Enhancing Cryptographic Application Performance on RISC-V Processors with B and K Extensions

Harsh Mayank Dabhi
Electronics and computer engineering
Dublin city university
Dublin, Ireland
harshmayank.dabhi2@mail.dcu.ie

Abstract—**The paper presents a comprehensive literature survey on enhancing the performance of cryptographic algorithms on RISC-V processors through Instruction Set Architecture (ISA) extensions. The increasing adoption of RISC-V in resource-constrained embedded systems creates a pressing need for computationally efficient security. Standard software implementations of algorithms like the Advanced Encryption Standard (AES) and Secure Hash Algorithm (SHA) are often a performance bottleneck. This review critically analyzes prior work that leverages the RISC-V Bit-Manipulation ('B') and the highly specialized Scalar Cryptography ('K') extensions to accelerate these critical workloads. The survey details specific optimization techniques, such as replacing memory-intensive S-Boxes with single-cycle cryptographic instructions, yielding performance gains for AES. It also examines the use of general-purpose bit-manipulation instructions for more modest but broadly applicable speedups for SHA-256. Methodologies for evaluating these extensions, including cycle-accurate simulation and analysis of hardware area overhead are synthesized. The findings consistently demonstrate that ISA extensions provide a powerful, area-efficient pathway to high-performance cryptography. This survey synthesizes these results to justify a project focused on implementing and evaluating these optimizations within a simulated RISC-V environment using Spike and QEMU**

I. Introduction

The RISC-V Instruction Set Architecture (ISA) has initiated a new era of processor innovation, distinguished by its open-source, royalty-free, and modular design philosophy[1]. This approach democratizes processor design, allowing academics and industries to create highly customized cores without prohibitive licensing fees. A key tenet of RISC-V is its extensibility; a minimal base integer ISA can be augmented with a rich set of optional, standardized extensions to tailor a processor for specific application domains. This modularity has made RISC-V a popular choice for the rapidly growing Internet of Things (IoT) and embedded systems markets, where devices are often severely constrained by computational power, memory, and energy budgets.

Within this resource-constrained environment, the need for robust security is a non-negotiable requirement. The execution of fundamental cryptographic algorithms, which ensure data confidentiality and integrity, presents a significant performance challenge. Primitives such as the Advanced Encryption Standard (AES) and the Secure Hash Algorithm (SHA) are computationally intensive when implemented purely in software on a general-purpose processor. For instance, a common operation like a 32-bit rotation, which is fundamental to the SHA-2 algorithm, must be emulated using a sequence of three separate instructions (two shifts and a logical OR) on a base RISC-V core, creating a significant performance bottleneck. Similarly, software optimizations for AES, such as using pre-computed T-Tables, can improve speed but at the prohibitive cost of up to 4KB of memory, a resource often unavailable on low-power microcontrollers.

The RISC-V standard directly addresses this performance gap through two key optional extensions. The first is the general-purpose Bit-Manipulation ('B') extension, which provides a comprehensive toolkit of instructions for efficient bit-level operations that are common across many algorithms, including cryptography. The second is the domain-specific Scalar Cryptography ('K') extension, which offers a set of highly optimized, powerful instructions designed to accelerate the core computational rounds of symmetric block ciphers and hash functions like AES and SHA.

This literature survey will conduct a critical review and synthesis of prior research that has implemented and evaluated the performance impact of these 'B' and 'K' extensions on RISC-V processors. The goal is to build upon these findings by implementing and rigorously evaluating these cryptographic optimizations in a standardized, accessible, and instruction-accurate simulated environment using tools like Spike and QEMU.

II. Review and Analysis of Prior Work

A. RISC-V Architecture and Its Extensions

The RISC-V architecture (as shown in Fig. 1)[1] has emerged as a transformative platform in the field of computing due to its open-source nature and modular design, which allows for the development of domain-specific extensions. Among these, the B (Bit-Manipulation) and K (Cryptography) extensions have been specifically designed to improve the performance of cryptographic and other computationally intensive workloads.

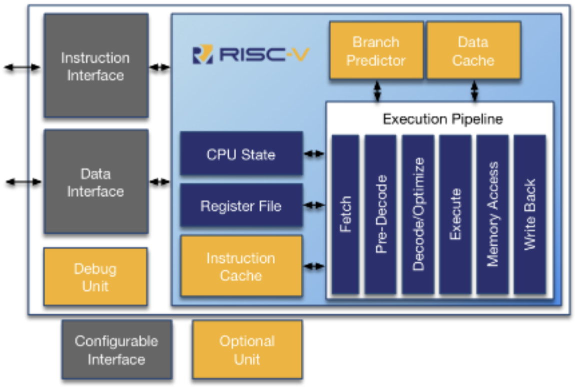

Fig. 1. RISC-V block diagram. Adapted from: https://roalogic.github.io/RV12/DATASHEET.html

B. *The RISC-V Bit-Manipulation ('B') Extension*

The 'B' extension is a general-purpose toolkit designed to improve performance and code density for a wide range of applications involving bit-level logic. As a ratified standard, it is composed of several sub-extensions, including Zba (Address generation), Zbb (Basic bit manipulation), Zbc (Carry-less multiply), and Zbs (Single-bit instructions). The work reviewed here focuses primarily on the Zbb subset, which provides a rich set of instructions that are directly beneficial to cryptography.

The work by Markov *et al.*[2] presents a from-scratch implementation of the Zbb extension as a tightly-coupled module for a 32-bit RISC-V core. They implemented all 18 instructions in the 32-bit Zbb standard, including `andn` (AND with negate), `orn` (OR with negate), `clz` (count leading zeros), `ctz` (count trailing zeros), `rol` (rotate left), `ror` (rotate right), and `rev8` (byte reverse). Their evaluation across 10 C-language benchmark programs demonstrated an average execution acceleration of 29.9% and an average code size reduction of 37.5%. This significant gain came at a notable hardware cost, with the number of FPGA logic cells increasing nearly tenfold (from 99 to 986) and registers increasing eightfold (from 103 to 821) for their minimal core. However, this still represents a very small fraction of the total resources on a modern FPGA.

Complementing this, Babu *et al.*[3] evaluated the 'B' extension's impact on larger, standardized benchmarks from the Embench suite. Their cycle-accurate Verilator simulation of a 32-bit core with 'B' support showed significant gains on computationally intensive tasks. Notably, on the `nettle-sha256` benchmark, they reported a speedup of 1.42x and a 20% reduction in executable size. On the `crc32` benchmark, the use of instructions like `rori`, `rol`, and `grevi` contributed to substantial performance gains. Their findings confirm that the 'B' extension provides a foundational layer of optimization, improving both speed and code density by replacing multi-instruction software sequences with efficient, single-cycle hardware instructions.

C. *The Scalar Cryptography ('K') Extension*

While the 'B' extension is a general-purpose tool, the Scalar Cryptography ('K') extension is a domain-specific accelerator designed for maximum performance on specific cryptographic algorithms. It is subdivided by algorithm, with `Zkne` targeting AES encryption, `Zknd` for AES decryption, and `Zknh` for SHA-2 hashing.

The work by Gewehr *et al*[6]. provides a thorough analysis of the `Zkne` and `Zknh` extensions implemented on a low-power, 2-stage Ibex RISC-V core. Their approach demonstrates the transformative potential of these instructions.

For AES acceleration, the `Zkne` extension provides the `aes32esmi` and `aes32esi` instructions. The `aes32esmi` instruction is particularly powerful, as it encapsulates the three most computationally intensive steps of an AES round—SubBytes, ShiftRows, and MixColumns—into a single-cycle operation. This completely obviates the need for software-based S-Box or T-Table lookups, which are either slow or memory-intensive. The result is a dramatic performance improvement. Compared to a baseline C implementation from the TinyCrypt library, the `Zkne`-accelerated version achieved a 42.57x speedup for AES-128 encryption and a 44.81x speedup for AES-256 encryption. These gains are an order of magnitude greater than what is achievable with general-purpose extensions and highlight the power of domain-specific instruction design.

For SHA-2 acceleration, the `Zknh` extension provides instructions that directly map to the standard's and functions. For example, `sha256sum0` computes `ROTR2(x)` `ROTR13(x)` `ROTR22(x)` in a single instruction. This directly replaces the three-instruction software sequence required to emulate rotation and avoids the associated register pressure. This hardware-level optimization resulted in a performance gain of 1.45x for the core compression function of SHA-256 and 1.74x for SHA-512. These results, when compared to the 1.42x gain seen from the 'B' extension on the entire `nettle-sha256` benchmark, suggest that `Zknh` provides a more potent acceleration for the core cryptographic workload.

D. *Benchmarking and Evaluation Methodologies*

The validity of the reported performance gains rests on a rigorous and well-defined evaluation methodology. The surveyed literature employs a consistent set of metrics and tools across different levels of abstraction.

- Key Performance Metrics: The primary metric is performance speedup, universally calculated as the ratio of clock cycles consumed by the baseline software implementation to that of the hardware-accelerated version. Code density is another critical metric for embedded systems, measured by analyzing the final executable's code size, static data size, and maximum stack usage. The hardware cost of the extensions is a crucial part of the trade-off analysis, quantified in terms of FPGA resource utilization (logic elements/LUTs and registers) or, for more accurate results, ASIC cell area (in µm² or kGE) after synthesis with a specific technology library. Finally, energy efficiency, measured in nanojoules (nJ) per cryptographic operation, is evaluated through gate-level power analysis of the post-synthesis netlist, and has shown gains of up to 28.91x for AES.

- Simulation and Prototyping Environments: The papers use a range of tools to validate their designs. For logical verification and cycle-accurate performance measurement at the Register-Transfer Level (RTL), simulators such as ModelSim and Verilator are used. For architectural exploration and software testing, Instruction Set Simulators (ISS) like RARS are employed. The work by Schlägl *et al.* [7]advocates for system-level evaluation using Virtual Prototypes (VPs) built with SystemC/TLM, which enable fast, instruction-accurate simulation of a full system, including peripherals. The proposed project's use of Spike (the official RISC-V ISS) and QEMU (a full-system emulator) aligns perfectly with this established practice of using ISSs for architectural performance analysis.

E. Role of Simulation in Performance Evaluation

Simulation tools play a vital role in evaluating the performance of RISC-V extensions, particularly in the absence of physical hardware. Tools like Spike, QEMU, and Gem5 are widely used to model the behavior of RISC-V processors with and without extensions.

1) ***Spike and QEMU:*** These simulators provide cycle-accurate modeling of RISC-V processors and enable researchers to evaluate the impact of the B and K extensions on cryptographic workloads. They are particularly useful for comparing execution times and energy consumption for different configurations.

2) ***Gem5 and Power Modeling:*** Gem5 extends the capabilities of standard simulators by integrating power models, allowing researchers to estimate energy efficiency alongside throughput and resource utilization. This holistic approach provides a comprehensive view of the trade-offs associated with implementing RISC-V extensions.

3) ***Challenges and Limitation:*** While simulators are invaluable for early-stage evaluation, they often fail to capture real-world constraints, such as thermal effects and hardware-level interactions. These limitations highlight the need for experimental validation on physical RISC-V hardware platforms.

F. Comparative Analysis of Approaches

The reviewed literature provides a clear distinction between software-only and hardware-accelerated approaches for cryptographic workloads:

- **Software-Only Solutions:** Ideal for general-purpose use cases where flexibility is prioritized over performance.
- **Instruction Set Extensions:** Suitable for high-performance applications, offering superior throughput and energy efficiency.

The B and K extensions in RISC-V demonstrate substantial potential to enhance cryptographic operations, particularly AES and SHA. However, their adoption must be guided by application-specific requirements, considering factors like energy constraints, hardware cost, and desired performance levels.

III. Relation of Prior Work to the Project Problem

The comprehensive body of reviewed literature provides a clear and compelling justification for the proposed project. It not only validates the project's core premise but also offers a detailed roadmap for its methodology and execution.

a) *Comparison of Prior Work with the Project Focus:* The central problem this project seeks to address the quantitative evaluation of RISC-V 'B' and 'K' extensions for accelerating cryptographic workloads is the central thesis of the works by Gewehr *et al.*[4], Babu *et al.*[3], and Markov *et al.*[2]. These papers provide direct, empirical evidence of the extensions' effectiveness, offering specific implementation details and performance results that the project aims to verify and analyze within a different, standardized simulation context. The literature confirms that this is a relevant and impactful area of research.

b) *Applicability and Adaptation of Prior Methods:* The methodologies and techniques described in the literature are highly applicable and will be directly adapted for this project. The architectural model of a tightly-coupled functional unit integrated into the processor pipeline will be the conceptual basis for the simulation. The project will implement the specific instruction-level optimizations identified as most effective: using the `Zkne` instructions for AES to eliminate table lookups, using the `Zknh` instructions for SHA-2 to eliminate rotation emulation, and using the general-purpose `Zbb` instructions as a comparative baseline for optimization. The benchmarking methodology will be directly adopted, focusing on the rigorous measurement and comparison of execution clock cycles and compiled code size, which are the most relevant metrics for an ISS-based evaluation.

c) *Unsuitability of Certain Approaches:* While most of the literature is directly relevant, some approaches are outside the project's defined scope. The coprocessor-based model for a MIPS ISA presented by Elmohr *et al.* [5] is architecturally distinct from the RISC-V pipeline model and is therefore not directly applicable. Similarly, the RISC-V Vector ('V') extension, as surveyed by Wang *et al.*[8], represents a different design philosophy aimed at data-parallel, high-performance computing (HPC) and scientific workloads. This is a fundamentally different approach from the lightweight, scalar, low-power extensions ('B' and 'K') that are the focus of this project and are more suitable for the target embedded systems domain.

d) *Proposed Solution Strategy:* Drawing on the insights from prior work, this project proposes a multi-faceted solution strategy that builds on the strengths of existing approaches while addressing their limitations:

1) Comprehensive Benchmarking Framework: The project will utilize RISC-V simulators, such as Spike or QEMU, to evaluate AES and SHA performance with and without the B and K extensions. Key metrics will include throughput, execution cycles, and energy consumption, providing a well-rounded performance assessment.
2) Realistic Workload Design: Test cases will emulate real-world cryptographic operations, including dynamic key expansion for AES, multi-message hashing for SHA, and mixed cryptographic workloads. These scenarios will provide insights into the practical applicability of the RISC-V extensions in diverse applications.
3) Holistic Analysis of Trade-Offs: The project will evaluate not only performance improvements but also the hardware costs associated with implementing the B and K extensions. This analysis will ensure that the findings are relevant to both high-performance and resource-constrained applications. Instead of focusing on speed of execution, the evaluation will utilize other benchmarking tools such as resource utilization, throughput, and efficiency metrics.

By leveraging state-of-the-art methods and addressing the gaps identified in the literature, this project aims to provide a nuanced and comprehensive evaluation of the RISC-V B and K extensions. The proposed solution strategy balances performance gains with hardware and energy trade-offs, offering valuable insights for future adoption of these extensions in cryptographic applications.

IV. Conclusion

This study aims to evaluate the performance acceleration achieved by leveraging the RISC-V architecture's B (Bit-Manipulation) and K (Cryptography) extensions for cryptographic workloads, specifically AES and SHA. The literature review highlights the significant benefits of these extensions in improving throughput, reducing execution cycles, and enhancing energy efficiency. However, it also reveals gaps in existing research, particularly in the holistic evaluation of multiple performance metrics and the applicability of findings to real-world cryptographic scenarios.

By building on state-of-the-art methods, this project proposes a comprehensive evaluation framework that integrates advanced power modeling and resource utilization analysis into RISC-V simulators. Additionally, the adoption of realistic workload scenarios will ensure the relevance of the results to practical applications. Through this approach, the project not only seeks to quantify the benefits of the B and K extensions but also to understand the trade-offs involved in their implementation, particularly in resource-constrained environments like IoT and embedded systems.

Ultimately, the findings of this project will contribute to a deeper understanding of how hardware-level enhancements in open-source architectures like RISC-V can transform cryptographic operations. By addressing existing gaps and providing actionable insights, this work aims to guide future research and development in the design and optimization of cryptographic solutions using extensible architectures.

V. References

[1] https://roalogic.github.io/RV12/DATASHEET.html

[2] D. Markov and A. Romanov, "Implementation of the RISC-V Architecture with the Extended Zbb Instruction Set," in *2022 International Ural Conference on Electrical Power Engineering (UralCon)*, 2022.

[3] P. S. Babu, S. Sivaraman, D. N. Sarma, and T. S. Warrier, "Evaluation of Bit Manipulation Instructions in Optimization of Size and Speed in RISC-V," in *2021 34th International Conference on VLSI Design and 2021 20th International Conference on Embedded Systems (VLSID)*, 2021.

[4] C. G. de A. Gewehr and F. G. Moraes, "Improving the Efficiency of Cryptography Algorithms on Resource-Constrained Embedded Systems via RISC-V Instruction Set Extensions," in *2023 36th SBC/SBMicro/IEEE/ACM Symposium on Integrated Circuits and Systems Design (SBCCI)*, 2023.

[5] M. A. Elmohr *et al.*, "Hardware Implementation of A SHA-3 Application-Specific Instruction Set Processor," in *2016 28th International Conference on Microelectronics (ICM)*, 2016.

[6] C. Gewehr *et al.*, "Hardware Acceleration of Authenticated Encryption with Associated Data via RISC-V Instruction Set Extensions in Low Power Embedded Systems," in *2024 IEEE 15th Latin America Symposium on Circuits and Systems (LASCAS)*, 2024.

[7] M. Schlägl, M. Stockinger, and D. Große, "A RISC-V 'V' VP: Unlocking Vector Processing for Evaluation at the System Level," in *Proc. Design, Automation & Test in Europe Conf. (DATE)*, 2024.

[8] J. Wang, L. Yu, W. Zhuang, X. Yang, S. Zhang, and Z. Qin, "Research on Vector Extension of Instruction Set Architecture," pp. 378–385, Oct. 2024, doi: https://doi.org/10.1109/cbase64041.2024.10824427.