Securing RISC-V Micro-architecture cores for High-security Applications in IoT against Power Side-Channel Attacks

A
thesis submitted
for the award of degree of

DOCTOR OF PHILOSOPHY

Submitted by

Titu Mary Ignatius

Under the supervision of

Prof. Roy Paily Palathinkal

Department of Electronics and Electrical Engineering
Indian Institute of Technology, Guwahati
Assam, India - 781039
December, 2024

Dedicated to
"My Mother" ,
whom I lost during the course of my PhD journey.
This accomplishment is for her, as she always dreamed of seeing this day. Her love and wishes remain my strength.

DECLARATION

*This is to certify that the thesis entitled "**Securing RISC-V Micro-architecture cores for High-security Applications in IoT against Power Side-Channel Attacks**", submitted by me to the Indian Institute of Technology, Guwahati for the award of the degree of **Doctor of Philosophy** is a bonafide work carried out by me under the esteemed supervision of **Prof. Roy Paily Palathinkal** . The contents of this thesis, in full or in parts, have not been submitted to any other institute or university for the award of any degree or diploma.*

Signed: 

Titu Mary Ignatius
Department of Electronics and Electrical Engineering
Indian Institute of Technology, Guwahati
Assam, India - 781039

Date: 25-12-24

CERTIFICATE

*This is to certify that the thesis entitled "**Securing RISC-V Micro-architecture cores for High-security Applications in IoT against Power Side-Channel Attacks**", submitted by **Titu Mary Ignatius (206102028)**, a Research Scholar in the Department of Electronics and Electrical Engineering, Indian Institute of Technology, Guwahati, for the award of the degree of **Doctor of Philosophy**, is an original research work carried out by her under my supervision and guidance. The thesis has fulfilled all the requirements as per the regulations of the institute and, in my opinion, has reached the standard needed for submission. The contents of this thesis, in full or in parts, have not been submitted to any other institute or university for the award of any degree or diploma.*

Signed: _____

Prof. Roy Paily Palathinkal
Department of Electronics and Electrical Engineering
Indian Institute of Technology, Guwahati
Assam, India - 781039

Date: 25-12-24

Acknowledgement

First of all, I would like to thank God Almighty, whose blessings made it possible for me to complete my PhD.

I extend my heartfelt gratitude to my husband, who stood by me throughout this journey. He was my greatest strength for completing my PhD and helping me overcome the immense loss of my mother and grandmother during my PhD. My son, who adjusted and studied independently, made it easier for me to focus on my work. I am also deeply thankful to my grandmother for her constant prayers. I am grateful to my parents, brother, and sister for their love and support throughout this journey.

I want to express my deepest and most sincere gratitude to my supervisor, Prof. Roy Paily Palathinkal, for his fundamental role in my doctoral work. Despite his busy schedule, he always guided me in the correct direction. I am especially thankful for his continuous support and patience towards me, and also for providing all the facilities I needed to carry out my research. The most striking feature during my time with him is his ability to understand his students' situations and provide a proper guidance. His sincerity and dedication to teaching are qualities, which I would like to follow when stepping into this profession.

I am very thankful to my doctoral committee members, Prof. Shaik Rafi Ahamed, Dr. Sonali Chouhan and Dr John Jose, for sparing their precious time to evaluate the progress of my work and providing valuable suggestions.

A special thanks to our lab in charge, Josephine Ma'am, for her constant support both personally and professionally. She was always there to listen to my problems and made me feel like a family. Her guidance and solutions during tough times were extremely valuable.

On the work front, I would like to thank my seniors, Dr. Thockchom Birjit Singha and Dr. Satyajit Bora, for their constant support and constructive feedback on my work. I also extend my thanks to Sampriti Ghosh, Anupam Kumari, Aditi Paul, Aggala Venakata Sai, Angara Aparna, Guljar Ansari, and Firoz Shah for the wonderful time we shared during our work together. I would also like to thank my lab mates, Emilin Ma'am, Arun Sir, Surabhi Mishra, Praveen Saraswat, Uday Maurya, Rahul Sharma, Ashagrie M. Kemie, Shiva Prasad, Radha rani and Tsiiveikho Saphou, for making the lab a happy space to work.

I am also thankful to all my friends at IIT Guwahati, Divya, Sivakumar, Feba, Ashwani, and Roshini Sister, who have been like family to us. Lastly, I would like to thank all my friends, family members, and teachers who have supported me throughout my life.

Guwahati *Titu Mary Ignatius*
December, 2024

Abstract

The rapid growth of IoT edge (IoTe) devices has created a demand for secure, low-power, and area-efficient embedded processors. While the Advanced Encryption Standard (AES) is commonly used to secure these processors, they remain vulnerable to Power Analysis Attacks (PAA). Owing to the resource constraints in IoTe devices, it is essential to design embedded processors with cryptographic capabilities that provide strong security against power attacks while maintaining low resource consumption, area, and power overheads. This thesis proposes secure 32-bit RISC-V micro-architectures tailored for high-security IoTe applications to meet the requirements of reliable and secure embedded processors for modern IoTe devices.

The design process begins by optimizing the pipeline stages and techniques to develop a compact low-power RISC-V core. Building on this foundation, two RISC-V crypto-cores are developed, each featuring a dedicated AES hardware unit integrated within the RISC-V pipeline and controlled by custom AES instructions. The first core employs a software pause (SW) technique, and the second uses a hardware pause (HW) technique, achieving AES encryption in 23 and 20 clock cycles, compared to 6,953 and 10,914 clock cycles for their baseline cores. The developed RISC-V crypto-cores achieve AES throughputs of 1.25 Gbps and 1.73 Gbps, with clock frequencies limited of 223.71 MHz and 269.54 MHz, respectively. This integration reduces side-channel leakage, enhancing the intrinsic resilience of these RISC-V crypto-cores. To further improve security by obscuring AES power consumption patterns, a novel Non-Linear Cyclic Variable Clock Feistel Bridge-Inspired Countermeasure (NCVCFB) is designed. Operating in tandem with each AES round, it operates additional random rounds in the first and last AES rounds, disrupting PAA-targeted power samples.

Additionally, the thesis explores tightly and loosely coupled AES integration techniques with RISC-V. The tightly coupled approach embeds AES within the RISC-V micro-architecture, extending the ISA for encryption instructions. While this approach offers better security and reduced area, it requires more efforts for integration and requires RISC-V compiler modifications. In contrast, the loosely coupled approach uses an external address-enabled AES accelerator IP, offering greater flexibility. Finally, an ultra-low-power Random Clock Self Complementary (RCSC) countermeasure is introduced. It enhances security by introducing variability in both amplitude and time domains through complementary AES operations and inserting random delays in random rounds of AES, resulting in a variable number of clock cycles for each encryption.

Hardware security metrics, including Measurements To Disclose (MTD), Signal-to-Noise Ratio (SNR), Mutual Information (MI), and Test Vector Leakage Assessment (TVLA), confirm that both the NCVCFB and RCSC countermeasures significantly improve PAA resilience with minimal area and power overheads. The NCVCFB-secured RISC-V core achieved remarkably low area and power overheads of 1.05% and 1.23%, respectively, while maintaining frequency and resisting PAA with 2 million plaintexts, making it ideal for compact, low-power, high-speed IoT applications. The RISC-V core with RCSC-secured AES IP incurs only a 0.407% power overhead and 7.33% hardware cost increase, demonstrating strong PAA resilience with 1 million plaintexts and making it well-suited for ultra-low-power IoT applications. Designs are validated on ASIC using Synopsys and Cadence tools at UMC 65nm technology node and on FPGA using the SASEBO platform.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

AES	Advanced Encryption Standard
ALU	Arithmetic Logic Unit
ASIC	Application Specific Integrated Circuit
CPA	Correlation Power Analysis
CPD	Critical Path Delay
CSR	Control and Status Register
CSV	Comma Separated Variable files
DBP	Dynamic Branch Prediction
DMIPS	Dhrystone MIPS
EDA	Electronic Design Automation
FoM	Figure of Merit
FPGA	Field Programmable Gate Array
FSDB	Fast Signal Data Base file
GE	Gate Equivalent
GPRs	General Purpose Registers
GSa/s	Giga Samples per second
HD	Hamming Distance
HDL	Hardware Description Language
HP	High Performance
HW	Hamming Weight

ID	Instruction Decode
IE	Instruction Execute
IF	Instruction Fetch
IoT	Internet of Things
ISA	Instruction Set Architecture
LFSR	Linear Feedback Shift Register
LUT	Look Up Table
MC	Mix Columns
MI	Mutual Information
MIPS	Millions of Instructions Per Second
MTD	Measurements To Disclose
NCVCFB	Non-linear Cyclic Variable Clock Feistel Bridge-inspired
NIST	National Institute of Standards and Technology
NSFunc	New Substitution Function
PAA	Power Analysis Attack
PRNG	Pseudo Random Number Generator
PSCA	Power Side Channel Attack
RCSC	Random Clock Self-Complimentary
RISC-V	Reduced Instruction Set Computer-V
RW	Register Write
SASEBO	Side-channel Attack Standard Evaluation BOard
S-box	Substitution Box
SCA	Side Channel Attack
SNR	Signal to Noise Ratio
TP	Throughput
TVLA	Test Vector Leakage Assessment
UMC	United Microelectronics Corporation
VCD	Value Change Dump file
VLSI	Very Large Scale Integration

CHAPTER

1

INTRODUCTION

Contents

1.1 Background and Motivation

The rapid growth of the Internet of Things (IoT) has transformed traditional devices into smart systems, ushering in a new era of connected devices capable of communicating, processing, and responding to data with unprecedented scale and precision. From smart homes to healthcare, industrial automation, and beyond, IoT applications now permeate nearly every facet of modern life. Earlier IoT devices relied on cloud computing, where all data processing and storage occurred on centralized cloud servers. However, the exponential growth in global data generation, coupled with the demand for reduced latency and improved performance, has driven a paradigm shift toward IoT edge (IoTe) computing, where data processing and storage occur near to the end user, significantly reducing latency and improving performance by offloading workloads from cloud servers. By 2025, it is projected that 75% of all data will be processed on IoTe devices, emphasizing the need for systems that balance ultra-low power consumption, computational efficiency, and cost-effectiveness. IoTe devices are often battery-powered with limited memory, necessitating the design of highly efficient processors that maximize resource utilization without compromising performance.

Currently, the processor industry is dominated by ARM, widely used in portable devices, and Intel x86, prevalent in personal computers and servers. However, their proprietary nature restricts customization and access for individual hardware designers. To address these limitations and promote innovation in computer architecture, UC Berkeley developed RISC-V [1], an open source and flexible ISA. Among existing processor architectures, the simplicity, compactness, and customizability of RISC-V make it well suited for IoT applications, enabling designers to optimize configurations for low-power, resource-constrained environments.

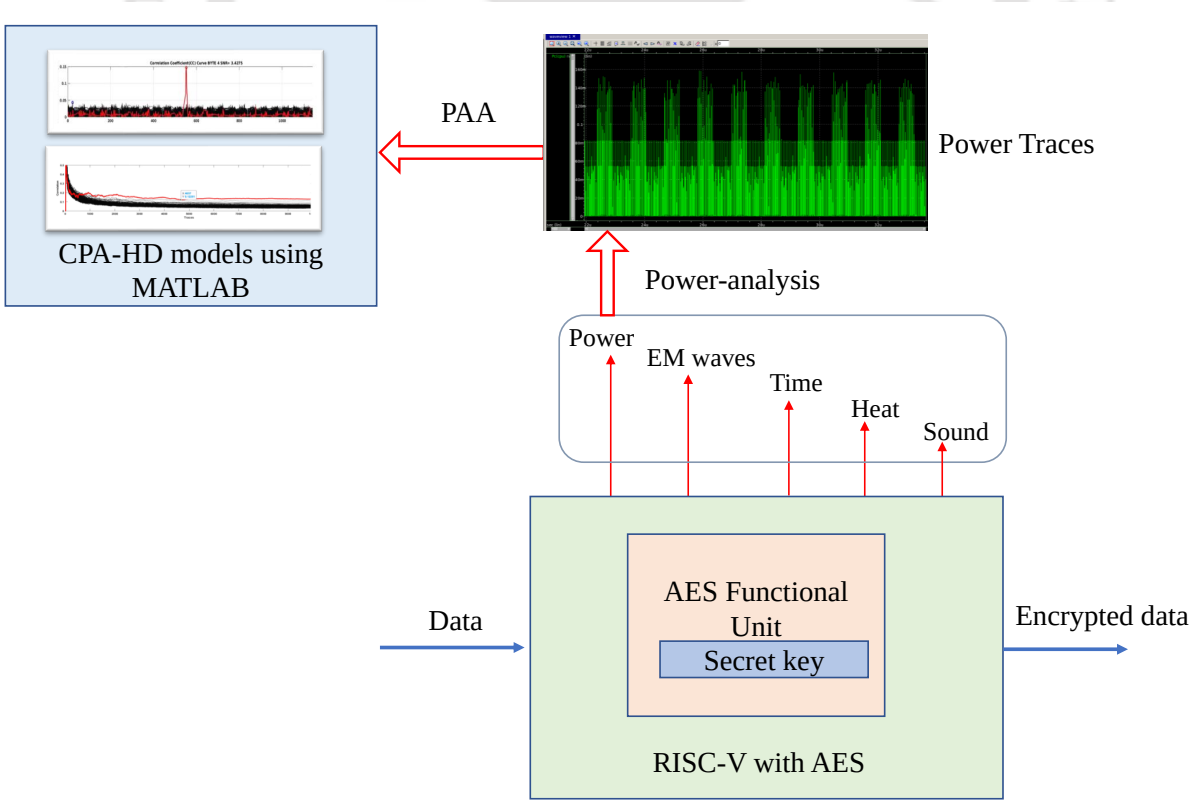


Figure 1.1: Side Channel Attacks

The extensive connectivity of IoT devices introduces new security challenges, as they often handle sensitive information and operate in untrusted environments. Ensuring the security and privacy of these devices has become a paramount concern, particularly against sophisticated attacks targeting the underlying hardware. The customizability of RISC-V enables designers to tailor its microarchitectural features to specific security requirements by incorporating cryptographic capabilities such as Advanced Encryption Standard (AES) [2]. Although RISC-V with cryptographic capabilities offers a solid foundation for IoTe applications, it is also exposed to security risks.

With attackers increasingly targeting IoTe devices, Side Channel Attacks (SCA) have emerged as one of the most serious threats, as shown in Figure 1.1. It exploits unintended information leakage, such as power variations, time, heat, electromagnetic waves etc., by physically accessing these devices during cryptographic operations to extract sensitive data, such as cryptographic keys, thereby compromising system security. IoTe devices are especially vulnerable to Power Analysis Attacks (PAA) due to their limited resources and ease of physical access, making the IoTe security a concern.

As IoT applications become increasingly critical and demand high-security assurances, it is essential to enhance the resilience of RISC-V cores against PAA. Addressing these vulnerabilities will be the key to ensuring security and reliability of IoTe devices in future.

1.2 Problem Statement

Most modern IoTe devices are battery-powered, and operate under stringent power constraints. Since these devices are built around a processor, the power consumption of these processor has to be as low as possible to extend the battery life of the devices. At the same time, the processor should also have good processing speed and robust security. Achieving this balance is vital for modern IoT systems, where seamless synchronization with high-speed networks and robust protection against security threats are of utmost importance. These challenges are addressed by incorporating cryptographic capabilities with these embedded processors.

Despite the widespread use of cryptographic techniques such as Advanced Encryption Standard (AES), Rivest-Shamir-Adleman (RSA) and Secure Hash Algorithm (SHA) for securing embedded processors in IoT environments, many IoTe devices remain vulnerable to security breaches. AES introduced by NIST in 2001, is widely recognized for its robustness, efficiency, and adaptability to resource-constrained environments. As a symmetric encryption algorithm, AES is faster and less resource-intensive than asymmetric techniques like RSA or ECC, making it a cornerstone of IoT communication protocols such as IEEE 802.15.4, ZigBee, Sigfox, LoRaWAN, and Z-Wave. However, software implementations of AES on general purpose embedded processors are resource-intensive and significantly increase the processor's workload requiring thousands of clock cycles, leading to high energy consumption and long encryption times. These limitations have spurred the adoption of AES hardware accelerators, which provide improved performance and lower latency, making them well-suited for real-time IoT applications.

Literature portrays widespread design techniques for implementing AES algorithms on general purpose embedded processors, or integrating AES hardware accelerators predominantly targeting high throughput to conform to today's high-speed communication. Despite advancements in integrating AES with embedded processors, existing designs largely focus on achieving high throughput while neglecting critical aspects like security and efficiency under constrained resources. Moreover, these papers often neglect to focus on the vulnerabilities of

these cores to PAAs. While some studies consider designing secured processor by incorporating countermeasures to mitigate power attacks, these solutions introduce substantial area and power overheads, rendering them unsuitable for resource-constrained IoTe devices.

The limited availability of research analyzing the susceptibility of embedded processors to PAAs during data encryption, as well as the development of secure RISC-V processors to mitigate these attacks, highlights a critical gap in the field. This thesis initially aims to address these gaps by designing compact, low-power, and efficient RISC-V processors with cryptographic capabilities tailored for high-security applications. The study explores various integration techniques for embedding AES hardware with RISC-V cores, analyzing their performance and vulnerability to PAAs. To enhance security further, ultra-low-power compact countermeasures are developed to protect these processors against PAAs, while ensuring minimal area and power overheads, thus meeting the stringent requirements of IoTe devices. The proposed designs undergo rigorous evaluation on ASIC and FPGA platforms, measuring hardware security metrics such as Measurements To Disclose (MTD), Signal-to-Noise Ratio (SNR), Mutual Information (MI), and Test Vector Leakage Assessment (TVLA) to ensure their high resilience against PAA, detailed in chapter 2.

1.3 Overview of the foundational components of the thesis: RISC-V and AES

This section delves into the fundamentals of RISC-V and AES, which are essential for developing RISC-V with cryptographic capabilities for high-security applications, which serve as the foundation of the thesis.

1.3.1 RISC-V

Computer architecture design involves various levels of abstraction, as illustrated in Figure 1.2. Among these, the Instruction Set Architecture (ISA) serves as a crucial interface, which is used by hardware and software designers to develop a processor.

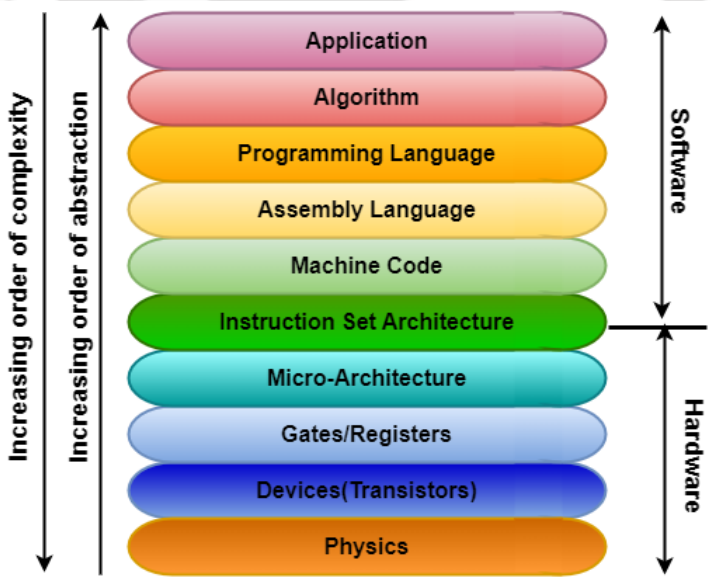


Figure 1.2: Levels of abstraction in computer architecture

RISC-V, originally created for computer architecture research and education, has evolved into a widely adopted free and open ISA for industry. It offers a modular design with a small base integer ISA, suitable for customization and optional standard extensions for general-purpose software development. RISC-V supports both 32-bit and 64-bit address space variants, making it suitable for direct native hardware implementation. Its highly extensible architecture allows developers to define user-specific ISA extensions and create specialized variants tailored to diverse applications. Additionally, optional variable-length instructions enhance performance, reduce code size, and improve energy efficiency. Unlike proprietary ISAs like Intel x86 and ARM, which are complex and often restricted to specific market domains, RISC-V is simpler, more flexible, and adaptable across diverse applications. By offering an open and extensible standard, RISC-V allows developers to innovate freely, making it a versatile choice for modern computing needs.

The RISC-V ISA consists of a mandatory base integer ISA and optional extensions for customization. The base ISA, inspired by early RISC designs, excludes branch delay slots and supports optional variable-length instruction encoding. It provides a minimal set of instructions, essential for compilers, assemblers, linkers, and operating systems, serving as a foundation for customized ISAs. The base ISA comes in two main variants, RV32I and RV64I, supporting 32-bit and 64-bit user-level address spaces respectively, while RV32E, a subset of RV32I, is tailored for micro-controllers. The base integer ISA is referred to as "I" (with RV32 or RV64 prefixes indicating the integer register width) and contains integer computational instructions, integer loads, integer stores and control-flow instructions which is mandatory for all RISC-V implementations. RISC-V supports extensive customization through standard and non-standard instruction-set extensions as shown in Figure 1.3.

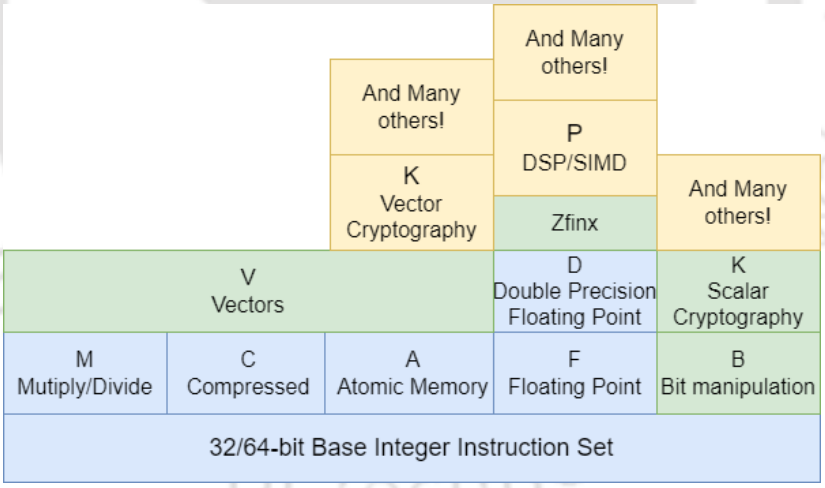


Figure 1.3: Various standard extensions of RISC-V

Standard extensions in RISC-V are designed for general-purpose use, ensuring compatibility with other extensions. Key standard extensions include: "M", which provides integer multiplication and division instructions; "A", enabling atomic instructions for interprocessor synchronization through read-modify-write operations on memory; "F", supporting single-precision floating-point registers, operations, and associated load/store instructions; and "D", expanding on the F extension with double-precision floating-point operations. A general-purpose scalar ISA combining the base integer ISA and these extensions (IMAFD) is denoted as "G" (e.g., RV32G, RV64G). This modular approach of RISC-V enables flexible and scalable processor design for diverse applications. This adaptability positions RISC-V

as a powerful alternative to proprietary ISAs, fostering broad applicability across modern computing domains.

1.3.2 AES

AES is a symmetric block cipher that encrypts 128-bit data blocks, ensuring secure communication between the sender and receiver. It supports three configurations: AES-128, AES-192, and AES-256 based on key lengths of 128, 192, and 256 bits, requiring 10, 12, and 14 rounds of encryption, respectively. This work focuses on AES-128, which uses a 128-bit key. The encryption process begins with a whitening step, where the plaintext is XORed with the secret cipher key, followed by 10 rounds of transformation. Each round consists of four main operations: SubBytes, ShiftRows, MixColumns, and AddRoundKey, with the final round omitting the MixColumns step.

AES transforms plaintext into an unrelatable ciphertext by repeatedly applying round operations. The deeper the plaintext gets into the algorithm, or the more the round operations act upon it, the lesser is its correlation between the intermediate round outputs and the plaintext. The 128-bit plaintext is stored in the form of a 4×4 state matrix with each matrix element representing a byte. The operations in each round are:

1. **SubBytes:** Substitutes each byte in the state matrix by some predetermined values obtained by substitution box (S-box). While lookup table (LUT)-based S-boxes are simple to implement, they demand significant hardware resources, leading to high area and power consumption. Compact S-box designs, based on Galois Field (GF) mathematics, which minimize hardware usage, are preferred for efficient AES implementations.The SubBytes operation is crucial for introducing nonlinearity in AES, enhancing its cryptographic strength.

2. **ShiftRows:** Permutes the state matrix by cyclically shifting rows. The first row remains unchanged, while the second, third, and fourth rows are shifted left by one, two, and three positions, respectively.

3. **MixColumns:** Treats each column of the state matrix as a polynomial over GF(2^8), and multiplies it by a fixed polynomial ($03x^3 + 01x^2 + 01x + 02$), achieving diffusion.

4. **AddRoundKey:** This operation XORs the output of the MixColumns with the corresponding round key generated by the key scheduler, which is derived from the cipher key.

Unique round keys, essential for each round, are generated by key scheduler from the cipher key using a g-function, which performs a 1-byte circular left shift on SubWord (substitution of a word using S-boxes) and xoring with round constant (Rcon). These round keys can be precomputed and stored in memory or generated dynamically during encryption. This iterative process ensures that as encryption progresses, the correlation between intermediate states and the plaintext diminishes, producing completely uncorrelated ciphertext.

1.4 Scope and Research Objectives

This research focuses on designing secure, high-performance RISC-V crypto-cores tailored for IoTe devices as shown in Figure 1.4, to address the aforementioned challenges posed by PAAs, through focusing the following objectives:

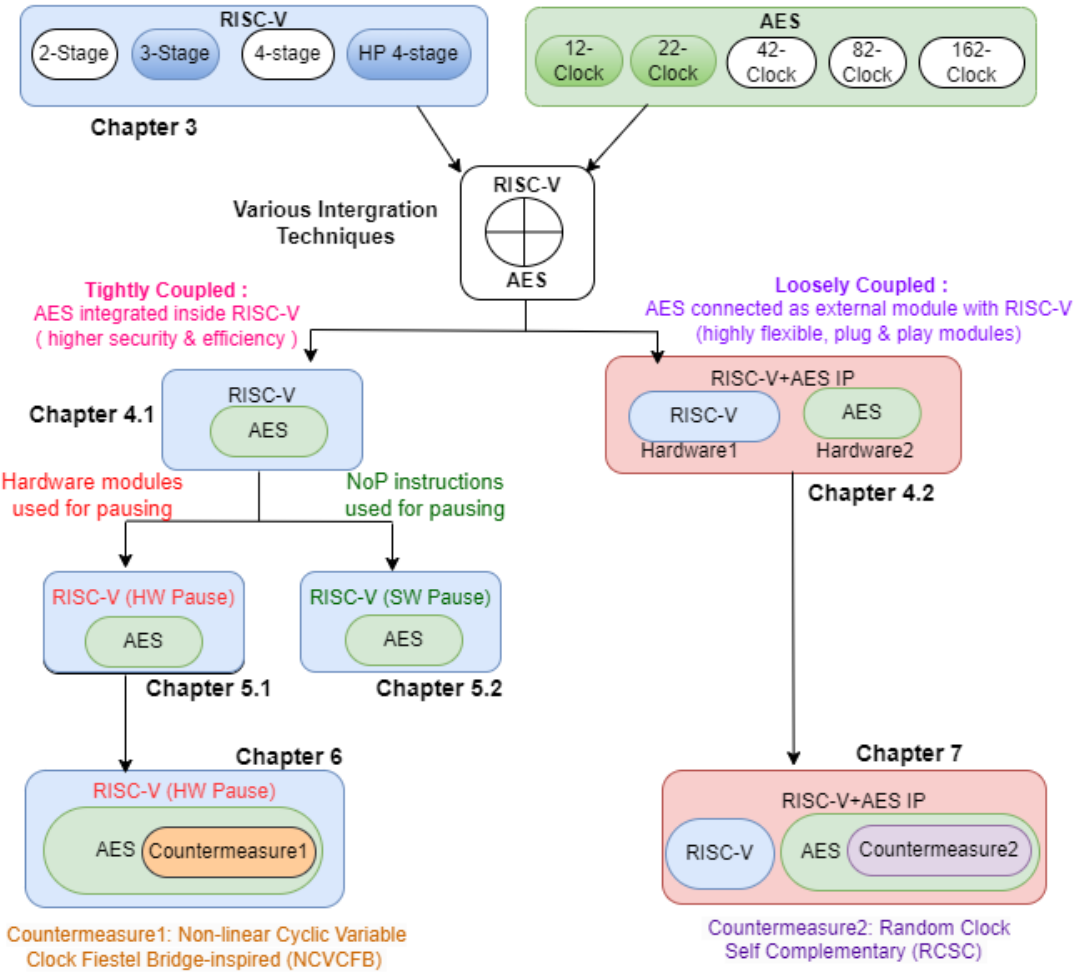


Figure 1.4: Objectives and Scope of the work

1. **RISC-V processor optimization:** Exploring enhancement of RISC-V micro-architectures through pipeline optimizations, advanced branch prediction mechanisms, and multiple write ports in the register bank to enable parallel writes, improving computational efficiency while maintaining minimal area and power consumption.
2. **Integration of AES Accelerators with RISC-V:** Development and exploration of tightly and loosely coupled AES integration techniques with RISC-V processors to optimize latency, area, energy efficiency and security.
   (a) Loosely Coupled: A loosely coupled AES integration adopts a plug-and-play approach, where the AES hardware operates as an independent IP. The processor sends data to the AES unit, and the AES unit performs encryption and return the encrypted data. Since the AES unit operates independently, it can be "plugged" into the system with minimal effort. This makes it modular and easy to integrate or replace without significant modifications to the processor. This modular design simplifies integration, enables re-usability across platforms, reduces design complexity, facilitates scalability in multi-core or multi-processor systems, making it ideal for flexible and versatile implementations, but increasing area and latency.
   (b) Tightly Coupled: AES is embedded into the processor pipeline and activated using custom instructions, offering lower latency and higher security, but requiring more effort for integration.

3. **RISC-V crypto-cores for high security applications:** Designing low-power, high-performance RISC-V crypto-cores by embedding AES hardware directly into the pipeline and activating it using custom AES instructions to improve the performance of security applications. The design incorporates two pipeline pausing techniques to synchronize with other pipeline stages of RISC-V while the AES unit is active:
   (a) Software Pause (SW Pause): Utilizes NOP instructions to temporarily halt the pipeline during AES operation.
   (b) Hardware Pause (HW Pause): Incorporates additional hardware circuitry to dynamically pause the pipeline during AES activity.

   It evaluates the impact of these pausing methods and the number of pipeline stages on overall performance and security of the cores, highlighting trade-offs between processor complexity and resilience to PAA.
4. **Implementation of two countermeasure to enhance resistance against PAA with minimal overheads:** Implementation of innovative countermeasures such as Non-Linear Cyclic Variable Clock Feistel Bridge-Inspired (NCVCFB) or Random Clock Self Complimentary (RCSC), to enhance resistance against PAAs while maintaining minimal overhead. Thereby providing insights into designing secure RISC-V, efficient IoTe devices that balance performance, power consumption, and security.
   (a) Non-Linear Cyclic Variable Clock Feistel Bridge-Inspired Countermeasure (NCVCFB): Utilizes a modified Feistel bridge rolling architecture for enhanced security with minimum area and power.
   (b) Ultra-Low Power Random Clock Self-Complimentary Countermeasure (RCSC): This technique inserts random delays during AES operations and performs self-complimentary AES operations on random data to obscure power traces, improving resistance to PAAs.

   The effectiveness of these countermeasures is verified by assessing their ability to prevent key recovery under PAA.

This study evaluates the vulnerability of RISC-V cryptographic cores to PAA and proposes a extremely small countermeasure that balances security and efficiency.

1.5 Methodology

The entire methodology of PAA for the developed RISC-V crypto cores, as illustrated in Figure 1.5, outlines a structured approach for designing, implementing, and evaluating cryptographic cores on both ASIC and FPGA platforms, emphasizing the generation and analysis of power traces to effectively identify and mitigate security vulnerabilities.

The proposed RISC-V core microarchitecture, featuring new AES instruction set extensions (ISE) for high-security applications, is implemented in Verilog using the Xilinx Vivado tool. High-level programs written in C or Assembly, along with modified C programs for AES algorithms, are compiled using the modified RISC-V GNU compiler to generate corresponding hex files. These hex files are loaded into the memory (D-cache and I-cache) of both python and RTL (Verilog simulators) environment for verification. In the Python environment, the instructions are executed, and the status of registers and memory read/write operations after each instruction is recorded in a file called the "software status register". Similarly, the compiled code is executed in the Verilog simulator, where the register and

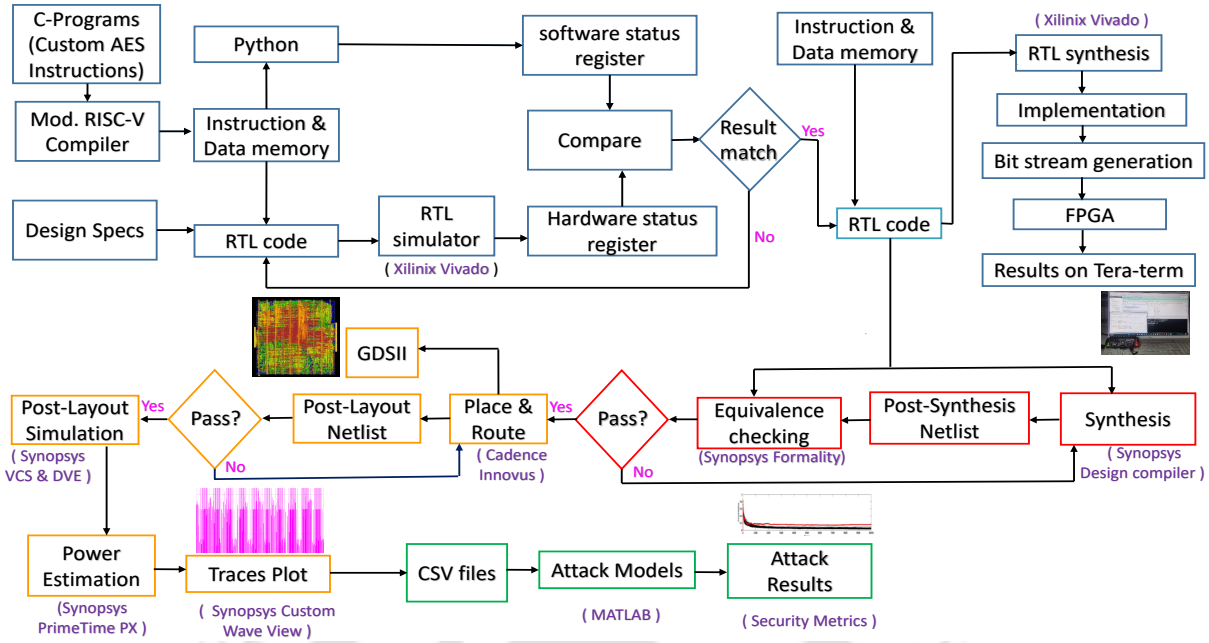


Figure 1.5: Methodology used for PAA attack of the developed crypto-cores

memory read/write status is stored in a separate file known as the "hardware status register". The two files generated from RTL simulator and the Python environment are compared to check whether the file matches or not. On matching of results from both environments, it is concluded that the designed cores are working properly. These designed cores are further imported on FPGA and their functionality is verified by dumping the bit file generated on the FPGA board and printing the output using UART on Tera Term.

After verifying the functionality of the developed cores on FPGA, Electronic Design Automation (EDA) tools are employed to import the design to ASIC using UMC 65 nm technology. The RTL code is synthesized into a gate-level netlist using Synopsys Design Compiler (DC), which is then validated for logic equivalence with the RTL design using Synopsys Formality. The verified netlist is imported into Cadence Innovus for physical layout design, encompassing several steps such as floor planning, power planning, standard cell placement, clock tree synthesis, routing, static timing analysis, and design rule checks. This process results in the generation of a GDSII file for fabrication and a post-layout netlist incorporating parasitic resistance and capacitance. The post-layout netlist undergoes functional verification using Synopsys VCS and Synopsys DVE, where a Value Change Dump (VCD) file is generated, which stores digital signal transitions during testbench simulations. The post-layout netlist along with VCD file are used by Synopsys PrimeTime PX to estimate and analyze power dissipation. An event-based Fast Signal Data Base (FSDB) file is then generated, capturing signal toggles in binary format. This FSDB file is visualized in Synopsys Custom WaveView to generate power traces, depicting power values over time. The power traces are sampled at a rate of 1 GSa/s, exported as a Comma Separated Variable (CSV) files, and analyzed in MATLAB using HD attack models to assess the design's resilience against Power Analysis Attacks (PAA).

For real-time evaluation, a hardware setup using the Side-channel Attack Standard Evaluation Board (SASEBO), shown in Figure 1.6, is employed for PAA. SASEBO is a specialized FPGA platform designed for side-channel analysis of cryptographic cores, providing cleaner power trace patterns compared to general-purpose FPGAs, which often introduce significant noise. It includes two FPGAs, a cryptographic FPGA dedicated to AES operations, uses a

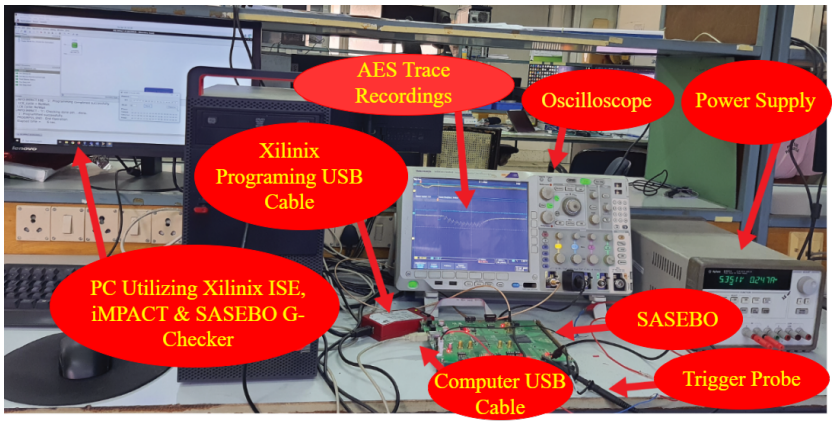


Figure 1.6: Hardware setup on SASEBO for PAA

Spartan-3A XC313400DSP, and a control FPGA for managing control operations on board, employs a Spartan-3AN XC350AN. The Verilog-based RTL design is synthesized and programmed onto the FPGA using the generated bit files. The functionality is validated, and power traces are captured using an oscilloscope. These traces are sampled at a sampling rate of 1 GSa/s and stored as CSV files, which is then exported to MATLAB to evaluate the cores' vulnerability or resilience to PAA.

1.6 Research Contributions

The contributions of this thesis are organized into five major working chapters, each building towards the development of highly secured RISC-V crypto-cores against PAA.

The chapter 2 provides a comprehensive survey of existing RISC-V cores developed primarily for high-security applications and various countermeasure techniques. It introduces the fundamental concepts behind Power Side Channel Attacks and hardware security metrics used to evaluate resistance/vulnerabilities to PSCA, including MTD, SNR, MI, and TVLA.

Chapter 3 investigates various ultra-low-power RISC-V cores with different pipeline configurations, analyzing the impact of pipeline stages on overall performance and efficiency. The analysis aims to identify the most suitable RISC-V architecture for developing RISC-V Crypto-core for high security applications, by considering the trade-offs in area, power, and processing capabilities. Hardware costs are estimated through ASIC synthesis using the UMC 65 nm technology node. Performance results of various benchmark programs, including Dhrystone, CoreMark, and Embench, are also presented to provide a comprehensive assessment of these cores' efficiency.

Chapter 4 discusses the tightly and loosely coupled integration techniques of AES with RISC-V. In tightly coupled integration technique, the AES module is embedded directly within the RISC-V micro-architecture, extending the ISA to support encryption instructions. In contrast, the loosely coupled integration technique implements the AES module externally as an address-enabled accelerator IP. These chapter provides performance metrics for both designs and evaluates their security metrics against PAA, identifying the optimal AES integration technique for developing a resilient RISC-V.

Two novel RISC-V crypto-core architectures that integrate an AES hardware functional unit directly within the RISC-V pipeline, utilizing custom AES instructions are proposed in chapter 5. The first design employs a software pause (SW) technique, while the second utilizes a hardware pause (HW) technique. This chapter explores the performance of these cores for high security applications, focusing on metrics such as AES throughput, encryption

time, energy consumption, area, and power. Additionally, it examines each core's resistance to PAA and investigates how increased processor complexity and pipeline techniques impact the performance and security metrics of these cores.

Following the identification of an optimal AES integration technique, Chapter 6 introduces a Non-Linear Cyclic Variable Clock Feistel Bridge-Inspired Countermeasure (NCVCFB) based on a modified Feistel bridge architecture. This countermeasure employs a New Substitution Function (NSFunc) and implemented as rolling architecture that operates in tandem with each round of AES encryption, effectively masking AES power consumption patterns. The design achieves minimal area and power overheads while providing substantial resilience to power analysis attacks, demonstrating to withstand up to 2 million traces.

Chapter 7 focuses on securing a loosely coupled AES-integrated RISC-V against PAA by employing the Random Clock Self Complementary (RCSC) countermeasure. The RCSC countermeasure enhances security by introducing variability in both amplitude and time domains, by using random data for complementary operations and variable number of clock cycles for encrypting each data. As the loosely coupled AES integration technique with RISC-V offers greater flexibility for upgrading and replacing the RISC-V core, this chapter begins by integrating the RCSC-secured AES with a 3stage RISC-V (RV32I), followed by a HP 4stage RISC-V (RV32I), and finally a HP 4stage RISC-V (RV32IM). The progressive integration allows for a detailed analysis of the effects of increased processor complexity on area, power, and security metrics.

Finally, Chapter 8 presents the conclusions drawn from the work conducted and provides insights into potential directions for future research and development.

CHAPTER

2

LITERATURE SURVEY

Contents

The proliferation of IoT devices in diverse domains such as healthcare, automotive, and smart infrastructure has led to a significant increase in the generation and processing of sensitive data at the edge. With the constrained computational and power resources of IoT edge devices, ensuring secure and efficient data handling is a critical challenge. Power Analysis Attacks (PAA), introduced by Kocher et al. [3], exploit variations in power consumption during cryptographic operations to extract sensitive information, posing a serious threat to IoT systems. Traditionally, IoT systems relied on cloud computing, where data collected from devices was transmitted securely to cloud servers for processing using cryptographic methods. However, with the exponential growth of data and the adoption of edge computing, much of the processing now occurs directly on edge devices with small embedded processors. This paradigm shift necessitates securing not just the data but also the embedded processors themselves, as processor security is now a fundamental requirement in IoT environments. Recognizing this critical need, our research is focused on developing and implementing secured embedded processors for IoTe applications, addressing the evolving challenges like side channel attacks in IoTe computing environments. This chapter provides a concise overview of various low-power RISC-V architectures, low-power AES architectures, power analysis attack models, and the security metrics employed in this work to analyze the security of the developed designs. It also discusses existing countermeasures against power analysis attacks, emphasizing their limitations.

2.1 RISC-V and Its Role in IoTe Devices

To meet the stringent constraints of IoTe, the embedded processors used, should have minimal area and power consumption while offering flexibility for future design upgrades to accommodate evolving requirements. RISC-V, developed by UC Berkeley, is an open and extensible Instruction Set Architecture (ISA) derived from the Reduced Instruction Set Computer (RISC) architecture. Its modular design features a compact base ISA, RV32I, with optional extensions that allow scalability and customization. These features make RISC-V particularly suited for low-cost, low-power embedded devices, while its extensibility enables enhancements to address evolving needs, including security and high-performance computing.

RISC-V, the 5^{th} generation of RISC processors, is an open-source and extensible ISA, widely adopted in IoT applications for its modular design and low-power capabilities. The ISA's flexibility enables seamless integration of custom instructions and accelerators, making it a preferred choice for secure micro-architectures.

Table 2.1: Classification of RISC-V ISA

Instr	31—25	24—20	19—15	14—12	11—07	6—0
R-Type	Func7	rs2	rs1	Func3	rd	Opcode
I-Type	imm[11],imm[10:5]	imm[4:1],imm[11]	rs1	Func3	rd	Opcode
S-Type	imm[11],imm[10:5]	rs2	rs1	Func3	imm[4:1],imm[0]	Opcode
SB-Type	imm[12],imm[10:5]	rs2	rs1	Func3	imm[4:1],imm[11]	Opcode
U-Type	imm[31:20]		imm[19:15]	imm[14:12]	rd	Opcode
UJ-Type	imm[20],imm[10:5]	imm[4:1],imm[11]	imm[19:15]	imm[14:12]	rd	Opcode

The architecture supports three base ISAs: RV32I, RV32E, and RV64I differentiated by register width and memory address space. RV32I, the 32-bit base ISA with 47 instructions, meets the fundamental needs of modern processors. RV32E, a reduced version of RV32I with fewer registers, is optimized for embedded systems where area is a major concern. RISC-V follows a load-store architecture, where operations are performed using general-purpose

registers and memory access occurs solely through load and store instructions. The RV32I ISA instructions organized into six formats: R-type (register-register), I-type (immediate), S-type (store), SB-type (branch), U-type (upper immediate), and UJ-type (jump) as shown in Table 2.1. A key feature of this encoding is the fixed position of register specifiers, enabling parallel register fetch and instruction decoding. Its 7-bit opcodes reserve the two least significant bits for ISA extensions, improving code density.

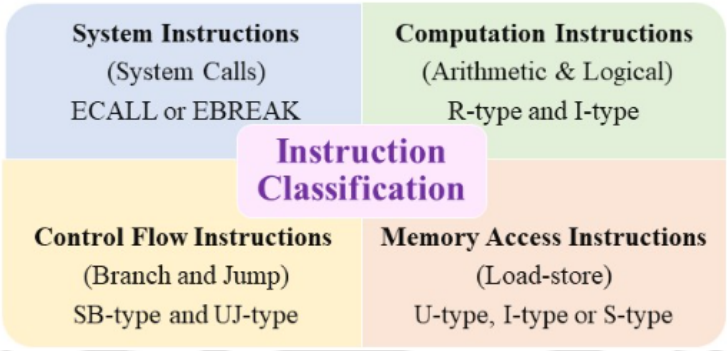


Figure 2.1: Types of Instructions in RISC-V

RISC-V instructions are broadly classified into four categories, each addressing specific aspects of program execution, as shown in Figure 2.1. Computation Instructions handle arithmetic, logical, and comparison operations using R-type and I-type formats, enabling essential data manipulation tasks. Memory Access Instructions facilitate data transfer between CPU registers and memory using U-type, I-type, and S-type formats for loading and storing data, ensuring proper data handling. Control Flow Instructions manage program execution by directing control through conditional branching by SB-type instructions based on comparisons or unconditional control transfers by UJ-type jump instructions for tasks like procedure calls and loops. Finally, System Instructions supports system-level operations, such as ECALL for system calls, EBREAK for debugging, and CSR instructions for accessing, reading and modifying control status registers.

Several RISC-V cores have been developed for IoT applications, focusing on balancing performance, power efficiency, and area constraints. The Rocket Core [4], was the first RISC-V microarchitecture and is based on the RV64G ISA. It features a six-stage, single-issue, in-order pipeline, delivering 10% better performance and 49% higher area efficiency compared to the ARM Cortex-A5 [5], making it highly suitable for high-performance and area-efficient IoT processors. The Z-Scale[6] also developed by UC Berkeley, is a compact a 32-bit core with a three-stage pipeline designed for resource-constrained environments, offering simplicity and efficiency for small-scale IoT applications. ETH Zurich developed the PULP platform [7], featuring clusters of ultra-low-power cores tightly coupled with shared data memory. This architecture excels in DSP applications, where low-voltage operation is critical. By reducing the supply voltage (as dynamic power scales with the square of voltage) and using multi-core designs, the PULP platform achieves remarkable energy efficiency without significant performance trade-offs. IIT Madras developed the Shakti series of processors in Bluespec System Verilog (BSV), including Shakti-E [8], a three-stage in-order core for low-power devices like smart cards and IoT sensors. The VexRiscv processor [9], is a configurable five-stage pipeline RISC-V core that supports the RV32IM ISA, equipped with branch prediction and cache modules. These features enhance its performance and suitability for low-power applications enabling efficient instruction and data management. The PicoRV32

[10] is a tiny, reconfigurable RISC-V core designed for ultra-low-power IoT devices. It supports the RV32IMC ISA, achieves high operating frequencies, and is optimized for both area and power efficiency, making it ideal for resource-constrained applications. RI5CY [11] is a four-stage pipelined processor based on the RV32IM ISA, designed for DSP applications. By updating its ISA with new MAC instructions, boosts the performance and increases the effectiveness of signal-processing algorithms. The Zero-riscy and Micro-riscy cores [12], designed for arithmetic and control tasks, target ultra-low-power IoT applications. Zero-riscy, a two-stage pipelined core based on RV32IM ISA, balances computational performance and area optimization. In contrast, Micro-riscy is tailored for environments with extreme area and power constraints, using the RV32E ISA. It achieves this by omitting separate hardware for multiplication and division, reducing general-purpose registers from 32 to 16 and utilizing prefetch buffer for compressed instructions. While Zero-riscy prioritizes performance, Micro-riscy focuses on achieving minimal power and area usage. The Asynchronous RISC-V Core [13], features a four-stage pipeline and achieves three times lower dynamic power consumption compared to its synchronous design by eliminating the power-hungry global clock distribution. The processor employs adaptive micro-pipelining for power and performance optimization. It uses local handshake signals and event-driven mechanisms for synchronization which will minimize the chip activity and thereby reduce power consumption. Although asynchronous processors are generally 15% slower than their synchronous counterparts, the proposed design compensates with better power efficiency and improved performance due to its adaptive pipeline architecture, making it robust and energy-efficient, ideal for IoT devices. These diverse RISC-V cores exemplify the flexibility of the ISA in meeting the varied requirements of IoT applications, from high performance to ultra-low power consumption.

2.2 Importance of AES in IoTe Devices

Cryptography forms the backbone of secure communication in IoT systems, ensuring data confidentiality by encrypting plaintext into ciphertext and decrypting it back using a cipher key. Numerous cryptographic algorithms, such as Rivest-Shamir-Adleman (RSA) [14], Secure Hash Algorithm (SHA) [15], Elliptic Curve Cryptography (ECC) [16], PRESENT, and CLEFIA have been developed for securing data. RSA, while highly secure, is resource-intensive, and PRESENT and CLEFIA, though lightweight, offer lower security levels. Among the available encryption algorithms, the Advanced Encryption Standard (AES) introduced by the National Institute of Standards and Technology (NIST) in 2001, is considered to be the most suitable for IoT edge devices due to its symmetric key cryptography and optimal balance of security, efficiency, and low resource consumption. AES has been integrated into IoT standards such as IEEE 802.15.4, ZigBee, Sigfox, LoRaWAN, and Z-Wave for secure communication in IoT applications.

Advanced Encryption Standard (AES) is a symmetric encryption algorithm that uses shared keys for both encryption and decryption, requiring the secret key to be known by both the sender and receiver. It processes 128-bit data blocks /plaintext, using cipher keys of variable lengths of 128, 192, and 256 bits. The 128-bit AES employs 10 rounds with128-bit plaintext and 128-bit keys for encryption. The process begins with a whitening step, where the key is XORed with the plaintext and is followed by 10 rounds. Each of the first nine rounds includes four operations: SubBytes, ShiftRows, MixColumns, and AddRoundKey. The final round excludes MixColumns. Round keys are generated from the initial secret key using a key scheduler. There are various architectures of AES implementations, mentioned

in [17] like rolling architecture, unrolling architecture, fine grain pipelining etc. Unrolling Architecture employs separate hardware for each round with pipeline registers in between, achieving high throughput but significantly increasing area and power consumption around ten times. Fine-Grained Pipelining places pipeline registers within each round step to further enhance throughput but at a steep cost in area and power. Whereas in Rolling AES Architecture the same hardware is used for all rounds, minimizing area and power at the cost of lower throughput. By strategically designing the SubBytes function researchers can further optimize AES architectures and compact, low-power AES designs with high throughput and improved resilience to PAA can be achieved. Singha et al. [18] provide a literature survey of numerous AES designs based on different S-box design techniques like normal basis, polynomial basis, and LUT-based. The usage of LUT-based S-box in AES designs is prevalent in literature [18],[19],[20] due to its ease and simplicity in implementation. But these implementations using conventional LUT-based S-boxes requires storing of 256 bytes and additional circuitry for address decoding and fetching, consuming huge area. Such extensive hardware demands make this approach unsuitable for IoT applications, where minimizing area and power consumption is crucial. Recent research has shifted towards LUT-less logic gates-based S-box designs leveraging Galois Field (GF) theory to reduce the S-box area by minimizing gate count. Various compact AES designs employing CFA S-box [17] and Masoleh S-box [21] came as replacement for LUT S-box. The Masoleh S-box-based AES improves area and power efficiency by simplifying the S-box implementation using optimized combinational logic, which reduces gate count, memory usage, and circuit complexity compared to traditional LUT-based AES implementations. This not only saves area but also minimizes switching activity, leading to lower dynamic power consumption and significantly improving the overall efficiency of the AES core.

AES has traditionally been implemented in software, running on general-purpose processors. Software-based implementations of AES [22], [23], although flexible, are unsuitable for IoT applications due to high latency, large energy consumption, and vulnerability to side-channel attacks. Dedicated AES hardware accelerators [24–28], address these challenges by offloading encryption tasks from the CPU, offering significant improvements in throughput, latency, and energy efficiency. Hardware implementations offer faster encryption times and lower overhead compared to their software counterparts, making them ideal for real-time IoT applications. Notable advancements include Intel's AES-New Instructions (AES-NI) [29] and ARM's AES extensions in Armv8-A ISA [30]. While these implementations reduce vulnerabilities and improve encryption performance, integrating AES hardware securely with the processor remains a challenge. Intel introduces intel's Advanced Encryption Standard New Instructions (AES-NI) in 2010, a set of processor instructions designed to accelerate the performance of the AES encryption and decryption algorithms. AES-NI enhances data security by enabling faster and more efficient cryptographic operations and is supported by a wide range of Intel processors, including various models of Intel Xeon and Core processors.

AES accelerators can be integrated with RISC-V processors using either tightly coupled or loosely coupled architectures. The paper [31] presents an energy-efficient, reconfigurable RISC-V SoC that integrates a loosely coupled encryption module capable of executing various cryptographic algorithms. In [32], an advanced AES accelerator is integrated as an IP core with RISC-V, capable of operating in multiple AES modes without requiring ISA extensions or compiler modifications. This simplifies synchronization but results in increased area and power consumption. Other studies explore tightly coupled AES integration, such as [33], which enhances the security of Hummingbird E203 core by adding an AES co-processor. In [34], custom RISC-V instructions are used to integrate an S-box, significantly reducing mem-

ory and energy usage compared to software AES implementations. FAC-V [35] demonstrates that tightly coupling AES hardware into the RISC-V architecture improves performance but requires more effort for synchronization and integration.

2.3 Side-Channel Attacks on IoTe Devices

With the widespread adoption of IoTe devices, which are often deployed in environments with limited physical protection, attackers are increasingly targeting these edge devices, making them vulnerable to numerous attacks, particularly side-channel attacks (SCAs) [3]. These attacks exploit unintended leakages, such as variations in power, timing, temperature, and electromagnetic emissions, to extract the secret key, thereby the sensitive data. Among these, power side-channel attacks pose significant threats, as they exploit variations in device's power consumption to infer sensitive data such as cryptographic keys [36].

2.3.1 Power Analysis Attacks

Power Analysis Attacks (PAA) introduced by Kocher et al. [3], exploit the correlation between the sensitive data and the power consumption of the processor to recover the secret information, especially the cryptographic keys by analyzing numerous traces. Correlation Power Analysis (CPA), a widely used PAA, involves statistical analysis of power traces to infer cryptographic keys.

The power consumption in AES encryption is mainly due to the dynamic power consumption of CMOS gates, which is data-dependent and primarily caused by gate switching. One key component of dynamic power is the charging power of the load capacitance, represented in Equation 2.1. Here $p_{chrg}(t)$ indicates the instantaneous charging power, α is the activity factor, C_L represents the load capacitance, and V_{DD} denotes the supply voltage.

$$P_{chrg} = \frac{1}{T}\int_0^T p_{chrg}(t)\,dt \;\; = \alpha.f.C_L.V_{DD}^2 \tag{2.1}$$

Another factor contributing to dynamic power is the occurrence of short-circuit currents during output switching, as detailed in the equation 2.2. Here, $p_{sc}(t)$ denotes the instantaneous short-circuit power. I_{peak}, represents the peak current observed while switching, and t_{sc} signifies the duration for which the short circuit occurs.

$$P_{chrg} = \frac{1}{T}\int_0^T p_{sc}(t)\,dt \;\; = \alpha.f.V_{DD}.I_{peak}.t_{sc} \tag{2.2}$$

CPA attacks exploit this dynamic power to recover the 128-bit encryption key by adopting a divide-and-conquer strategy, i.e, breaking the key into 16 bytes and recovering each byte individually [36]. While a brute-force attack on the entire 128-bit key, with its 2^{128} possibilities, would take a trillion years for a computer to recover the key, recovering each key byte individually with only 256 (2^8) possibilities significantly simplifies the key-guessing process and reduces the time required to recover the key. CPA attacks typically target the Hamming Weight (HW) or Hamming Distance (HD) during AES encryption. In this work, the 10^{th} round of AES is attacked utilizing the Hamming Distance (HD) model. For each ciphertext, a sequence of inverse operations of the last round of AES (AddRoundKey, InvShiftRows, and InvSubBytes) is performed and utilizing the 10^{th} round key, the original key is recovered and thereby the data encrypted. The Hamming Distance (HD) is analyzed according to equation 2.3.

$$h_{i,j} = HD(v_{i,j}, c_i) = HW(v_{i,j} \oplus c_i) \tag{2.3}$$

In this context, h represents the hypothetical power consumption values, c refers to the encrypted ciphertext, and v signifies the hypothetical intermediate values from the final round of encryption, which serve as the attack point targeted by attackers. The HD values are calculated from the HW of the result obtained by XORing corresponding v and c values.

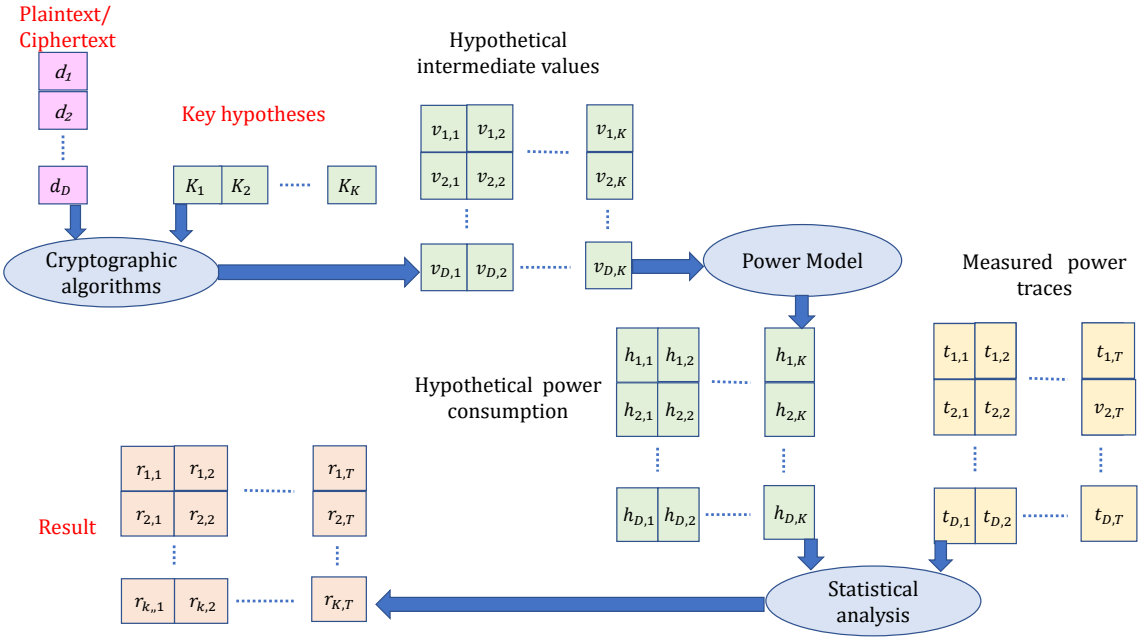


Figure 2.2: CPA Attack Model

CPA attack [36, 37] involves 5-steps as shown in Figure 2.2.

1. The primary step of a CPA attack is to select an intermediate result of the cryptographic algorithm that is executed by the attacked device. This intermediate result derived is a function of $f(d, k)$, where 'd' is a known non-constant data value and 'k' is a small part of the key. In most attack scenarios, 'd' is either the ciphertext or plaintext.
2. The subsequent step of a CPA is to compute the power consumption of the cryptographic device during the encryption or decryption of various data blocks. The actual power traces, while performing cryptographic operations for each data, are recorded as $t_{ij} = (t_{1,1}, \ldots\ldots t_{D,T})$ and are stored as a matrix $T(D\ X\ T)$, where 'D' represents the data and 'T', trace length.
3. The third step is to compute a hypothetical intermediate value for every possible choice of key. For all data, 'd', and 'k', key hypotheses, the hypothetical intermediate values, *f(d, k)*, are evaluated by an attacker, resulting in a matrix $V(D\ X\ K)$, whose elements are represented by equation 2.4.

$$v_{ij} = f(d_i, k_j); i = 1, \ldots D; j = 1, \ldots K \tag{2.4}$$

4. The next step is to map the intermediate values V_{ij} to hypothetical power consumption values h_{ij}, using Hamming Distance (HD) or Hamming Weight (HW) power models. It is represented by a matrix $H(D\ X\ K)$.

   The HW power model computes the number of bits that are high in a register to obtain the key using the following equation 2.5.

$$h_{ij} = HW(v_{ij}, d_i) \tag{2.5}$$

where, 'h' is the hypothetical power traces, 'd' is the plaintext used for AES encryption, 'v' represents hypothetical intermediate values corresponding to the first round input.

The HD power model computes the number of bit flips between two consecutive values held in the register, thereby obtaining cipher keys using the equation 2.6.

$$h_{ij} = HD(v_{ij}, d_i) = HW(v_{ij} \oplus d_i) \tag{2.6}$$

In HD model, 'h' is the hypothetical power traces, 'd' is the Ciphertext obtained after AES encryption and 'v' represents hypothetical intermediate values corresponding to the last round. The HD of hypothetical power traces is obtained by calculating HW of the simply XORed result of 'v' and 'd'.

5. In the final step, the attacker compares the hypothetical power values of each key hypothesis with the recorded power traces by comparing 'H' and 'T' matrices. The outcome is a matrix $R(D\ X\ K)$, where each element, r_{ij}, is the result after comparing elements h_i and t_j.

Correlational Power Analysis (CPA) attack is a type of PAA, where 'correlation coefficient' is utilized as an evaluation parameter to determine a linear relationship between the hypothetical power values, h_{ij}, with the real power traces t_{ij}. The resultant matrix 'R' is comprised of estimated correlation coefficients, r_{ij}, for $i = 1...\ K$, and $j = 1...\ T$, between hypothetical power traces, 'H', and real power traces, 'T', based on the 'D' data blocks processed. It is denoted mathematically using equation 2.7.

$$r_{i,j} = \frac{\sum_{d=1}^{D}(h_{d,i} - \bar{h_l}).(t_{d,j} - \bar{t_j})}{\sqrt{\sum_{d=1}^{D}(h_{d,i} - \bar{h_i})^2.\sum_{d=1}^{D}(t_{d,j} - \bar{t_j})^2}} \tag{2.7}$$

where 'h' and 't' represent their respective matrix elements.

2.3.2 Hardware Security Metrics

The section provides detailed evaluation of all the four security metrics [36]: MTD , SNR, MI and TVLA. Out of the four security metrics evaluated, SNR and MTD are based on CPA attacks as they are evaluated using Correlational coefficients.

Measurement To Disclose (MTD)

MTD provides each byte of the key and specifies the number of power traces needed to recover the last key byte. A design remains secure until all key bytes are retrieved. Higher MTD values indicate stronger security, as they require more traces for key recovery, whereas lower MTD values suggest vulnerability to power attacks.

MTD uses statistics of population, which is comprised of numerous 'samples', where the mean of the parameter in the population is represented by μ. The mean of each samples in a group of random variables, $\overline{R}$, is denoted as $\overline{r}$. In accordance with the law of large numbers, as the number of trace counts increases, the estimation of $\overline{R}$ improves. So, for the average value to more precisely represent the mean value, additional traces, 'n,' are required. The statistical concept of 'confidence interval' helps to understand the approximation of $\overline{r}$ towards $\overline{\mu}$. Confidence intervals are based on hypothesis tests, wherein a certain hypothesis is tested to check whether it is true or not. The second hypothesis is known as the alternative hypothesis, and is defined as H_1: $\mu \neq \mu_0$ in contrast to the first hypothesis, H_0, which is known as the null hypothesis H_0: $\mu = \mu_0$. A set of values for a selected parameter that are

all substantially near to a given value is referred to as a confidence interval. All values μ_0 of the null hypothesis that fall within the confidence interval of μ are accepted. A confidence interval of 0.99 indicates, there is 99% probability that the number will fall within the defined range. This idea serves as the foundation for MTD's correct Key guess.

Signal to Noise (SNR)

SNR measures the ratio between the correlation coefficient of the correct key byte and the highest correlation coefficient among incorrect key bytes during CPA. For secure designs, SNR should be below 1, indicating that exploitable signal leakage is lower than noise. Unprotected designs typically exhibit SNR values greater than 1, where the correlation coefficient of the correct key is significantly higher than those of incorrect keys.

The total power P_{total} [38], consumed by the device while performing cryptographic functions is the sum of operation-dependent power P_{op}, data-dependent power P_{data}, noise power P_{noise}, and constant power P_{const}.

$$P_{total} = P_{op} + P_{data} + P_{noise} + P_{const} \tag{2.8}$$

P_{const} power is independent of the operations performed and data processed, whereas P_{op} and P_{data} are exploitable power P_{exp} and depend on the operations performed and data processed, respectively. P_{noise} depends on switching noise $P_{sw:noise}$ and electronic noise $P_{el:noise}$. Hence, attacker exploits P_{exp} power, sum of P_{op} and P_{data} to recover the key.

$$P_{total} = P_{exp} + P_{sw:noise} + P_{el:noise} + P_{const} \tag{2.9}$$

SNR is another crucial security metric. SNR is the ratio between signal power to noise power and is defined as follows:

$$SNR = \frac{\sigma^2_{signal}}{\sigma^2_{noise}} = \frac{\sigma^2_{P_{exp}}}{\sigma^2_{P_{sw:noise}} + \sigma^2_{P_{el:noise}}} \tag{2.10}$$

where, σ^2_{signal} and σ^2_{noise} are the variance of the exploitable signal power and noise power components, respectively.

When taking the correlation coefficient into account, SNR is defined as the ratio of the highest correlation coefficient for the correct key to the highest correlation coefficient corresponding to any other (wrong) key throughout the course of processing data. It can be stated mathematically as:

$$SNR = \frac{\max_{t\in[1...T]} \rho_{t,k_{correct}}}{\max_{t\in[1...T],k\in[1...K]\setminus k_{correct}} \rho_{t,k}} \tag{2.11}$$

where, t: corresponds to a particular instant of sampled power, T: refers to the number of samples taken per clock cycle, $\rho_{t,k_{correct}}$ and $\rho_{t,k}$ are correlation coefficients for correct and wrong key guesses respectively. Higher SNR in PAA makes it easy to distinguish exploitable power P_{exp} from background noise power, suggesting the cryptographic design is leaky. For unprotected design, SNR is always greater than 1.

Mutual Information (MI) analyses

The dependence between two random variables is generally measured by MI analyses. MI uses Shannon's entropy concept to find the information leaked by the crypto-core processor. Protected designs exhibit low MI values with no distinct peaks at the correct key byte,

while unprotected designs show a pronounced peak at the correct key, indicating significant leakage. For protected design it is in milli range.

The information leaked by the crypto-core is evaluated using conditional entropy instead of using correction coefficients. The MI of two random variables X and Y on a discrete space χ, γ is given:

$$I(X:Y) = \sum_{x\varepsilon\chi, y\varepsilon\gamma} P_r[X=x, Y=y] \log_2(\frac{P_r[X=x, Y=y]}{P_r[X=x].P_r[Y=y]}) \tag{2.12}$$

Test Vector Leakage Assessment (TVLA)

TVLA was introduced by Goodwill et al. [39] as a security metric for leakage assessment of crypto-cores. Welch's t-test is used to calculate the t-value. Secure designs have t-values within the range of ±4.5, while unprotected designs exhibit t-values exceeding these limits, indicating significant leakage.

It determine whether two sample sets are from the same population by comparing their means. In accordance with the test, the key, plaintext, and ciphertext are employed to record power traces. Two distinct data groups are created from the acquired traces. The t-test is computed on two subsets of data: a random set of data, N_A and a fixed set of data, N_B, and is expressed as follows:

$$T(K) = \frac{X_A - X_B}{\sqrt{\frac{S_A^2 - S_B^2}{N_A - N_B}}} \tag{2.13}$$

where, X_A and X_B are the averages of the traces in respective groups, and the S_A and S_B are the standard deviation of all the traces in the corresponding group.

Security metrics indicates that as security strengthens, MTD increases, while SNR, MI, and TVLA values decrease. Unprotected designs exhibit low MTD (<100K traces), high SNR (>1), distinct MI peaks, and t-values exceeding ±4.5, indicate significant vulnerability. In contrast, protected designs demonstrates high MTD (> 100K traces), low SNR (<1), minimal MI (in milli range), and t-values within ±4.5, reflect enhanced security and reduced leakage.

2.4 Countermeasures against Power Analysis Attacks

Both software-based AES implementations on general-purpose processors [40], [41] and standalone AES hardware [42–44] are highly vulnerable to security breaches by power analysis attacks due to the significant power leakage from cryptographic operations. Integrating AES accelerators within RISC-V cores reduces vulnerabilities, but even with this integration vulnerabilities to PAAs remain a significant concern. Addressing vulnerabilities to PAAs in RISC-V cores requires robust countermeasures, as existing methods often incur significant area and power overheads. With the increasing adoption of RISC-V in IoT, it is crucial to implement efficient countermeasures that maintain performance and energy efficiency, while providing significant security.

Masking and hiding are two widely used countermeasures against power analysis attacks as shown in Figure 2.3. Hiding reduces data exposure by minimizing the correlation between power consumption and processed data, and is further categorized into uniform hiding and random consumption hiding. The uniform hiding approach ensures uniform, data-independent power dissipation by designing circuits to consume equal energy in each

Figure 2.3: Various Countermeasure Techniques

clock cycle, regardless of the processed data. This technique is typically implemented at the gate level, as uniform power consumption at this level often propagates to the system level. Random consumption hiding, on the other hand, introduces randomness and noise into power consumption patterns. Masking technique complements these hiding techniques by using random mask data to alter intermediate values, ensuring that computations in cryptographic circuits are performed on masked data instead of the original, sensitive data. Masking techniques enhance security by splitting sensitive data into multiple shares, either at the algorithmic or circuit level. Higher-order masking provides stronger protection by increasing the number of shares, but this comes with greater implementation complexity.

This section surveys hardware and software countermeasures based on these techniques to mitigate power dissipation leakage, focusing on reducing the correlation between power dissipation and processed data to protect the circuit's secret key. It includes a cost-benefit analysis of area, power, and security trade-offs for various methods and examines the feasibility of implementing these strategies in hardware or software. Software countermeasures have a limited design space as they are more vulnerable, whereas hardware countermeasures offer a broader design space to address side-channel information leakage.

1. **Hardware-Based Countermeasures:** Hardware-based approaches aim to address PAAs by introducing design modifications at hardware or circuit level to obscure power consumption patterns and make them less correlated to the underlying cryptographic operations.
   (a) **Dual-Rail Pre-Charge Logic (DPL) [45]:** A secure gate-level encoding scheme where each bit is represented by two rails the real value and its complement. It operates in two phases: a pre-charge phase, which resets circuits to a uniform state mitigate power-based attacks, and an evaluation phase for executing logic. This design reduces leakage exploitable by Hamming Distance (HD) models.
   (b) **Masked Dual Rail Pre-Charge Logic (MDPL) [46]:** Combines masking with dual-rail pre-charge logic to resist PAA by employing data-independent switching activity and ultilizing single random mask bit to address routing imbalances.

However, it incurs significant area overhead and suffers from mask bit leakages.

(c) **Time Enclosed Logic (TEL) [47]:** Implements three phases within a clock cycle: pre-charge (return-to-zero), evaluation, and post-evaluation (all rails set to logical 1). By enclosing the evaluation phase between two encoding steps, TEL minimizes leakage from memory effects and reduces vulnerability to attackers with limited time resolution or bandwidth.

(d) **Delay-based Dual-Rail Pre-Charge Logic (DDPL) [48]:** Utilizes time-domain encoding to eliminate memory effects and offers strong resistance against PAA, even with unbalanced capacitances.

(e) **Wave Dynamic Differential Logic (WDDL) [49]:** This technique equalizes power consumption by ensuring uniform switching activity across logical gates. WDDL facilitate secure circuit design through standard-cell compatibility, but often comes with significant area and power overhead, making it less ideal for resource-constrained devices.

(f) **Sizing and Place and Route for semi full-custom topologies [50]:** To address the imbalance between rails in Dual-Rail Precharge Logic (DPL) circuits, strategies such as optimized transistor sizing and improved Place and Route techniques are employed to enhance security and performance.

(g) **On-Chip Current Flattening:** Proposed by Ekarat et al. [51], this technique mitigates PSCAs, by maintaining constant power consumption across all operations, making it challenging to correlate power variations with cryptographic activities. Similarly, Ratanpal et al. [52] utilize an integrated filter to achieve data-independent, uniform current consumption. While effective, these approaches come with increased silicon area requirements and higher overall power usage.

(h) **Detaching power supply:** Telandro et al. [53] propose an on-chip voltage regulator to decouple external and internal power supply currents. Similarly, Gornik [54] introduced a circuit with buffering capacitance to isolate the main power supply from the internal power supply driving logic gates. Although these strategies are highly effective against power attacks, they result in substantial power consumption.

(i) **Random Delay Insertion (RDI) [55]:** This technique introduces programmable delay elements into the data-path to randomize execution timing and power dissipation, effectively mitigating attacks but increasing area overhead by up to 100%.

(j) **Randomization Techniques [56]:** Approaches such as Dynamic Voltage Frequency Scaling (DVFS) disrupt the correlation between power consumption and cryptographic operations, effectively mitigating power attacks, but at the expense of area, power and performance degradation.

(k) **Globally Asynchronous and Locally Synchronous (GALS) architecture [57]:** Combines random clocking and parallel processing in a pipeline architecture to introduce noise into global power consumption.

(l) **Dual-core parallel processing [58]:** This method involves the simultaneous processing of actual data and its complementary counterpart using two separate processing cores. By executing the complementary operations in parallel, the power consumption patterns of the two cores effectively cancel each other out, reducing the correlation between power variations and the processed data. This approach introduces randomness into the power profile, making it significantly harder for attackers to extract sensitive information through side-channel analysis.

Although dual-core parallel processing enhances security, it comes at the cost of increased hardware complexity and power consumption, as two processing units operate simultaneously.

(m) **Dummy rounds and operations:** Dummy rounds are added to cipher algorithms (e.g., Moucha et al. [59]) to obscure the actual cryptographic operations. Similarly, Baylis et al. [60] introduce extra computational steps on randomly generated data using ring oscillators, which create noise in power consumption to hinder side-channel attacks. However, these methods require additional hardware, such as ring oscillators, which significantly increase area and power costs. Furthermore, the noise introduced by these techniques can be mitigated through advanced pre-processing methods, reducing their overall effectiveness.

(n) **Clock-Based Strategies [61]:** Lagasse et al. proposed randomized clock techniques to enhance security by dynamically selecting one of four phased clock signals. These clock signals are generated using a linear feedback shift register (LFSR), introducing variability in timing and making it harder for attackers to correlate power consumption with operations. This method provides strong security enhancements while maintaining a smaller area footprint compared to alternative approaches.

(o) **Attenuated Signature Noise Injection (ASNI) [62]:** The ASNI technique effectively suppresses current variations by injecting noise into the power signature, disrupting side-channel analysis. It achieves this with a low area overhead and demonstrates immunity to correlation power analysis (CPA) attacks, even up to 1 million traces.

(p) **Asynchronous AES Designs [63]:** Chong et al. present the first asynchronous AES countermeasure design featuring dual hiding techniques: amplitude moderation and time moderation to alter power signatures and enhance resistance to attacks. While these approaches significantly enhance security, they often entail trade-offs in cost, implementation complexity, and residual vulnerabilities.

(q) **Random Frequency Tuning Countermeasure (RFTC) [64]:** Developed by Darshana et al., this approach randomizes execution times for each encryption, reducing resource usage and power requirements compared to conventional masking techniques. However, it significantly impacts the performance.

(r) **Random Dynamic Frequency Scaling (RDFS) [65]:** In this method, cryptographic accelerators use Random Dynamic Frequency Scaling which operates in 219,000 distinct frequencies, offering strong resistance to power analysis with minimal performance impact, but at expense of huge area.

(s) **SecureV Processor [66]:** This high-performance, secure RISC-V core incorporates features like dynamic code translation, control unit enhancements, and adaptive memory management to improve security without significant chip redesign.

(t) **Data Flow Integrity (DFI) [67]:** Introduced by Lang et al., DFI strengthens RISC-V processor security by ensuring data flow integrity, with minimal performance degradation. This is achieved by securing critical data paths in the processor architecture.

2. **Software-Based Countermeasures:** Software-level countermeasures aim to enhance security through algorithmic modifications or runtime strategies, often offering greater flexibility but with trade-offs in performance and reduced resistance to attacks.

(a) **Algorithmic Masking [68]:** This method combines cryptographic computations with random values at the algorithm level to obscure sensitive data. While effective against first-order attacks, masking schemes remain vulnerable to higher-order attacks.

(b) **Code Randomization [69]:** Randomizing instruction execution order in AES algorithms to modify power consumption patterns. This method requires compiler-level integration and has been implemented on RISC-V cores with minimal performance overhead.

(c) **SIMD instructions [70]:** Replacing LUT-based S-boxes with SIMD instructions in AES, improves performance and security, especially in high-order masking. These approaches aim to enhance security but implementation complexity increases.

(d) **Dynamic Key Refreshing [71]:** In this method, cryptographic devices utilize dynamic keys instead of static keys, which are refreshed after a specific number of plaintexts to safeguard against PAA.

(e) **PARAM Processor [72]:** The Power Attack ResistAnt Microprocessor (PARAM) improves resistance to PAAs by identifying and securing high-leakage modules in the SHAKTI-C RISC-V processor using Power Attack Leakage Analyzer (PLAN). PLAN evaluates power consumption and identifies leakage causes, enabling targeted countermeasures like HDL code modifications and obfuscation techniques to break power-data correlations. PARAM achieves superior protection with lower area and power overhead compared to traditional masking or hiding methods.

(f) **Protecting RISC-V with Masking [73]:** This approach applies software masking countermeasure techniques to high-leakage modules in RISC-V such as the ALU, register bank, and memory. Random values generated by Pseudo Random Number Generator (PRNG) obscure data, but transfer of unmasked plaintext and ciphertext remains vulnerable.

(g) **Masking Techniques [74]:** Masking protects sensitive data by splitting it into random multiple shares and performing arithmetic or logic operations like addition, multiplication or XOR on these shares to obscure the power. The protection level 'd' determines the number of required shares (d+1) and impacts the implementation cost in terms of chip area and randomness. There are different masking techniques like Boolean Masking, Threshold Implementations and Domain-Oriented Masking.

   i. **Boolean Masking (BM):** This technique uses XOR operations to mask data with random values but suffers from glitches leading to potential data leakage. Removing these glitches requires complex reordering of logic blocks.

   ii. **Threshold Implementations (TI):** It eliminates glitches, so that the data is always independent by adhering to properties like correctness, non-completeness, and uniformity. TI increases protection but requires significant logic gates, thereby consumes huge area and power.

   iii. **Domain-Oriented Masking (DOM):** It ensures security-critical computations are made independent of the underlying data by dividing the data into multiple shares and assigning each share to a specific domain. Additional registers are employed to prevent glitches, enhancing security. The number of shares required is determined by the desired protection level(d+1), providing

robust resistance to PAA. DOM provides a balance of higher protection with lower area and power overhead compared to BM and TI.

Software-level countermeasures, while more flexible and less resource-intensive, often degrade performance and provide limited security against advanced attacks. On the other hand, hardware-level countermeasures significantly improve security but introduce trade-offs, such as increased area and power consumption due to the additional circuitry required. These challenges are particularly critical for battery-powered IoT devices, which operate under stringent constraints on space, energy efficiency, and performance. Balancing these trade-offs to achieve efficient and secure designs remains a significant challenge for researchers and engineers.

2.5 Chapter summary

This chapter reviewed the state-of-the-art in various low power RISC-V micro-architectures, cryptographic techniques, and countermeasures against power side-channel attacks, forming the basis for the developing proposed secure RISC-V cores presented in subsequent chapters.

CHAPTER

3

IMPACT OF PIPELINING ON LOW POWER IoT APPLICABLE RISC-V ISA CORE MICRO-ARCHITECTURES

Contents

3.1 Introduction

Internet of Things (IoT) edge devices are experiencing exponential growth in the current era. Depending on the application and specific computational needs, different types of processor cores are available, ranging from high-performance super-scalar cores for DSP applications to tiny ultra-low power processors designed for limited performance. Today, many core vendors offer a wide range of micro-architectures designed to address various performance, area, and power constraints. For example, ARM provides small-budget compact low-power cores like Cortex-M0 [75] and high-performance super-scalar cores such as Cortex-M4 [76], which feature branch speculation and single-cycle MAC operations. Recently, low-power heterogeneous multi-core architectures have been developed for IoT applications, using tightly coupled DSP clusters at low voltage to enhance energy efficiency. Since dynamic power decreases with the square of the supply voltage, operating at lower voltages reduces power consumption. Performance degradation by lowering voltages can be recovered by using multiple cores.

Since the majority of IoT devices are battery-driven, the flawless design of ultra-low power processors is of utmost relevance. Low-power processor designs for IoT applications based on RISC-V ISA[1], developed by UC Berkeley in 2010, are gaining attraction in industry and academia due to their simple and extensible nature. The Rocket core (RV64G) [4], a 6-stage RISC-V micro-architecture, outperforms the ARM Cortex-A5 [5] in both performance and area efficiency. For simpler designs, UC Berkeley developed the Z-Scale [6] with a 3-stage pipeline, while ETH Zurich created PULP [7], an ultra-low-power RISC-V SoC platform for DSP applications. IIT Madras developed the Shakti-F core [8], optimized for low-power usage, as part of their Shakti processor series. High-performance cores like VexRiscv[9] and PicoRV32 [10], are designed for complex IoT applications. Area-optimized cores such as Zero-riscy [12] and Micro-riscy suitable for arithmetic and control tasks were also developed. An asynchronous 4-stage RISC-V core [13] was also developed, which significantly reduces dynamic power consumption by replacing the global clock with local handshake signals. Selecting the appropriate RISC-V ISA and number of pipeline stages for IoT devices depends on the specific application's performance requirements, power consumption, and area constraints. Not all applications need high-end RISC-V processors with numerous pipeline stages and complex extensions. Simpler designs like RV32I are well-suited for low-power, area efficient applications, making it ideal for embedded systems in resource-constrained IoT environments. It is widely used in applications such as smart home devices, health-monitoring smartwatches, smart meters tracking energy or water usage, automated irrigation systems based on soil moisture, livestock health and location monitor sensors, portable health monitors that collect and analyze patient health data like glucose meters or ECGs, tele-health solutions for remote diagnostics, and air quality sensors for pollution monitoring. For simple, low-power IoT and embedded systems, RV32I with 3-4 pipeline stages strikes a good balance between performance and efficiency. For applications requiring more intensive arithmetic operations, the RV32IM extension, which includes multiplication and division, is a suitable enhancement. Thus, for IoT Devices, RV32I or RV32IM with 3-4 stages is highly suitable to achieve fairly good performance with power and area efficiency. The high-end applications that handle data-intensive tasks may benefit from the RV64I/RV64IMAFD ISA and with 6 or more pipeline stages, though this comes at the cost of increased complexity, power consumption, and area usage.

This chapter focuses on designing area-optimized, ultra-low power RISC-V cores with varying pipeline stages to analyze the impact of pipelining on the overall performance and select the best core for designing a high-security RISC-V Crypto-core. The chapter is orga-

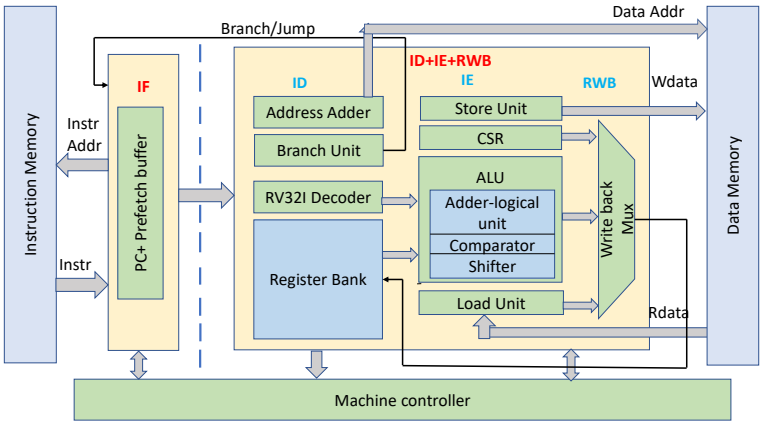
nized as follows: Section 3.2 focus on 1) design of three area-optimized ultra-low power cores based on RV32I: 2stage, 3stage, and 4stage which differ only in the number of pipeline stages. 2) design of HP 4stage: a high-performance 4-stage pipelined RISC-V core that utilizes Dynamic Branch Prediction (DBP) to achieve better performance, compared to the standard 4stage core. 3) Finally, converting the HP 4stage RV32I core to RV32IM core with multiplication and division extension for arithmetic-intensive applications. Section 3.3 analyze 1) the impact on area, power, maximum frequency, Dhrystone score, and overall performance of developed cores with increase in number of pipeline stages and pipeline techniques. 2) examining the effect of supply voltage variations and operating corners (best-case and worst-case) on these cores 3) assessing the effect of scaling on area and power while porting the proposed designs from 90nm to 40nm technology node. This structured approach aims to identify the best pipeline architectures for different IoT applications, culminating in the development of RISC-V Crypto-cores for secure, high-performance applications.

3.2 RISC-V Micro-architectures

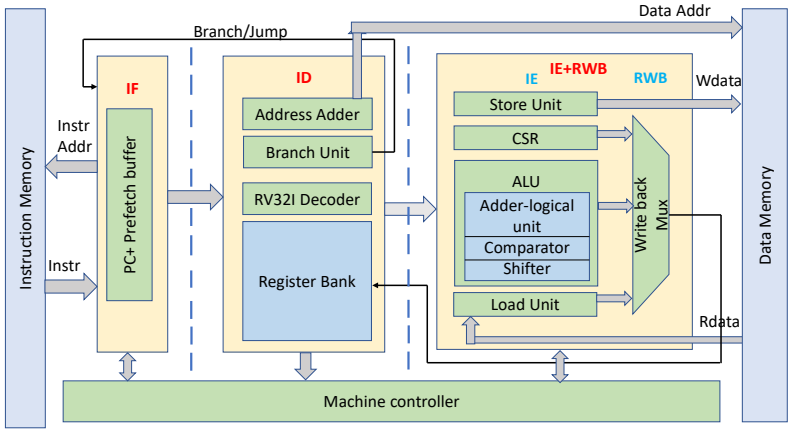
In this section, five different RISC-V architectures are detailed. The initial four proposed cores support the 32-bit base integer instruction format of RISC-V ISA (RV32I), comprising of 47 instructions. The 2stage, 3stage, and 4stage are pipelined RISC-V architectures comprising of two, three, and four pipeline stages, respectively. The only difference between these nearly identical cores is the addition of more buffers with the increase in the number of pipeline stages. Control hazards in these cores due to pipelining are removed by either branch forwarding or stalling. The comparative study of these cores offers a realistic assessment on the impact of the number of pipelining stages on the cores' overall performance. The HP 4stage RISC-V employs DBP and an additional write port for the register bank to prevent control hazards, boost operating frequency, and obtain high performance. It discusses how the additional features and pipeline strategies, such as stalling, branch forwarding, and DBP, might impact the overall performance of the RISC-V. The HP 4stage is extended with an M extension and its impact on the core's performance is evaluated. These cores could be used as the foundation for all advanced micro-architectures for supporting other extensions and features of the RISC-V ISA.

3.2.1 2stage core: 2-Stage Pipelined RISC-V Core

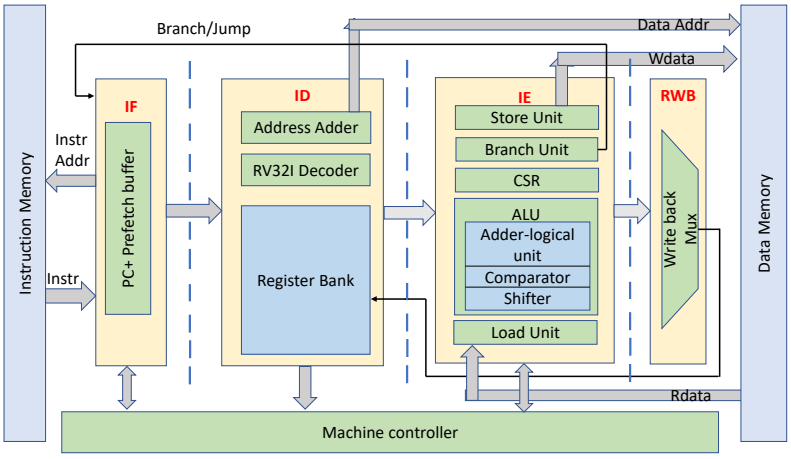
The 2-stage pipelined RISC-V microarchitecture is depicted in Figure. 3.1a. It consists of two stages: the first stage is Instruction Fetch (IF) and the second stage is Instruction Decode and Execute with Register Write Back (IDE-RWB). The IF stage fetches the instructions from instruction memory using a pre-fetch buffer and stores them until the next stage is ready to process them. The Program Counter (PC) is basically incremented by four in the normal execution of instructions or updated with a new target address when conditional or unconditional jump instruction occurs. When branch and jump instructions occur, the pipeline is stalled and the PC is not updated until the branching condition is evaluated in order to eliminate control hazards. Since it is a 2- stage pipelined architecture, the penalty of control instructions is only one. The IDE+RWB stage decodes instructions, reads operands from the register bank, executes them using the ALU, and writes back results into the register bank using Write Back (WB) Mux. The Athematic Logical Unit (ALU) has a 32-bit adder, a 32-bit shifter, and a logic unit to perform arithmetic and logical operations. Control Status Register (CSR) are used to check the status of control registers and to calculate the



(a) 2-stage RISC-V core



(b) 3-stage RISC-V core



(c) 4-stage RISC-V core

Figure 3.1: Various RISC-V Core architectures based on the number of pipeline stages

number of cycles, the number of instructions per program, etc. The load-store unit is used to access the memory. It either loads the data into the registers or stores the data from the register in memory. Except for the Load-Store unit, no other unit is interfaced with

the memory. The ALU results, CSR values, or data from the load unit are written back to the destination register of General Purpose Registers (GPRs) using WB Mux. This 2-stage RISC-V architecture reduces synchronized buffers for pipelining, which reduces area and clock-controlled switching, resulting in less overall power consumption.

3.2.2 3stage core: 3-Stage Pipelined RISC-V Core

A 3-stage pipelined RISC-V micro-architecture is shown in Figure. 3.1b. The first IF stage contains a pre-fetch buffer and PC. The second ID stage contains a decoder, address adder, register bank, and branch unit. The third IE+RWB stage contains an ALU, CSR, load unit, store unit, and a WB Mux. The control hazards, caused by control and transfer instructions are eliminated by using branch forwarding technique. The branch penalty of the usual 3-stage pipeline processor is two. In the proposed 3stage core, by forwarding the execution of the branch instruction to the ID stage, the branch penalty is effectively reduced to one and thereby boosting the performance. Almost all instructions are performed in three stages. The first stage fetches the instructions and updates the PC. The second stage decodes the instructions and generates corresponding control signals. The third stage executes requisite operations using ALU or CSR based on the control signals generated in the second stage. The Write-back Mux selects either the ALU result, CSR values, load data from memory, or PC return address that is to be stored in the register bank. In Load-store instructions, the 2^{nd} stage performs decoding of the instructions and computation of the memory address. The address adder in the decode stage is utilized to compute the memory address, which allows load-store instructions to be finished in one clock cycle. In this design, a separate branch unit in the ID stage will evaluate the branch condition and generate a branch-taken signal prior to the IE stage, which reduces the number of clocks required for branch instruction and thereby increase the IPC of the core.

3.2.3 4stage core: 4-Stage Pipelined RISC-V Core

A compact 4-stage pipelined RISC-V micro-architecture is shown in Figure. 3.1c. Instructions are fetched in the IF stage, decoded in the ID stage, executed in the IE stage, and the results are written back into the GPRs using the WB Mux in the Register Write Back (RWB) stage. The previously developed 3stage core is converted into a 4stage core by adding buffers in between the execution unit and register write-back unit. To eliminate control hazards, the stalling technique is employed rather than the branch forwarding technique. The increase in pipeline stages help to decrease the critical path delay, which raises the maximum operating frequency. Since stalling technique is used, the penalty of pipelining is 3 and hence IPC is reduced.

Table 3.1: Impact of pipeline stages on the designed cores

Cores	**Area** [μm^2]	**Dyn. power** [μ**W**]	**Max. Freq** [**MHz**]	**Dhrystone** [**DMIPS/MHz**]	**FOM** [$\frac{(DMIPS/MHz).MHz}{uWmm\hat{2}}$]
2stage	23355.00	901.2	100.40	1.81	8.63
3stage	25367.76	1048.7	128.87	1.86	9.01
4stage	27927.72	1205.8	139.66	1.34	5.56

* Dynamic power at 100 MHZ operating frequency

The effect of the number of pipeline stages on the overall efficiency of cores is demonstrated in Table 3.1. The area gets increased by 19.2% from 2stage to 4stage core, which is mainly

due to an increase in buffers. The dynamic power for 4stage was found to be increased by 33.7% than 2stage due to larger clock switching of synchronized buffers. As pipeline stages increased, 4stage achieved 39.1% higher operating frequency than the 2stage. The increase in operating frequency is due to the reduction of critical path delay resulting from the inclusion of buffers. In summary, based on above results, it is inferred that as the number of pipeline stages increases, the maximum operating frequency also rises, but at the cost of increased area and power.

3.2.4 HP 4stage core: High-performance 4-Stage Pipelined Core

Figure. 3.2a depicts the micro-architecture of a High-performance 4-stage pipelined RISC-V. The IF stage fetches the instructions and updates the PC. It has a 2-bit Dynamic Branch Predictor (DBP), which will evaluate the actual branching behaviour by recording the recent past history of branching in a Branch History Table (BHT) and make predictions assuming future behaviour will be the same. If the prediction goes wrong, it updates the BHT with new values and stalls the pipeline all the way while re-fetching the instructions. A 2-bit prediction counter is used to weigh up the accuracy of predictions and hardware complexity. The four states of the 2-bit prediction counter are strongly branching not taken, weakly branching not taken, weakly branching taken, and strongly branching taken. Branch prediction is assumed to be false in the first two states and true in the last two states. If the prediction goes wrong, the whole pipeline is flushed, and the correct instruction is loaded again. The ID stage has a decoder and a register bank. The register bank has two read ports as well as two write ports. If two write requests come in the same clock cycle, the requests are allotted to the two write ports of the register bank and thus avoiding stalling and thereby increasing the performance. Unconditional jump instructions don't require any register write or read operations and hence it is forced to finish in the decode stage itself and thereby eliminating the requirement of flushing in the execution stage for the unconditional jump, which in turn improves core performance. The IE stage has an ALU that performs addition, subtraction, logical operations, and address calculation; a Comparator to perform signed and unsigned comparisons; Barrel shifters for shift operations; a Data controller which generates control signals for memory read/write operations; Branch unit for generating branch enable signals for branching and CSR. The fourth stage, Register Write (RW) is used basically to write back the results after execution or to load data from memory into the register bank. It has a load-store unit to load and store data into registers from memory and vice-versa. Almost all of the instructions are completed in four stages. The same ALU in the IE stage, which performs arithmetic and logical operations, is used for calculating memory read-write address and branch target address.

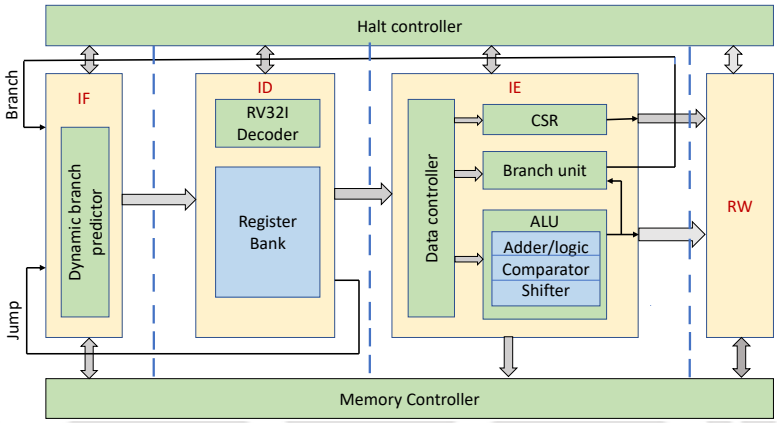
Table 3.2: Comparison of HP 4stage with a basic 4stage core

Cores	**Area** [μm^2]	**Dyn. power** [μ**W**]	**Max. Freq** [**MHz**]	**Dhrystone** [**DMIPS/MHz**]	**FOM** [$\frac{(DMIPS/MHz).MHz}{uWmm\hat{2}}$]
4stage	27927.72	1205.8	139.66	1.34	5.56
HP 4stage	39774.96	1382.6	186.92	1.68	5.71

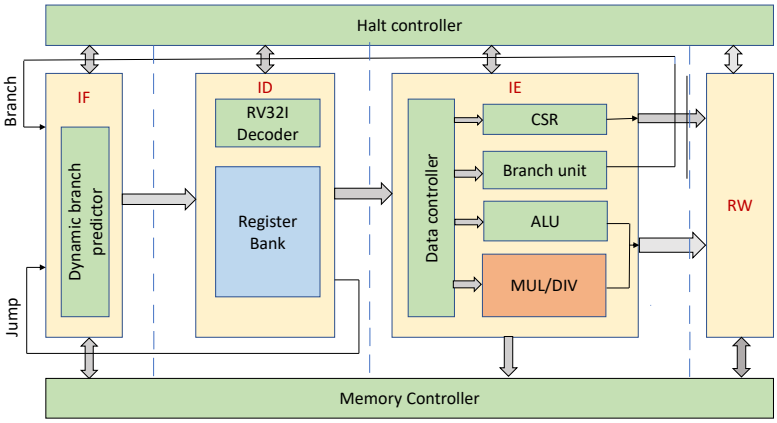
* Dynamic power at 100 MHZ operating frequency

The HP 4stage, utilizes DBP and additional write ports for the register bank to yield better performance and higher Dhrystone score. These improvements are done in 4stage core as it achieves the highest frequency among prior designs. While both cores have four

pipeline stages, the basic 4stage uses stalling technique with a single write port, whereas the HP 4stage uses more sophisticated pipeline techniques like DBP and an additional write port for better performance. The HP 4stage's advanced techniques enable a 33.83% increase in frequency and a 25.3% improvement in Dhrystone score. However, these enhancements come with a 14.66% increase in power consumption and a 42.6% larger area than the basic 4stage core, as shown in Table 3.2. The 4-stage HP RISC-V core consumes more area and dynamic power than the baseline 4-stage pipelined RISC-V core, but this is compensated by its significantly higher operating frequency and Dhrystone score, as shown in Table 3.2. The performance improvements are achieved by key architectural enhancements, including the integration of a 2-bit DBP in the IF stage, the elimination of the address adder from the ID stage to minimize critical path delay, and the addition of extra write ports in the register bank to improve performance. The Figure of Merit (FoM) reflects this trade-off, with the HP core attaining a slightly higher FoM of 5.71 compared to 5.56 for the baseline, despite its increased resource utilization.



(a) HP 4-stage based on RV32I



(b) HP 4-stage based on RV32IM

Figure 3.2: HP 4-stage: High-performance 4-Stage Pipelined Cores

The baseline 4-stage RISC-V core is well-suited for low-power IoTe applications like sensors and fitness trackers, where energy efficiency and compactness are key factors. In contrast, the HP 4-stage RISC-V core is ideal for performance-intensive IoTe tasks such as smart home devices, industrial controllers, and advanced health monitors, where speed and real-time processing are crucial.

3.2.5 HP 4stage core with Multiplication extension

The HP 4-stage core, based on the RV32I architecture, is extended with integer multiplication and division capabilities to develop the HP 4-stage RV32IM, tailored for arithmetic-intensive applications as shown in Figure. 3.2b. This M extension integrates a multiplier and divider hardware unit in the execution (IE) stage, with decoder modifications to support eight additional instructions (four for multiplication and four for division). Since 32-bit multiplication and division are expensive process in terms of Critical Path Delay (CPD), they are executed over multiple clock cycles to minimize the CPD. The RV32IM architecture employs a 2-cycle Baugh-Wooley signed multiplier: in the first cycle, partial products are computed and buffered, and in the second cycle, these results are added to produce the final product. For division, a Radix-16 divider unit is used, which requires eight clock cycles to complete each operation. This multi-cycle approach ensures efficient handling of multiplication and division within the architecture while maintaining slight degradation in frequency.

Table 3.3: Comparison of HP 4stage with M extension (RV3IM) with HP 4stage (RV32I)

Cores	**Area** [μm^2]	**Dyn. power** [μ**W**]	**Max. Freq** [**MHz**]	**Dhrystone** [**DMIPS/MHz**]	**FOM** [$\frac{(DMIPS/MHz).MHz}{uWmm\hat{2}}$]
HP 4stage (RV32I)	39774.96	1382.6	186.92	1.68	5.71
HP 4stage (RV32IM)	63142.56	1840.5	168.92	1.71	2.49

* Dynamic power at 100 MHZ operating frequency

Table 3.3 summarizes the changes in area, power, maximum operating frequency, and performance while upgrading the HP 4-stage RV32I core to RV32IM. Extending the HP 4-stage RV32I core to RV32IM increases area by 58.75% and power by 33.12%. The multiplier and divider circuits have larger critical path than other components like the ALU and CSR, which reduces the maximum operating frequency of the RV32IM core compared to the RV32I core. However, this addition of hardware for multiplication and division significantly boosts benchmark performance. The Dhrystone score (DMIPS/MHz) shows a slight improvement from 1.68 to 1.71, as it involves more control-transfer instructions than multiplication or division. In contrast, the CoreMark score (CoreMark/MHz) increases substantially from 2.74 to 3.57, due to the higher presence of multiplication and division operations in CoreMark. This enables the RV32IM core to complete the CoreMark benchmark program with fewer instructions and clock cycles, leading to a notable performance gain.

3.3 Results and Discussion

The processor cores designed with different pipeline stages are integrated with 32KB memory along with a UART module for data transmission. A comparative analysis was conducted on performance metrics such as dhrystone score, maximum frequency, dynamic power, and hardware utilization, along with their variations across technology nodes and supply voltages.

The whole processor was designed using Verilog HDL. Various C programs like factorials, prime factors, matrix multiplications and small benchmark programs like Dhrystone, Embench programs, etc, were converted into hex files using RISC-V GNU tool-chain and RISC-V cross-compilers. These hex files were loaded into Block RAM, and the outputs for the C programs from the FPGA were validated by porting the results to the Tera-term using UART. Cores were synthesized at the UMC 65nm node using Synopsys tools for accurate power, area, and efficiency metrics, as shown in Table 3.4. Despite different pipeline stages,

Table 3.4: ASIC Implementation results of the Developed Cores

Various Parameters	Different RISC-V and commercial cores										
	Developed RISC-V cores					Existing RISC-V cores					
						[77]	[12]	[78]	[79]	[80]	[12]
Processor Cores	2stage RV32I	3stage RV32I	4stage RV32I	HP 4stage RV32I	HP 4stage RV32IM	RISC-V basic	RI5CY	Open RISC	CortexM3 ARMT32	SiFive E24 RV32IMAFC	Zero Riscy
Technology Node	65nm	65nm	65nm	65nm	65nm	65nm	65nm	65nm	90nm	55nm	65nm
Vdd[V]	1.08	1.08	1.08	1.08	1.08	1.08	1.2	1.2	1.2	NA	1.2
Area[mm2]	0.0234	0.0253	0.0279	0.0398	0.0631	0.068	0.0586*	0.064	0.119	0.083	0.0272*
Area[kGE]	16.22	17.62	19.39	27.62	43.85	NA	40.7*	NA	NA	NA	18.9*
Freq[MHz]	100.40	128.87	139.66	186.92	168.92	357	333	362	216	210	333
Dynamic Power[µW/MHz]	9.012	10.487	12.058	13.826	18.405	28.68	24.9	33.8	32.8	67.8	11
Dhrystone[DMIPS/MHz]	1.81	1.86	1.34	1.68	1.71	NA	NA	NA	1.52	1.38	NA
Coremark/MHz	1.27	1.31	1.53	2.74	3.57	2.43	NA	2.66	3.4	3.1	NA
FoM [(DMIPS/MHz)(MHz)/(µWmm2)]	8.63	9.01	5.56	5.71	2.49	NA	NA	NA	NA	NA	NA

*area is derived from a 100MHz target netlist, -Dynamic power is normalized w.r.t clock frequency and +NA means Not Available

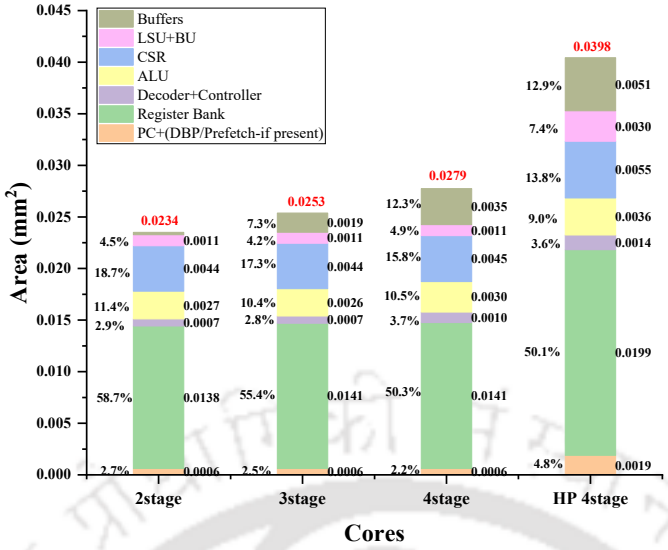
the 2stage, 3stage, and 4stage cores had a smaller area than most listed cores due to simple pipelining techniques and minimal additional hardware, though with lower frequency. The designed HP 4stage core occupies larger area than the Zero-Riscy core [12] due to its additional pipeline stages, use of DBP techniques, and multiple write ports in the register banks to enable parallel data writes. These features reduce CPD, increase frequency, and improve performance, achieving higher Instructions Per Cycle (IPC). It also illustrates that the dynamic power of all developed cores is lower than that of the majority of existing cores due to the selective enabling of the required modules and simple pipeline techniques.

Area

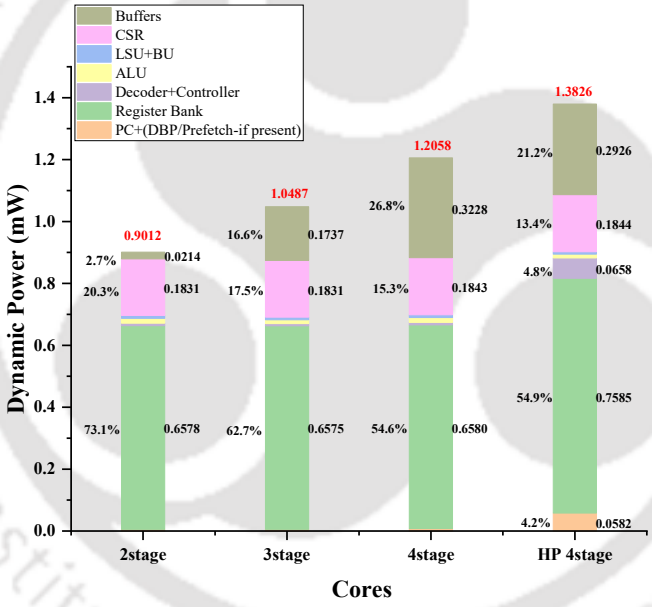
Figure. 3.3a shows that the HP 4-stage core occupies the largest area, at 1.7x of the 2-stage core. This increased area is due to its DBP and additional write ports in the register bank. In contrast, the 2stage, 3stage, and 4stage cores use smaller pre-fetch buffers, resulting in more compact designs. As the number of pipeline stages increases, more buffers are required, which in turn results in increased area. The register bank contributes approximately 50% of the total area across all cores. Buffer area also increases with pipeline stages, from 0.9% in the 2-stage to 12.9% in the HP 4-stage core. Additionally, the HP 4-stage core's fetch unit area doubles from 2.2% to 4.8% due to the replacement of the pre-fetch buffer with DBP.

Power

The 2-stage core has the lowest dynamic power, being 1.5x lower than the HP 4-stage, 1.3x lower than the 4-stage, and 1.2x lower than the 3-stage core, due to its minimal pipeline stages, resulting in fewer buffers, and reduced clock switching. Figure. 3.3b shows the total power distribution over various components of each core while operating at 100MHz. It has been observed that the maximum power is consumed by the register bank. As compared with other developed cores, HP 4stage core had larger power consumption in the IF unit and decoder-controller unit. This is mainly because, the IF unit of HP 4stage uses advanced branching techniques like DBP instead of pre-fetch buffers, which results in a significant consumption of power. The power consumption of buffers was observed to decrease drastically with the decrease in the number of pipeline stages. This is due to the fact that with a decrease in the number of pipeline stages, the number of buffers also decreases resulting in reduced clock switching. Hence, for ultra-low power and less-intensive IoT applications, it is always better to go for a core with fewer pipeline stages.



(a) Area distribution

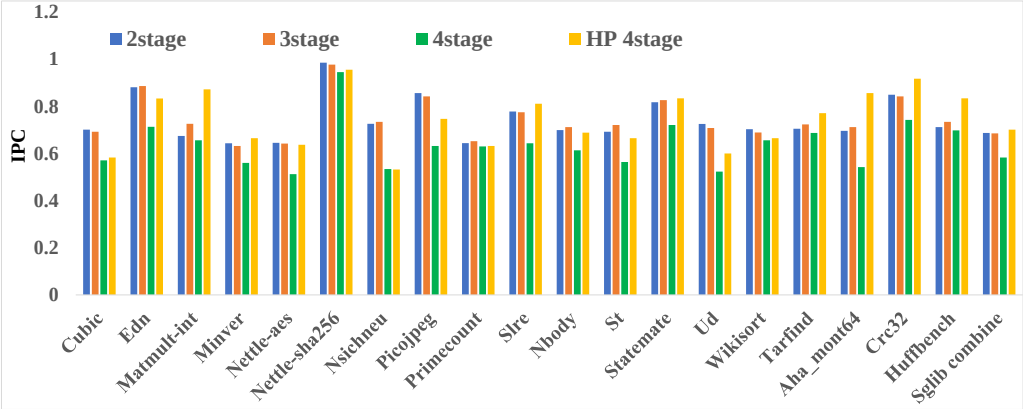


(b) Power distribution

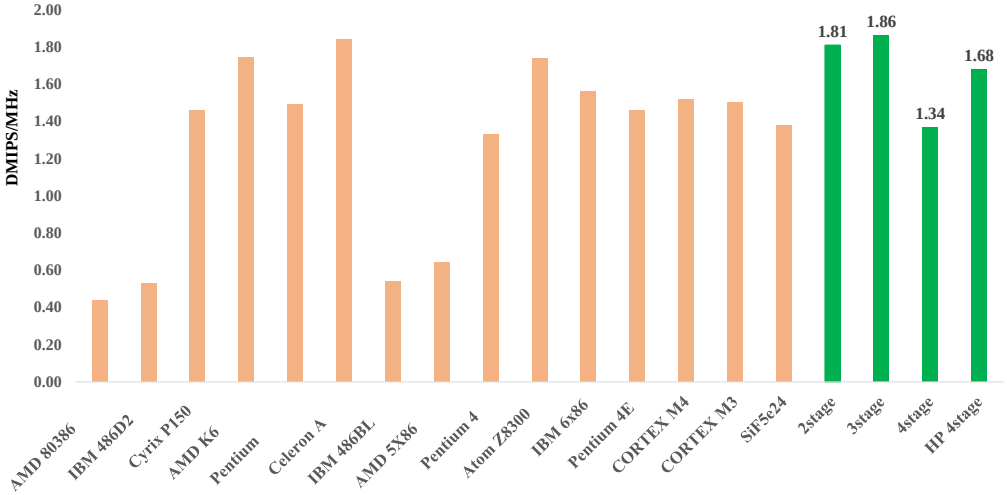
Figure 3.3: Area and Power distribution of the different cores

Workloads and Performance

Various benchmark programs like Dhrystone and Embench benchmark programs were run on the developed cores for measuring performance. Dhrystone (DMIPS) is a synthetic benchmark program developed by Reinhold P. Weicker that contains a set of integer programs. It was widely used for quite a long time in the industry and academy for evaluating the performance of cores. Embench programs are open-source lightweight benchmark suites used for the performance evaluation of IoT embedded cores. It contains various actual application programs which intensively perform branch, memory, and integer compute-type instructions.



(a) IPC of various Embench programs of developed core



(b) Comparison of Dhrystone scores of developed cores

Figure 3.4: Workloads and Performance of developed cores

The IPC of all the developed cores for different Embench programs was evaluated and the results of the comparison are shown in Figure. 3.4a. It implies that with the increase in the number of pipeline stages, the IPC decreases. The 4stage has the least IPC as it is a 4-stage pipelined architecture and uses simple stalling technique for conditional and unconditional jumps. The 3stage core achieved almost equal IPC of 2stage for most Embench programs because of the branch forwarding from the IE stage to the ID stage and the additional address adder for the memory address calculation. For control transfer programs, HP 4stage had better IPC than 4stage as the DBP was used instead of stalling technique. 3stage core with a Dhrystone score of 1.86 DMIPS/MHz outperforms the other cores in performance evaluation, as shown in Figure. 3.4b. This is because the address calculation and branching condition evaluation are carried out in the decoding stage itself, using a separate address adder and branch unit instead of the ALU in the IE stage. 2stage had the lowest operating frequency, followed by 3stage, 4stage, and HP 4stage. As expected, with an increase in the number of pipeline stages and the use of DBP, the critical path delay of HP 4stage decreased, resulting in it to emerge with the maximum operating frequency among the developed cores.

An additional parameter termed Figure of Merit (FoM) is used to evaluate the overall performance of the cores. The FoM is directly proportional to the operating frequency and Dhrystone score whereas it is inversely proportional to area and power. The FoM of all the

developed cores is calculated using the equation below;

$$FoM = \frac{(Maximum\ Frequency) * (Dhrystone\ score)}{(Area) * (Dynamic\ power)}$$

The best core must possess the maximum frequency and the highest Dhrystone, coupled with the least area, and power that in turn reflects in higher FoM.

Table 3.5: Power estimation of Developed cores at 100MHz

Cores	PDynm-PLkg @0.9V	PDynm-PLkg @1.08V
2stage	0.606 - 4.652	0.901 - 7.14
3stage	0.715 - 4.92	1.048 - 7.68
4stage	0.8296 - 5.47	1.206 - 8.49
HP 4stage	0.943 - 6.85	1.382 - 11.25

*PDynm-Dynamic Power(mW) and PLkg-Leakage Power(μW)

The effect of technology scaling and supply voltage variation on Area and Power

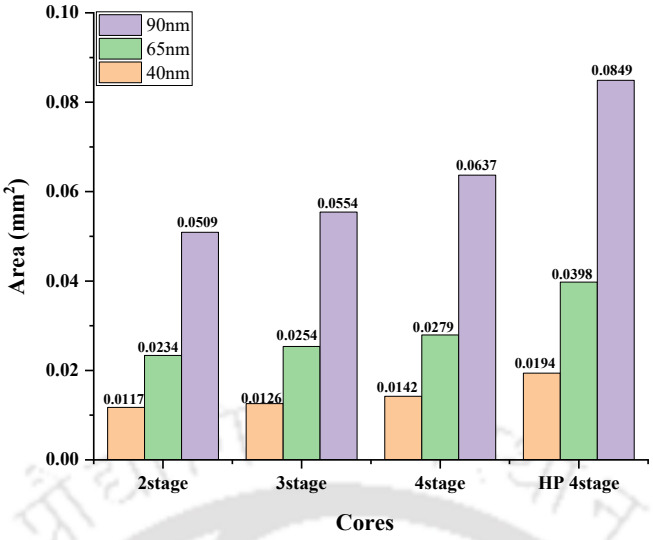
The dynamic power and leakage power variation with the change in supply voltage is depicted in Table 3.5. As the supply voltage was increased from 0.9 V to 1.08 V, the dynamic power requirement for all the developed cores went up by 1.4x times. The leakage power also got increased around 1.6x times for all cores. This is because the dynamic power is directly proportional to the square of the supply voltage.

Technological dependence on area and power was evaluated for all the developed cores. Each core was initially designed using UMC 65nm technology, which was later ported to UMC 40nm technology as well as UMC 90nm technology. All the developed cores had the least area in the 40nm technology node, which was 4.4x lower area than the 90nm technology node and 2.1x lower area than the 65nm technology node, respectively, as shown in Figure. 3.5a. The scaling of the technology node basically reduces the dimensions of MOSFETs, which results in a significant reduction in the area of cores. The variation of power consumption across the technology node while all cores are operating at 100MHz is depicted in Figure. 3.5b, and it is observed that the power of all the cores got reduced by around 2.4x when the designs were scaled down from 90nm technology to 40nm technology. As the technology was scaled down, the transistor size and supply voltage decreased, which in turn reduced dynamic power.

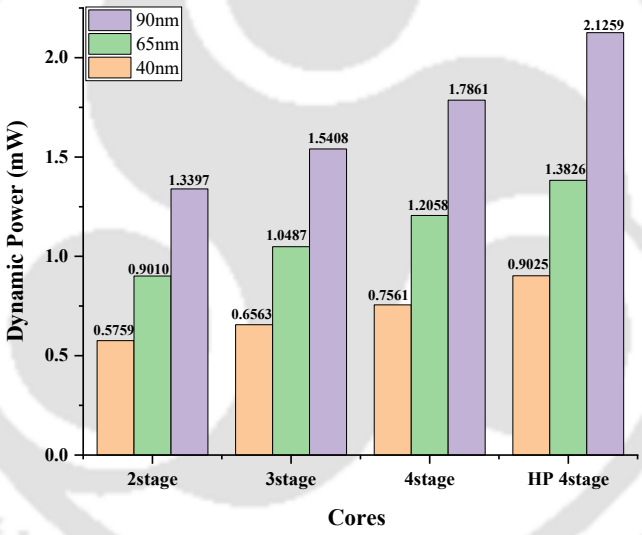
Performance variation for the designed cores across the process corners

The developed cores are tested in both best-case (fast-fast) and worst-case (slow-slow) process corners. Analyzing performance variation across process corners is essential for ensuring that the designed cores meet timing, power, and functional requirements under diverse manufacturing, temperature, and voltage conditions. This evaluation helps to confirm that the developed cores remain reliable and efficient across best-case (BC), worst-case (WC), and typical conditions, which is critical for optimizing yield and ensuring consistency in production. Ultimately, corner analysis ensures that the design operates securely and reliably in real-world applications, minimizing the need for costly adjustments.

Table 3.6 details the performance variations of all developed cores across different process corners. In the best-case corner, the cores achieve a higher maximum operating frequency but consume more power compared to the worst-case corner. Conversely, in the worst-case corner, the cores consume less power, though their maximum operating frequency is reduced



(a) Area dependency on Technology Scaling



(b) Power dependency on Technology Scaling

Figure 3.5: Area and Power Dependency of the developed cores on Technology Scaling

across both corners, the cores show comparable area with only slight variation. Similar trends in area, power, and maximum frequency were observed for all developed cores in both the worst-case and best-case corners. Among the designs, the 2stage core has the smallest area and power consumption, followed by the 3stage, 4stage, and HP 4stage cores. However, the 2stage core also has the lowest maximum operating frequency, while the 3stage, 4stage, and HP 4stage cores achieve progressively higher frequencies.

The pipeline stage numbers and pipelining techniques play a pivotal role in the design of a processor. The effect of the number of pipeline stages on the efficiency of cores is demonstrated by 2stage, 3stage, and 4stage cores. HP 4stage primarily illustrates the effects of pipeline techniques on the variation of the overall parameters of the cores. Figure. 3.6

Table 3.6: Performance variation for the designed cores across the process corners

Processor Cores	Area [µm2]	Area [kGE]	Freq [MHz]	Dynamic Power [µW/MHz]
Worst-Case (WC) corner				
2stage RV32I	23355.00	16.22	100.40	9.012
3stage RV32I	25367.76	17.62	128.87	10.487
4stage RV32I	27927.72	19.39	139.66	12.058
HP 4stage RV32I	39774.96	27.62	186.92	13.826
HP 4stage RV32IM	63142.56	43.85	168.92	18.405
Best-Case (BC) corner				
2stage RV32I	23483.88	16.31	193.05	9.993
3stage RV32I	25227.72	17.51	239.81	11.618
4stage RV32I	28779.40	19.99	251.89	14.358
HP 4stage RV32I	39842.72	27.67	298.51	15.324
HP 4stage RV32IM	63200.52	43.89	265.96	20.434

clearly portrays the effect on area, dynamic power, and FoM of the designed cores with pipeline stages and pipeline techniques. A designer can achieve a greater operating frequency and hence a higher overall throughput by increasing the number of pipeline stages, but at the expense of additional area and power. Along with the increased number of pipeline stages and use of modern pipeline techniques like DBP, the HP 4stage was able to achieve almost 86% higher frequency than the 2stage core, but dynamic power consumption and area went up by 53.4% and 70%, respectively. The data and control hazards also got increased with the increase in pipeline stages, thereby reducing the IPC and Dhrystone scores. The above-said curtailing effect in Dhrystone and IPC was reversed in HP 4stage core with the utilization of modern pipeline techniques.

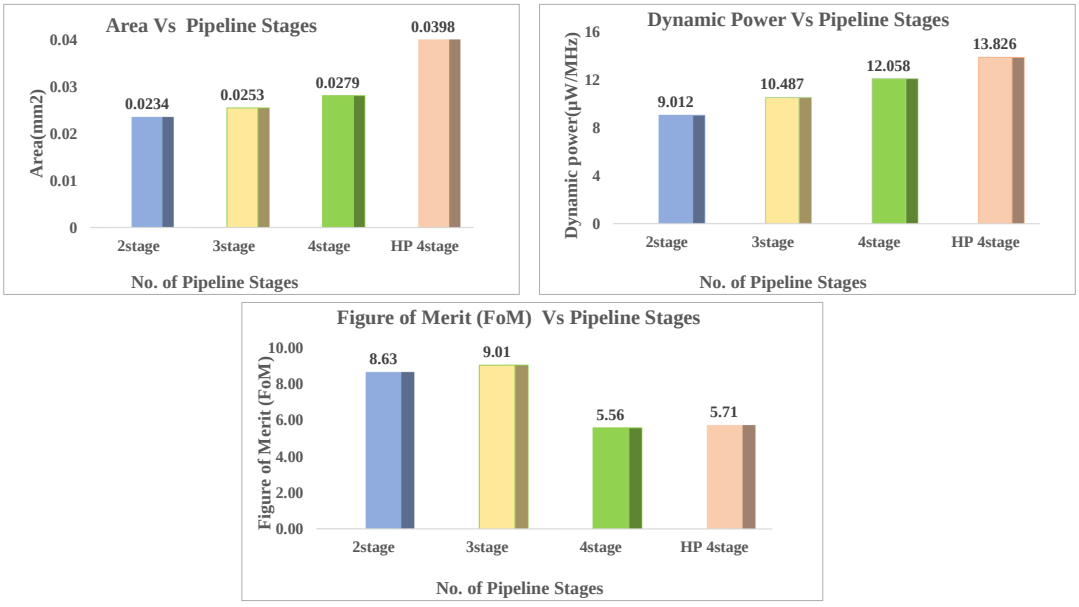


Figure 3.6: The overall performance of the designed cores

3.4 Chapter summary

This chapter presents RISC-V micro-architectures optimized for low-power applications, focusing on variations in pipeline stages and techniques. These cores outperform processors like the Cortex-M4, Cortex-M3, and IBM6X86 in performance, area efficiency, and dynamic power consumption. The 2-stage core, with its minimal pipeline, has the smallest area and lowest dynamic power, while the 3-stage core, utilizing address adders and branch forwarding techniques, achieves the highest Dhrystone score. The HP 4-stage core, equipped with a 4-stage pipeline and utilizes additional techniques like DBP, attained the highest operating frequency among the developed cores. For arithmetic-intensive applications, the HP 4-stage core is extended from RV32I to RV32IM, enhancing performance but at the expense of area and power overheads and a slight frequency reduction, due to additional multiplier divider hardware modules. This demonstrates the importance of tailoring RISC-V micro-architectures and extending the ISA to meet application-specific needs. With the rise of IoT devices and their vulnerability to attacks, securing embedded processors is increasingly crucial. In the following chapters, we will explore the ways to enhance security by integrating cryptographic modules, expanding the RISC-V ISA with cryptographic instructions, or incorporating a cryptographic accelerator IP.

CHAPTER

4

COMPARATIVE ANALYSIS OF AES HARDWARE INTEGRATION TECHNIQUES WITH RISC-V

Contents

4.1 Introduction

In power-constrained IoTe devices, integrating AES hardware module with RISC-V offers a more efficient solution than running AES algorithms purely in software on the RISC-V processor. This integration can be achieved by either tightly coupling the AES module with RISC-V through custom Instruction Set Extensions (ISE) or loosely coupling it as an AES accelerator IP. This chapter presents two methods for integrating AES with 32-bit RISC-V core and analyzes their impact on hardware utilization and security metrics. The first method extends the RISC-V ISA with custom AES instructions, while the second integrates AES as an address-enabled accelerator IP. Both approaches achieve reductions in energy consumption and encryption time compared to pure software AES-based solution, enhancing overall performance. A 128-bit, 12-clock AES based on the Masoleh S-box, is integrated in both methods, achieving minimal area and power overhead. Validation on UMC 65nm technology showed comparable performance metrics and resource efficiency across both designs. Notably, while both methods are effective, the AES accelerator IP offers greater flexibility, making it preferable for securing future multi-core RISC-V SoCs against PAA.

With the advent of Artificial Intelligence (AI) in IoTe, vast amount of data are processed at edge devices, making data security a critical concern. Unsecured data can lead to inaccurate AI predictions and security breaches. AES encryption safeguards data integrity and reliability in critical fields like healthcare, smart cities, and autonomous systems. In software-based AES implementations [40], data encryption often requires thousands of clock cycles, leading to increased energy consumption and longer processing times. In contrast, dedicated AES hardware accelerators [81, 82] significantly reduce clock cycles, providing an efficient alternative. Extending the processor's ISA to support AES operations directly, as seen in Intel's AES-NI [83] and ARM's Armv8-A [30] AES instructions, is an effective method for enhancing encryption performance.

RISC-V, a flexible open-source architecture is a popular choice for IoTe devices due to its compact, energy-efficient design and customizable ISA. To secure RISC-V cores in IoT devices, AES accelerators can be either tightly or loosely integrated. Previous work in literature demonstrates various approaches of integrating AES with RISC-V. Ustav et.al [31] presents a RISC-V SoC with cryptographic accelerator to execute different cryptography algorithms such as AES, Elliptic Curve Cryptography (ECC) and the Secure Hash Algorithm (SHA). An advanced AES accelerator supporting multiple AES modes is integrated with RISC-V as an IP in [32]. The tight coupling of an AES accelerator with RISC-V via custom instructions, offers better latency but requires more complex integration. The AES coprocessor in [33], uses Direct Memory Access (DMA), and enhances the Hummingbird E203 core for concurrent data processing. A custom S-box instruction in [34] reduces energy consumption by up to 100 times compared to software-based AES. By the integration of AES hardware, FAC-V [35] shows significant performance improvements. Tightly or loosely integrating AES with RISC-V can significantly reduce memory access and energy consumption compared to software-based AES solutions. Despite the use of software and hardware cryptographic techniques, IoTe devices still remain vulnerable to Power Analysis Attacks (PAA) [3]. Software-based cryptography on general-purpose processors was initially preferred for its flexibility and lower hardware cost, but is more susceptible to attacks than hardware implementations. Standalone AES hardware implementations [84] are also extremely vulnerable as large quantum of information leakage is derived from the power traces while carrying out cryptographic operations. Integrating AES with RISC-V core, makes PAA more challenging because all the components share a significant amount of off-chip power, which makes it diffi-

cult to isolate the significant, exploitable power of cryptographic engines from unexploitable power of various components.

This chapter provides an in-depth analysis of PAA vulnerabilities of various AES integration techniques with RISC-V in terms of hardware cost, security metrics, and performance. The chapter is organized as follows: Section 4.2 details the two AES integration techniques. In Integration Technique I, the AES module is embedded directly within the RISC-V microarchitecture, extending the ISA to support encryption instructions. In Integration Technique II, the AES module is implemented externally as an address-enabled accelerator IP. Section 4.3 then evaluates the hardware performance and security metrics for both integration techniques and helps to determine the most effective approach for resource-constrained IoTe.

4.2 Various AES integration methods with RISC-V

RISC-V's extendable ISA makes it ideal for integrating dedicated and custom-tailored hardware devices for specific applications. Integration of AES with a RISC-V processor improves performance and efficiency compared to running AES algorithms on the processor. Loosely coupled setups use a memory-mapped interface for co-processors, while tightly coupled designs extend the ISA with custom instructions to enable co-processor functionality.

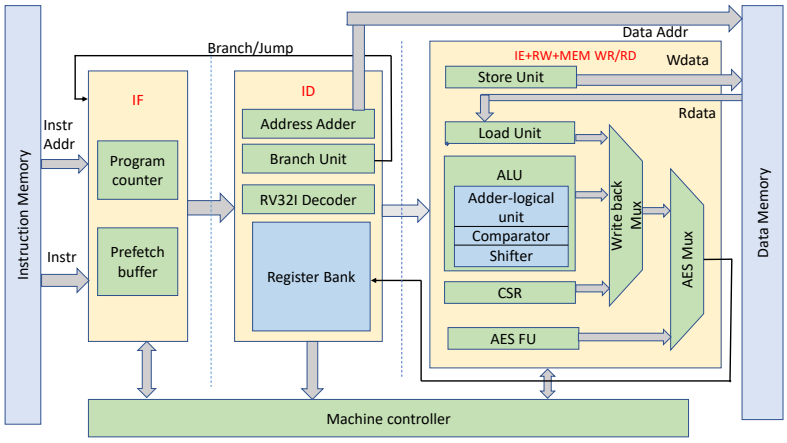
4.2.1 RISC-V Crypto-core with ISA extensions for encryption

In this implementation, the AES functional unit (FU) is tightly integrated within the execution stage (IE) of the RISC-V core, leading to efficient communication between the processor and the AES FU using custom instructions of type R. These instructions handle the transfer of data between the general-purpose registers of the core and the AES FU. The two AES custom instructions, AESLP and AESSC, which are added to RISC-V ISA are presented in Table 4.1. The AESLP loads 32 bits of the 128-bit plaintext into the AES FU, depending on the rd value of the AESLP instruction. Once the complete 128-bit plaintext has been loaded into the AES FU, the AES enable signal is activated, initiating the encryption process. Once the ciphertext is ready, a Data Valid (DVld) signal is activated, and AESSC stores each 32-bit segment of the 128-bit ciphertext back into the register bank. Since the AES FU is incorporated into the execution stage, it is crucial that the area, power consumption, and critical path delay of the AES design remain minimal to ensure there is no degradation in processor performance. To achieve this, a 12-clock rolling AES architecture based on the Masoleh S-box has been integrated with the RISC-V core, as illustrated in Figure. 4.1.

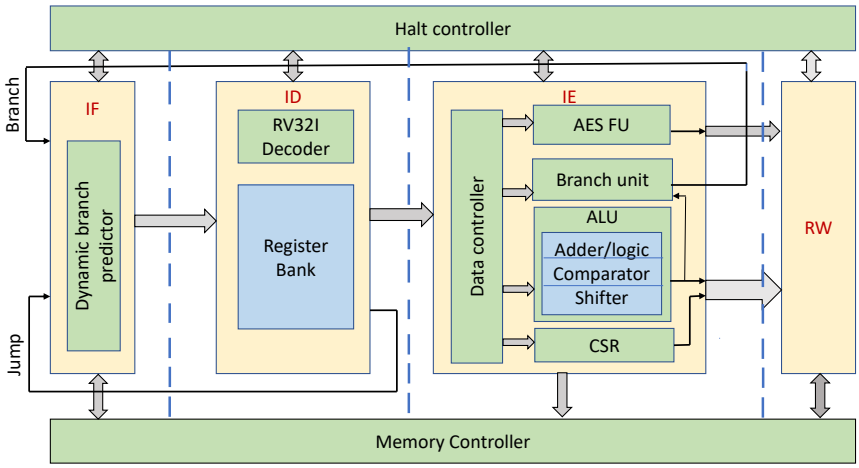
Table 4.1: Customized AES instructions for RISC-V

Instr	**31-25**	**24-20**	**19-15**	**14-12**	**11-7**	**6-0**
	Func7	**rs2**	**rs1**	**Func3**	**rd**	**Opcode**
AESLP	0010000	rs2	rs1	110	rd	1101011
AESSC	0010000	rs2	rs1	111	rd	1101011

The HP 4stage RISC-V Crypto-core, with a 12-clock AES FU, requires 20 cycles for encryption: 12 cycles for the AES operation and 8 cycles to transfer 32-bit chunks of plaintext and ciphertext between the register bank and AES FU. Similarly, the 3-stage RISC-V Crypto-core needs 23 cycles: 12 for AES and 11 for synchronization and data transfer. The AES C programs, as shown in algorithm 1 are modified to utilizing these new custom instructions and



(a) 3-stage RISC-V Crypto-core



(b) 4-stage HP RISC-V Crypto-core

Figure 4.1: RISC-V Crypto-core with ISA extensions

the RISC-V compiler is recompiled to support these modifications. This integration improves the processor's inherent resilience to PAA by leveraging the activity of other functional units (FUs) in the execution (IE) stage. While the AES FU is active, these other FUs in RISC-V continue to perform operations without updating the register bank. This additional activity, contributes to additional power which helps to obscure the power signature of AES, making it more difficult for attackers to isolate and analyze the AES power traces. A more detailed explanation of this integration technique (RISC-V Crypto-cores with ISA extensions) is provided in Chapter 5.

Algorithm 1 Snippet for AES encryption based on HW pause

```
int pt4 = 0x3243f6a8;
int pt3 = 0x885a308d ;
int pt2 = 0x313198a2 ;
int pt1 = 0xe0370734 ;
asm volatile ("aeslp zero, %[b], %[c] ;": : [b] "r" (pt2), [c] "r" (pt1) );
asm volatile ("aeslp %[a], %[b], %[c] ;": : [b] "r" (pt4), [c] "r" (pt3) );
asm volatile ("aessc %[a], %[b], %[c] ;": [a] "=r" (ct0) : [b] "r" (1), [c] "r" (0));
asm volatile ("aessc %[a], %[b], %[c] ;": [a] "=r" (ct1) : [b] "r" (1), [c] "r" (0));
asm volatile ("aessc %[a], %[b], %[c] ;": [a] "=r" (ct2) : [b] "r" (2), [c] "r" (0));
asm volatile ("aessc %[a], %[b], %[c] ;": [a] "=r" (ct3) : [b] "r" (3), [c] "r" (0));
```

4.2.2 Address-enabled AES Accelerator IP integrated RISC-V

The AES accelerator, which performs key expansion and encryption, is integrated with the RISC-V as an IP block, facilitated by an interface module and a control unit. The interface module manages data transfers between the AES accelerator, the CPU, and the memory. The control unit generates the necessary enable and disable signals for the AES module, the RISC-V core, and the memory. To ensure a fair comparison, the same AES module used in the RISC-V crypto-core is integrated outside RISC-V core as an IP address of the AES accelerator as shown in Figure 4.2, using an address-enabled mechanism, where peripherals are activated by specific data addresses. The data memory is divided into segments, with the main program starting at address 32'hx00000200, addresses below it are reserved for activating peripheral IPs, such as the AES accelerator and UART. At address 32'hx00000030, the AES Enable (AES_{En}) signal activates the AES accelerator IP, initiating the transfer of 128-bit plaintext to the AES module as 32-bit chunks. After the full 128-bit plaintext is loaded and the address reaches 32'hx0000003C, the Key-Ready (K_{Rdy}) and Data-Ready (D_{Rdy}) signals are asserted, and the AES modules start encryption during which the RISC-V core is paused for synchronization. The Write-AES-Enable (W_{Enaes}) signal for the data cache (D-cache) is activated when the Data Valid (DVld) signal becomes high, signalling the encryption completion and the ciphertext is ready. The ciphertext is stored back to the corresponding four locations of the D-cache in consecutive four-clock cycles.

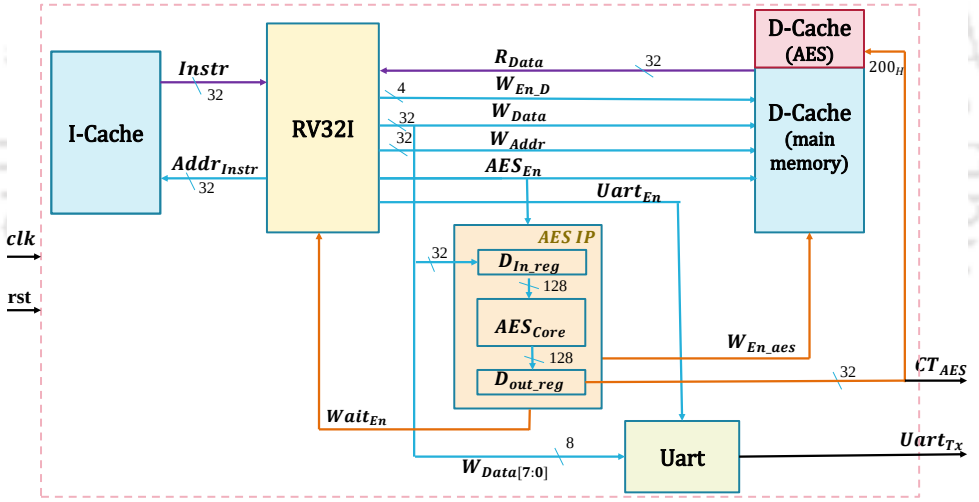


Figure 4.2: AES accelerator IP integrated outside with RISC-V

The detailed diagram of AES accelerator IP integrated RISC-V depicted in Figure 4.3, illustrates that the AES module operates with a 128-bit data width, while the RISC-V core has a 32-bit data width, requiring 4 clock cycles to transfer the 128-bit plaintext to the AES module. Once the data is fully transferred, the AES IP is enabled, and the encryption process begins. During the AES encryption, the RISC-V core is paused to maintain proper synchronization with other peripherals like UART by a Wait Enable ($Wait_{En}$) signal, which is sent to the Program Counter (PC) module, effectively halting the fetching of new instructions. The RISC-V core's ALU in IE stage, continues calculating the memory address of previous load instruction without updating values, consuming additional power and overshadowing the AES module's power consumption. Once the AES encryption is completed, a Dvld signal is triggered, indicating that the ciphertext is ready. This signal then activates the W_{Enaes} signal, which allows the ciphertext to be written into a dedicated data memory location,

requiring 4 clock cycles. After the ciphertext is successfully written to memory, the $Wait_{En}$ signal is deactivated, allowing the RISC-V core to resume normal operation.

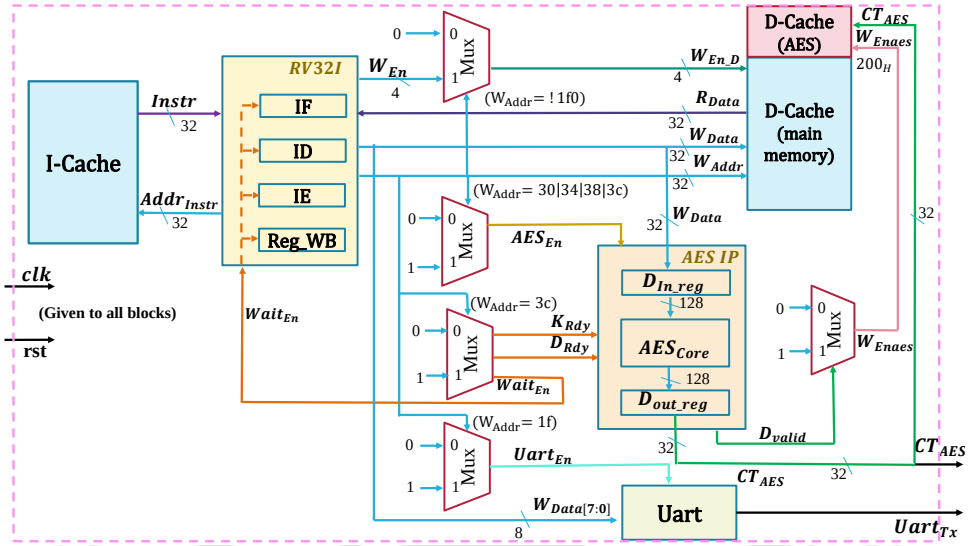


Figure 4.3: Detailed architecture of AES accelerator IP integrated outside with RISC-V

The encryption of single 128-bit plaintext using the AES IP integrated RISC-V core requires 28 clock cycles: 4 clock cycles to transfer the plaintext to the AES IP, 12 clock cycles for the AES encryption, 4 clock cycles to transfer the ciphertext to memory, 8 additional clock cycles to synchronize with other peripherals, such as the UART. The AES accelerator IP is triggered based on specific memory addresses. Therefore, the C-programs for AES encryption are written in such way to activate the AES IP according to these addresses, as described in Algorithm 2. This integration provides controlled activation of AES module which enhances security and efficiency. It is compatible with existing RISC-V compilers eliminating the need for recompilation, since no new instructions are needed to enable AES thereby simplifying the integration process. Additionally, the AES accelerator IP can be utilized by other cores in multi-core architectures.

Algorithm 2 Snippet for AES encryption by AES IP integrated RISC-V

```
volatile uint32˙t* REG1 = (volatile uint32˙t*) 0x00000030U;
volatile uint32˙t* REG2 = (volatile uint32˙t*) 0x00000034U;
volatile uint32˙t* REG3 = (volatile uint32˙t*) 0x00000038U;
volatile uint32˙t* REG4 = (volatile uint32˙t*) 0x0000003CU;
volatile uint32˙t* REG5 = (volatile uint32˙t*) 0x00000050U;
volatile uint32˙t* REG6 = (volatile uint32˙t*) 0x00000054U;
volatile uint32˙t* REG7 = (volatile uint32˙t*) 0x00000058U;
volatile uint32˙t* REG8 = (volatile uint32˙t*) 0x0000005cU;
var1 = 0x3243f6a8 ;
var2 = 0x885a308d ;
var3 = 0x313198a2 ;
var4 = 0xe0370734 ;
*REG1 = var1;
*REG2 = var2;
*REG3 = var3;
*REG4 = var4;
 var1 = *REG5;
 var2 = *REG6;
 var3 = *REG7;
 var4 = *REG8;
```

4.3 Hardware Performance and Security Metrics

The hardware performance and security metrics of all the developed designs are thoroughly evaluated in this section.

4.3.1 Hardware Resource Analysis

Two AES integration methods with the RISC-V core were analyzed focusing on performance metrics and hardware resource utilization using Synopsys design tools, at UMC 65 nm technology node. The analysis is aimed to identify the optimal integration method based on comparative results, scalability potential, and security requirements.

Table 4.2 and Table 4.3 summarizes various performance parameters of the developed cores using two integration techniques with its baseline RISC-V cores. Comparison of these integration techniques with its baseline RISC-V cores reveals significant reductions in energy consumption and notable performance improvements during AES encryption. For encrypting 128-bit data, the 3-stage RISC-V core without cryptographic capabilities, requires 6,953 clock cycles, whereas 3-stage RISC-V crypto-core and AES accelerator IP integrated with 3-stage RISC-V requires only 23 and 28 clock cycles, respectively. Similarly, the 4-stage HP RISC-V core without cryptographic capabilities requires 10,914 clocks to encrypt 128-bit data, while the 4-stage HP RISC-V crypto-core and AES accelerator IP integrated with 4-stage HP RISC-V complete encryption in 20 and 28 clocks, respectively. As illustrated in Tables 4.2 and 4.3 , integrating AES hardware with RISC-V using either Integration Technique-1 or Technique-2 increases dynamic power compared to their respective baseline RISC-V cores. This increase in dynamic power is primarily due to the added AES hardware and associated switching activity. However, these integration techniques drastically reduce the number of clock cycles needed for AES encryption. Since encryption time is directly proportional to the number of clock cycles and inversely proportional to the clock frequency, this significant reduction in clock cycle count leads to a substantial decrease in encryption time. Thus, these integration techniques drastically reduce the energy consumption and provide a substantial increase in throughput. Integrating AES as an IP with RISC-V core requires more area and reduces maximum frequency compared to AES FU embedded inside RISC-V.

Table 4.2: Performance metrics of Proposed integration technique with 3-stage RISC-V

Design	Area (μm^2)	kGE+	Dyn. power* (µW/MHz)	Delay (ns)	Max. freq. (MHz)	No. of clks	Throughput (Gbps)
Baseline core: 3stage RISC-V	25227.72	17.51	11.62	4.17	239.81	6953	0.0044
Integration technique-1: RISC-V Cryptocore (3stage RISC-V+AES-Integrated inside)	41522.76	28.83	17.08	4.47	223.71	23	1.25
Integration technique-2: RISC-V with AES IP (3stage RISC-V+AES-Integrated outside)	43205.04	29.82	15.48	4.54	220.41	28	1.01

*Evaluated for Best Case (BC) corner and kGE means kilo Gate Equivalents

Table 4.3: Performance metrics of Proposed integration technique with 4-stage HP RISC-V

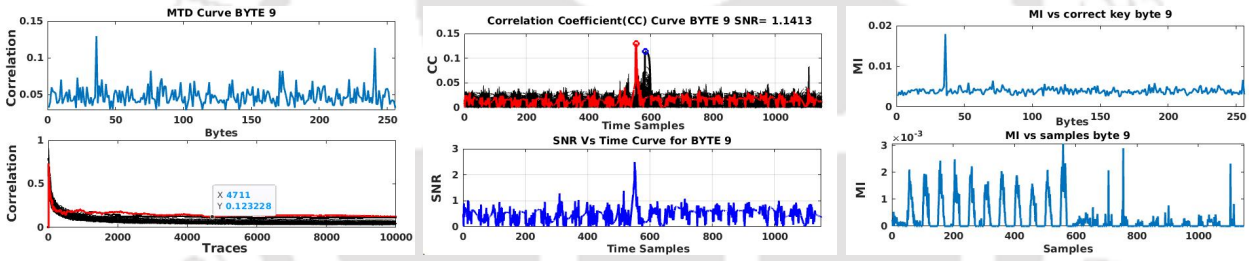
Design	Area (μm^2)	kGE+	Dyn. power* (µW/MHz)	Delay (ns)	Max. freq. (MHz)	No. of clks	Throughput (Gbps)
Baseline core: 4stage HP RISC-V	39842.72	27.67	15.32	3.35	298.51	10914	0.0035
Integration technique-1: RISC-V Cryptocore (4stage HP RISC-V+AES-Integrated inside)	58059.72	40.31	22.2	3.71	269.54	20	1.73
Integration technique-2: RISC-V with AES IP (4stage HP RISC-V+AES-Integrated outside)	59346.63	41.21	19.81	3.75	266.67	28	1.22

*Evaluated for Best Case (BC) corner and kGE means kilo Gate Equivalents

In this work, the focus is on low-power architecture design. Therefore, the BC/FF corner is of particular interest, as it results in the maximum power consumption, making it the most critical corner for evaluating worst-case power behavior. Also, to provide a technology-independent way to compare the size and complexity of the designs, we have evaluated the area in terms of kGE. kGE (kilo Gate Equivalents) is a standard metric for estimating digital circuit area, where 1 kGE equals the area of 1,000 basic 2-input NAND gates in a given technology node. In UMC 65nm, the actual area of one GE(a basic 2-input NAND gate) is 1.44 μm^2. As a result, 1 kGE would correspond to 1,440μm^2.

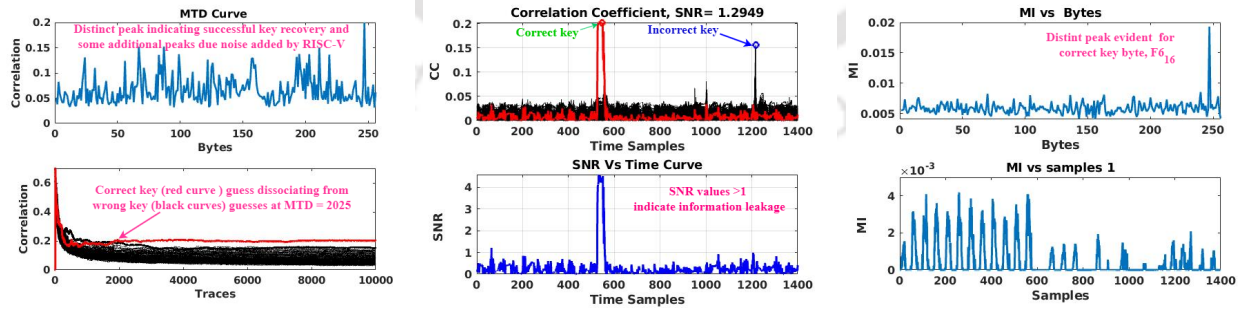
4.3.2 Hardware Security Metrics

The developed AES-integrated RISC-V cores, designed for high-security applications, are evaluated for it's vulnerability to PAA using key hardware security metrics: MTD, SNR, MI, and TVLA. MTD reflects the number of traces needed to recover the key byte; unprotected designs show successful recovery when the red line diverges from sea of black lines. SNR, measures the strength of correlation between correct and incorrect key guesses and should be less than 1 for secure designs, and greater than 1 in unprotected designs, with correct key correlations surpassing incorrect keys. MI measures mutual dependency between two random variables. Protected designs show low MI without distinct peaks, while unprotected designs display a distinct peak at the correct key, indicating leakage. TVLA, identifies leakage, with secure designs having t-values within ±4.5, whereas unprotected designs exceed these limits.



(a) MTD plot (b) SNR plot (c) MI plot

Figure 4.4: PAA results of last recovered byte9 for 3stage RISC-V Crypto-core

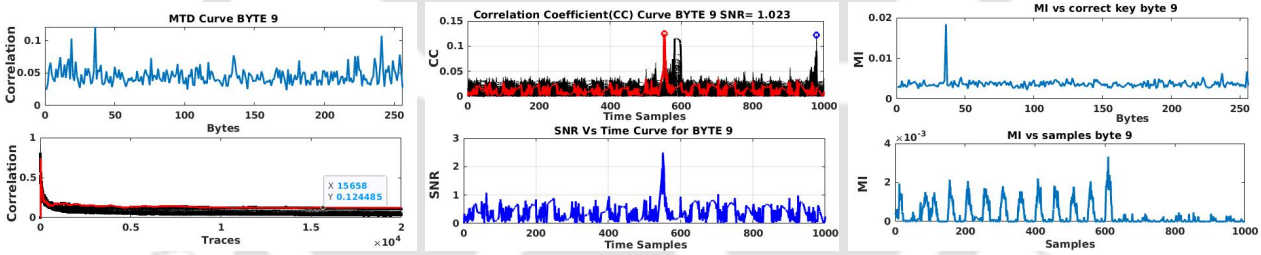


(a) MTD plot (b) SNR plot (c) MI plot

Figure 4.5: PAA results of last recovered byte1 of AES IP integrated 3stage RISC-V

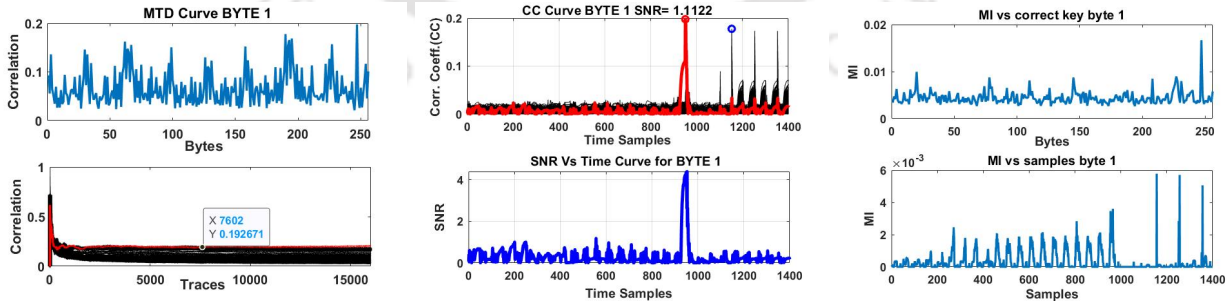
As security improves, MTD increases, while SNR, MI, and TVLA decrease. The Figures. 4.4 to 4.7 present the PAA results and provides a clear trend of security metrics for tightly and loosely coupled AES integration techniques with 3-stage and HP 4-stage RISC-V cores.

Measurement To Disclose (MTD): is a key security metric for evaluating resistance against PAA, indicating the number of power traces required to recover the secret key. As security improves, a greater number of power traces are required to recover key, which in turn increases MTD. A higher MTD implies improved security. As shown in Figure 4.4a, tightly coupled AES integration with the 3-stage RISC-V requires 4,711 traces to recover the key, compared to only 2,025 traces in the loosely coupled architecture in Figure 4.5a. This similar pattern continues with the HP 4-stage RISC-V, where the tightly coupled AES integration in Figure 4.6a achieves a significantly higher MTD of 15,658, while the loosely coupled AES integration with RISC-V in Figure 4.7a requires 7,602 traces. These results clearly demonstrate that tightly coupled integration enhances security.
Signal-to-Noise Ratio (SNR): is inversely proportional to MTD, with lower SNR values indicating higher security. For the 3-stage RISC-V core, the tightly coupled design exhibits an SNR of 1.141 (Figure 4.4b), and 1.295 in the loosely coupled version (Figure 4.5b). Similarly, in the HP 4-stage RISC-V core, the tightly coupled integration shows a lower SNR of 1.023 (Figure 4.6b), compared to 1.112 in the loosely coupled design (Figure 4.7b), aligning with the trend of improved security.
Mutual Information (MI): which quantifies the leakage of sensitive data, also decreases as security increases. The tightly coupled AES in the 3-stage RISC-V achieves an MI of 3.51 m (Figure 4.4c), whereas the loosely coupled version shows a higher MI of 4.37 m (Figure 4.5c). For the HP 4-stage RISC-V, MI values further confirm this trend with MI of 3.06 m for tightly coupled (Figure 4.6c) and 3.92 m for loosely coupled integration (Figure 4.7c).



(a) MTD plot (b) SNR plot (c) MI plot

Figure 4.6: PAA results of last recovered byte9 for HP 4stage RISC-V Crypto-core



(a) MTD plot (b) SNR plot (c) MI plot

Figure 4.7: PAA results of last recovered byte1 of AES IP integrated HP 4stage RISC-V

Test Vector Leakage Assessment (TVLA): results further support this trend. According to Table 4.4, the tightly coupled AES with 3-stage RISC-V achieves a TVLA of 85.47, compared to 91.10 for the loosely coupled version. Table 4.5 shows that for the HP 4-stage

RISC-V, the tightly coupled integration results in a TVLA of 77.37, while the loosely coupled setup reaches 89.24.

Table 4.4: Hardware security metrics of the integration techniques using 3-stage RISC-V

Hardware Security Metrics						
Design	**No:of Bytes recovered**	**Last Byte recovered**	MTD_{max}	SNR_{min}	MI_{min}	$\|TVLA_{max}\|$
Standalone AES	16	9	521	2.64	6.24*10´3	97.37
RISC-V Cryptocore (3Stage RISC-V+AES-Integrated inside)	16	9	4,711	1.15	3.51*10´3	85.47
RISC-V with AES accelerator IP (3Stage RISC-V+AES-Integrated outside)	16	1	2025	1.295	4.37*10´3	91.10

*The values are evaluated considering all bytes

Table 4.5: Hardware security metrics of integration techniques using HP 4stage RISC-V

Hardware Security Metrics						
Design	**No:of Bytes recovered**	**Last Byte recovered**	MTD_{max}	SNR_{min}	MI_{min}	$\|TVLA_{max}\|$
RISC-V Cryptocore (4Stage HP RISC-V+AES-Integrated inside)	16	9	15658	1.023	3.06*10´3	77.37
RISC-V with AES accelerator IP (4Stage HP RISC-V+AES-Integrated outside)	16	1	7602	1.1122	3.92*10´3	89.24

*The values are evaluated considering all bytes

Tables 4.4 and 4.5 provides a comparative analysis of the variations in hardware security metrics for the designed cores utilizing different integration techniques, discussed in this work. Results indicate that while AES-integrated RISC-V designs offer greater security than standalone AES implementation, they remain susceptible to PAA, as all key bytes were eventually recovered. The Table 4.4 shows, the AES integrated as an external IP outside the 3-stage RISC-V core showed improvements over the standalone AES, with an MTD increase of 3.89x and reductions in SNR, MI, and TVLA by 50.94%, 29.96%, and 6.43%, respectively. Direct integration of AES within the 3-stage RISC-V core achieved even better security, with an MTD increase of 9.04x and reductions in SNR, MI, and TVLA by 56.44%, 43.75%, and 12.22%, respectively, relative to the unprotected AES. Similarly, the Table 4.5 shows the AES integrated externally with the 4-stage HP RISC-V core yielded an MTD increase of 14.59x and reductions in SNR, MI, and TVLA by 57.87%, 37.18%, and 8.35%. Direct integration within the 4-stage HP RISC-V core further enhanced security, with a 30.05x MTD increase and reductions in SNR, MI, and TVLA by 61.25%, 50.96%, and 20.54% compared to the unprotected AES.

The above MTD, SNR, MI and TVLA results clearly demonstrate that tightly coupled AES integration offers better protection against PAA than loosely coupled integration. Tightly coupled AES integration achieves higher MTD and lower SNR, MI, and TVLA across both RISC-V architectures, demonstrating superior resistance to PAA. Security is enhanced in tightly coupled designs where AES hardware is embedded within the RISC-V core, making AES power signatures less accessible to attackers than in loosely coupled designs, where AES hardware is integrated externally outside RISC-V as an IP, in loosely coupled designs. Moreover, the HP 4-stage RISC-V provides improved security metrics compared to the 3-stage design, as its increased pipeline stages and complexity allows more module in RISC-V to operate in parallel with AES, thereby further obscuring AES power consumption patterns, which in turn increases security of the core against PAA.

Thus, from a security perspective, it can be concluded that the standalone AES offers the lowest level of security, followed by the AES integrated as an external IP outside the RISC-V core, with the highest security achieved when AES is integrated directly within the RISC-V core's execution stage. The standalone AES is highly vulnerable, as the entire power is contributed only by AES operations. However, in both integration techniques, when AES is integrated with RISC-V, the total power depends on both AES and RISC-V operations, which will inherently increase the security level. When AES is integrated inside RISC-V, the on-chip power measured will have less significance on AES power compared to the AES integrated outside RISC-V as an IP. This is because the AES power is more accessible to attackers in the case of AES integrated externally outside RISC-V than AES integrated inside RISC-V. This makes the RISC-V coupled with AES accelerator IP slightly more vulnerable. The AES integrated within the RISC-V core achieved better security metrics compared to the AES integrated externally as an IP with RISC-V.

In tightly coupled designs, the AES functional unit is integrated directly within the execution stage of the RISC-V core, and the RISC-V ISA is extended with custom instructions to support AES operations. This approach requires modifications to the RISC-V compiler and proper handling of pipeline synchronization. In loosely coupled designs, the AES is implemented as an address-enabled AES accelerator IP outside the RISC-V core. In this approach, the AES IP is activated, and the encryption process starts upon accessing specific address, eliminating the need for custom instructions to initiate AES. Thus, the AES can be easily integrated with RISC-V without any modification in its ISA and RISC-V compilers. Also, as AES is integrated outside RISC-V core as an IP while extending this architecture to multi-core RISC-V SoC, other RISC-V cores can also utilize the same AES IP. Additionally, when one RISC-V core is using the AES IP, other cores can continue their operations, helping to obscure the AES power usage and enhancing hardware security metrics. For future upgrades to multi-core SoCs in IoTe devices, a loosely coupled AES accelerator IP with a dedicated memory interface is preferable over integrating AES directly within the RISC-V core, as it provides better flexibility and supports multi-core architectures.

4.4 Chapter Summary

This chapter presents two integration techniques of AES with RISC-V processor to enhance throughput and reduce energy consumption, compared to standard RISC-V cores running AES algorithms in software. In the first approach, a cryptographic RISC-V core is designed by embedding an AES functional unit directly into the execution (IE) stage of the RISC-V core, with custom AES instructions added to the RISC-V ISA. The second method involves integrating AES as an external, address-enabled AES accelerator IP module, loosely coupled with the RISC-V core. All key bytes were recovered for both methods with fewer than 20K power traces. Following this evaluation, the RISC-V crypto-core with AES embedded inside RISC-V is ideal for high-performance, resource-constrained, and highly secure applications due to its lower area overhead, higher performance, and higher intrinsic resilience compared to an externally integrated AES accelerator IP with RISC-V. The tightly integrated RISC-V crypto-core is more detailed in chapter 5, followed by chapter 6 focusing on further enhancing performance and security of the developed RISC-V crypto-core, by using an extemely small Non-linear Cyclic Variable Clock Feistel Bridge-Inspired Countermeasure.

Conversely, the loosely coupled AES as an IP with the RISC-V core is preferable for applications requiring future scalability, such as multi-core SoCs, as it offers greater flexibility and does not require modifying the RISC-V compiler or adding new instructions to support

AES. This approach allows the AES module to operate independently of the processor's pipeline, avoiding the tight integration issues that can lead to under-utilization of resources in multi-core SoCs. It provides better adaptability for different workloads, as the AES FU can be invoked only when needed, freeing up the core to perform other tasks concurrently. Additionally, it enhances power efficiency by reducing the need for continuous synchronization with the AES FU. It also simplifies the overall architecture, allowing for easier upgrades or replacements of the AES module without significant changes to the processor core, making it a highly modular and maintainable solution. For these reasons, Chapter 7 will focus on the AES IP integrated with RISC-V, and further secured with a novel Random Clock Self-Complementary Countermeasure to protect against PAA.

CHAPTER

5

POWER SIDE CHANNEL ATTACKS ON RISC-V CRYPTO-CORE FOR HIGH-SECURITY APPLICATIONS

Contents

5.1 Introduction

With the advancement of IoTe devices, protecting sensitive data processed on these devices has become increasingly critical. This chapter aims to enhance processor's built-in resilience against Power Analysis Attack (PAA) by expanding pipeline stages, employing diverse pipeline techniques, and integrating additional features. It introduces 32-bit RISC-V core micro-architectures with inbuilt cryptographic capabilities, extending the RISC-V ISA with custom AES instructions to reduce energy consumption, code size, and encryption time compared to software AES solutions. Among the developed RISC-V cores in chapter 3, the 3stage and HP 4stage RISC-V cores are selected to develop secure RISC-V crypto-cores. As custom AES instructions require multiple clock cycles to execute, pausing other pipeline stages during AES operation is essential. This chapter presents two RISC-V crypto-cores that integrate AES module directly within the processor pipeline, utilizing two pausing methods. Core1 is a 3-stage pipelined core with software pause, while Core2 is a 4-stage pipelined core with hardware pause. Both architectures are evaluated for their ability to secure data through AES encryption and examines the impact of pausing techniques and pipeline stages on security metrics.

With the growing demand for data processing at IoT edge nodes, attackers are increasingly targeting these devices through various side-channel attack techniques. This shift from general-purpose computers to edge devices has underscored the need for secure data communication, where cryptography has become essential. The AES [2] is a symmetric key cryptographic method that uses the same key for both encryption and decryption. This makes AES to operate with fewer computational resources, resulting in faster data encryption and lower latency, which is ideal for devices with limited processing power, such as IoT edge nodes [85]. Usually, AES is implemented as a as pure software implementations on general-purpose embedded processors. However, the software AES implementations [22],[23] requires thousands of clock cycles to encrypt the data, resulting in higher energy consumption, longer encryption time, and larger code size. Furthermore, software implementations are more susceptible to PAA. For real-time, power-efficient IoT applications, key parameters such as latency, throughput, and low energy consumption are critical. AES hardware accelerators [24–28] offer a promising solution over pure software implementations. Dedicated AES hardware modules, integrated within the processor and supported by ISA extensions, provide optimized encryption performance. For instance, Intel's AES New Instructions (AES-NI) [83] provides seven new instructions to accelerate AES operations, including encryption, decryption, key generation, and matrix multiplication. ARM introduced similar AES instructions in the Armv8-A ISA [30]. Integrating AES hardware into a processor pipeline enhances performance and energy efficiency but requires careful design to avoid impacting the processor's timing performance.

This chapter provides deeper insights into power side-channel attacks on the designed low-power, high-performance embedded processors with cryptographic capabilities for IoT edge devices. It describes two novel RISC-V crypto-core architectures that integrate an AES hardware functional unit within the RISC-V pipeline, utilizing custom AES instructions. The integration of AES into RISC-V processors bolsters security and energy efficiency, enabling secure communication without significantly impacting device performance. The chapter further explores how increasing processor complexity affects vulnerability to PAA. The entire chapter is organized in the following manner: Section 5.2 outlines the design of an area-efficient and low-power AES hardware implementation. Section 5.3 describes two RISC-V crypto-core architectures that integrate this AES hardware unit and extend the RISC-V ISA

to accelerate encryption. The first design utilizes a software pause (SW) mechanism, while the second employs a hardware pause (HW) method. Section 5.4 presents a performance analysis of these cores, focusing on metrics such as throughput, encryption time, energy consumption, area, and power. Section 5.5 evaluates the resistance to PAA of these developed cores, using four hardware security metrics: MTD, SNR, MI, and TVLA, and explores how increased processor complexity impacts these metrics.

The designs are validated on FPGA and developed using UMC 65nm technology node. During AES encryption, power traces are extracted using Synopsys PrimeTime PX and analyzed with MATLAB power attack model, successfully recovering all key bytes. Both cores achieve notable throughput improvements, with Core1 and Core2 achieving higher throughput of 2.02× and 2.83×, respectively, than the Arm CryptoCell-312. The chapter demonstrates that despite their vulnerabilities, the integration of AES with RISC-V architecture significantly improves their intrinsic resilience against PAA.

5.2 Selection of AES hardware

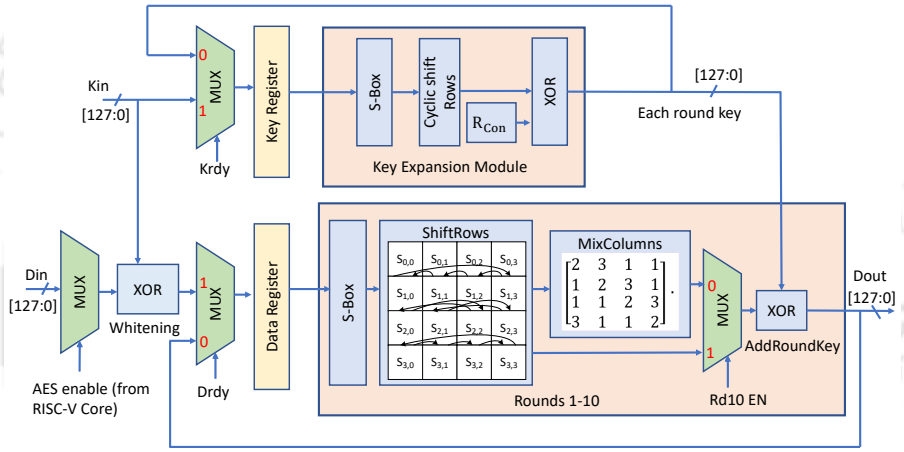


Figure 5.1: 128-bit AES architecture

As the AES hardware is to be integrated into the RISC-V core's pipeline, a compact and energy-efficient AES core has been designed. The block diagram of the 128-bit AES encryption unit used in the proposed cores is shown in Figure 5.1. The design is based on rolling architecture, and the results after each stage are stored back into a register. The initial stage is known as whitening (AddRoundKey), where the key is XORed with plaintext, which is followed by ten rounds. The first nine of the ten rounds have four steps: SubBytes, ShiftRows, MixColumns, and AddRoundKey, while the last round has MixColumns excluded. The key for each round is obtained using the key expansion module. We have designed a 12-clock 128-bit AES, where the S-box is instantiated 16 times for each byte substitution. Therefore, the selection of the S-box should be taken wisely for area-efficient AES designs, as the S-box contributes to the majority of the area. Singha et al. [18] provide a literature survey of numerous S-box designs based on various design techniques like normal basis, polynomial basis, and LUT-based. The LUT-based S-box is commonly used in AES designs because of its ease of implementation, but it occupies a humongous area, making these designs unsuitable for resource-constrained IoTe devices. The conventional LUT-based S-box, requires storing 256 precomputed S-box values as Look-Up-Tables in memory requiring additional ROM and address decoding circuitry. The Masoleh S-box-based AES design achieves significant reductions in area and power consumption compared to LUT-based and Canright S-box designs

due to several architectural optimizations. It eliminates memory overhead by computing S-box values directly with combinational logic, avoiding the use of Look-Up Tables. It uses deeper tower field representation of GF(2^8) as GF(((2^2)^2)^2), with each subfield expressed as normal basis improving efficiency, as squaring becomes simple bit rotations and multiplication requires fewer logic gates. The design further minimizes gate count and logic depth through sub-expression elimination and shared logic, while strategically using smaller gates like NAND and NOR to reduce area. Additionally, it replaces standard multiplexers with faster, more compact inverting multiplexers. These optimizations collectively reduce area, power consumption, and delay, making the Masoleh S-box more efficient for AES hardware implementations as shown in Table 5.1.

The AES architectures using LUT S-box [20], Canright S-box [86] and Masoleh S-box [21] were designed in 65nm technology, and a comparative analysis of these designs is made in Table 5.1. The throughput of the designs is calculated by the equation 5.1.

$$Throughput = \frac{(No.\ of\ bits\ processed)}{(No.\ of\ clks) * (Critical\ Path\ Delay)} \tag{5.1}$$

The Masoleh-based AES design proved to be the most efficient, with 1.93× less area and 1.18× lower power consumption compared to the LUT-based AES design, and 1.25× less area and 1.1× lower power than the Canright-based design. The Masoleh-based AES design has 1.15× lower maximum frequency and throughput than the LUT-based AES, but this is acceptable due to significant area and power savings. Meanwhile, Masoleh-based AES design achieves 1.37× higher frequency and throughput than the Canright-based AES, making it more efficient for high-performance low-power applications.

Table 5.1: Comparison of the designed 128-bit AES

128-bit AES Units	Tech. [nm]	Area [µm²]	kGE	Dynamic power* [µW/MHz]	No. of clks	Delay [ns]	Max. Freq [MHz]	Throughput [Gbps]
AES (LUT S-box)	65	26007.12	18.06	4.80	12	1.57	636.94	6.79
AES (Canright S-box)	65	16815.62	11.7	4.43	12	2.49	401.61	4.28
AES (Masoleh S-box)	65	13503.48	9.38	4.07	12	1.81	552.49	5.89

*Dynamic power is normalized w.r.t clock frequency

Table 5.1 shows that the Masoleh S-box-based AES design has the lowest area and power consumption while maintaining reasonably good throughput. Thus, Masoleh S-box based AES hardware was chosen for integrating with RISC-V to provide better performance and lower area and power overhead.

5.3 Novel RISC-V Crypto-core Micro-architectures

This section presents two tightly coupled AES integrated RISC-V architectures, Core1 and Core2, for high-security applications based on software pause and hardware pause. To enhance encryption performance, proposed area-efficient AES hardware module based on the Masoleh S-box is incorporated into the Instruction Execution (IE) stage. New custom AES instructions, detailed in Table 5.2, are added to the RISC-V ISA to facilitate plaintext loading, ciphertext storing, and AES activation. The instruction decoder in the ID stage is modified to handle these new AES instructions, specifically AESLP (Load Plaintext) and AESSC (Store Ciphertext). The AESLP instruction loads 128-bit plaintext into the AES encryption unit in two steps using 32-bit registers. Using the ‘rd‘ field, AESLP determines

whether to load the lower or upper 64 bits of plaintext. When 'rd' is zero, it loads the lower 64 bits; a non-zero 'rd' loads the upper 64 bits and triggers the AES enable signal, initiating encryption. The designed AES hardware provides a latency of 12 clock cycles for encryption and is synchronized with other stages of RISC-V pipeline through either software or hardware pause techniques. Core1 uses a software pause technique by inserting 12 NOP instructions to stall the pipeline during AES operation. Core2 uses a hardware pause technique, where the AES enable signal halts other pipeline stages and prevents register writes until AES completion, signalled by a Data Valid (DVld) signal. Both cores store the 128-bit ciphertext back in the register bank in 32-bit chunks using AESSC instructions after encryption, but Core2's dual-write port register bank enhances performance. While software pause techniques involve simpler synchronization through assembly code modification, hardware pause techniques offer greater flexibility by accommodating various clock AES architectures without altering assembly codes.

Table 5.2: Customized AES instructions for RISC-V

Instr	**31-25**	**24-20**	**19-15**	**14-12**	**11-7**	**6-0**
	Func7	**rs2**	**rs1**	**Func3**	**rd**	**Opcode**
AESLP	0010000	rs2	rs1	110	rd	1101011
AESSC	0010000	rs2	rs1	111	rd	1101011

Integrating custom AES instructions into the RISC-V ISA delivers substantial benefits in encryption speed, energy efficiency, code size, and security. By incorporating AES hardware module directly into the RISC-V architecture and extending its ISA with custom AES instructions, encryption of data is executed in fewer clock cycles, which significantly reduces both encryption time and energy consumption compared to software-based AES solutions. This approach, also minimizes the size of AES programs, saving memory space in resource constrained IoT environments. Additionally, custom AES instructions enhance security by reducing the exposure of AES hardware to power attacks, utilizing parallelism and pipelining by implementing rolling AES architecture and securely handling cryptographic keys within the hardware, thereby making power analysis attacks more challenging.

5.3.1 Core1: 3-Stage RISC-V crypto-core based on SW pause

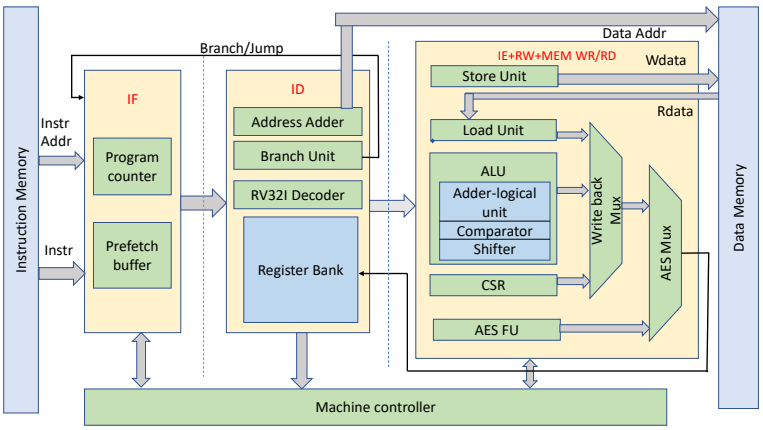


Figure 5.2: Core1: A 3-Stage RISC-V crypto-core (SW pause)

The proposed design is a 3-stage pipeline RISC-V crypto-core micro-architecture based on Software (SW) pause for high-security applications, shown in Figure 5.2. The 1^{st} stage

contains a pre-fetch buffer and Program Counter (PC). The 2^{nd} stage includes a decoder, address adder, register bank, and branch unit. The decoder decodes instructions and generates corresponding control signals for all RV32I instructions and for the two new AES instructions. The address adder is responsible for calculating both data and instruction addresses, while the branch unit evaluates branch conditions and issues enable signals for branching as needed. The register bank, which contains thirty-two 32-bit registers, is equipped with two read ports, a single write port, and an enable port that manages write operations. The 3^{rd} stage contains the ALU, CSR, AES Functional Unit, Load unit, Store unit, and a Write-back Mux. The AES Functional Unit performs 128-bit AES encryption for handling secure data operations. The load and store units are responsible for data transfer between memory and the register bank, with only these units directly interacting with memory. Write-back Mux selects ALU result, CSR values, load data or PC return address, which is to be stored in the register bank depending on the control signals generated based on the instruction. An additional multiplexer is added to the execution stage, which allows switching between data from the Write-back Mux and the AES Functional Unit output, enabling the pipeline to choose between normal data processing path or encrypted data paths.

Algorithm 3 Snippet for AES encryption based on SW pause

```
int pt4 = 0x3243f6a8;
int pt3 = 0x885a308d ;
int pt2 = 0x313198a2 ;
int pt1 = 0xe0370734 ;
asm volatile ("aeslp zero, %[b], %[c] ;": : [b] "r" (pt2), [c] "r" (pt1) );
asm volatile ("aeslp %[a], %[b], %[c] ;": : [b] "r" (pt4), [c] "r" (pt3) );
asm volatile ("aessc %[a], %[b], %[c] ;": [a] "=r" (ct0) : [b] "r" (1), [c] "r" (0));
asm volatile ("aessc %[a], %[b], %[c] ;": [a] "=r" (ct1) : [b] "r" (1), [c] "r" (0));
asm volatile ("aessc %[a], %[b], %[c] ;": [a] "=r" (ct2) : [b] "r" (2), [c] "r" (0));
asm volatile ("aessc %[a], %[b], %[c] ;": [a] "=r" (ct3) : [b] "r" (3), [c] "r" (0));
asm volatile ("NOP" );
asm volatile ("NOP" );
asm volatile ("NOP" );
asm volatile ("NOP" );
asm volatile ("NOP" );
asm volatile ("NOP" );
asm volatile ("NOP" );
asm volatile ("NOP" );
asm volatile ("NOP" );
asm volatile ("NOP" );
asm volatile ("NOP" );
asm volatile ("NOP" );
```

In the execution stage, the ALU and CSR operations are completed in single clock cycle. However, the AES Functional Unit requires 12 clock cycles to finish encryption. To synchronize the stages without additional hardware, a software-based pause method is implemented by inserting 12 NOP instructions into the program, as detailed in Algorithm 3. This stalls the pipeline until the AES operation is completed. This approach of using a software pause enables secure AES processing without increasing hardware complexity, aligning the pipeline stages efficiently. While Core1's software pause technique is simpler and easier to implement, it increases code size and creates synchronization issues if the number of NOP instructions inserted in AES software program is incorrect. The 12 NOPs are essential in the software pause technique because the rolling AES architecture reuses the same AES round hardware unit for all ten AES rounds, completing AES encryption only in 12 clock cycles. If fewer NOPs are used, the RISC-V core may proceed fetching and executing new data before encryption is complete, causing synchronization failure and incorrect encryption results. Core2 avoids this issue by using a hardware-generated pause signal, which enables more efficient and reliable synchronization, making it better suited for high-performance applications.

5.3.2 Core2: 4-Stage RISC-V crypto-core based on HW pause

This section proposes enhancing processor's intrinsic resilience against PAA, by different strategies such as increasing the number of pipeline stages and incorporating features like hardware pause circuits for synchronization, DBP in the IF stage, and an extra write port for the register bank. The design process began with upgrading a 3-stage RISC-V pipeline to a 4-stage pipelined RISC-V, optimizing it for higher performance and maximum operating frequency. The proposed 4-stage pipelined RISC-V crypto-core micro-architecture based on HW pause, shown in Figure 5.3, was designed by integrating AES into the IE stage of the newly designed 4-stage pipelined RISC-V core micro-architecture. The core comprises four stages: IF, ID, IE, and Register Write (RW). The IF stage includes a 2-bit DBP for predicting branch conditions for branch instructions, thereby reducing the stalling of the pipeline for branching and increasing the performance of the core. The ID stage's decoder is modified to support the new AES instructions and generate control signals for AES operations. The register bank has dual read and write ports to handle concurrent write requests, thus reducing pipeline stalls. When up to two write requests occur in a single clock cycle, they are processed simultaneously. In the IE stage, a dedicated AES functional unit (FU) is integrated to accelerate data encryption, operating with a 12-clock latency while other units, like the Arithmetic Logic Unit (ALU) and Control Status Register (CSR), have single-clock latency. The RW stage primarily writes values from memory read operations, AES FU, and ALU/CSR into the register bank, ensuring efficient data handling within the register bank.

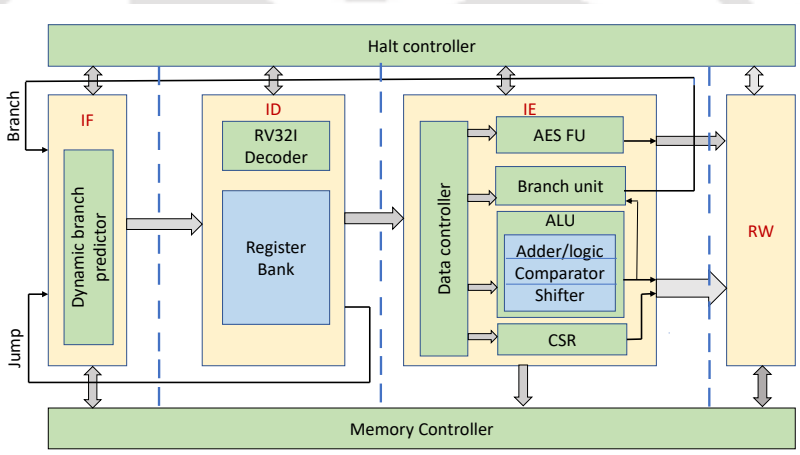


Figure 5.3: Core2: A HP 4Stage RISC-V crypto-core (HW pause)

As shown in the snippet of algorithm 4, two AESLP instructions are used to load the 128-bit data. A non-zero rd field in AESLP generates an AES enable signal, which activates the AES FU and initiates AES operations. During AES operations, a hardware pause signal is triggered to prevent data from being written into the register bank until the ciphertext is ready, ensuring proper synchronization. Upon completion of AES, a DVld signal is generated, indicating the ciphertext is ready and deactivates the hardware pause signal to zero, after which the ciphertext is stored as 32-bit chunks across four register locations using custom AESSC instructions. The Core2 achieves high performance and frequency due to the presence of more pipeline stages, DBP etc., however at the expense of increased area and power.

This section showcases two RISC-V crypto-cores, balancing performance, power consumption, hardware resources, flexibility, and complexity, allowing selection based on application requirements. Core1 and Core2 differ in pipeline stages, pausing techniques, control hazard mitigation techniques, register bank write ports, synchronization circuits, and active func-

Algorithm 4 Snippet for AES encryption based on HW pause

```
int pt4 = 0x3243f6a8;
int pt3 = 0x885a308d ;
int pt2 = 0x313198a2 ;
int pt1 = 0xe0370734 ;
asm volatile ("aeslp zero, %[b], %[c] ;": : [b] "r" (pt2), [c] "r" (pt1) );
asm volatile ("aeslp %[a], %[b], %[c] ;": : [b] "r" (pt4), [c] "r" (pt3) );
asm volatile ("aessc %[a], %[b], %[c] ;": [a] "=r" (ct0) : [b] "r" (1), [c] "r" (0));
asm volatile ("aessc %[a], %[b], %[c] ;": [a] "=r" (ct1) : [b] "r" (1), [c] "r" (0));
asm volatile ("aessc %[a], %[b], %[c] ;": [a] "=r" (ct2) : [b] "r" (2), [c] "r" (0));
asm volatile ("aessc %[a], %[b], %[c] ;": [a] "=r" (ct3) : [b] "r" (3), [c] "r" (0));
```

tional units during AES operation. Core1, a 3-stage pipeline, uses software pausing with NOP instructions to synchronize pipeline stages while AES is active, with branch decision made in the ID stage, utilization of pre-fetch buffer in IF stage and a dedicated address adder in the ID stage for address calculations. Core1's SW Pause technique requires fewer hardware resources, offering simplicity and flexibility in managing synchronization without altering the hardware. This approach, while straightforward to implement and suitable for applications with modest performance needs, introduces potential latency due to the sequential nature of software control. Core1's software pause slightly increases power but not enough to mask AES power, resulting in a modest security improvement. In contrast, Core2 employs a 4-stage pipeline with hardware pausing, halting pipeline updates during AES encryption and using DBP in the IF stage to reduce control hazards, additional write ports and separate memory control instructions. It relies on the same ALU used for computation in the IE stage for address calculations. Core2's HW Pause technique with dedicated hardware signals for pausing pipeline stages during AES operations, ensures precise synchronization and minimizes latency effectively. This method enhances performance, throughput, and reliability in applications requiring frequent AES computations, albeit at the cost of increased hardware complexity and potential area and power overheads. These measures generate additional power from other FUs of the RISC-V architecture to conceal the AES power signature, thereby enhancing the intrinsic resilience of the RISC-V crypto-core against PAA.

5.4 Hardware Resource Analysis

The proposed designs are initially validated on FPGA board and by the ASIC flow method in UMC 65nm technology node. Subsequently, a comprehensive assessment of various performance parameters and hardware resource utilization of the proposed designs is conducted across various operating frequency and the technology nodes, including UMC 40 and UMC 90, to analyse trends. As the cores are designed for high-security applications, AES C-programs as well as the modified C-programs with new AES instructions, are used to evaluate the functionality of the cores.

5.4.1 Performance metrics evaluation of the proposed cores

The proposed RISC-V core designs, Core1 and Core2, are evaluated on an ASIC platform, focusing on key performance metrics like throughput, energy consumption, encryption time, and Gate Equivalent (GE). Table 5.3 compares these designs with existing processor cores with cryptographic capabilities. The proposed Core1 and Core2 are implemented using 28.83 kGEs and 40.31 kGEs, whose areas are lower than the available cores in the market. Notably, Core1 occupies the smallest area among the cryptographic cores, with 1.34×, 1.55× and 1.11× reductions in area compared to the designs by B.Marshall et.al Variant-3 [87], Utsav

Table 5.3: ASIC implementation results of the developed cores in UMC65nm Technology

Processor Cores	**Core1**	**Core2**	**B. Marshall et.al** [87]	**Utsav et.al** [31]	**Arm CryptoCell-312** [88]	**Pan et. al** [33]	**de Araujo et. al** [89]
ISA	RV32I Ext	RV32I Ext	RV32I Ext	RV32I SoC	ArmISA	RISCV Ext	RISCV Ext
Technology node	65nm	65nm	-	65nm	40nm	65nm	28nm
Area[mm^2]	0.0415	0.0580	0.0555	0.0639	-	0.04516	-
kGE	28.83	40.31	38.5	44.6	-	31.94	38.13
Dyn. Power[µW/MHz]	17.08	22.20	-	40.36	-	71.9	-
Critical path delay[ns]	4.47	3.71	-	-	5	-	2
Max. Freq[MHz]	223.71	269.54	-	78	200	100	500
Dhrystone[DMIPS/MHz]	1.86	1.68	-	0.96	-	-	-
CoreMark/MHz	1.31	2.13	-	-	-	-	-
Throughput [Gbps]	1.25	1.73	-	-	0.61	0.471	2.08
AES Energy [nJ]	3.93	4.43	-	6.21	-	-	-

*Dynamic power is normalized w.r.t. clock frequency for the fair comparison with other existing cores

et.al [31] and Pan et al [33], respectively. Core2 also exhibits a reduced area relative to Utsav et.al [31] and is comparable to other existing cores. Both cores shows lower dynamic power and higher maximum frequency compared to similar designs. These improvements are achieved through key architectural optimizations. First, the cores integrate a compact AES designed using Masoleh S-boxes, which is faster and more area-efficient than the larger LUT-based S-boxes commonly used in other designs. Second, a rolling architecture is utilized, where a single AES round hardware block is reused across all 10 AES rounds, instead of duplicating it for each round. These design choices reduce both the critical path delay and overall hardware resource utilization, resulting in higher maximum operating frequency and lower dynamic power. The lightweight AES coprocessor by Pan et al. [33] operates at 100 MHz with a throughput of 0.471 Gbps, while Core1 and Core2 achieve throughputs 2.65× and 3.67× higher than Pan et al [33]. Arm CryptoCell-312 [88], a security platform, with AES integrated with ARM processor, process the data at 3.07 $bits/cc$, yielding a maximum throughput of 0.61 Gbps at 200 MHz. In contrast, Core1 processes 5.56 $bits/cc$ at 223.71 MHz, resulting in a throughput of 1.24 Gbps, and Core2 processes at 6.4 $bits/cc$ at 269.54 MHz, achieving 1.73 Gbps. These results in 2.02× and 2.83× increases in throughput over CryptoCell-312, respectively. de Araujo et al. [89] achieved slightly higher throughput than our cores, which is mainly due to increase in maximum frequency, by implementing their core in a lower technology node.

A comparison of the proposed RISC-V Crypto-cores with its baseline RISC-V cores in Table 5.4 highlights substantial performance gains, with custom AES instructions reducing code size by factors of 2.22 for Core1 and 2.27 for Core2. This reduction is pivotal in embedded systems, decreasing memory usage and accelerating execution due to fewer instructions, thereby enhancing overall performance. Core1 and Core2 significantly reduce the clock cycles required to encrypt a 128-bit plaintext to just 23 and 20 cycles, respectively, compared to 6,953 and 10,914 cycles over their baseline cores. This leads to throughput improvements of 284× for Core1 and 494× for Core2 over their baselines, with Core1 achieving 1.25 Gbps compared to 0.0044 Gbps and Core2 reaching 1.73 Gbps compared to 0.0035 Gbps. Encryption time is reduced by factors of 282 for Core1 and 492 for Core2, while energy consumption during encryption drops by 20× for Core1 and 37× for Core2. These performance gains come with certain area and power overheads: Core1 and Core2 have area overheads of 1.64× and 1.45×, respectively, and power overheads of 1.47× and 1.44× respectively. However, these overheads are deemed acceptable, particularly considering the substantial performance gains achieved by the proposed cores. Core2 experiences relatively lower overheads due to higher area and power consumption of its baseline core. This is due to the fact that as the num-

ber of pipeline stages and features of the RISC-V core increase, its overall area and power consumption also grow, making the relative overhead of integrating AES with the RISC-V appear smaller.

Table 5.4: Hardware performance metrics of Core1 and Core2 with baseline cores

Design	Core1 (Baseline)	Core1 (Baseline+AES ISA)	Core2 (Baseline)	Core2 (Baseline+AES ISA)
Area (µm²)	25227.72	41522.76	39842.72	58059.72
kGE+	17.51	28.83	27.67	40.31
Code size	4.904	2.204	4.904	2.156
Dyn. power* (µW/MHz)	11.62	17.08	15.32	22.20
Delay (ns)	4.17	4.47	3.35	3.71
Max. freq. (MHz)	239.81	223.71	298.51	269.54
No. of clks	6953	23	10914	20
Encryp. time (µs)	28.99	0.1028	36.56	0.0742
AES energy (nJ)	80.78	3.93	167.25	4.43
Throughput(Gbps)	0.0044	1.25	0.0035	1.73

*evaluated in BC corner

In summary, Core2 demonstrates superior overall performance compared to Core1. Core2 achieved a 1.21× increase in maximum operating frequency and reduction of three clock cycles for each 128-bit data encryption compared to Core1. This improvement is due to the increased pipeline stages leading to a decrease in critical path delay and additional register bank write ports, which facilitate parallel write operations, reducing clock cycle counts. This reduction in required clock cycles for encryption, coupled with a decrease in critical path delay, enabled Core2 to achieve 1.4× higher throughput and a 1.39× decrease in encryption time compared to Core1. These improvements were attained at the expense of increased area and power relative to Core1.

5.4.2 Performance metrics of cores across operating frequency

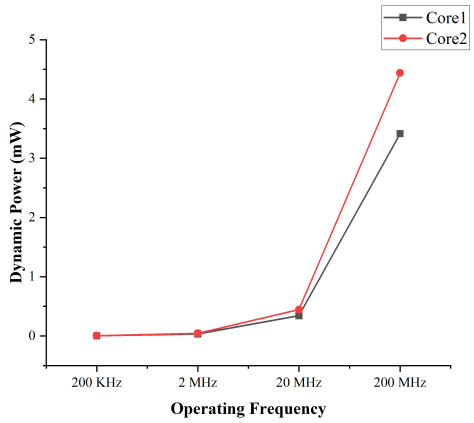
Hardware performance metrics of the proposed designs across varying operating frequencies, shown in Figure 5.4 reveal that both cores exhibit negligible changes in metrics like area and critical path delay. However, dynamic power experiences a significant increase with rising operating frequency. This phenomenon can be attributed to the direct proportionality between dynamic power and operating frequency, as depicted by the equation 5.2.

$$P_{Dynm} = \frac{1}{T}\int_0^T P_{Dynm}(t)\,dt \;= \alpha.f.C_L.V_{DD}^2 \tag{5.2}$$

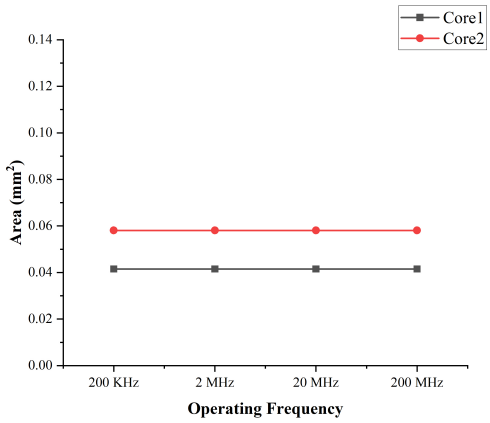
where, $P_{Dynm}(t)$ denotes the instantaneous dynamic power component, α represents the switching activity factor, f is the clock frequency, C_L symbolizes the load capacitance, and V_{DD} stands for the supply voltage. Hence in this work, we normalize the dynamic power with respect to operating clock frequency for the fair evaluations.

5.4.3 Performance metrics of cores across technology nodes

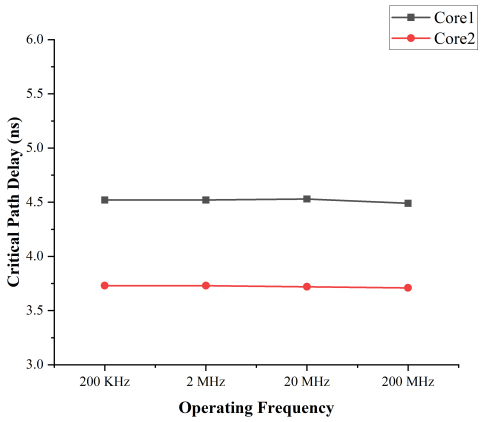
At the UMC 65nm technology node, Core2 achieved 1.21× increase in maximum operating frequency, at the expense of increased area and power relative to Core1. Figure 5.5 illustrates various hardware performance metrics, such as area, power, and maximum frequency for Core1 and Core2 across different technology nodes. The performance trends observed at the 65nm node remain consistent at 40nm and 90nm, validating that Core2 exhibits superior performance and maximum operating frequency compared to Core1. However, this improvement in Core2's performance is accompanied by increased area and power consumption. The



(a) Power vs operating frequency



(b) Area vs operating frequency



(c) CPD vs operating frequency

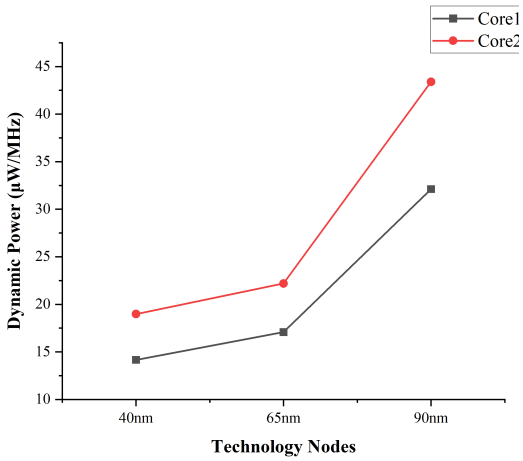
Figure 5.4: Performance parameters of developed cores across different operating frequencies

performance of both Core1 and Core2 sees enhancements as we scale down from the 90nm to the 40nm technology node, by 1.39× and 1.51×, respectively.

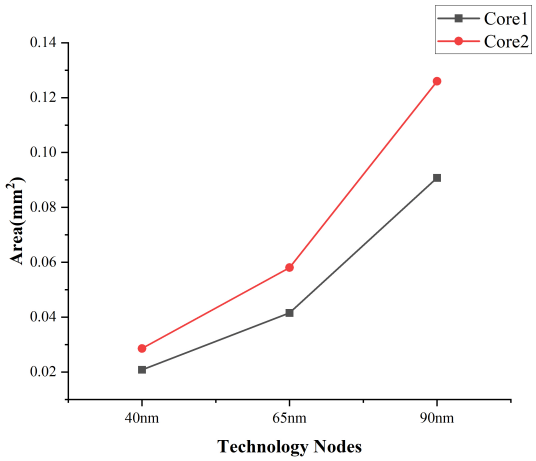
Upon migrating the designs from the 90nm to the 40nm technology node, the area of Core1 and Core2 decreases by factors of 4.2 and 4.5, respectively, with Core2 achieving a slightly greater reduction. This indicates that at lower technology nodes, the relative area difference between Core2 and Core1 narrows. The power consumption of both Core1 and Core2 decreases approximately by a factor of 2.26 and 2.29, respectively, when scaled down from 90nm to 40nm technology. Overall, Figure 5.5 illustrates that scaling to lower technology nodes significantly improves the performance, area efficiency, and power consumption of both cores. Furthermore, there is a narrower gap in terms of area and power between the two cores at lower technology nodes compared to higher ones. Notably, Core2 exhibits a greater increase in maximum frequency compared to Core1 as we move to lower technology nodes.

5.5 Hardware Security Metrics

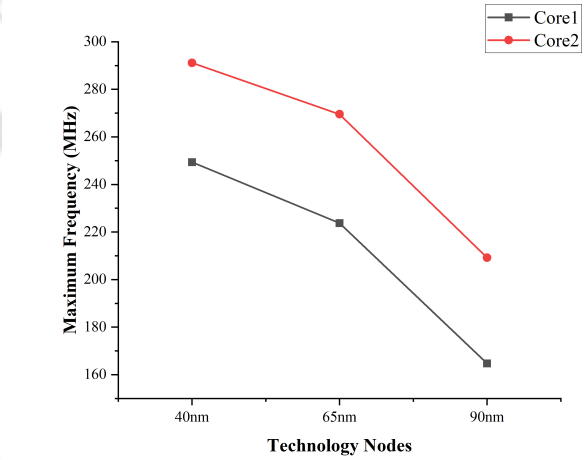
The developed cores, while performing AES encryption, are vulnerable to power side-channel attacks. In this analysis, the Hamming Distance (HD) power model from equation 5.4 is used, specifically targeting the last round of AES. It compares the hypothetical power values with



(a) Power vs technology nodes



(b) Area vs technology nodes



(c) Max.Frequency vs technology

Figure 5.5: Performance parameters of developed cores across different technology nodes

the real power values obtained during AES encryption of multiple plaintexts by RISC-V crypto core. Since the attack happens at the input of the 10^{th} round, AddRoundKey, Inverse S-box, and Inverse ShiftRows are performed to obtain the 10^{th} round intermediate data as shown in equation 5.3.

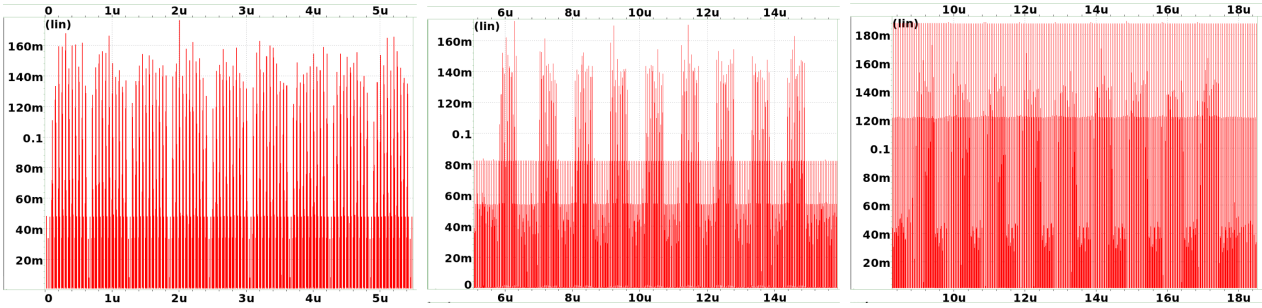
$$v_{i,j} = Sbox^{-1}(ShiftRows^{-1}(k_{i,j} \oplus c_i)) \tag{5.3}$$

and hypothetical power values are denoted by equation 5.4.

$$h_{i,j} = HD(v_{i,j}, c_i) = HW(v_{i,j} \oplus c_i) \tag{5.4}$$

where 'v' is the intermediate result of the 10^{th} round AES input, which is a function of $f(c, k)$, 'c' is the ciphertext, and 'k' represents the last round key values.

The power traces of the designed cores during encryption, obtained using PrimeTime PX are sampled at every 1 ns, and exported as CSV files for analyzing in MATLAB to assess the four security metrics. These metrics provide a comprehensive view of security, with standalone AES evaluated first, followed by Core1 and Core2. Standalone AES was found to be highly vulnerable, with key recovery achieved within fewer than 1,000 traces, as its

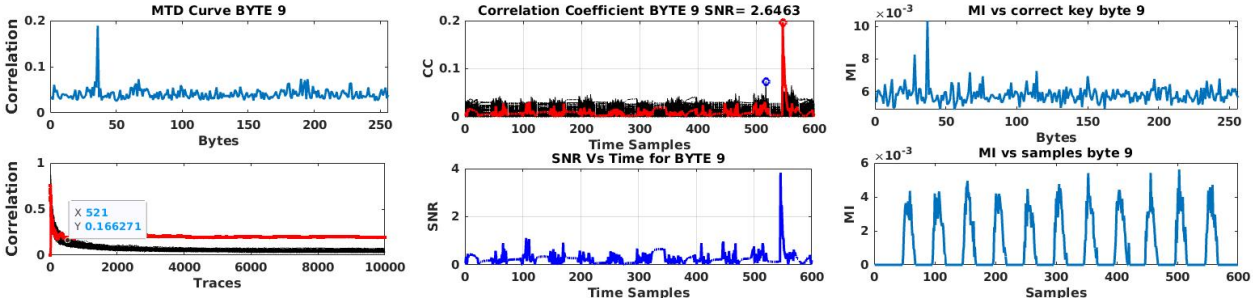


(a) Power traces of AES (b) Power traces of Core1 (c) Power traces of Core2

Figure 5.6: Power traces of Standalone AES and RISC-V Crypto-Cores

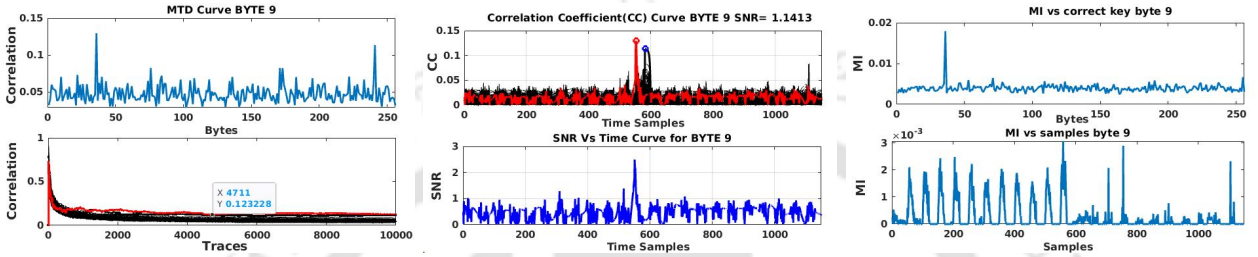
power traces were distinct for each plaintext. In contrast, Core1 and Core2, with additional power contributions from other RISC-V components, displayed improved security, particularly Core2, in which AES power is more effectively masked. Core2's hardware pause halts register updates while keeping other functional units active, generating additional dynamic power to obscure AES power. In contrast, Core1, using a software pause with NOP instructions, executes a simple addition operation with register R0 without updating any values, to synchronize pipeline stages. This adds only a small amount of power from the ALU, which is insufficient to effectively mask the AES power consumption, as shown in Figure 5.6. The increased complexity of Core2, with more pipeline stages and parallel paths, further enhances security by adding more noise and making it harder to correlate AES-specific power patterns. Expanding pipeline stages create greater instruction overlap, increasing power from other modules, thus improving resilience against PAAs.

Four security metrics: MTD, SNR, MI, and TVLA, were evaluated for standalone AES, Core1, and Core2. Among the four security metrics evaluated, SNR and MTD are based on CPA attacks, as they are evaluated using correlational coefficients. MTD indicates the number of power traces required to recover a key byte. A design is considered protected until all key bytes are recovered. In unprotected designs, the MTD as shown in Figure 5.7a represents the number of traces required to recover the last key byte, with the successful key recovery indicated by the red line separating from the sea of black lines. SNR is defined as the ratio between the correlation coefficient of the correct key byte and the highest correlation coefficient among the incorrect key bytes during the data processing. For communication applications, high SNR is typically preferred for faithful signal reproduction. In this context, the signal power is employed as the exploitable leakage power for PAA. Hence, for secure designs, the SNR should be below 1, indicating that signal leakage power is lower than noise power; unprotected designs exhibit SNR values above 1. For unprotected designs as shown in Figure 5.7b, the correct key's correlation coefficient is higher than that of incorrect keys. With increased security, incorrect keys' correlation coefficients rise, nearing or exceeding those of the correct key. MI, uses Shannon's entropy to measure dependency between two random variables, with lower MI values in protected designs, where no distinct peak appears at the correct key byte. In contrast, unprotected designs show a distinct peak at the correct key, indicating leakage as illustrated in Figure 5.7c. TVLA depicted in Figure 5.10 applies Welch's t-test to assess leakage, with secure designs having t-values within ± 4.5 and unprotected designs showing values beyond these limits. MTD recovers each key byte, while MI and SNR not only recover each key byte but also indicate whether the design is leaky. In contrast, TVLA provides only leakage information based on t-values. As security increases, MTD increases, while SNR, MI, and TVLA decreases.



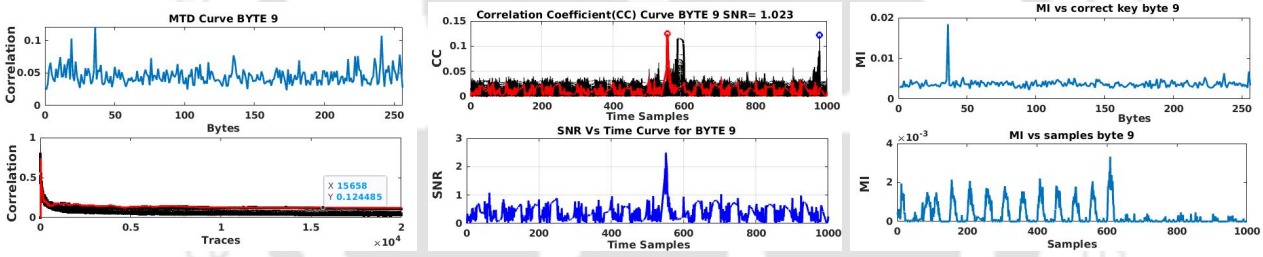
(a) MTD plot (b) SNR plot (c) MI plot

Figure 5.7: PAA results of byte 9 for standalone AES (Masoleh S-box)



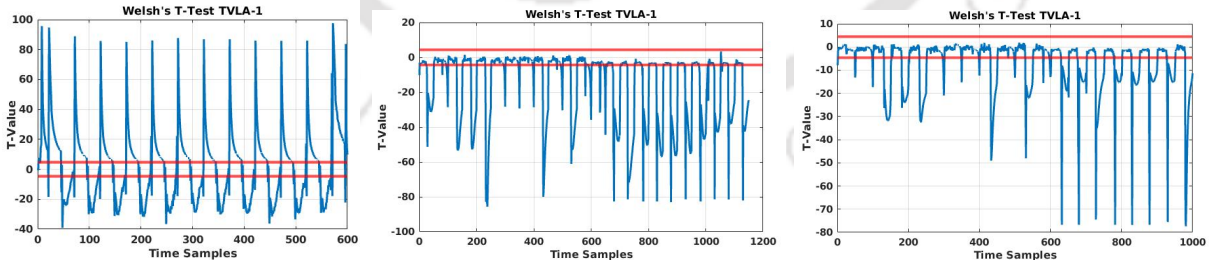
(a) MTD plot (b) SNR plot (c) MI plot

Figure 5.8: PAA results of byte 9 for Core1



(a) MTD plot (b) SNR plot (c) MI plot

Figure 5.9: PAA results of byte 9 for Core2



(a) TVLA result of AES (b) TVLA result of Core1 (c) TVLA result of Core2

Figure 5.10: TVLA results of standalone AES and RISC-V crypto-cores

Results reveal that standalone AES is the most vulnerable, as the entire power consumption is attributed by AES cryptographic operations. Embedding AES within RISC-V cores in Core1 and Core2 significantly improves security by masking AES power with contributions from other components. As core complexity and pipeline stages increase, additional power from non-AES modules further conceal AES power, enhancing security metrics further. Dur-

ing testing, all cores were operated at 20 MHz frequency with a sampling rate of 1 GSa/s. The HD power model targets to recover the 10^{th} round key of AES, thereby derive actual key. Standalone AES takes 12 clocks to complete AES, hence recorded 600 samples per encryption of each plaintext. MTD analysis indicated that all AES key bytes were recovered within 1,000 samples, with the last recovered byte (byte 9) taking 521 traces, confirming high vulnerability. The core2, with a more complex design, requires 20 clocks out of these, 12 clocks are for AES operations, and the remaining 8 clocks are used to transfer, yielding 1,000 samples of which 600 samples correspond to AES operations, and the remaining samples are used for data transfer between the AES FU and register bank. The MTD, SNR, and MI results corresponding to Byte 9 of each design are shown in Figure 5.7, Figure 5.8 and Figure 5.9. The TVLA for the all developed core is in shown Figure 5.10.

Table 5.5 summarizes the security metric improvements across the developed designs. Using MTD analysis, all the key bytes of the standalone AES, Core1 and Core2, were successfully recovered. Standalone AES, with an MTD of 521, is highly vulnerable. Core1 improves MTD to 4,711, and Core2 achieves the highest at 15,658, indicating increased security due to more complexity. This indicates that larger number of traces are necessary to retrieve the key as the core's complexity increases. MTD improvements for Core1 and Core2 are 9.04× and 30.1× over standalone AES, with Core2 showing a 3.82× improvement over Core1. SNR and MTD have an inverse relationship, meaning that as designs become more secure, SNR decreases. Hence, SNR decreases with improved security. Standalone AES has the highest SNR at 2.64, Core1 at 1.15, and Core2 at 1.02, corresponding to improvements of 2.2× and 2.6× over standalone AES, with Core2 showing a 1.13× improvement compared to Core1. MI values follow similar trends, with standalone AES at 6.24 milli, Core1 at 3.51 milli, and Core2 at 3.06 milli, showing improvements of 1.78× and 2.04× over standalone AES, with Core2 achieving a 1.15× improvement over Core1. In terms of TVLA, which indicates the level of leakage in designs, standalone AES displays the highest value at 97.37, followed by Core1 at 85.47, and Core2 at 77.37, showing 1.14× and 1.26× improvements over standalone AES, with Core2 achieving a 1.11× improvement over Core1. Among the designs, Core2 demonstrates the highest MTD and lowest SNR, MI, and TVLA values, indicating enhanced security and a lower likelihood of key recovery. As pipeline stages increase, more modules contribute to overall power, further obscuring AES-specific power consumption and improving security metrics. Standalone AES, with power purely from AES operations, remains the most vulnerable, with keys recoverable with minimal traces.

Table 5.5: Hardware security metrics of Core1 and Core2

Hardware Security Metrics				
Design	MTD_{max}	SNR_{min}	MI_{min}	$\lvert TVLA_{max}\rvert$
Standalone AES	521	2.64	$6.24*10^{-3}$	97.37
Core1	4,711	1.15	$3.51*10^{-3}$	85.47
Core2	15,658	1.02	$3.06*10^{-3}$	77.37

*Evaluated considering all bytes

In summary, the complexity of the cores mainly depends on the number of pipelining stages, parallel operations, and pipeline pausing techniques while the AES FU is active. As the complexity of the developed core increases, the chance of getting the key is reduced, but still, the keys can be recovered at a higher number of traces. The key recovery becomes more challenging, as evidenced by the increased MTD, reduced SNR, and MI.

Table 5.6: Summary of relative advantages and disadvantages of the proposed Core1 & Core2

Feature	Core1 (SW Pause)	Core2 (HW Pause)
No: of pipeline stages	3-Stage	4-Stage
Pipeline Stages	IF, ID, IE+RW	IF, ID, IE, RW
Maximum operating Frequency	Larger critical paths due lesser pipelines stages, resulting in lower frequency.	Smaller critical paths, due increased pipeline stages, reulting in higher frequency.
Performance	Single write ports, normal techniques for handling control hazards, lower throughput	Dual write ports, advanced techniques (DBP) for handling control hazards, higher throughput
Area & Power	Lower area and power due to simple design	Slightly higher area & power due to added logic & advanced features
Synchronization	Utilize software pause by inserting NOPs in the programs. Synchronization is simpler & requires less hardware resources.	Generate hardware pause signal, while AES is active. It achieves higher performance, at the expense of increased area and complexity for synchronization.
Code Size	Larger due to extra NOPs in programs	Smaller as no NOPs are added in programs, instead use hardware pause signal
Security Metrics	Moderate (better than standalone AES)	Higher security due to improved masking of AES power, as more modules of RISC-V operates in parallel with AES
Ideal Use Case	Low-power, low-area IoT applications	Real-time, high-performance, secure applications

The relative advantages and disadvantages of the proposed Core1 and Core2 are summarized in Table 5.6. The table highlights the impact of pipeline depth and pausing techniques on the performance and security metrics of the proposed cores.

5.6 Chapter summary

This chapter focus on enhancing the cryptographic capabilities of IoT edge devices by integrating an AES hardware module within the RISC-V processor through ISA extensions. To achieve this, two RISC-V cryptographic core micro-architectures, Core1 and Core2, have been developed. These cores incorporate an AES Functional Unit (FU) into the 32-bit RISC-V processor, enabling efficient, low-power encryption. This reduces energy consumption while enhancing security by minimizing the direct attack surface and increasing resistance to power side-channel attacks. While both cores feature tightly coupled AES modules within the RISC-V architecture, Core1 and Core2 mainly differ in the number of pipelining stages and the pipeline pausing techniques when the AES FU is active. It is observed that as the cores' complexity increases, leakage is reduced, making key recovery more difficult for attackers. This tightly coupled AES RISC-V architectures achieve significantly higher throughput and lower energy consumption during AES encryption, with lower utilization of hardware resources, compared to its AES software implementation using its baseline cores. The hardware security analysis using PAA models depicts that as the complexity of the RISC-V cores increases, the cores become more secure to power attacks, but still, they are vulnerable to PAA. This chapter primarily contributes to a deeper understanding of the type of countermeasures to be designed to secure RISC-V architectures against PAA. It enables us to tailor countermeasure designs based on the RISC-V architecture and specific application requirements, rather than directly employing an enormous countermeasure. Thus, the security metrics can be improved multiple folds without significant area or power costs. Building on this foundation, upcoming chapters will explore the development of countermeasures with minimal area and power overheads. These countermeasures are intended to secure embedded processors in IoTe devices against PAA and achieve an MTD of over one million, without requiring extensive countermeasure design. This approach strives to improve the resilience of IoTe devices against attacks, ensuring robust security in resource-constrained environments.

CHAPTER

6

NON-LINEAR CYCLIC VARIABLE CLOCK FEISTEL BRIDGE-INSPIRED COUNTERMEASURE FOR SECURING RISC-V CRYPTO-CORE AGAINST POWER ATTACKS

Contents

6.1 Introduction

With the increasing popularity of IoTe devices, RISC-V emerges as the most suitable architecture for various applications. Since attackers have direct physical access to these IoTe devices, securing these RISC-V [1], designs are of major concern. The AES [2] stands out as the most effective cipher for securing these RISC-V processors in resource-constrained IoTe applications. Thus, the AES hardware accelerator can be seamlessly integrated with RISC-V processors, either in a tightly [33], [35] or loosely [32], [31] coupled manner. In tightly coupled designs, compact AES hardware with limited capabilities are usually integrated into the processor pipeline to minimize its impact on the processor's timing performance. Even though the embedded processors are secured in these IoTe devices, by integrating AES, they are still susceptible to PAA. Therefore, it is necessary to employ countermeasure designs that utilise masking [73], [90] or hiding techniques [91, 92] to enhance the devices' security. Hiding techniques minimize the exposure of sensitive data, while masking techniques masks the results with random data, to obscure the relationship between sensitive data and observable values. Additionally, various other countermeasure techniques like employing dynamic keys instead of static keys [71], implementing on-chip current flattening technique [51], and cryptographic accelerators of RISC-V operating at Random Dynamic Frequencies [64] are available. While these countermeasure techniques bolster RISC-V security, they come with trade-offs in terms of increased area and power consumption due to the integration of additional circuitry. This is particularly challenging for battery-powered IoT devices operating under strict area and power constraints.

In this chapter, an extremely small Non-Linear Cyclic Variable Clock Feistel Bridge-Inspired Countermeasure (NCVCFB) with minimum overheads is implemented. It provides variation in both amplitude and temporal domains of signal to enhance AES security in RISC-V based IoTe devices, specially to address the vulnerability against PAA. The NCVCFB countermeasure employs the rolling architecture and will operate in tandem with each round of AES to obscure power consumption patterns. It will execute additional random number of rounds cyclically in the first and last round of AES, based on the randomly generated number. The newly designed, secured RISC-V achieved remarkably low area and power overheads of 1.05% and 1.23%, respectively, without affecting the maximum operating frequency. The PAA was performed using the power traces captured from post-layout design on ASIC platform at UMC 65nm technology node as well as on the experimental hardware setup employing a Side-channel Attack Security Evaluation Board (SASEBO). The resilience of the secured RISC-V architecture against PAA was tested by subjecting it to 2 Million traces, and none of the bytes got recovered.

The chapter is organized in the following manner: Section 6.2 provides an intricate examination of several AES designs and selection of the most suitable AES architecture based on optimal area, power, throughput and security. Section 6.3 presents the novel Non-linear Cyclic Variable Clock Feistel Bridge-Inspired Countermeasure (NCVCFB) technique. Section 6.4 thoroughly analyzes the resource utilization and security metrics of the NCVCFB countermeasure integrated with standalone AES and its application in securing RISC-V.

6.2 Selection of AES hardware

This section focuses on selecting a compact, power-efficient AES hardware module that delivers reasonable throughput while offering improved intrinsic resilience to PAA. To further enhance security, the NCVCFB countermeasure is integrated into this compact AES ar-

chitecture, minimizing the overall area and power overhead, while maintaining fairly good throughput, and achieving high resistance to PAA.

Table 6.1: Comparison of the designed 128-bit AES architectures with state of the art

AES Architectures	Performance Metrics								Security Metrics	FoM
	Tech. [nm]	No. of clks	Area [mm²]	kGE	Dynamic power* [µW/MHz]	Delay [ns]	Max. Freq [MHz]	Throughput [Gbps]	MTD	[Gbps/ (um2)*(W)]
Different type of S-Box based AES										
12-Clock AES (LUT S-box)	65	12	0.026007	18.06	4.80	1.57	636.94	6.79	630	342.89
12-Clock AES (Canright S-box)	65	12	0.016815	11.7	4.43	2.49	401.61	4.28	427	245.43
12-Clock AES (Masoleh S-box)	65	12	0.013503	9.38	4.07	1.81	552.49	5.89	521	558.50
Different Clock AES										
22-Clock AES (Masoleh S-box)	**65**	**22**	**0.012921**	**8.97**	**4.17**	**1.8**	**555.56**	**3.23**	**873**	**524.18**
42-Clock AES (Masoleh S-box)	65	42	0.012136	8.38	4.29	1.8	555.56	1.69	982	321.01
82-Clock AES (Masoleh S-box)	65	82	0.012004	8.34	4.501	1.79	558.66	0.87	1297	209.33
162-Clock AES (Masoleh S-box)	65	162	0.012719	8.85	4.987	1.79	558.66	0.4414	1652	114.81
Existing AES										
Singh, JSSC'20[93]	130	-	0.275	-	136.25	-	80	-	800	-
Bellizia, TVLSI'18[91]	65	44	0.0416	20	-	-	20	58.2 Mbps	72	-
Huang,TCAD'19[94]	65	10	0.0636	-	-	-	372	4.76	1100	-
Utsav, JSSC'19 [31]	65	11	0.015	10.6	-	-	-	-	-	-
Lui, ESSCIRC'11 [95]	90	10	0.0979	15	29.95	3.92	255	2.99	-	-
Tokunaga , JSSC'10 [96]	130	11	0.35	-	333.2	5	200	2.56	4000	-
David, [97]	180	11	0.79	-	-	3.03	330	3.84	-	-
Agwa, ISCAS'17 [98]	130	-	0.296	-	214.8	1.5	666	2.601	-	-
Bui. TVLSI'17 [99]	28	44	-	8.6	2	100	10	28 Mbps	-	-
Lu. IEEE TrustCom'18 [100]	28	213	0.0028	-	-	20	50	30.05 Mbps	-	-

*Dynamic power is normalized w.r.t clock frequency

In 128-bit AES, the 16-bytes are substituted with corresponding values in SubBytes step using S-boxes. The power attack primarily focuses on the power-intensive nonlinear SubBytes operation. By strategically designing the SubBytes function, a compact, low-power AES with high throughput and improved resilience to PAA can be achieved. It requires mainly two steps, first is the proper selection of S-box and second is spreading of SubBytes operations over multiple clock cycles, by utilizing fewer S-boxes, instead of 16 S-boxes at a time. Firstly, the S-box, should be chosen wisely, while designing an area-efficient AES system, as it is occupying the majority of the area. The usage of LUT-based S-box in AES designs is prevalent in literatures [18],[19],[20] due to its easiness and simplicity in implementation. But the implementations using conventional LUT-based S-boxes requires storing 256 bytes and additional circuitry for address decoding and fetching. Such extensive hardware demands make this approach unsuitable for IoT applications, where minimizing area and power consumption is crucial. Compact AES designs using Canright and Masoleh S-boxes have been proposed as replacements for the LUT S-box. For a fair comparison, AES implementations with these S-boxes were designed in UMC 65nm, with results compared to existing architectures, as shown in Table 6.1. The Masoleh-based AES achieved a significant area reduction and the lowest power consumption, while maintaining reasonably good throughput. All three designs showed comparable MTD values. Based on these parameters, the FoM was calculated using equation 6.1.

$$FoM = \frac{T * MTD}{A * P} \tag{6.1}$$

where, T denotes throughput, A represents area, P stands for power, and MTD represents the number of traces required to disclose the key. The Masoleh-based AES design has demonstrated superior performance compared to other AES implementations. Therefore, it has been utilized to implement AES architectures with different clock cycle configurations.

In conventional 12-clock AES rolling architecture, all 16 S-boxes of the SubBytes operation are instantiated and implemented in a single clock cycle to maximize throughput,

thereby the performance. However, this comes at the cost of significantly increased area and huge information leakage, requiring complex resource-intensive countermeasure to obscure the power due to these 16 S-boxes. To address these limitations, AES designs with higher clock cycles are implemented, which reduces the number of simultaneously active S-boxes by distributing the SubBytes operation over multiple clock cycles. To minimize the area and reduction in the leakage caused by SubBytes, the S-box is implemented in the power of 2 ($2^0 = 1$, $2^1 = 2$, $2^2 = 4$, $2^3 = 8$, and $2^4 = 16$) across multiple clock cycles by utilizing fewer number of S-boxes in each cycle, such as 2-clock, 4-clock, 8-clock, and 16-clock implementations. At the architectural level, the SubBytes function can be implemented with different number of clocks, as shown in Figure 6.1. In 1-clock cycle SubBytes implementation, all 16 S-boxes are instantiated simultaneously, attaining high performance, but at expense of huge area. For a 2-clock cycle implementation, 8 S-boxes are reused twice across the two clock cycles, balancing throughput, power consumption and resource utilization. The 4-clock cycle design sequentially invokes 4 S-boxes over 4 cycles, offering a middle ground between speed and resource utilization. In an 8-clock cycle setup, 2 S-boxes are reused across 8 cycles, emphasizing area efficiency over speed. Lastly, a 16-clock cycle configuration triggers one S-box per cycle over 16 cycles, drastically reducing the throughput. In the 162-clock AES design, both dynamic power and area increase despite using fewer S-boxes. This is primarily due to the overhead introduced by additional registers required for data synchronization and storage, which adds to switching activity and hardware complexity.

Table 6.1 shows that 22-clock AES architecture strikes an effective balance by using 8 S-boxes at a time across 2 clock cycles for the SubBytes operation. This 22-clock AES offers fairly good throughput at much lesser area and better security than 12-clock AES designs. The 22-clock AES design, compared to the 42, 82, and 162-clock designs, achieves relatively lower dynamic power while avoiding the severe throughput degradation that affects overall performance in the higher-clock designs. As a result, it achieves an efficient compromise between resource utilization, performance, and side-channel resistance. For battery operated IoT applications, a 22-clock AES design is optimal as it balances power, area, and performance. This design employs fewer S-boxes (8 running in parallel), making it easier for a smaller countermeasure to conceal AES power.

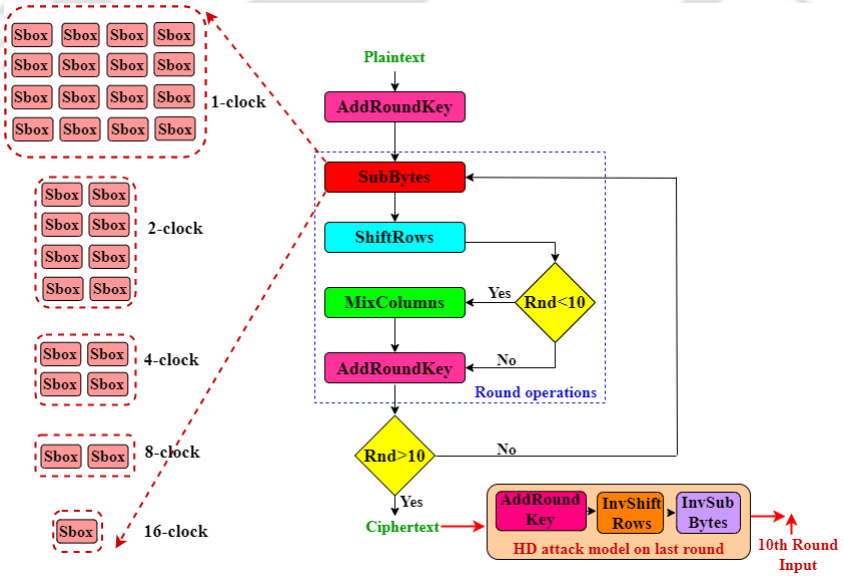


Figure 6.1: Architectural variation in different clock AES and HD attack model

6.3 Novel Non-linear Cyclic Variable Clock Feistel Bridge-Inspired Countermeasure

CPA attacks recover keys by correlating real and hypothetical power samples at every instant of time for multiple plaintext while the cryptographic devices are encrypting or decrypting data. To secure RISC-V core during cryptographic operations, power patterns are varied in both amplitude and time domains. Random operations are performed along with AES operations to alter amplitude, and AES rounds are executed non-periodically by inserting random operations in between each AES rounds to shift data in the time domain. By varying the power in amplitude domain and deliberately misaligning it in time domain during the SubBytes operations, the CPA attack's primary target is effectively disrupted. This prevents the hypothetical power values from matching the actual values, thereby thwarting the attack.

The S-box which relies on non-linear operations is the primary contributor of leakage power in CPA. In 12-clock AES design, 16 S-boxes operate in parallel, resulting in significant leakage power, thus the initial step involves converting the 12-clock AES to a 22-clock AES, followed by implementing a small countermeasure on it.

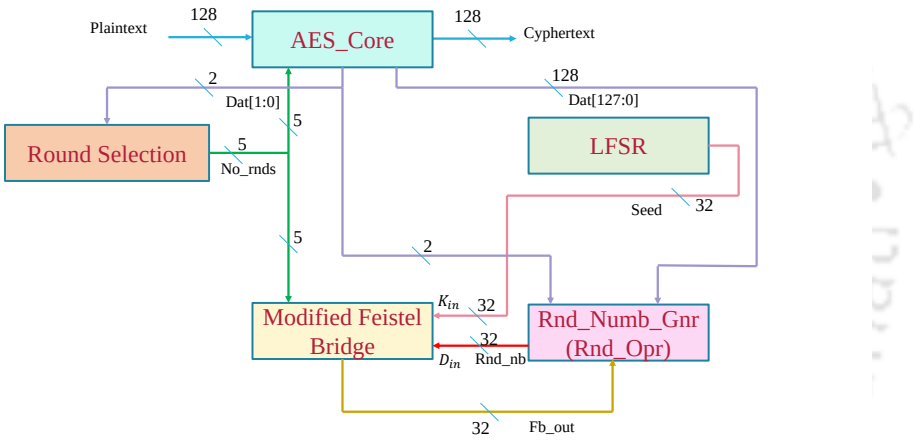


Figure 6.2: 32-bit NCVCFB Countermeasure Architecture

The Non-linear Cyclic Variable Clock Feistel Bridge-Inspired Countermeasure (NCVCFB) design in Figure 6.2, aims to develop a very small countermeasure based on Feistel Bridge. In a conventional 32-bit Feistel bridge architecture, a 32-bit data and key are used, typically involving four rounds where the 32-bit data is split into two halves, L_{in} and R_{in}, which are cross-coupled in each round. During each round, R_{in} and the key are processed through a function, with the output XORed with L_{in} to become R_{in} for the next round, while R_{in} is used as L_{in} in the next round. To minimize area usage, the standard Feistel bridge is first transformed into a rolling architecture, optimizing area at the cost of additional clock cycles. This rolling Feistel bridge is further modified to achieve the desired countermeasure design.

The NCVCFB countermeasure is a compact 32-bit architecture highly suitable to secure RISC-V processors, with minimal overheads. It comprises of four main blocks: the round selection block, the random operator-based RNG, the LFSR, and the modified Feistel Bridge, as depicted in Figure. 6.2.

(1) Round Selection Block: determines the number of rounds the modified Feistel bridge should execute, ranging from 1 to 5, thereby providing variations in temporal domain. The decision for additional rounds in the Modified Feistel bridge is based on the last 2-bits of AES

round input for the first and last rounds. Normally, the Modified Feistel bridge runs for 1 round, taking 1 clock cycle, and operates in parallel with each AES round. Depending on the derived number of rounds, the modified Feistel bridge runs an additional 1 to 4 rounds, during which the AES round is temporarily halted. The modified Feistel bridge runs for additional rounds only in AES's first and last rounds, not in the intermediate rounds. This is because, CPA attacks either first or last round power samples of AES depending on HW or HD attack model. This selective application introduces temporal variation without significantly reducing throughput, and enhancing the overall security of AES encryption. This approach is more efficient than implementing additional rounds of modified Feistel bridge in all AES rounds, as it effectively boosts security while maintaining performance.

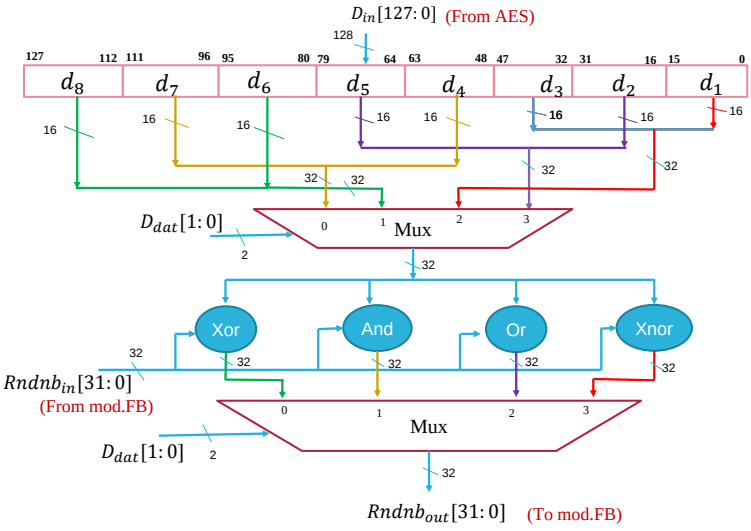


Figure 6.3: Random Operator-Based Random Number Generator

(2) Random Operator-Based Random Number Generator (Rnd-Opr RNG) block: generates random numbers using different non-linear and linear operators as shown in Figure 6.3. The output from the modified Feistel Bridge serves as one input to the Rnd-Opr RNG block, while the other input is a randomly selected 32-bit segment from the 128-bit AES data generated in each round. The operation performed on these two inputs either AND, OR, XOR, or XNOR is determined by the last two bits of the AES data in each round. The resulting random data from this operation serves as the 32-bit input data for the next round in the modified Feistel bridge. Among these logic operations, only XOR is a linear operation because its output changes in a predictable, consistent way with changes in input, and the original data can be recovered by applying XOR operation again. In contrast, AND, OR, and XNOR are non-linear operations, because output can vary unpredictably with input changes, and the original inputs cannot be recovered by performing the same operations.

(3) Linear Feedback Shift Register (LFSR) block: generates a 32-bit Seed that is used as an input for the modified Feistel Bridge block. A LFSR is a digital circuit commonly used to generate pseudo-random bit sequences. The LFSR requires a one-time initial input, known as an initial seed which is stored in a register. Once initialized, the LFSR operates independently and does not require continuous external inputs, during its operation to generate its output.

The circuit diagram of the 32-bit LFSR, depicted in Fig 6.4, consists of a series of 32 one-bit registers (flip-flops) connected sequentially, forming a shift register where each bit is shifted to the next position with every clock cycle. These registers are labelled from Q_1 (most significant bit) to Q_{32} (least significant bit). At each clock pulse, the content of each

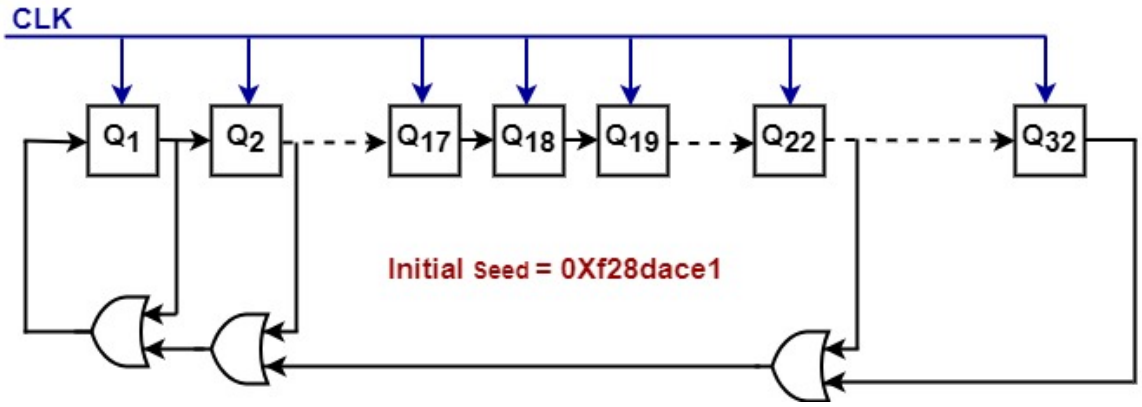


Figure 6.4: Architecture of 32-bit LFSR

register shifts to the next higher register, and a new bit is computed and fed into Q_1. This new bit is generated by applying XOR logic to the outputs of specific registers, called taping points. The choice of these tap points is critical in determining the maximum length of the pseudo-random sequence. To ensure that the LFSR generates the longest possible pseudo-random sequence with maximum non-repeatability of 2^{32}-1 , two essential conditions must be satisfied. First, the tap points used must correspond to a primitive feedback polynomial. Second, the LFSR must be initialized with a non-zero initial seed. In this case, the chosen primitive feedback polynomial is $x^{32}+x^{22}+x^2+x^1+1$, which defines the tap points at Q_{32}, Q_{22}, Q_2, and Q_1. The outputs from these specific registers are XORed together, and the resulting bit is fed back into Q_1, forming the feedback mechanism of the LFSR. The initial seed selected for this LFSR is 0Xf28dace1 and used with a proper primitive feedback polynomial, $x^{32}+x^{22}+x^2+x^1+1$ to obtain the maximum non-repeatability of the seed.

(4) The 32-bit Modified Feistel Bridge: forms the core of this innovative countermeasure designed to obscure the AES power signature. It takes two inputs, a 32-bit data D_{in} provided by the Random Operator-Based RNG and a 32-bit key K_{in}, derived from the LFSR. It employs a series of modified Feistel functions which thoroughly mix and diffuse the random data, contributing significant power to hide S-box power. The output of this modified Feistel bridge is fed back to the Random Operator-Based RNG as an input. This entire countermeasure forms a feedback loop, interconnecting all modules, including AES. Consequently, it introduces temporal variation and disrupts the amplitude domain of the AES trace pattern by injecting random noise, making the design highly resistant to PAA. These four blocks work together to create a secure, compact and efficient countermeasure, leveraging the unique properties of non-linear cyclic variable clocks and the Feistel bridge architecture to protect against PAA.

6.3.1 32-bit Modified Feistel Bridge

In the actual Feistel bridge, 32-bit input data D_{in} and a 32-bit key K_{in} are used and operate over 4 fixed stages. The data is divided into two halves, the right input R_{in} and the left input L_{in}, with R_{in} directly passed as L_{in} for the next stage. Three key changes are made in the modified Feistel bridge, as shown in Figure 6.5. Firstly, the number of stages executed in the modified Feistel bridge can vary. Generally, from 2^{nd} to 9^{th} rounds of AES, the modified Feistel bridge completes its operation in single stage, and the output after this stage is the final output. However, during the first and last rounds of AES, the modified Feistel bridge

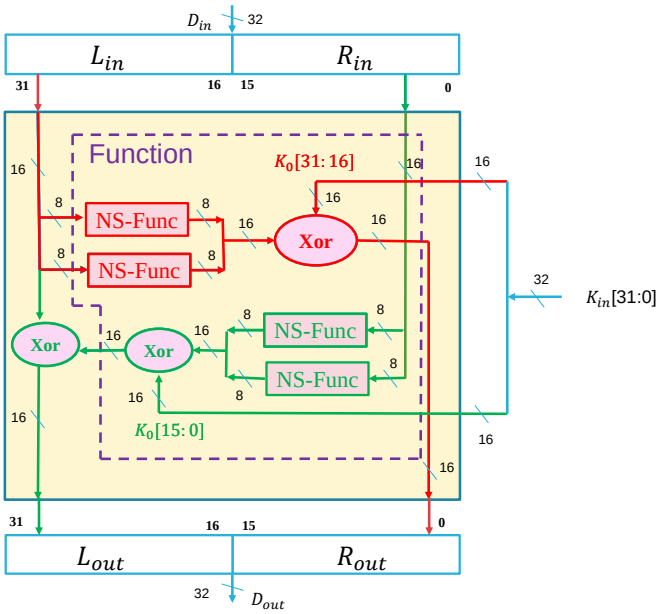


Figure 6.5: 32-bit Modified Feistel Bridge

operates with a variable number of stages, producing the final output after 2, 3, or 4 stages, depending on the specific round number selected.

Secondly, the 32-bit data is divided into two 16-bit parts, R_{in} and L_{in}. Each 16-bit R_{in} is processed through two New Substitution Function (NS-Func) modules, transforming it into a new 16-bit data, which is then XORed with the least significant 16 bits (LSB) of the key. This result is further XORed with L_{in} and used as the L_{in} input for the next round. In contrast, a standard 32-bit Feistel bridge directly uses R_{in} as the L_{in} input for the next round. Similarly, each 16-bit L_{in} is processed through two NS-Func modules, converting it into new 16-bit data, which is XORed with the most significant 16 bits (MSB) of the key and used as the R_{in} input for the next round.

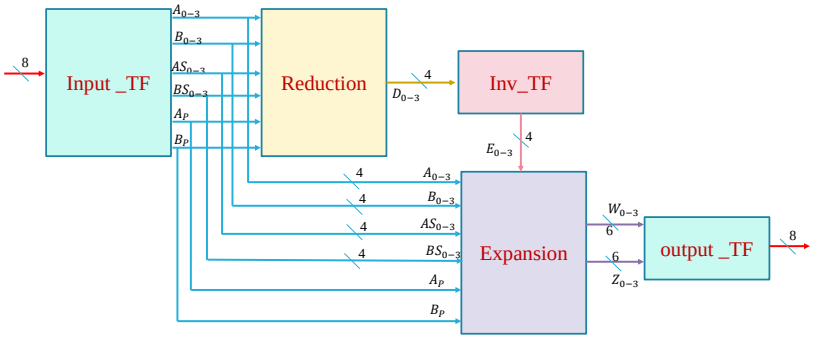


Figure 6.6: 8-bit New Substitution Function (NS-Func)

Thirdly, the actual Function (Func) that uses both D_{in} and K_{in} in a standard Feistel bridge is replaced by a New Substitution Function (NS-Func). This NS-Func processes 8-bit data in, converting it into new 8-bit data. The NS-Func as shown in Figure 6.6 consists of five blocks, including the Input Transfer Block, Reduction Block, Inverse Transfer Block, Expansion Block, and Output Transfer Block.

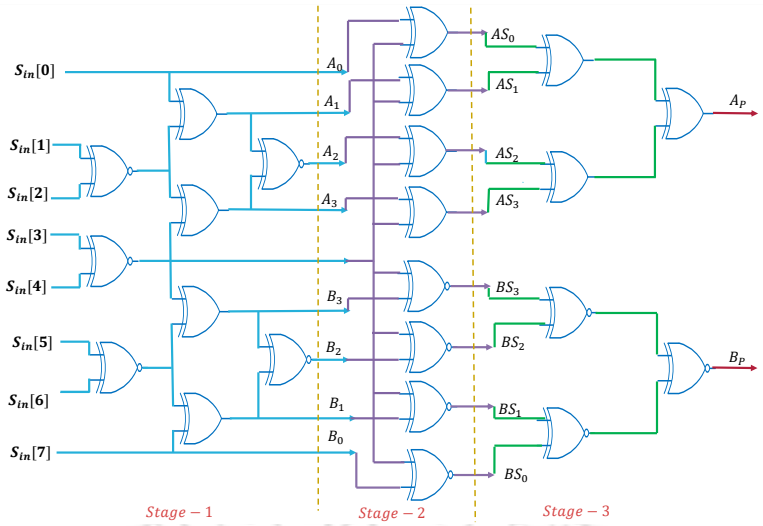


Figure 6.7: Input transfer block

The Input transfer block, shown in Figure 6.7, contains only linear gates XOR and XNOR. It takes 8-bit inputs and generates 18-bit outputs through a three-stage architecture. The first stage uses alternating XNOR and XOR gates to produce an 8-bit output. In the second and third stages, the upper half contains XOR gates, and the lower half contains XNOR gates. The second stage generates another 8-bit output, while the third stage produces a 2-bit output. These outputs are then sent to the Reduction Block.

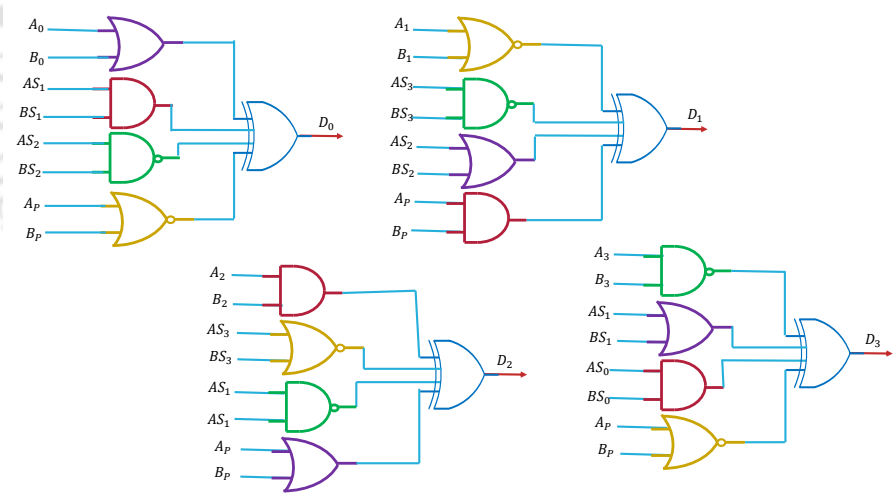


Figure 6.8: Reduction Block

The reduction block is mainly used for the reduction of bits, as depicted in Figure 6.8. It converts these 18-bits outputs from input transfer block into 4-bits. It utilizes non-linear gates like AND, OR NOR, and NAND gates and the final results are XORed. Each of these gates plays a role in manipulating the data and to reduce its bit-length. The output of reduction block is given to Inverse transfer Block.

The Inverse transfer Block, shown in Figure 6.9 is designed to transform a 4-bit data input into another 4-bit data output through an OR-AND function architecture. Initially, the four inputs received from the Reduction Block are processed using XOR and XNOR gates, generating intermediate outputs. These outputs are further given to an OR-AND structure to produce the final 4-bit output of the Inverse Transfer Block.

The Expansion Block, detailed in Figure 6.10 processes inputs from the Input Transfer Block and the 4-bit data from the Inverse Transfer Block, converting them into two sets of 6-bit data. These outputs are sent directly to the Output Transfer Block.

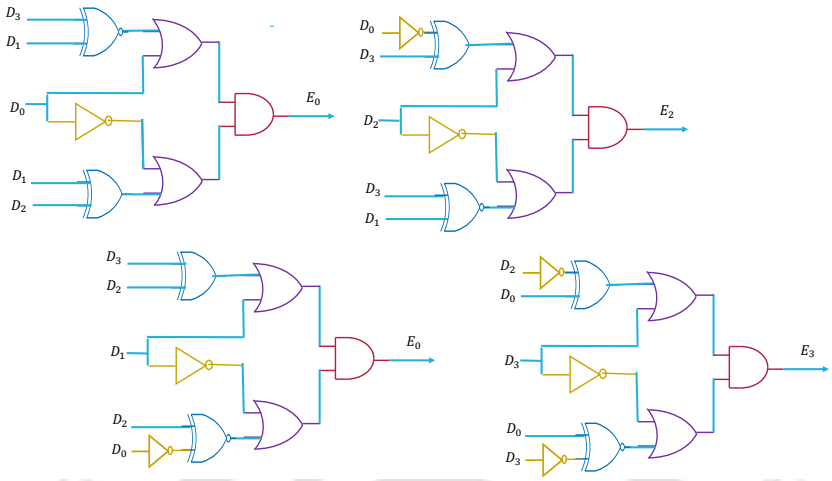


Figure 6.9: Inverse transfer Block

The Output Transfer Block depicted in Figure 6.11 converts the two 6-bit data outputs from the Expansion Block into an 8-bit data output. This block employs a 3-stage architecture composed solely of linear gates. In the first stage, XOR gates are used. The second stage utilizes XNOR gates, and the final stage incorporates both XOR and XNOR gates.

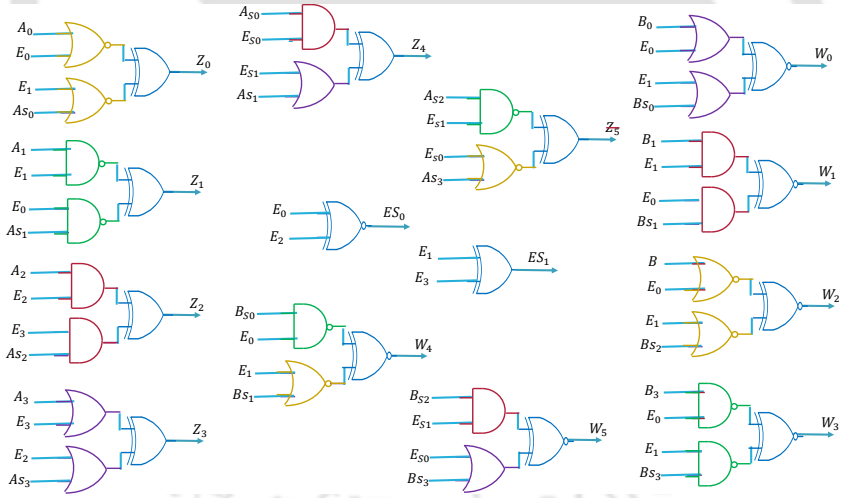


Figure 6.10: Expansion Block

The structured approach of NS-Func ensures an efficient transformation of the 8-bit input data into the 8-bit output data. The inclusion of multiple stages in the combinational path of the NS-Func is a strategic design choice driven by the need to operate modified Feistel bridge in parallel with AES.

The S-box in AES as a major source of power leakage due to its highly nonlinear, bit-level transformations, which cause significant variations in power consumption. To mitigate this vulnerability, the New Substitution Function (NSFunc) in the modified Feistel Bridge of the NCVCFB countermeasure is designed as illustrated Figure 6.6, with its detailed gate-level representation for each block provided in Figures 6.7 -6.11. The primary purpose of designing the NS-Func in such away is to ensure that the NSFunc operates in parallel with each round

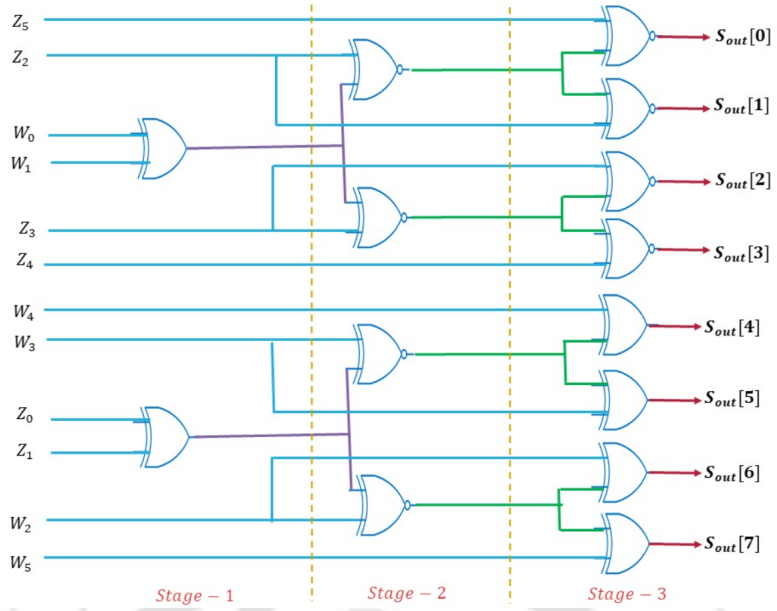


Figure 6.11: Output Transfer Block

of AES, thereby the power leakage caused by the S-Box is effectively masked by the additional power generated by the NS-Func in the Modified Feistel Bridge. Its effectiveness in concealing power leakage is driven by the following key design elements:

(1) Synchronous Activation of NS-Func with S-Box: To mask power leakage, NS-Func operates in parallel with the S-Box, ensuring it is active throughout the S-Box's operation. This synchronization is achieved by matching the Critical Path Delay (CPD) of the NS-Func to that of the S-Box. Since the Masoleh S-Box used in AES is a purely combinational circuit comprising five stages, the NS-Func is also implemented in five stages, ensuring comparable CPD. This ensures simultaneous operation, effectively masking power leakage by introducing additional power activity that overlaps with the S-Box's power consumption.

(2) Randomization of Data-Dependent Power Consumption: The NS-Func is designed in such that it produces comparable power to the S-Box but with a significantly smaller area, making it suitable for resource-constrained environments. This is achieved by random gate-switching activity, as dynamic power consumption depends primarily on switching. PAA exploit the dependency of a device's power consumption on the data it processes and the operations it performs. To mitigate this, the S-Box in AES processes actual data, while the NS-Func operates on random data generated by a Random Operator-Based Random Number Generator in the NCVCFB countermeasure. By randomizing the data processed by the NS-Func, the overall power variation becomes less correlated with the S-Box's processed data, effectively reducing the vulnerability to power attacks.

(3) Randomization of Operation-Dependent Power Consumption by proper Gate-Level Design of NS-Func: The power consumption of the S-Box is primarily driven by gate switching. The Masoleh S-Box uses Linear gates in the first and last stages and Non-linear gates in the intermediate stages. To complement this, the NS-Func adopts a similar five-stage design but uses complementary gates to balance switching activity between the S-Box and NS-Func, effectively randomizing power consumption. Moreover, the NS-Func employs a multi-level OR-AND logic gate architecture in its Inverse Transfer Block to randomise switching activity and reduce short-circuit current. This gate-level design ensures that the NS-Func's power consumption is comparable to, but distinct from, the S-Box's power consumption, effectively masking the S-box's power signature.

(4) Enhanced Power Concealment in Critical AES Rounds: Power attacks primarily target the first and last rounds of AES. To further obscure power leakage, the NCVCFB countermeasure operates additional varying random rounds in the first and last rounds of AES, randomizing AES power patterns in both amplitude and temporal domain.

The strategic gate-level design of the NS-Func in the modified Feistel bridge significantly alters the AES power pattern by being active during the entire S-Box operation. Thus, when both S-Box and NS-Func operate simultaneously, their combined power randomizes the power patterns, effectively concealing AES power leakage. This makes it challenging for attackers to correlate power variations with the actual data processed.

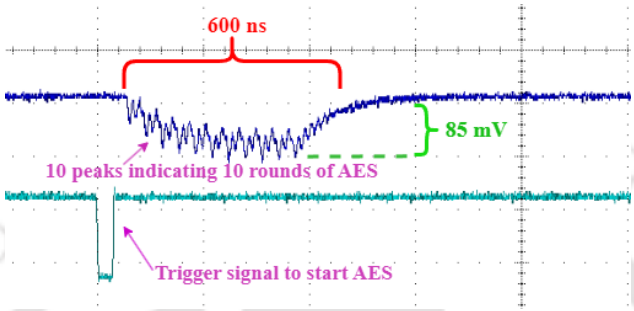


Figure 6.12: Power patterns of unprotected AES on SASEBO

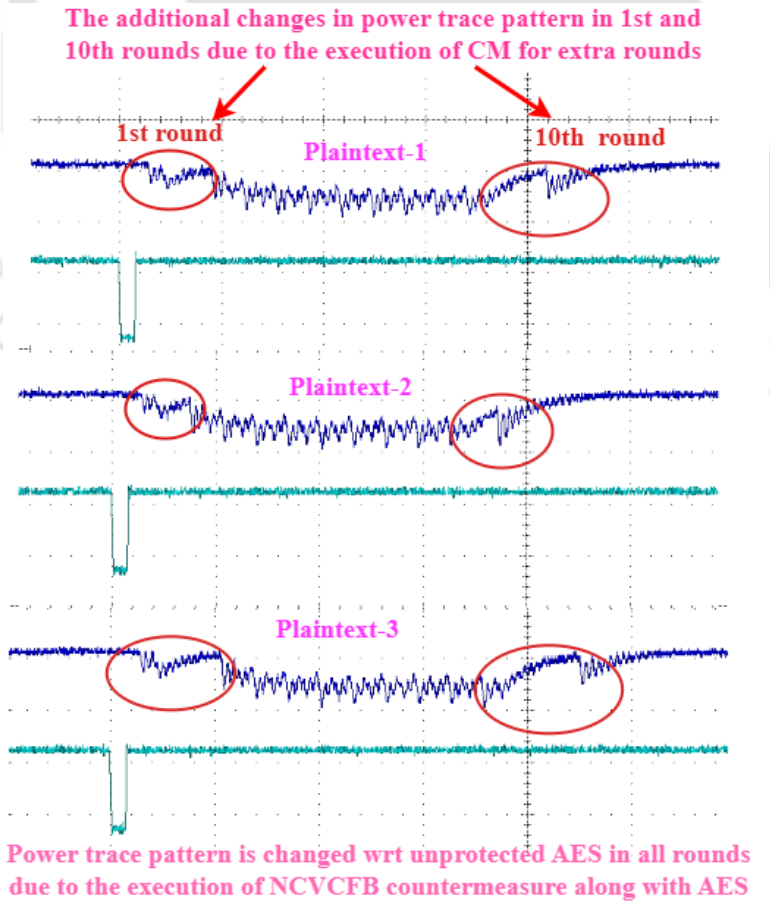
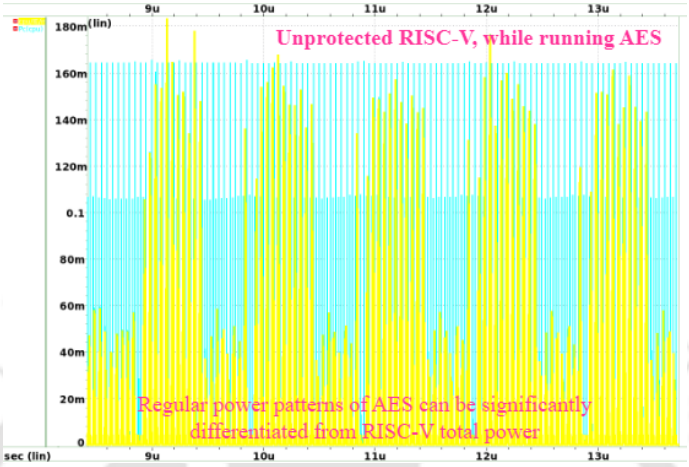


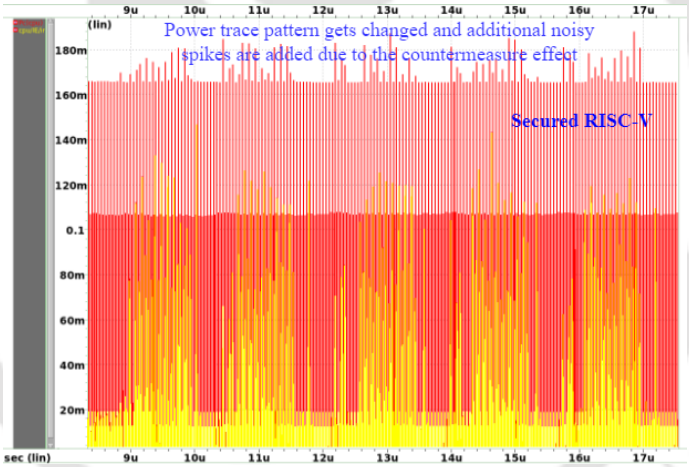
Figure 6.13: Power patterns of NCVCFB protected AES on SASEBO

6.4 Countermeasure Analysis

Initially, the vulnerability of the standalone NCVCFB countermeasure integrated AES, operating at 20 MHz, was tested on SASEBO and on ASIC platforms for 1 Million traces. Security metrics were evaluated by collecting 1,500 power samples (50 ns clock period × 30 clocks) per plaintext at a sampling rate of 1 GSa/s.



(a) Power traces of the Unprotected RISC-V against PAA



(b) Power traces of the NCVCFB Secured RISC-V against PAA

Figure 6.14: Unprotected vs NCVCFB Protected ASIC traces

Power trace patterns of NCVCFB-protected AES on SASEBO show significant amplitude variation compared to unprotected AES, due to the parallel operation of NCVCFB countermeasure circuit. The NS-Func in the modified Feistel bridge of the NCVCFB countermeasure circuit significantly alters the AES power pattern by remaining active throughout the S-box operation, achieving a comparable critical path delay (CPD) of 0.96 ns and power consumption of 114.83 nW/MHz to the AES S-box's 1.01 ns and 123.67 nW/MHz. Moreover, the NS-Func achieves this comparable delay and power with a smaller area of around 260.64 μm^2, compared to the S-box area of 315.36 μm^2. This results in increased switching activity within each clock cycle. Additionally, there is an amplitude surge during the first and last round of AES because of the extra NCVCFB countermeasure rounds, resulting in varying AES operating time, adding complexity to the power profile and enhancing security. While the unprotected design in Figure 6.12 has consistent patterns in all AES rounds, the pro-

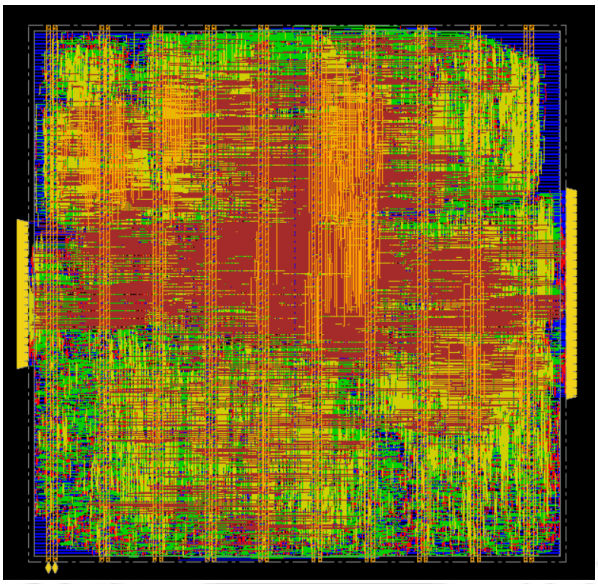


Figure 6.15: Layout of the NCVCFB secured RISC-V design

tected design in Figure 6.13 produces varying patterns and additional noise for each plaintext as well as each round of AES, demonstrating the countermeasure's effectiveness.

The security metrics, MTD, SNR, MI and TVLA of the NCVCFB countermeasure are subsequently evaluated for both the platforms to justify its resilience and the results are displayed in Figure. 6.16 and 6.17. In MTD plots, the correct key byte (red line) is indistinguishable from incorrect key guesses (black lines), with no distinct peak, indicating secure key protection and confirming unsuccessful key recovery attempts on both platforms. SNR results indicate a non-leaky design, with incorrect key guesses (blue circle) yielding higher correlations than the correct key (red circle), and SNR value around 0.5. The MI plots show no distinct peak for the correct key byte, with minimal MI values in milli range, reflecting low side-channel leakage. The t-values in TVLA plots Figure 6.19a and Figure 6.19b remain within the safe threshold limits of -4.5 to +4.5 for all samples, confirming that the design is non-leaky and highly resilient on both platforms.

Table 6.2: Performance metrics of NCVCFB countermeasure Protected AES

Design	Area (μm^2)	kGE+	Dyn. power* (µW/MHz)	Delay (ns)	Max. freq. (MHz)	No. of clks	Throughput (Gbps)
12-Clock AES (S-box Masoleh)	13503.48	9.38	4.07	1.81	552.49	12	5.89
22-Clock AES (S-box Masoleh)	12921.56	8.97	4.17	1.8	555.56	22	3.23
NCVCFB Countermeasure Protected AES	**13455.5**	**9.34**	**4.25**	**1.8**	**555.56**	**23-30**	**3.09 - 2.37**

* Dynamic power is normalized w.r.t clock frequency

Compared to the unprotected 22-clock AES design, the NCVCFB protected AES showed less than 5% overhead, with a 4.16% increase in area and a 2.03% increase in power due to the random operator RNG and modified Feistel bridge. The AES throughput varied between 4.35% and 26.7% due to dynamic clock cycles for each plaintext, offering strong security with minimal performance impact, as detailed in Table 6.2.

Subsequently, the NCVCFB countermeasure was incorporated into the unprotected RISC-V crypto core to create a secured RISC-V against PAA. Figure 6.15 shows the layout of the secured RISC-V designed created using Cadence Innovus, with post-layout power traces for five plaintexts of both unprotected and secured RISC-V crypto-cores illustrated in Figure 6.14. The power traces of secured RISC-V shown in Figure 6.14b, clearly demonstrates that

Table 6.3: Performance metrics of NCVCFB CM secured RISC-V with baseline core

Design	Area (μm^2)	kGE+	Dyn. power* (µW/MHz)	Delay (ns)	Max. freq. (MHz)	No. of clks	Throughput (Gbps)
RISC-V Cryptocore (RISC-V+ 12-Clock AES)	58059.72	40.31	22.2	3.71	269.54	20	1.73
RISC-V Cryptocore (RISC-V+ 22-Clock AES)	57404.63	39.86	22.86	3.71	269.54	30	1.15
NCVCFB Secured RISC-V (RISC-V+22-Clock AES+CM)	58006.88	40.28	23.14	3.71	269.54	31-38	1.113 - 0.908

* Dynamic power is normalized w.r.t clock frequency

the power pattern has been significantly altered compared to the baseline RISC-V crypto-core, shown in Figure 6.14a. Additionally, the AES power is completely concealed within the total power consumption due to the added noise power from the NCVCFB countermeasure, confirming the effectiveness of the protection. The secured RISC-V, running at 20MHz and requiring 38 clock cycles for encryption, 1,900 power samples (50ns clock period * 38 clock cycles) were gathered per plaintext, while sampling at 1 GSa/s for security evaluation.

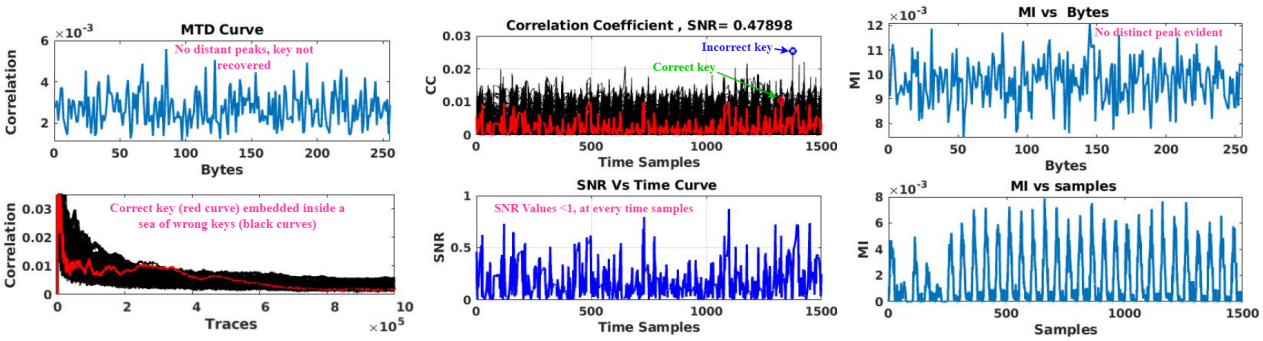
Table 6.4: Comparison of proposed countermeasure with state-of-the-art

Cores	Tech. Node	Countermeasure Type	AES Power @ frequency, Voltage	Overhead (%)			Analysis Method	MTD
				Area	Power	Perf.		
This work	**65nm**	**Proposed NCVCFB Countermeasure Secured RISC-V**	**462.83 µW @ 20 MHz, 1.08 V**	**1.05**	**1.23**	**3.23 - 21.05**	**CPA,MI, TVLA**	**> 2M**
This work	**65nm**	**Proposed NCVCFB Countermeasure Protected standalone AES**	**85.054 µW @ 20 MHz, 1.08 V**	**4.16**	**2.03**	**4.35 - 26.67**	**CPA,MI, TVLA**	**> 1M**
TCAS-II'22[101]	180nm	On-demand current equalizer	@ 50 MHz, 1.8 V	32.88	21.17	0	CPA	> 126K
JSSC'18[102]	130nm	Integrated Voltage regulator	10.5 mW @ 40 MHz, 0.9 V	-	5	3.33	CPA,TVLA	> 100K
CHESS'10 [103]	ATmega16	Software Random Delay	-	0	0	953 cc	CPA	> 150K
JSSC'10 [96]	130nm	Switched capacitor current equalizer	33.2 mW @ 100 MHz, 1.2 V	42	33	50	DPA	> 10M
TVLSI'18 [91]	65nm	Secure Double Rate registers	- @ 20 MHz, -	33	180	0	CPA,MI,TVLA	> 100K
JSSC'23 [104]	Intel 4	Multiplicative masking scheme	- @100 MHz, 0.5 V	65	-	4	TVLA,CPA.DNN	> 850M
JSSC'20[105]	14nm	Heterogeneous S-boxes with linear masked mixcolumns, and dual-rail key addition	11 mW @ 708 MHz, 0.75 V	28	23	0.7	CPA,TVLA	> 12M
VLSI'21[106]	65nm	Equalizer LDO with attack-detection scheme	0.0354 mW, @ 3.3 MHz, 0.45 V	51.58	1.84	0	CPA	> 10M
JSSC'20 [93]	130nm	Digital LDO regulator with SNI	10.9 mW @ 80 MHz, 0.84 V	36.9	32	10.4	CPA, TVLA,CEMA	> 6.8M
TVLSI'20 [107]	65nm	Randomization Technique	336 mW @ 800 MHz, 1.2	2.3	3.5	4	CPA,CNN, TVLA	> 2M

*Lesser overheads desirable

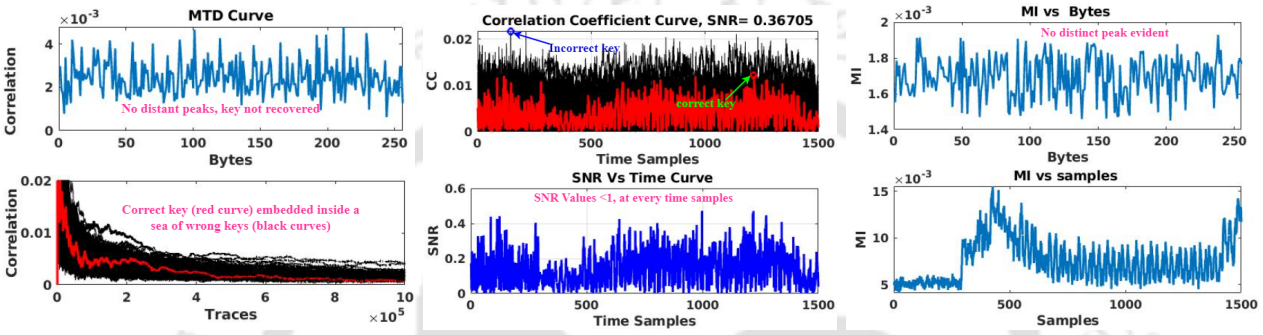
The MTD analysis over 2 Million traces depicted in Figure 6.18a, of the NCVCFB-secured RISC-V, shows no key recovery, as the red line is obscured among incorrect guesses (black curves). The t-values in the TVLA plots remained within the threshold limits of -4.5 to +4.5 for all samples. The NCVCFB-secured RISC-V showed lower SNR and MI values compared to the NCVCFB-integrated AES, indicating significantly higher security for the secured RISC-V crypto-core in Figure 6.18. The results showed that the device remained highly secure even with this large trace count. This increased security is due to the additional noise from NCVCFB countermeasure along with other RISC-V components, effectively concealing the AES power signature.

The various performance parameters of this newly designed secured RISC-V are evaluated against the baseline RISC-V crypto core in Table 6.3. Compared to the baseline RISC-V crypto-core, the NCVCFB-secured RISC-V incurs minimal overhead, with a 1.05% area increase and 1.23% power increase, without affecting maximum operational frequency. Though there is a slight performance degradation due to variable throughput ranging from 3.23% to 21.05%, caused by the varying number of clock cycles required for data encryption, this variability significantly enhances security, resulting in a highly secure RISC-V design against PAA.



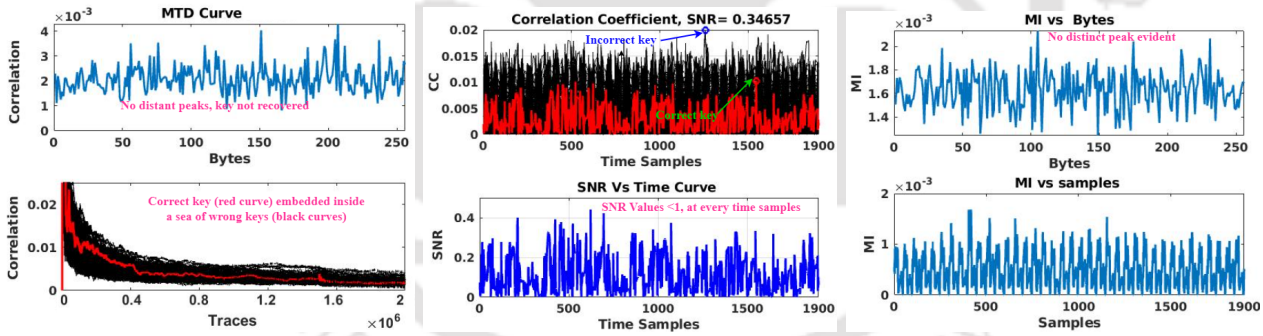
(a) MTD plot (b) SNR plot (c) MI plot

Figure 6.16: ASIC-based security metrics of NCVCFB countermeasure protected AES



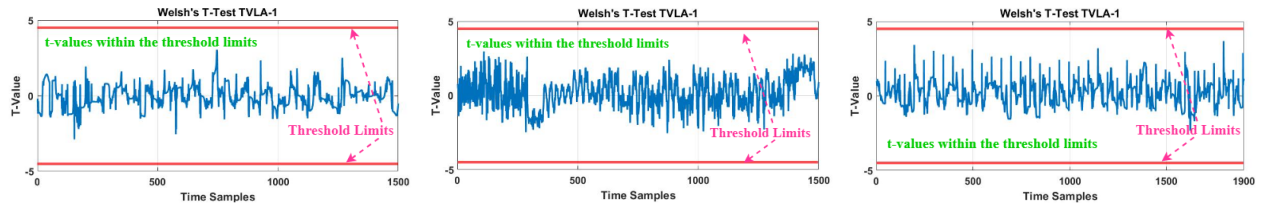
(a) MTD plot (b) SNR plot (c) MI plot

Figure 6.17: SASEBO-based security metrics of NCVCFB countermeasure protected AES



(a) MTD plot (b) SNR plot (c) MI plot

Figure 6.18: ASIC-based security metrics of NCVCFB secured RISC-V crypto-core



(a) Protected AES (ASIC) (b) Protected AES(SASEBO) (c) Secured RISC-V (ASIC)

Figure 6.19: TVLA plots of proposed NCVCFB countermeasure

Table 6.4 illustrates that the proposed NCVCFB countermeasure outperforms its state-of-the-art counterparts by achieving minimal area and power overheads, along with security testing that confirms the design's resilience against 2 Million traces. Thus, the developed countermeasure is both compact and effective, offering robust protection against attacks with exceptionally low area and power overheads.

6.5 Chapter summary

The initial focus of the chapter was to design a compact AES with significant throughput and improved intrinsic resilience against power attacks by developing a rolling AES architecture utilizing a compact Masoleh S-box and splitting the SubBytes operation over two clock cycles, using only eight S-boxes at a time. To further enhance the security, a novel NCVCFB countermeasure was developed, providing variation in both amplitude and temporal domains. The standalone NCVCFB countermeasure integrated AES was tested with 1 Million traces, with none of the bytes being recovered, indicating high security. Subsequently, the NCVCFB secured RISC-V was developed by incorporating the NCVCFB countermeasure integrated AES with the RISC-V architecture, achieving minimal area and power overheads of 1.05% and 1.23%, respectively, with no change in operating frequency. It exhibited a slight varying degradation in performance ranging from 3.23% to 21.05%, while providing high resilience to CPA, as evidenced by a MTD of over 2 Million, SNR below 0.5, MI in the milli range, and $|t|$ values less than 4.5 for TVLA. The newly designed NCVCFB countermeasure secured RISC-V processor, addresses the vulnerabilities against PAA and thereby enhancing the security in IoTe devices.

CHAPTER

7

SECURING AES INTEGRATED RISC-V WITH RANDOM CLOCK SELF COMPLEMENTARY COUNTERMEASURE

Contents

7.1 Introduction

The growth of IoT across industries, has made secure communication crucial, especially in resource-limited, power-constrained SoCs. In this chapter, a secured loosely coupled cryptographic RISC-V was designed using the Random Clock Self Complementary (RCSC) countermeasure to protect against PAA. RCSC introduces variability in amplitude and time domain by using random numbers for complementary operations and variable number of clock cycles for encrypting each data. The design is tested and power traces for PAA were generated on both ASIC and FPGA platforms.

Initially the RCSC countermeasure secured AES was integrated as an IP with 3-stage RISC-V(RV32I). The design achieves minimal power overhead of 0.407% and hardware costs of 7.33%, with strong resilience to PAA. Subsequently, the RCSC-secured AES design was integrated with 4-stage HP RISC-V(RV32I) and 4-stage HP RISC-V(RV32IM) cores to analyze its effects on area, power, and security metrics as processor complexity increased. It was observed that as the processor complexity increases, the area and power overheads associated with the RCSC countermeasure remain negligible. Additionally, with higher processor complexity, MTD increases and SNR decreases, indicating enhanced resilience of the designs towards PAA.

7.2 Novel Random Clock Self Complementary (RCSC) Countermeasure

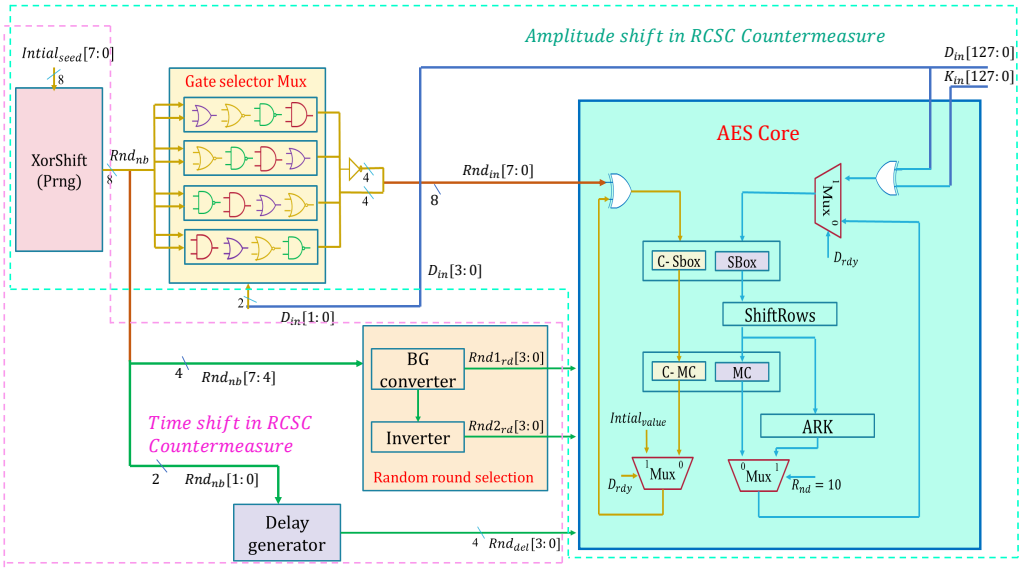


Figure 7.1: Complete Architecture of RCSC Countermeasure

The complete architecture of the proposed ultra-low power Random Clock Self Complementary Countermeasure (RCSC), shown in Figure 7.1, introduces both time and amplitude variations to resist PAA. The PAA using Hamming Distance (HD) model, analyses bit flips between the input and output of the S-box in the final round of AES. Resilience to this HD model is achieved by extending the AES S-box inputs and outputs to 16 bits and performing complementary operations for both the S-box and MixColumns. The least significant 8-bits holds AES data, while the most significant 8-bits holds random numbers generated by

Xorshift Pseudo Random Number Generator (PRNG). For a 128-bit AES, each 16 S-boxes, stores corresponding 8-bits of AES data in the LSB of the register, whereas the same random data is shifted and reused in the MSB of the register across different S-boxes to optimize area. The exact complementary operation of S-box is performed on the random data by Complementary S-box (CS-box) and a similar modification is applied to MixColumns (MC).

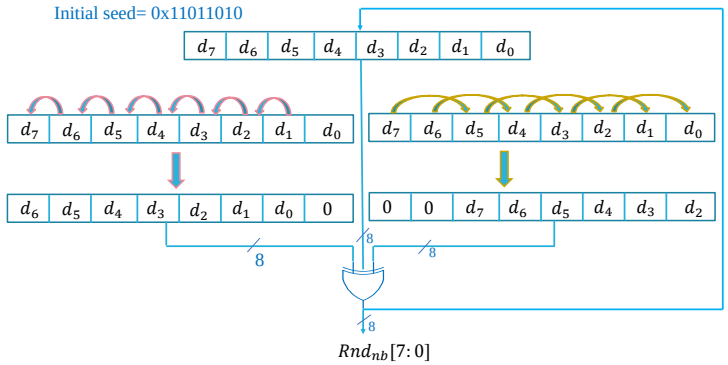


Figure 7.2: Xorshift Pseudo Random Number Generator

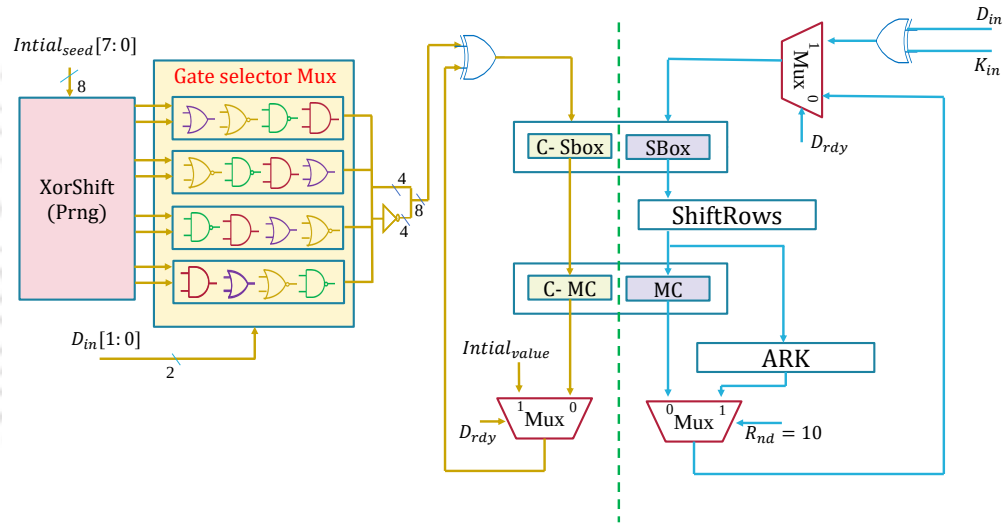


Figure 7.3: Amplitude shift in RCSC Countermeasure

The PRNG employed in this RCSC countermeasure is the Xorshift random number generator, depicted in Figure 7.2. The Xorshift PRNG developed by George Marsaglia in 2003, is highly efficient and fast, by using bitwise XOR and shift operations to rapidly produce random sequences. This speed and efficiency make it ideal for real-time security applications, offering a significant improvement over the older Linear Feedback Shift Register (LFSR) PRNG method, introduced in 1973. The Xorshift PRNG, efficiently generates an 8-bit random number by shifting the bits of an integer to left and with zeros padded on the right, and similarly shifting right with zeros added on the left, then performing XOR operations on the shifted values and actual number to generate a new random number.

XorShift PRNG generates 8-bit random numbers that are integrated directly into the AES data path. In order to provide variation in the amplitude domain, the XorShift PRNG's output is further processed through a gate selector multiplexer (MUX), that chooses between AND, OR, NOR, or NAND gates to produce a 4-bit value, which is combined with its complement to form an 8-bit random number. In the first AES round, this value is XORed

with the initial seed which is fed as random data to the Complementary S-box (CS-box) and further processed by the Complementary MixColumns (C-MC). In subsequent rounds, this output of the C-MC is XORed with the random data obtained by gate selector multiplexer (MUX) and the resultant random data is again fed back to CS-box. These dynamic operations introduce variations in power trace pattern along the amplitude axis. Figure 7.3 illustrates the architecture which introduce this variation in the amplitude domain.

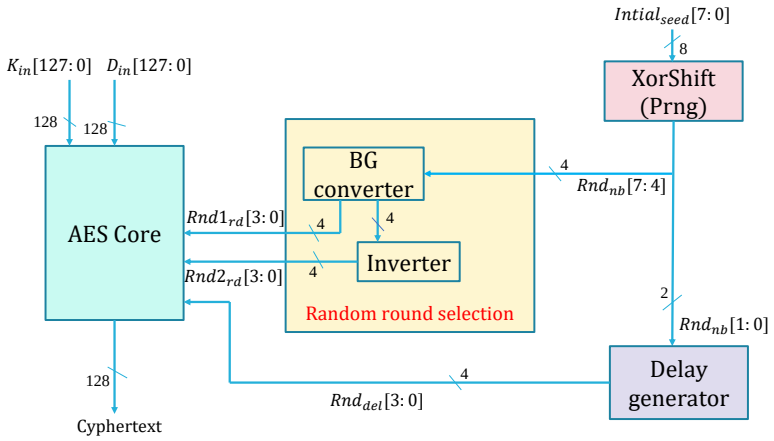


Figure 7.4: Time shift in RCSC Countermeasure

To introduce variation in the time axis as shown in Figure 7.4, random delays ranging from 1 to 4 cycles are inserted into two random rounds of AES. Thus, each plaintext is processed with a random number of clock cycles. The MSB 4-bits of Xorshift PRNG is given to a Binary to Grey code converter, which converts it into corresponding grey code. Depending on the obtained grey value and its complementary values, the delay is inserted in the corresponding two random rounds of AES from 1-10. The delay is inserted in the last round, if the value is greater than 9, and the delay is inserted in the first round if the value is equal to zero. The number of delays to be inserted in the chosen rounds of AES is determined by the LSB 2-bits of Xorshift PRNG. These random delays disrupt the alignment of 10^{th} round output of AES, making PAA even harder.

7.3 Random Clock Self Complementary (RCSC) Countermeasure Analysis

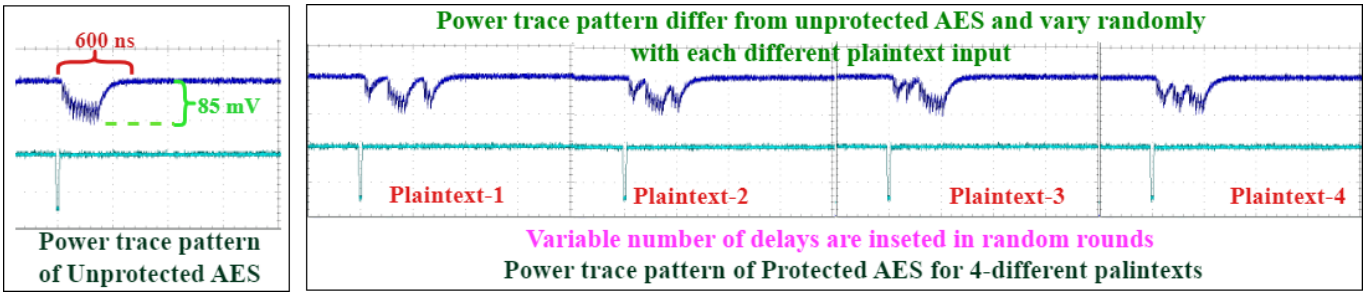


Figure 7.5: Power patterns of unprotected AES and RCSC protected AES on SASEBO

The hardware performance and security metrics of standalone AES protected with the Random Clock Self Complimentary countermeasure were analyzed on both FPGA (real environment) and ASIC (UMC 65nm technology). The RCSC protected AES was initially integrated with the 3stage RISC-V and later coupled with more complex RISC-V processors to assess the impact of increased complexity on these metrics.

Figure 7.5, shows the power traces of both unprotected and RCSC countermeasure-protected designs, captured using the Side-channel Attack Security Evaluation Board (SASEBO), a specialized FPGA for SCA. The unprotected design exhibits a consistent power consumption pattern across all AES rounds, while the RCSC-protected design displays varying power traces for each plaintext, indicating the effectiveness of the countermeasure in mitigating SCA. A similar trend is observed in ASIC power trace patterns obtained after post-layout simulations, as depicted in Figure 7.6.

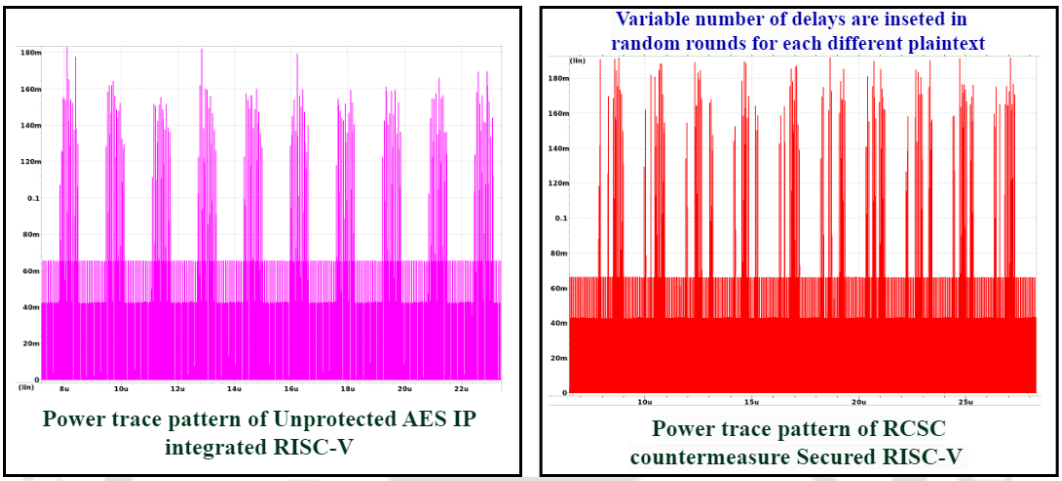


Figure 7.6: Unprotected Vs RCSC Protected ASIC Power trace patterns

The layout of RCSC countermeasure secured RISC-V, depicted in Figure 7.7, was generated using Cadence Innovus. Post-layout simulations were performed using the netlist obtained from this layout, and the corresponding power traces obtained from Synopsys PrimeTime PX were analyzed to evaluate the hardware security metrics.

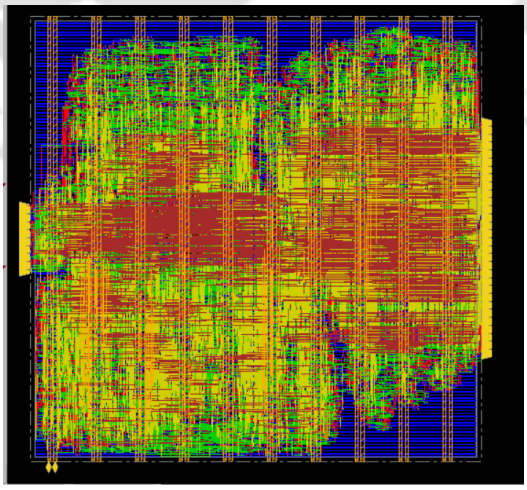
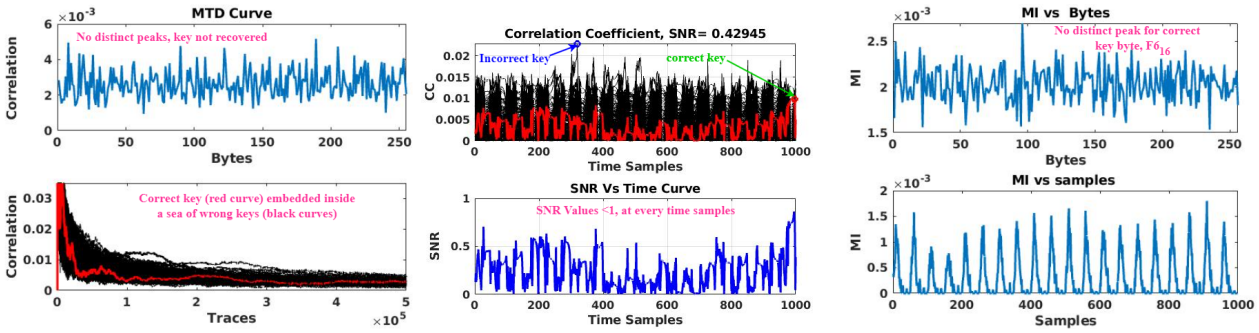


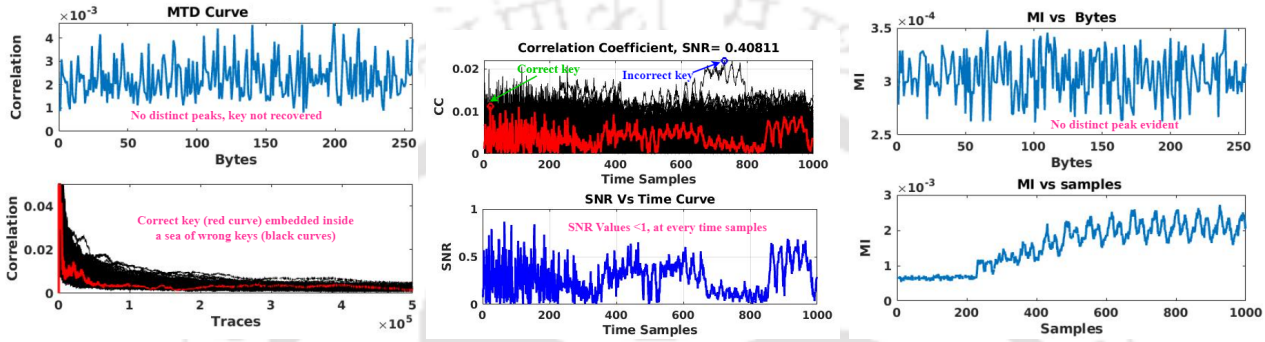
Figure 7.7: Layout of the RCSC secured RISC-V Core

Plots 7.8 and 7.9 illustrate the hardware security metrics: MTD, SNR, and MI of the RCSC countermeasure protected AES, evaluated over 500K traces on both ASIC and



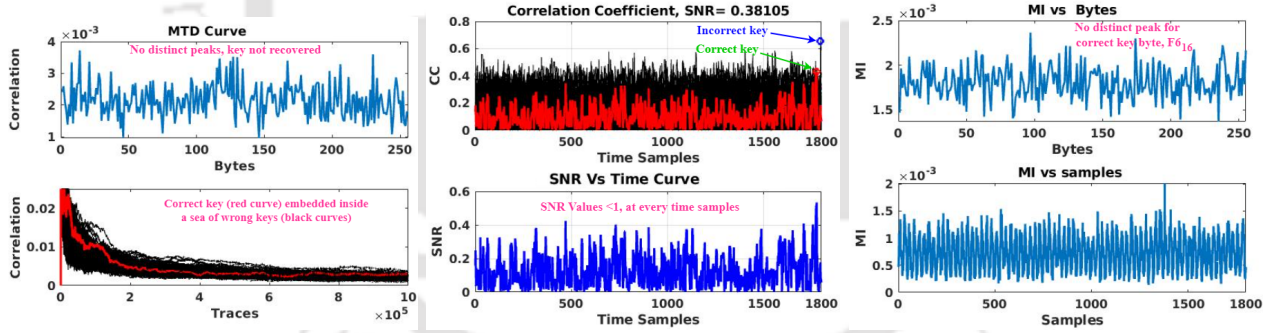
(a) MTD plot (b) SNR plot (c) MI plot

Figure 7.8: ASIC-based security metrics of RCSC countermeasure protected AES



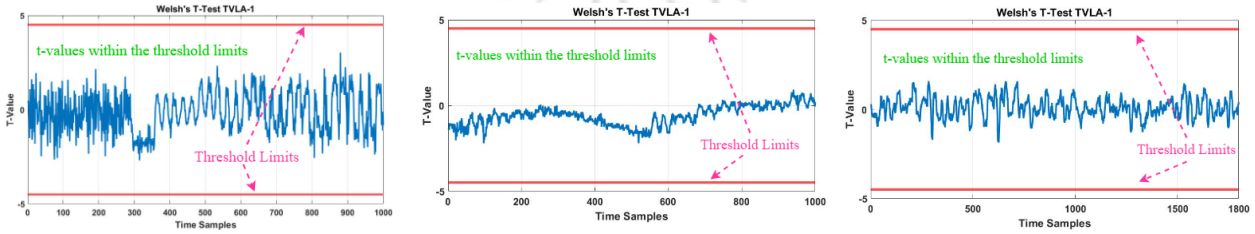
(a) MTD plot (b) SNR plot (c) MI plot

Figure 7.9: SASEBO-based security metrics of RCSC countermeasure protected AES



(a) MTD plot (b) SNR plot (c) MI plot

Figure 7.10: ASIC-based security metrics of RCSC Countermeasure secured RISC-V



(a) Protected AES (ASIC) (b) Protected AES (SASEBO) (c) Secured RISC-V (ASIC)

Figure 7.11: TVLA plots of proposed RCSC countermeasure

SASEBO platforms. The metrics confirm that the RCSC protected AES design is secure and non-leaky, with MTD > 500K, SNR < 1, MI in the milli range, and TVLA within ±4.5. The AES with the RCSC countermeasure is integrated as an IP with the 3-stage RISC-V processor. Plot 7.10 displays the MTD, SNR, and MI of the RCSC-secured RISC-V, evaluated over 1 million traces. The security metric plots indicate that integration with the RISC-V processor further improves MTD, SNR, MI, and TVLA values, enhancing the design's security, adaptability, and effectiveness, making it a robust solution for protecting sensitive data in IoTe devices.

The MTD plots for protected designs show that the correct key byte (red line) is indistinguishable from incorrect guesses (black lines), indicating unsuccessful key recovery attempts on both platforms. SNR values around 0.5 (below 1) further confirm the non-leaky nature of the design. MI plots in the milli range, without distinct peaks for the correct key byte, suggest minimal side-channel leakage. The TVLA results in Figure 7.11 remain within ±4.5 across all samples, reinforcing that the designs are non-leaky.

Table 7.1: Security metrics of the RCSC secured AES integrated 3-stage RISC-V

Designs	No:of Bytes Recovered/ Last Recovered Byte	MTD	SNR	MI	TVLA
AES-IP Integrated RISC-V	16/Byte1	2025	1.295	4.37*10ˆ-3	91.10
RCSC Secured AES-IP integrated 3-stage RISC-V (RV32I)	Nil	>1 M	0.381	1.98*10ˆ-3	<+/-4.5

Table 7.1 demonstrates significant improvements in hardware security metrics of RCSC countermeasure secured RISC-V compared to its baseline unprotected AES-integrated RISC-V. The RCSC-secured RISC-V did not reveal any key bytes even after 1 million traces, achieving an SNR around 0.5 and TVLA within ±4.5. In contrast, the unprotected AES-integrated RISC-V exposed all key bytes within 2025 traces, with an SNR above 1 and much higher TVLA values. As shown in Table 7.1, the proposed design achieved a notable increase in MTD, a 70.5% reduction in SNR, a 54.6% decrease in MI, and a 95% reduction in TVLA. This highlights the effectiveness of countermeasure.

Table 7.2: Performance metrics of the RCSC secured AES integrated 3-stage RISC-V

Design	Area (μm^2)	kGE+	Dyn. power* (µW/MHz)	Delay (ns)	Max. freq. (MHz)	No. of clks	Throughput (Gbps)
AES-IP Integrated 3-stage RISC-V (RV32I)	43205.04	29.82	15.482	4.54	220.41	28	1.0076
RCSC Secured AES-IP integrated 3-stage RISC-V (RV32I)	46373.76	32.01	15.545	4.54	220.41	30-36	0.940-0.783

Table 7.2 compares the performance metrics of the RISC-V secured with the RCSC countermeasure to the baseline unprotected AES-integrated RISC-V. The addition of the RCSC countermeasure results in an increase in area, with only a minimal increase in dynamic power due to the low number of registers and flip-flops used, which helps to reduce dynamic power consumption. The operating frequency remains unchanged with the inclusion of the countermeasure, while throughput varies due to the random clock cycles required for data encryption.

The overheads of the newly designed secured RCSC countermeasure RISC-V against other leading protected designs are shown in Table 7.3. The proposed design achieved an MTD > 1M with an area overhead of 7.33% and a negligible power overhead of 0.407%. The increase in dynamic power is negligible, as we use only 8-bit PRNG and requires only 8-bit registers. In contrast, other designs in Table 7.3, improved its security at the cost of considerable area

Table 7.3: Comparison of the proposed RCSC countermeasure with state-of-the-art

Cores	Tech. Node	Countermeasure Type	Power @ frequency, Voltage	Overhead (%)			Analysis Method	MTD
				Area	Power	Perf.		
This work	**65nm**	**Proposed RCSC Countermeasure Secured RISC-V SoC**	**@ 20 MHz, 1.08 V**	**7.33**	**0.407**	**6.67 - 22.22**	**CPA,MI, TVLA**	**> 1M**
TCAS-II'22[101]	180nm	On-demand current equalizer	@ 50 MHz, 1.8 V	32.88	21.17	0	CPA	> 126K
CHESS'10 [103]	ATmega16	Software Random Delay	-	0	0	953 cc	CPA	> 150K
JSSC'10 [96]	130nm	Switched capacitor current equalizer	33.2 mW @ 100 MHz, 1.2 V	42	33	50	DPA	> 10M
TVLSI'18 [91]	65nm	Secure Double Rate registers	- @ 20 MHz, -	33	180	0	CPA,MI,TVLA	> 100K

*Lesser overheads desirable

and power overheads. The proposed RCSC countermeasure is highly suitable for battery driven IoT applications, due to the enhanced security with minimal power requirement.

Table 7.4: Performance of RCSC secured AES integrated 4-stage HP RISC-V(RV32I)

Design	Area (μm^2)	kGE+	Dyn. power* (µW/MHz)	Delay (ns)	Max. freq. (MHz)	No. of clks	Throughput (Gbps)
AES-IP Integrated 4-stage HP RISC-V (RV32I)	59346.63	41.21	19.813	3.75	266.67	28	1.219
RCSC Secured AES-IP integrated 4-stage HP RISC-V (RV32I)	61410.96	42.65	19.819	3.75	266.67	30-36	1.138 - 0.948

The same RCSC protected AES is integrated with both 4-stage HP RISC-V based on RV32I and a 4-stage HP RISC-V based on RV32IM. The hardware performance metrics for both designs are presented in Table 7.4 and Table 7.5.

Table 7.5: Performance of RCSC secured AES integrated 4-stage HP RISC-V(RV32IM)

Design	Area (μm^2)	kGE+	Dyn. power* (µW/MHz)	Delay (ns)	Max. freq. (MHz)	No. of clks	Throughput (Gbps)
AES-IP Integrated 4-stage HP RISC-V (RV32IM)	79484.40	55.20	24.731	3.76	265.96	32	1.0638
RCSC Secured AES-IP integrated 4-stage HP RISC-V (RV32IM)	81776.41	56.79	24.736	3.76	265.96	34 - 40	1.0013 - 0.851

The impact on area and power overhead due to the RCSC countermeasure is illustrated in Figures 7.12 and 7.13. These figures clearly show that as the processor complexity increases, the relative area and power overhead from the RCSC countermeasure decreases.

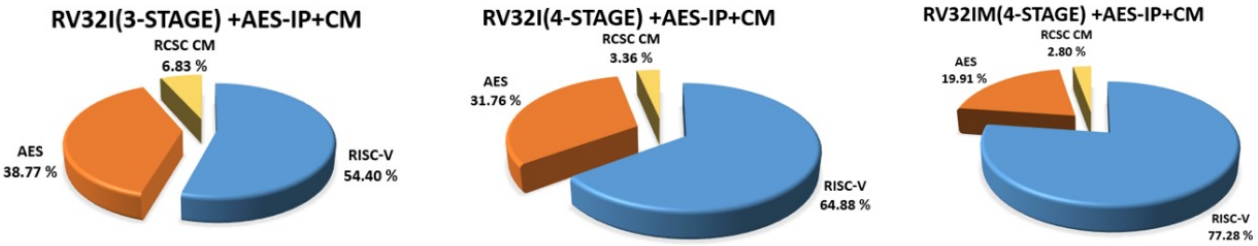


Figure 7.12: Area overheads while integrating RCSC countermeasure with various RISC-V

Figures 7.14a and 7.14b illustrate improvements in MTD and SNR metrics when the RCSC countermeasure is integrated with a 3-stage RISC-V, a 4-stage HP RISC-V (RV32I), and a 4-stage HP RISC-V (RV32IM). These results indicate that as processor complexity increases, intrinsic resilience to power analysis attacks also improves, as more modules are active during AES operation. Consequently, even a small countermeasure can provide high security against power attacks.

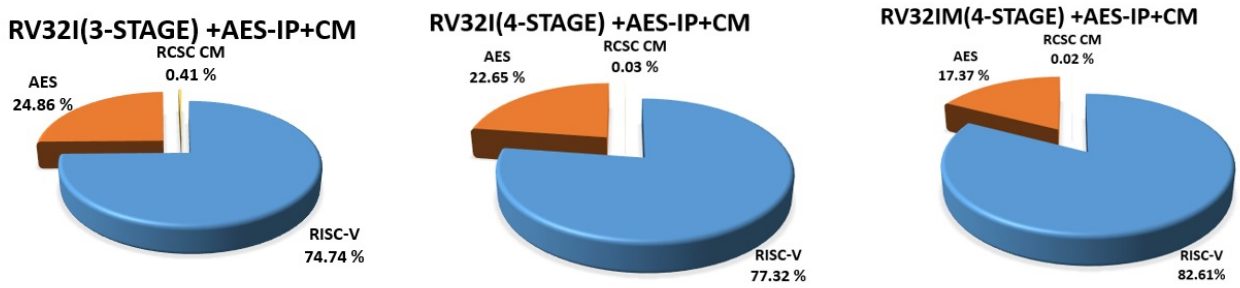
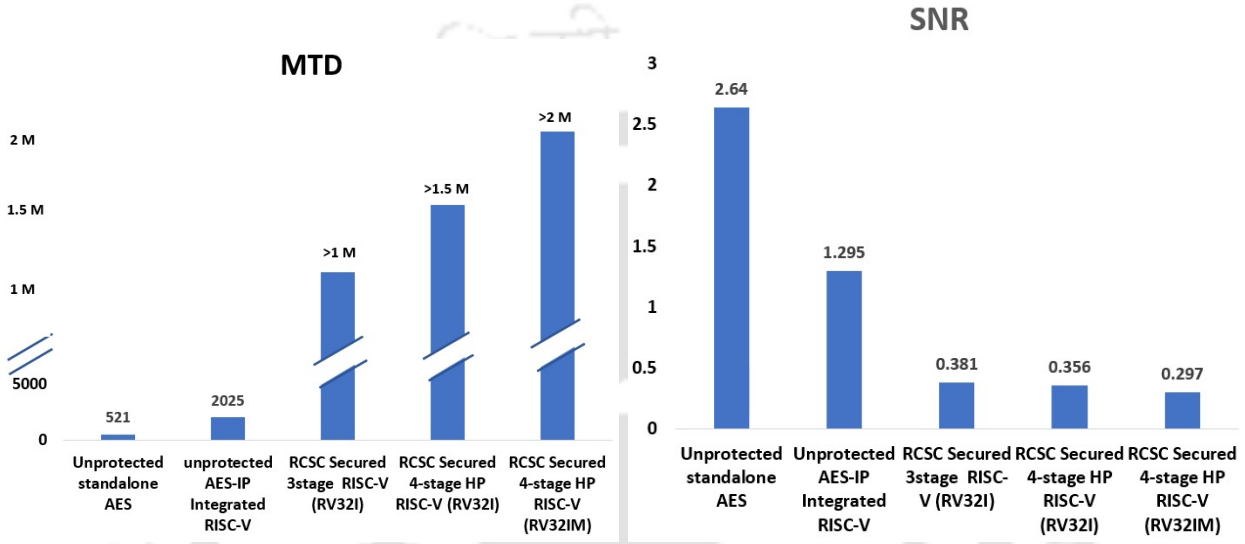


Figure 7.13: Power overheads while integrating RCSC countermeasure with various RISC-V



(a) MTD of RCSC Countermeasure across cores (b) SNR of RCSC Countermeasure across cores

Figure 7.14: Security metrics while integrating RCSC countermeasure with various RISC-V

7.4 Chapter summary

This chapter proposed a novel RCSC countermeasure that enhances security by introducing variability in both amplitude and time domains. It achieves this by performing complementary S-box and MixColumns operations on random numbers and by adding random delays in two random rounds of AES. The standalone AES protected by the RCSC countermeasure was tested with 500K traces, successfully preventing any key byte recovery. When integrated with a 3-stage RISC-V processor and tested with 1 million traces, the RCSC-protected AES again showed no key byte recovery, with higher MTD. The ultra-low power RCSC countermeasure secured RISC-V, with minimal power overhead of 0.407%, is best suitable for battery-powered IoT applications. Additionally, integrating the RCSC-protected AES with more complex processors demonstrated even greater resilience against power analysis attacks.

CHAPTER

8

CONCLUSION AND FUTURE DIRECTIONS

8.1 Summary of Contributions

This thesis successfully addressed the critical challenges of securing embedded processors in IoTe devices against power attacks. The research focused on developing highly secured RISC-V architectures by first enhancing the intrinsic resilience of RISC-V crypto-cores through optimized integration techniques, followed by the design of compact and efficient countermeasures. The proposed designs provided strong protection against PAA with minimal area and power overhead, making it well-suited for high-security IoTe devices.

With RISC-V being the heart of IoTe devices, the initial focus was on selecting a RISC-V core that strikes the right balance between compact size, low power consumption, and high performance. This was achieved by optimizing the pipeline stages and implementing efficient pipeline techniques. After thorough evaluation, the HP 4stage RISC-V core emerged as the optimal choice.

The next step of thesis explored various AES integration techniques with RISC-V micro-architectures, focusing on tightly coupled and loosely coupled approaches. The loosely coupled design provides flexibility, making it a viable option for multi-core IoT applications. Meanwhile, the tightly coupled approach offers superior security and efficiency, making it ideal for high-security, high-performance applications. However, it involves greater efforts for integration due to the need for ISA extensions and compiler modifications.

The subsequent step involved more detailed way of developing tightly coupled RISC-V crypto-cores by integrating a dedicated AES hardware unit into the RISC-V pipeline and leveraging custom AES instructions for efficient operation. Two variations were implemented: one using software pause (SW) technique and the other employing hardware pause (HW) technique. This integration significantly improved efficiency, achieving AES encryption in just 20 and 23 clock cycles compared to thousands of clock cycles in baseline cores. These results demonstrate an improvement in AES throughput and intrinsic resilience while maintaining resource efficiency.

A key contribution of this work is the development of novel countermeasures to address side-channel leakage effectively. The Non-Linear Cyclic Variable Clock Feistel Bridge-Inspired Countermeasure (NCVCFB), is based on modified rolling fiestel bridge which operates in parallel with AES in each round. Since power attack happens in the first and last round of AES, the modified Feistel bridge operates for additional random rounds in the first and last round of AES corrupting the meaningful samples of AES, acquired by the attacker. Due to these techniques, the NCVCFB countermeasure enhance the security drastically. The ultra-low-power Random Clock Self Complementary (RCSC) countermeasure enhanced security by introducing variability in both amplitude and time domains. It was achieved by performing complementary S-box and MixColumn operations on random data and inserting random delays into two randomly selected AES rounds during encryption. The complimentary operations and the selective insertion of delays improved its security while minimizing the impact on throughput, as delays were not applied to all AES rounds.

The NCVCFB countermeasure secured RISC-V core achieved remarkable low area and power overheads of only 1.05% and 1.23%, respectively, with resilience to PAA using 2 million plaintexts. Its compact design and minimal impact on throughput make it ideal for high-speed IoT applications. Similarly, the RCSC countermeasure incurred a power overhead of just 0.407% and a hardware cost increase of 7.33%, providing strong resilience against PAA with 1 million plaintexts, making it highly suitable for ultra-low-power IoT environments.

This research establishes a robust foundation for future work in secure processor design. Directions for further exploration include integrating advanced memory systems, developing scalable multicore architectures, and extending cryptographic support to emerging standards such as Post-Quantum Cryptography, Ascon etc.

8.2 Future Directions

This research primarily focuses on optimizing and securing the core micro-architecture of embedded processors against Power attacks. However, several directions can extend the work for enhanced performance, security, and broader applicability:

1. **Development of Scalable Multi-Core Architectures:** Future research can explore the design of multi-core System-on-Chip (SoC) architectures that integrate secure RISC-V cores alongside cores optimized for tasks like AI/ML and other computational applications. Leveraging heterogeneous core integration can improve performance while maintaining robust security, offering flexibility for a wide range of IoT applications.

2. **Advanced Memory Systems:** Integrating advanced memory architectures, such as multi-level caches and DDR memory with security features, can enable efficient handling of larger programs and improve resilience against memory-based side-channel attacks. Secure data storage and access protocols can be developed to safeguard memory subsystems, further strengthening the security of IoT devices.

3. **Peripheral and Interface Integration:** The inclusion of peripherals like VGA, HDMI, and USB can enhance connectivity and real-time data exchange with external devices. Such additions will improve the usability and interaction of IoT devices in practical applications, facilitating seamless communication and improved user experience.

4. **Enhanced Security Features:** Future work can focus on unified countermeasures capable of addressing multiple attack models, including power side-channel attacks, timing attacks, and fault injections, within a single framework. Additionally, runtime obfuscation techniques, such as dynamic key masking and clock jittering, can be implemented to obscure power signatures effectively. Incorporating emerging cryptographic standards, such as Post-Quantum Cryptography (PQC) and lightweight ciphers like Ascon, can provide an alternative or complement to AES, ensuring robust security tailored to IoT requirements.

By addressing these areas, future research can significantly advance the development of secure, high-performance processors while paving the way for innovations in hardware security for IoT and other critical applications.

Table 8.1: Summary of the research

	Results					Impact
Selection of RISC-V Core	**RISC-V Cores**	**2stage**	**3stage**	**4stage**	**HP 4stage**	Selected: 1) HP 4-stage RISC-V (Maximum Freq:298.51 & Dhrystone:1.68) 2) 3-stage RISC-V (Highest Dhrystone:1.86)
	Area [µm²]	23483.88	25227.72	28779.4	39842.72	
	Dynamic Power [µW/MHz]	9.993	11.618	14.358	15.324	
	Max. Freq [MHz]	193.05	239.81	251.89	298.51	
	Dhrystone [DMIPS/MHz]	1.81	1.86	1.34	1.68	
RISC-V Crypto-core based on (SW pause or HW pause)	**RISC-V Cores**	**3-stage**	**3-stage RISC-V Crypto-core (SW)**	**HP 4stage**	**4-stage HP RISC-V Crypto-core (HW)**	1) RISC-V Crypto-core (SW): AES TP: 1.25Gbps, MTD:4711, SNR: 1.15 2) HP RISC-V Crypto-core (HW): AES TP: 1.73 Gbps, MTD:15658, SNR: 1.02 selected: 4-stage RISC-V Crypto-core (HW) due to higher throughput and MTD
	Area [µm²]	25227.72	41522.76	39842.72	58059.72	
	Dynamic Power [µW/MHz]	11.62	17.08	15.32	22.2	
	Max. Freq [MHz]	239.81	223.71	298.51	269.54	
	No: of Clocks	6953	23	10914	20	
	Throughput[TP] [Gbps]	0.0044	1.25	0.0035	1.73	
AES Integration techniques with RISC-V (loosely coupled or tightly coupled)	**Integration Techinques**	**Tightly Coupled (3-stage RISC-V)**	**Loosely Coupled (3-stage RISC-V)**	**Tightly Coupled (4-stage HP RISC-V)**	**Loosely Coupled (4-stage HP RISC-V)**	1)3stage RISC-V Tightly coupled TP:1.25Gbps, MTD:4711, SNR:1.15 2)3stage RISC-V Loosely coupled: TP:1.01Gbps, MTD:2025, SNR:1.295 3)4stage HP RISC-V Tightly coupled TP:1.73Gbps, MTD:15658, SNR:1.023 b)4stage HP RISC-V Loosely coupled TP:1.22Gbps, MTD:7602, SNR:1.112 Selected: loosely coupled-flexibility.
	Area [µm²]	41522.76	43205.04	58059.72	59346.63	
	Dynamic Power [µW/MHz]	17.08	15.48	22.2	19.81	
	Max. Freq [MHz]	223.71	220.41	269.54	266.67	
	No: of Clocks	23	28	20	28	
	Throughput[TP] [Gbps]	1.25	1.01	1.73	1.22	
Non-Linear Cyclic Variable Clock Feistel Bridge-inspired (NCVCFB) Countermeasure	**NCVCFB**	**4-stage HP RISC-V Crypto-core**		**NCVCFB secured RISC-V Crypto-core**		Area overhead and Power overheads <5%, MTD>2 million traces, SNR=0.346.
	Area [µm²]	58059.72		58006.88		
	Dynamic Power [µW/MHz]	22.2		23.14		
	Max. Freq [MHz]	269.54		269.54		
	No: of Clocks	20		31-38		
	Throughput[TP] [Gbps]	1.73		1.113 - 0.908		
Random Clock Self Complementary (RCSC) Countermeasure	**RCSC**	**Loosely Coupled (3-stage RISC-V)**		**RCSC secured RISC-V**		Area overhead <8%, Power overhead <1% , MTD>1 million traces, SNR=0.381
	Area [µm²]	43205.04		46373.76		
	Dynamic Power [µW/MHz]	15.48		15.545		
	Max. Freq [MHz]	220.41		220.41		
	No: of Clocks	28		30-36		
	Throughput[TP] [Gbps]	1.01		0.940-0.783		

PUBLICATIONS

Published

1. **T. M. Ignatius** and R. P. Palathinkal, "Non-Linear Cyclic Variable Clock Feistel Bridge-Inspired Countermeasure for Securing RISC-V Crypto-Core Against Power Attacks," in *IEEE Transactions on Circuits and Systems I: Regular Papers*, doi: 10.1109/TCSI.2025.3536284.

2. **T. M. Ignatius** and R. P. Palathinkal, "Securing RISC-V SoC With Random Clock Self Complementary Countermeasure," in *IEEE Embedded Systems Letters,* doi: 10.1109/LES.2025.3567317.

3. **T. M. Ignatius**, T. Birjit Singha and R. Paily Palathinkal, "Power Side-Channel Attacks on Crypto-Core Based on RISC-V ISA for High-Security Applications," in *IEEE Access*, vol. 12, pp. 150230-150248, 2024, doi: 10.1109/ACCESS.2024.3477961.

4. **T. M. Ignatius**, S. Bora and R. P. Palathinkal, "Impact of Pipelining on Low Power IoT applicable RISC-V ISA Core Micro-architectures," 2024 IEEE 17th International Symposium on Embedded Multicore/Many-core Systems-on-Chip (MCSoC), Kuala Lumpur, Malaysia, 2024, pp. 223-230, doi: 10.1109/MCSoC64144.2024.00045.

Accepted

1. **Accepted Paper**, titled "Securing RISC-V with Randomized Clock countermeasure against PAA" *for presentation at* ***SRC TECHCON 2025****, to be held* ***September 7–11, 2025****, at the Renaissance Hotel in Austin, Texas..*

Submitted

1. **Titu Mary Ignatius**, Thockchom Birjit Singha and Roy Paily Palathinkal, "Integration of AES and RISC-V Hardwares and Securing it by Xorshift PRNG-based Gray Code Randomized Clock Countermeasure" in *IEEE Transactions on Dependable and Secure Computing*

BIBLIOGRAPHY

[1] A. Waterman, Y. Lee, D. A. Patterson, and K. Asanović, "The RISC-V instruction set manual, volume i: User-level isa, version 2.0," tech. rep., EECS Department, University of California, Berkeley, May 2014.

[2] V. Rijmen and J. Daemen, "Advanced Encryption Standard," *Proceedings of Federal Information Processing Standards Publications, National Institute of Standards and Technology*, pp. 19–22, 2001.

[3] P. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Advances in Cryptology CRYPTO' 99*, Springer Berlin Heidelberg, 1999.

[4] Y. Lee, "Rocket chip SoC based on RISC-V generator in chisel." `https://www2.eecs.berkeley.edu/Pubs/TechRpts/2016/EECS-2016-17.pdf`.

[5] "ARM CORTEX A-5 Technical reference manual, tech. rep." `https://developer.arm.com/Processors/Cortex-A5`.

[6] A. M. Yunsup Lee, Albert Ou, "Z-Scale: Tiny 32-bit RISC-V Systems." `https://riscv.org/wp-content/uploads/2015/06/riscv-zscale-workshop-june2015.pdf`.

[7] D. Rossi, F. Conti, A. Marongiu, A. Pullini, I. Loi, M. Gautschi, G. Tagliavini, A. Capotondi, P. Flatresse, and L. Benini, "PULP: A Parallel ultra low power platform for next generation IoT applications," in *2015 IEEE Hot Chips 27 Symposium (HCS)*, pp. 1–39, 2015.

[8] S. Gupta, N. Gala, G. S. Madhusudan, and V. Kamakoti, "Shakti-f: A fault tolerant microprocessor architecture," in *2015 IEEE 24th Asian Test Symposium (ATS)*, pp. 163–168, 2015.

[9] "Vexriscv based on RISC-V ISA." https://github.com/SpinalHDL/VexRiscv.

[10] C. Wolf, "Picorv32 based on RISC-V ISA." `https://github.com/cliffordwolf/picorv32`.

[11] M. G. Andreas Traber and P. D. Schiavone, "RI5CY: User manual." `https://www.pulp-platform.org/docs/ri5cy_user_manual.pdf`.

[12] P. Davide Schiavone, F. Conti, D. Rossi, M. Gautschi, A. Pullini, E. Flamand, and L. Benini, "Slow and steady wins the race? a comparison of ultra-low-power RISC-V Cores for internet-of-things applications," in *2017 27th International Symposium on Power and Timing Modeling, Optimization and Simulation (PATMOS)*, pp. 1–8, 2017.

[13] Z. Li, Y. Huang, L. Tian, R. Zhu, S. Xiao, and Z. Yu, "A low-power asynchronous RISC-V Processor with propagated timing constraints method," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 68, no. 9, pp. 3153–3157, 2021.

[14] A. E. Cohen and K. K. Parhi, "Architecture optimizations for the RSA public key cryptosystem: A tutorial," *IEEE Circuits and Systems Magazine*, vol. 11, no. 4, pp. 24–34, 2011.

[15] C.-H. Lin, Y.-S. Yeh, S.-P. Chien, C.-Y. Lee, and H.-S. Chien, "Generalized secure hash algorithm: SHA-X," in *2011 IEEE EUROCON - International Conference on Computer as a Tool*, pp. 1–4, 2011.

[16] A. U. Ay, E. Öztürk, F. R. Henríquez, and E. Savaş, "Design and implementation of a constant-time FPGA accelerator for Fast Elliptic Curve Cryptography," in *2016 International Conference on ReConFigurable Computing and FPGAs (ReConFig)*, pp. 1–8, 2016.

[17] B. Singha, R. P. Palathinkal, and S. R. Ahamed, "Implementation of AES using composite field arithmetic for IoT Applications," in *2020 Third ISEA Conference on Security and Privacy (ISEA-ISAP)*, pp. 115–121, 2020.

[18] T. B. Singha, R. P. Palathinkal, and S. R. Ahamed, "Securing AES Designs Against Power Analysis Attacks: A Survey," *IEEE Internet of Things Journal*, vol. 10, no. 16, pp. 14332–14356, 2023.

[19] X. Gao, E. Lu, L. Li, and K. Lang, "LUT-based FPGA Implementation of SMS4/AES/Camellia," in *Fifth IEEE International Symposium on Embedded Computing*, pp. 73–76, 2008.

[20] X. Luo, Y. Qi, Y. Wan, Q. Wang, and H. Zhang, "A Fast AES Encryption method based on single LUT for industrial wireless network," in *2014 International Conference on Identification, Information and Knowledge in the Internet of Things*, pp. 158–161, 2014.

[21] A. Reyhani-Masoleh, M. M. I. Taha, and D. Ashmawy, "New area record for the AES Combined S-Box/inverse S-Box," *2018 IEEE 25th Symposium on Computer Arithmetic (ARITH)*, pp. 145–152, 2018.

[22] D. A. Osvik, J. W. Bos, D. Stefan, and D. Canright, "Fast software AES Encryption," pp. 75–93, Springer Berlin Heidelberg, 2010.

[23] P. Schwabe and K. Stoffelen, "All the AES You need on Cortex-M3 and M4," in *Selected Areas in Cryptography – SAC 2016* (R. Avanzi and H. Heys, eds.), pp. 180–194, Springer International Publishing, 2017.

[24] W. Zhao, Y. Ha, and M. Alioto, "AES architectures for minimum-energy operation and silicon demonstration in 65nm with lowest energy per encryption," in *2015 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 2349–2352, 2015.

[25] R. Ueno, S. Morioka, N. Miura, K. Matsuda, M. Nagata, S. Bhasin, Y. Mathieu, T. Graba, J.-L. Danger, and N. Homma, "High throughput/gate AES hardware architectures based on datapath compression," *IEEE Transactions on Computers*, vol. 69, no. 4, pp. 534–548, 2020.

[26] P. N. Khose and V. G. Raut, "Implementation of AES algorithm on FPGA for low area consumption," in *2015 International Conference on Pervasive Computing (ICPC)*, pp. 1–4, 2015.

[27] M.-H. Dao, V.-P. Hoang, V.-L. Dao, and X.-T. Tran, "An energy efficient AES Encryption core for hardware security implementation in IoT systems," in *2018 International Conference on Advanced Technologies for Communications (ATC)*, pp. 301–304, 2018.

[28] A. Jain, D. Jain, A. Katiyar, and G. Kaur, "An inner round pipeline architecture hardware core for AES," in *2022 2nd Asian Conference on Innovation in Technology (ASIANCON)*, pp. 1–5, 2022.

[29] S. Tillich and J. Großschädl, "Power analysis resistant AES Implementation with instruction set extensions," in *Cryptographic Hardware and Embedded Systems - CHES 2007* (P. Paillier and I. Verbauwhede, eds.), (Berlin, Heidelberg), pp. 303–319, Springer Berlin Heidelberg, 2007.

[30] "ARMV8 architecture reference manual, for ARMV8-A , 2020.."

[31] U. Banerjee, A. Wright, C. Juvekar, M. Waller, Arvind, and A. P. Chandrakasan, "An energy-efficient reconfigurable DTLS cryptographic engine for securing Internet-of-Things applications," *IEEE Journal of Solid-State Circuits*, vol. 54, no. 8, pp. 2339–2352, 2019.

[32] P. Nannipieri, S. D. Matteo, L. Baldanzi, L. Crocetti, L. Zulberti, S. Saponara, and L. Fanucci, "VLSI Design of advanced-features AES cryptoprocessor in the framework of the european processor initiative," *IEEE TVLSI*, vol. 30, no. 2, pp. 177–186, 2022.

[33] L. Pan, G. Tu, S. Liu, Z. Cai, X. Xiong, and M. A. Jan, "A lightweight AES Coprocessor based on RISC-V custom instructions," *Sec. and Commun. Netw.*, vol. 2021, jan 2021.

[34] C. Duran and E. Roa, "A 10pj/bit 256b AES-SoC Exploiting memory access acceleration," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 69, no. 3, pp. 1612–1616, 2022.

[35] T. Gomes, P. Sousa, M. Silva, M. Ekpanyapong, and S. Pinto, "FAC-V: An FPGA-Based AES coprocessor for RISC-V," *Journal of Low Power Electronics and Applications*, vol. 12, no. 4, 2022.

[36] S. Mangard, E. Oswald, and T. Popp, "Power analysis attacks - revealing the secrets of smart cards," 2007.

[37] E. Brier, C. Clavier, and F. Olivier, "Correlation power analysis with a leakage model," in *Cryptographic Hardware and Embedded Systems - CHES 2004* (M. Joye and J.-J. Quisquater, eds.), (Berlin, Heidelberg), pp. 16–29, Springer Berlin Heidelberg, 2004.

[38] M. Avital, I. Levi, O. Keren, and A. Fish, "Cmos based gates for blurring power information," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 63, no. 7, pp. 1033–1042, 2016.

[39] G. Goodwill, B. Jun, J. Jaffe, and P. Rohatgi, "A testing methodology for side channel resistance," 2011.

[40] D. Jayasinghe, A. Ignjatovic, and S. Parameswaran, "Scrip: Secure random clock execution on soft processor systems to mitigate power-based side channel attacks," in *2019 IEEE/ACM, ICCAD*, pp. 1–7, 2019.

[41] R. Munny and J. Hu, "Power side-channel attack detection through battery impedance monitoring," in *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1–5, 2021.

[42] W. Cao, F. Huang, M. Zheng, and H. Hu, "Attacking FPGA-based dual complementary AES implementation using HD and SD Models," in *2020 16th International Conference on Computational Intelligence and Security (CIS)*, pp. 278–282, 2020.

[43] R. Menicocci, A. Trifiletti, and F. Trotta, "A logic level countermeasure against CPA side channel attacks on AES," in *Proceedings of the 20th International Conference Mixed Design of Integrated Circuits and Systems - MIXDES 2013*, pp. 403–407, 2013.

[44] Y. Nomata, M. Matsubayashi, K. Sawada, and A. Satoh, "Comparison of side-channel attack on cryptographic cirucits between old and new technology FPGAS," in *2016 IEEE 5th Global Conference on Consumer Electronics*, pp. 1–4, 2016.

[45] K. Tiri, M. Akmal, and I. Verbauwhede, "A dynamic and differential CMOS logic with signal independent power consumption to withstand differential power analysis on smart cards," in *Proceedings of the 28th European Solid-State Circuits Conference*, pp. 403–406, 2002.

[46] T. Popp and S. Mangard, "Masked dual-rail pre-charge logic: DPA-Resistance without routing constraints," in *Cryptographic Hardware and Embedded Systems – CHES 2005* (J. R. Rao and B. Sunar, eds.), (Berlin, Heidelberg), pp. 172–186, Springer Berlin Heidelberg, 2005.

[47] D. Bellizia, G. Scotti, and A. Trifiletti, "TEL Logic style as a countermeasure against side-channel attacks: Secure cells library in 65nm uppercaseCMOS and experimental results," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 65, no. 11, pp. 3874–3884, 2018.

[48] M. Bucci, L. Giancane, R. Luzzi, G. Scotti, and A. Trifiletti, "Delay-based dual-rail precharge logic," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 19, no. 7, pp. 1147–1153, 2011.

[49] Y. Zhang, G. Wang, Y. Ma, and J. Li, "A comprehensive design method based on WDDL and dynamic cryptosystem to resist DPA Attack," in *2011 International Conference on Intelligence Science and Information Engineering*, pp. 333–336, 2011.

[50] K. Bhattacharya and N. Ranganathan, "A linear programming formulation for security-aware gate sizing," in *Proceedings of the 18th ACM Great Lakes Symposium on VLSI*, GLSVLSI '08, (New York, NY, USA), p. 273–278, Association for Computing Machinery, 2008.

[51] E. Laohavaleeson and C. Patel, "Current flattening circuit for DPA countermeasure," in *2010 IEEE HOST*, pp. 118–123, 2010.

[52] G. Ratanpal, R. Williams, and T. Blalock, "An on-chip signal suppression countermeasure to power analysis attacks," *IEEE Transactions on Dependable and Secure Computing*, vol. 1, no. 3, pp. 179–189, 2004.

[53] V. Telandro, E. Kussener, A. Malherbe, and H. Barthelemy, "On-chip voltage regulator protecting against power analysis attacks," in *2006 49th IEEE International Midwest Symposium on Circuits and Systems*, vol. 2, pp. 507–511, 2006.

[54] A. Gornik, A. Moradi, J. Oehm, and C. Paar, "A hardware-based countermeasure to reduce side-channel leakage: Design, implementation, and evaluation," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, no. 8, pp. 1308–1319, 2015.

[55] Y. Lu, M. P. O'Neill, and J. V. McCanny, "FPGA Implementation and analysis of random delay insertion countermeasure against dpa," in *2008 International Conference on Field-Programmable Technology*, pp. 201–208, 2008.

[56] Z. Tao, R. Sun, and J. Dong, "Software countermeasures against DVFS fault attack for AES," in *2023 10th International Conference on Dependable Systems and Their Applications (DSA)*, pp. 575–582, 2023.

[57] R. Soares, N. Calazans, F. Moraes, P. Maurine, and L. Torres, "A robust architectural approach for cryptographic algorithms using GALS pipelines," *IEEE Design Test of Computers*, vol. 28, no. 5, pp. 62–71, 2011.

[58] J. Ambrose, R. Ragel, S. Parameswaran, and A. Ignjatovic, "Multiprocessor information concealment architecture to prevent power analysis-based side channel attacks," *IET Computers Digital Techniques*, vol. 5, pp. 1–15, 2011.

[59] P. Moucha, S. Jeřábek, and M. Novotný, "Novel dummy rounds schemes as a DPA countermeasure in PRESENT Cipher," in *2020 23rd International Symposium on Design and Diagnostics of Electronic Circuits Systems (DDECS)*, pp. 1–4, 2020.

[60] A. Baylis, G. Stitt, and A. Gordon-Ross, "Overlay-based side-channel countermeasures: A case study on correlated noise generation," in *2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS)*, pp. 1308–1311, 2017.

[61] J. Lagasse, C. Bartoli, and W. Burleson, "Combining clock and voltage noise countermeasures against power side-channel analysis," in *2019 IEEE 30th International Conference on Application-specific Systems, Architectures and Processors (ASAP)*, vol. 2160-052X, pp. 214–217, 2019.

[62] D. Das, S. Maity, S. B. Nasir, S. Ghosh, A. Raychowdhury, and S. Sen, "ASNI: Attenuated signature noise injection for low-overhead power side-channel attack immunity," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 65, no. 10, pp. 3300–3311, 2018.

[63] K.-S. Chong, J.-S. Ng, J. Chen, N. K. Z. Lwin, N. A. Kyaw, W.-G. Ho, J. Chang, and B.-H. Gwee, "Dual-hiding side-channel-attack resistant FPGA-Based asynchronous-logic AES: Design, countermeasures and evaluation," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 11, no. 2, pp. 343–356, 2021.

[64] D. Jayasinghe, A. Ignjatovic, and S. Parameswaran, "RFTC: Runtime frequency tuning countermeasure using FPGA Dynamic reconfiguration to mitigate power analysis attacks," in *ACM/IEEE DAC*, pp. 1–6, 2019.

[65] B.-A. Dao, T.-T. Hoang, A.-T. Le, A. Tsukamoto, K. Suzaki, and C.-K. Pham, "Correlation power analysis attack resisted cryptographic RISC-V SoC with random dynamic frequency scaling countermeasure," *IEEE Access*, vol. 9, pp. 151993–152014, 2021.

[66] S. Pillement, M. M. Real, J. Pottier, T. Nieddu, B. L. Gal, S. Faucou, J. Béchennec, M. Briday, S. Girbal, J. L. Rhun, O. Gilles, D. G. Pérez, A. Sintzoff, and J. Coulon, "Securing a RISC-V architecture: A dynamic approach," in *2023 DATE*, pp. 1–5, 2023.

[67] L. Feng, J. Huang, L. Li, H. Zhang, and Z. Wang, "RVDFI: A RISC-V Architecture with security enforcement by high performance complete data-flow integrity," *IEEE Transactions on Computers*, vol. 71, 2022.

[68] L. Ordu and B. Ors, "Power analysis resistant hardware implementations of AES," in *2007 14th IEEE International Conference on Electronics, Circuits and Systems*, pp. 1408–1411, 2007.

[69] Y. Chen, A. Hajiabadi, R. Poussier, A. Diavastos, S. Bhasin, and T. E. Carlson, "Mitigating power attacks through fine-grained instruction reordering," *CoRR*, vol. abs/2107.11336, 2021.

[70] A. Miyajan, Z. Shi, C.-H. Huang, and T. F. Al-Somani, "An efficient high-order masking of AES using SIMD," in *2015 Tenth International Conference on Computer Engineering Systems (ICCES)*, pp. 363–368, 2015.

[71] Y. Gui, S. M. Tamore, A. S. Siddiqui, and F. Saqib, "A key update scheme for side-channel attack mitigation," in *2019 IEEE 16th International Conference on Smart Cities: Improving Quality of Life Using ICT IoT and AI (HONET-ICT)*, pp. 187–188, 2019.

[72] M. Arsath K F, V. Ganesan, R. Bodduna, and C. Rebeiro, "PARAM: A Microprocessor hardened for power side-channel attack resistance," in *2020 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pp. 23–34, 2020.

[73] E. D. Mulder, S. Gummalla, and M. Hutter, "INVITED: Protecting RISC-V against side-channel attacks," in *2019 56th ACM/IEEE Design Automation Conference (DAC)*, pp. 1–4, 2019.

[74] H. Gross, M. Jelinek, S. Mangard, T. Unterluggauer, and M. Werner, "Concealing secrets in embedded processors designs," in *Smart Card Research and Advanced Applications* (K. Lemke-Rust and M. Tunstall, eds.), (Cham), pp. 89–104, Springer International Publishing, 2017.

[75] "ARM CORTEX M-0 Technical reference manual, Tech. rep,." `https://developer.arm.com/documentation/ddi0432/latest/`.

[76] "ARM CORTEX M-4 Technical reference manual, Tech. rep,." `https://developer.arm.com/documentation/100166/0001`.

[77] M. Gautschi, P. D. Schiavone, A. Traber, I. Loi, A. Pullini, D. Rossi, E. Flamand, F. K. Gürkaynak, and L. Benini, "Near-threshold RISC-V Core with DSP Extensions for scalable IoT Endpoint devices," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 10, pp. 2700–2713, 2017.

[78] M. Gautschi, A. Traber, A. Pullini, L. Benini, M. Scandale, A. Di Federico, M. Beretta, and G. Agosta, "Tailoring instruction-set extensions for an ultra-low power tightly-coupled cluster of OpenRISC cores," in *2015 IFIP/IEEE International Conference on Very Large Scale Integration (VLSI-SoC)*, pp. 25–30, 2015.

[79] "ARM CORTEX M-3 Technical reference manual, Tech. rep,." `https://developer.arm.com/products/processors/cortex-m/cortex-m3`.

[80] "SIFIVE E24 Instruction manual." ://www.sifive.com/cores/e24.

[81] A. Satoh, S. Morioka, K. Takano, and S. Munetoh, "A compact rijndael hardware architecture with s-box optimization," in *ASIACRYPT*, 2001.

[82] A. Hodjat and I. Verbauwhede, "Minimum area cost for a 30 to 70 Gbits/s AES processor," in *IEEE Computer Society Annual Symposium on VLSI*, pp. 83–88, IEEE, 2004.

[83] "Intel advanced encryption standard (AES) new instructions set, 2012."

[84] Berna and B. Preneel, "Power analysis of an FPGA implementation of Rijndael: Is pipelining a DPA countermeasure," in *in Cryptographic Hardware and Embedded Systems*, pp. 217–227, Citeseer, 2004.

[85] F. Rahman, M. Farmani, M. Tehranipoor, and Y. Jin, "Hardware-assisted cybersecurity for IoT devices," in *2017 18th International Workshop on Microprocessor and SoC Test and Verification (MTV)*, pp. 51–56, 2017.

[86] D. Canright, "A very compact s-box for aes," in *Cryptographic Hardware and Embedded Systems – CHES 2005* (J. R. Rao and B. Sunar, eds.), (Berlin, Heidelberg), pp. 441–455, Springer Berlin Heidelberg, 2005.

[87] B. Marshall, G. R. Newell, D. Page, M.-J. O. Saarinen, and C. Wolf, "The design of scalar AES Instruction set extensions for RISC-V," *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, vol. 2021, pp. 109–136, 2020.

[88] "ARM Cryptocell-312." `https://developer.arm.com`.

[89] C. G. de Araujo Gewehr and F. G. Moraes, "Improving the efficiency of cryptography algorithms on resource-constrained embedded systems via risc-v instruction set extensions," in *36th SBC/SBMicro/IEEE/ACM Symposium on Integrated Circuits and Systems Design*, pp. 1–6, 2023.

[90] F. Lozachmeur and A. Tisserand, "A RISC-V Instruction set extension for flexible hardware/software protection of cryptosystems masked at high orders," in *2023 IEEE 66th IMWSCAS*, pp. 360–364, 2023.

[91] D. Bellizia, S. Bongiovanni, P. Monsurrò, G. Scotti, A. Trifiletti, and F. B. Trotta, "Secure double rate registers as an RTL Countermeasure against power analysis attacks," *IEEE VLSI*, vol. 26, pp. 68–76, 2018.

[92] M. Kar, A. Singh, S. Mathew, A. Rajan, V. De, and S. Mukhopadhyay, "Improved power-side-channel-attack resistance of an AES-128 core via a security-aware integrated buck voltage regulator," in *ISSCC*, 2017.

[93] A. Singh, M. Kar, V. C. K. Chekuri, S. K. Mathew, A. Rajan, V. De, and S. Mukhopadhyay, "Enhanced power and electromagnetic SCA Resistance of encryption engines via a security-aware integrated all-digital LDO," *IEEE Journal of Solid-State Circuits*, pp. 478–493, 2020.

[94] H. Huang, L. Liu, Q. Huang, Y. Chen, S. Yin, and S. Wei, "Low area-overhead low-entropy masking scheme (LEMS) against correlation power analysis attack," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 38, no. 2, pp. 208–219, 2019.

[95] P.-C. Liu, J.-H. Hsiao, H.-C. Chang, and C.-Y. Lee, "A 2.97 Gb/s DPA-resistant AES engine with self-generated random sequence," in *2011 Proceedings of the ESSCIRC (ESSCIRC)*, pp. 71–74, 2011.

[96] C. Tokunaga and D. Blaauw, "Securing encryption systems with a switched capacitor current equalizer," *IEEE Journal of Solid-State Circuits*, vol. 45, no. 1, pp. 23–31, 2010.

[97] D. Hwang, K. Tiri, A. Hodjat, B.-C. Lai, S. Yang, P. Schaumont, and I. Verbauwhede, "AES-Based security coprocessor IC with resistance to differential power analysis side-channel attacks," *IEEE Journal of Solid-State Circuits*, vol. 41, no. 4, pp. 781–792, 2006.

[98] S. Agwa, E. Yahya, and Y. Ismail, "Power efficient AES core for IoT constrained devices implemented in 130nm CMOS," in *ISCAS*, 2017.

[99] D.-H. Bui, D. Puschini, S. Bacles-Min, E. Beigné, and X.-T. Tran, "AES Datapath optimization strategies for low-power low-energy multisecurity-level internet-of-things applications," *IEEE TVLSI*, vol. 25, no. 12, pp. 3281–3290, 2017.

[100] M. Lu, A. Fan, J. Xu, and W. Shan, "A Compact, lightweight and low-cost 8-bit datapath AES Circuit for IoT Applications in 28nm CMOS," in *17th IEEE Conference On TrustCom/BigDataSE*, pp. 1464–1469, 2018.

[101] S.-H. Cheng, M.-H. Lee, B.-C. Wu, and T.-T. Liu, "A lightweight power side-channel attack protection technique with minimized overheads using on-demand current equalizer," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 69, no. 10, pp. 4008–4012, 2022.

[102] M. Kar, A. Singh, S. K. Mathew, A. Rajan, V. De, and S. Mukhopadhyay, "Reducing power side-channel information leakage of AES engines using fully integrated inductive voltage regulator," *IEEE Journal of Solid-State Circuits*, vol. 53, no. 8, pp. 2399–2414, 2018.

[103] J.-S. Coron and I. Kizhvatov, "Analysis and improvement of the random delay countermeasure of CHES 2009," in *Proceedings of the 12th International Conference on Cryptographic Hardware and Embedded Systems*, CHES'10, p. 95–109, Springer, 2010.

[104] R. Kumar, V. B. Suresh, S. Taneja, M. A. Anders, S. Hsu, A. Agarwal, V. De, and S. K. Mathew, "A 7-Gbps SCA-Resistant multiplicative-masked AES engine in intel 4 CMOS," *IEEE Journal of Solid-State Circuits*, vol. 58, no. 4, pp. 1106–1116, 2023.

[105] R. Kumar, V. Suresh, M. Kar, S. Satpathy, M. A. Anders, H. Kaul, A. Agarwal, S. Hsu, G. K. Chen, R. K. Krishnamurthy, V. De, and S. K. Mathew, "A 4900- μ m2 839-mb/s side-channel attack- resistant AES-128 in 14-nm CMOS with heterogeneous sboxes, linear masked mixcolumns, and dual-rail key addition," *IEEE Journal of Solid-State Circuits*, vol. 55, no. 4, pp. 945–955, 2020.

[106] S. J. Kim, D. Kim, A. Sharma, and M. Seok, "EQZ-LDO: A Near-zero EDP Overhead, >10m-attack-resilient, secure digital ldo featuring attack-detection and detection-driven protection for a correlation-power-analysis-resilient IoT Device," in *Symposium on VLSI Circuits*, 2021.

[107] J. Yang, J. Han, F. Dai, W. Wang, and X. Zeng, "A power analysis attack resistant multicore platform with effective randomization techniques," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 28, no. 6, pp. 1423–1434, 2020.