and it is crucial to enable adoption and ensure a long-lasting impact even in face of powerful future adversaries. At the same time the security architecture is challenging and inspires new research: energy efficiency has been neglected in cryptographic research over the past decades and PQC makes it even worse; TEEs and real-time systems, while plenty and versatile, have not been investigated jointly; and side-channel attacks and countermeasures need to be adapted and extended to new CIM technologies to keep IP safe.

Eventually, the entire security architecture will be practically demonstrated on FPGAs. This ensures the results indeed contribute towards placing the EU in a prime position to become a world-wide leader in AI-powered smart edge devices.

References

[1] "Edge computing market size to reach $155.90 billion by 2023," https://www.grandviewresearch.com/press-release/global-edge-computing-market, accessed: 24-10-28.

[2] "CONVOLVE Project Website," https://convolve.eu/, accessed: 2024-10-25.

[3] L. Mei, P. Houshmand, V. Jain, J. S. P. Giraldo, and M. Verhelst, "Zigzag: Enlarging joint architecture-mapping design space exploration for DNN accelerators," *IEEE Trans. Computers*, vol. 70, no. 8, pp. 1160–1174, 2021. [Online]. Available: https://doi.org/10.1109/TC.2021.3059962

[4] A. Symons, L. Mei, S. Colleman, P. Houshmand, S. Karl, and M. Verhelst, "Towards heterogeneous multi-core accelerators exploiting fine-grained scheduling of layer-fused deep neural networks," *CoRR*, vol. abs/2212.10612, 2022. [Online]. Available: https://doi.org/10.48550/arXiv.2212.10612

[5] P. C. Kocher, "Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems," in *Advances in Cryptology - CRYPTO '96, 16th Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 1996, Proceedings*, ser. Lecture Notes in Computer Science, N. Koblitz, Ed., vol. 1109. Springer, 1996, pp. 104–113. [Online]. Available: https://doi.org/10.1007/3-540-68697-5_9

[6] P. C. Kocher, J. Jaffe, and B. Jun, "Differential Power Analysis," in *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*, ser. Lecture Notes in Computer Science, M. J. Wiener, Ed., vol. 1666. Springer, 1999, pp. 388–397. [Online]. Available: https://doi.org/10.1007/3-540-48405-1_25

[7] K. Gandolfi, C. Mourtel, and F. Olivier, "Electromagnetic Analysis: Concrete Results," in *Cryptographic Hardware and Embedded Systems - CHES 2001, Third International Workshop, Paris, France, May 14-16, 2001, Proceedings*, ser. Lecture Notes in Computer Science, Ç. K. Koç, D. Naccache, and C. Paar, Eds., vol. 2162. Springer, 2001, pp. 251–261. [Online]. Available: https://doi.org/10.1007/3-540-44709-1_21

[8] E. Biham and A. Shamir, "Differential Fault Analysis of Secret Key Cryptosystems," in *Advances in Cryptology - CRYPTO '97, 17th Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 1997, Proceedings*, ser. Lecture Notes in Computer Science, B. S. K. Jr., Ed., vol. 1294. Springer, 1997, pp. 513–525. [Online]. Available: https://doi.org/10.1007/BFb0052259

[9] F. Buschkowski, G. Land, J. Richter-Brockmann, P. Sasdrich, and T. Güneysu, "HADES: Automated Hardware Design Exploration for Cryptographic Primitives," Cryptology ePrint Archive, Paper 2024/130, 2024. [Online]. Available: https://eprint.iacr.org/2024/130

[10] J. W. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J. M. Schanck, P. Schwabe, G. Seiler, and D. Stehlé, "CRYSTALS - kyber: A cca-secure module-lattice-based KEM," in *2018 IEEE European Symposium on Security and Privacy, EuroS&P 2018, London, United Kingdom, April 24-26, 2018*. IEEE, 2018, pp. 353–367. [Online]. Available: https://doi.org/10.1109/EuroSP.2018.00032

[11] D. Knichel, A. Moradi, N. Müller, and P. Sasdrich, "Automated Generation of Masked Hardware," *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, vol. 2022, no. 1, pp. 589–629, 2022. [Online]. Available: https://doi.org/10.46586/tches.v2022.i1.589-629

[12] N. Aragon, P. Barreto, S. Bettaieb, L. Bidoux, O. Blazy, J.-C. Deneuville, P. Gaborit, S. Ghosh, S. Gueron, T. Güneysu *et al.*, "Bike: bit flipping key encapsulation," 2022.

[13] D. Lee, D. Kohlbrenner, S. Shinde, K. Asanovic, and D. Song, "Keystone: An open framework for architecting trusted execution environments," in *Proceedings of the Fifteenth European Conference on Computer Systems*, ser. EuroSys'20, 2020.

[14] A. Waterman, K. Asanovic, and J. Hauser, "The risc-v instruction set manual, volume ii: Privileged architecture," *RISC-V Foundation*, pp. 1–4, 2019.

[15] A. Amid, D. Biancolin, A. Gonzalez, D. Grubb, S. Karandikar, H. Liew, A. Magyar, H. Mao, A. Ou, N. Pemberton, P. Rigge, C. Schmidt, J. Wright, J. Zhao, Y. S. Shao, K. Asanović, and B. Nikolić, "Chipyard: Integrated design, simulation, and implementation framework for custom socs," *IEEE Micro*, vol. 40, no. 4, pp. 10–21, 2020.

[16] K. Asanović, R. Avizienis, J. Bachrach, S. Beamer, D. Biancolin, C. Celio, H. Cook, D. Dabbelt, J. Hauser, A. Izraelevitz, S. Karandikar, B. Keller, D. Kim, J. Koenig, Y. Lee, E. Love, M. Maas, A. Magyar, H. Mao, M. Moreto, A. Ou, D. A. Patterson, B. Richards, C. Schmidt, S. Twigg, H. Vo, and A. Waterman, "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

[17] National Institute of Standards and Technology, "Module-lattice-based digital signature standard," U.S. Department of Commerce, Washington, D.C., Tech. Rep. Federal Information Processing Standards Publications (FIPS PUBS) 204, August 13, 2024, 2024.

[18] L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, P. Schwabe, G. Seiler, and D. Stehlé, "Crystals-dilithium: A lattice-based digital signature scheme," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pp. 238–268, 2018.

[19] M. J. Kannwischer, P. Schwabe, D. Stebila, and T. Wiggers, "Improving software quality in cryptography standardization projects," in *IEEE European Symposium on Security and Privacy, EuroS&P 2022 - Workshops, Genoa, Italy, June 6-10, 2022*. Los Alamitos, CA, USA: IEEE Computer Society, 2022, pp. 19–30. [Online]. Available: https://eprint.iacr.org/2022/337

[20] J. W. Bos, J. Renes, and A. Sprenkels, "Dilithium for memory constrained devices," in *Progress in Cryptology - AFRICACRYPT 2022*, L. Batina and J. Daemen, Eds. Cham: Springer Nature Switzerland, 2022, pp. 217–235.

[21] M. M. Real and R. Salvador, "Physical side-channel attacks on embedded neural networks: A survey," *Applied Sciences (Switzerland)*, vol. 11, no. 15, 8 2021.

[22] S. Hamdioui, L. Xie, A. N. H. Anh, M. Taouil, K. Bertels, H. Corporaal, H. Jiao, F. Catthoor, D. Wouters, L. Eike, and J. v. Lunteren, "Memrisor Based Computation-in-Memory Architecture for Data-Intensive Applications," in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2015*. New Jersey: IEEE Conference Publications, 2015, pp. 1718–1725.

[23] F. J. Mir, A. Aljuffri, S. Hamdioui, and M. Taouil, "Extracting weights of cim-based neural networks through power analysis of adder-trees," 2024, pp. 1–4.

[24] FreeRTOS, "Memory protection unit (mpu) support," 2024. [Online]. Available: https://www.freertos.org/Security/04-FreeRTOS-MPU-memory-protection-unit#upgrade-information

[25] ——, "Freertos-kernel/history.txt," 2024. [Online]. Available: https://github.com/FreeRTOS/FreeRTOS-Kernel/blob/main/History.txt

[26] W. Zhou, L. Guan, P. Liu, and Y. Zhang, "Good motive but bad design: Why arm mpu has become an outcast in embedded systems," 08 2019.

[27] K. Goossens, M. Koedam, A. Nelson, S. Sinha, S. Goossens, Y. Li, G. Breaban, R. van Kampenhout, R. Tavakoli, J. Valencia, H. Ahmadi Balef, B. Akesson, S. Stuijk, M. Geilen, D. Goswami, and M. Nabi, "NOC-based multi-processor architecture for mixed time-criticality applications," in *Handbook of Hardware/Software Codesign*, S. Ha and J. Teich, Eds. Springer, 2017, pp. 491–530.

[28] F. Conti, D. Rossi, A. Pullini, I. Loi, and L. Benini, "Pulp: A ultra-low power parallel accelerator for energy-efficient and flexible embedded vision," *J. Signal Process. Syst.*, vol. 84, no. 3, p. 339–354, Sep. 2016. [Online]. Available: https://doi.org/10.1007/s11265-015-1070-9

[29] 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.