International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18(16s) 2026 pp. 1480-1489
https://doi.org/10.70917/ijcisim-2026-4679
Article

A Survey of AES-256 and SHA-256 Hardware Architectures on FPGA

Vasudeva G.[1], Bharathi Gururaj[2], Mruthika N.[3], Raghavendra Chinmai S.[4], Sowjanya M. Sharma[5], Yashwanth S.[6]

[1] Department of Electronics and Communication Engineering, Dayananda Sagar Academy of Technology and Management, Bengaluru, Karnataka, India.
Email: devan.vasu921@gmail.com
[2] Department of Electronics and Communication Engineering, KS Institute of Technology, Bengaluru, Karnataka, India.
Email: bharathigururaj@gmail.com
[3] Department of Electronics and Communication Engineering, Dayananda Sagar Academy of Technology and Management, Bengaluru, Karnataka, India.
[4] Department of Electronics and Communication Engineering, Dayananda Sagar Academy of Technology and Management, Bengaluru, Karnataka, India.
[5] Department of Electronics and Communication Engineering, Dayananda Sagar Academy of Technology and Management, Bengaluru, Karnataka, India.
[6] Department of Electronics and Communication Engineering, Dayananda Sagar Academy of Technology and Management, Bengaluru, Karnataka, India.

Abstract: In the present day, data confidentiality and integrity are among the major concerns in the embedded and communication systems and hardware-based cryptographic systems are playing a significant role in this regard. Moreover, the flexible features of FPGA implementations of cryptographic algorithms, such as greater performance, higher parallelism, and greater protection for sensitive data, are better than software-based approaches. This survey discusses the latest FPGA-based architectures of the AES-256 and the SHA-256, along with the design methodologies, optimization techniques and integration strategies. An analysis of existing studies is conducted to understand the pros and cons of various architectures, resource usage, throughput and implementation methods. The survey reveals that most current designs focus on optimizing only one aspect of encryption and hashing separately, whereas integrated designs may require extra hardware because of the duplicated control and interface logic. From these observations, it can be deduced that an efficient resource usage, scalability and secure hardware implementation is possible by integrating the AES-256 and SHA-256 components into a modular FPGA architecture.
Keywords: AES-256, SHA-256, FPGA, Hardware Security, Cryptography, Verilog HDL, Embedded Systems

1 .Introduction

With the need for embedded systems and communication networks to be secured, data security is a major requirement, with both data security and data integrity an essential requirement. The Advanced Encryption Standard (AES) is a popular choice for encrypting data and the SHA-2 family, especially SHA-256, is widely used for message authentication and integrity checks [1, 2]. While software implementations are flexible, they tend to result in a higher latency, utilization of processors and exposure of key material. The Field-Programmable Gate Arrays (FPGAs) combine the ability to be re-programmed for future applications with the option of hardware implementations that are reconfigurable and more efficient because they allow for parallel processing and better security systems than software implementations. Along with the AES variants, AES-256 is the most secure variant with a 256-bit key and fourteen

rounds of encryption, which is suitable for long-term secure systems [1]. Likewise, SHA-256 produces a fixed size message digest of 256 bits and is still commonly used in secure communications protocols [2, 3]. To increase the

throughput, decrease resource usage and optimize cryptographic performance, several FPGA based architectures have been introduced recently. These studies vary however from each other in architectural design, implementation approach as well as evaluation metrics. This survey presents an overview of the traffic of hardware implementation of the AES-256 and SHA-256, explains the works' architecture, analyzes ongoing research gaps, and reveals future research directions to build efficient FPGA-based cryptographic systems.

2. Scope and Survey Method

This survey is concerned with the hardware implementations of the AES-256 and SHA-256 algorithms, and especially FPGA based cryptographic architectures. The above-mentioned sources of reviewed literature were IEEE Xplore, ACM Digital Library, Elsevier journals, and relevant NIST standards [1]–[21]. The selected studies were systematically analyzed based on target platform, hardware description language, architectural design, resource utilization, throughput, and security features. The surveyed works were next classified in regards to architectural approaches and a full comparison was made in relation to their performance, benefits and drawbacks. The research method used in this survey is presented in Figure 1 where the method of selecting, classification and evaluation of the reviewed research works are presented. Thus, as well as the performance parameters, trade-offs between computational efficiency, consumption of hardware resource, the need for power and implementation complexity are taken into account. Special emphasis is placed on FPGA architectures that use such concepts as pipelining, loop unrolling, parallel processing, resource sharing, and architectural optimization to gain computing power in cryptographic applications. These techniques are studied to understand their impact on throughput, latency, device/filling efficiency, and total hardware efficiency.

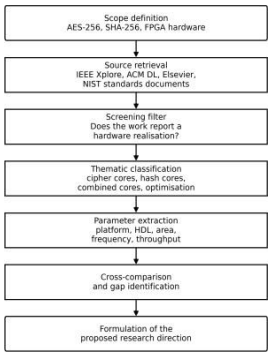

Fig. 1. Sequence followed in selecting, screening and classifying the surveyed works.

3. Literature Survey

A. Cipher Datapath Organisation

The architecture of an AES has been categorized into iterative, unrolled and pipelined architectural design, which have their own resource/performance trade-off. Iterative architectures re-use a single round of encryption, which leads to a reduction in area and an increase in the execution time. The idea was proved by Trang and Loi who showed it to be resource-efficient with moderate throughput [4]. Unrolled architectures are used to mitigate latency by replicating multiple iterations, but they may incur a higher combinational complexity that reduces operating frequency. Liu et al. optimized the datapath organization for pipeline balancing, which showed the effectiveness of optimizing pipeline through asymmetric stage partitioning [5]. Pipelined implementations are best for getting the data through quickly because they process multiple data blocks at once. Arshad et al. [6] and Kouzehgar and Moghadam [7] reported performance gains with the architecture, but with the usage of bigger FPGA resources. In general, the datapath organization can be customized depending on the application's throughput and hardware requirement.

B. Substitution-Box Realisation

One of the most important parts that have an impact on hardware performance is the AES substitution box (S-box). Memory-based lookup tables are fast and require a significant amount of FPGA memory, while the composite-field arithmetic alternatively uses memory-less arithmetic with logic, thus offering an improved area efficiency [8] and [9]. One alternative is to change the structure of the S-box to enhance performance, similar to the work done by Zodpe and Sapkal

[10] which changes the key expansion and S-box. However, changing the S-box in the standard AES from the AES-197 compliant gives less compatibility with FIPS-197 compliant systems. Furthermore, each of the implementations of the S-box has different level of resistance to the side channel attacks; thus the choice of implementation depends on the performance and security considerations.

C. Hash Compression Architectures

The arithmetic operations and message scheduling are heavily built into the design of the compression function, as opposed to AES.The design of SHA-256 is a sequential 64-round compression function, unlike AES. Recent architectures achieve the optimisation of storage by using a circular register file rather than a complete message memory, simplifying the

hardware used [11]. Another feature identified to extract performance enhancements was the critical path reduction obtained by optimizing addition scheduling by Chaves et al.,

[12] or the increase in throughput obtained by pipelining and loop unrolling by Rajan et al., [13] across several FPGA platforms. Other works were able to present flexible implementation of SHA-2 using multiple digest sizes [14]. The results of the above works show the importance of scheduling, pipelining and memory optimization in the improvement of the hardware performance of SHA-256.

Combined Cipher and Hash Architecture

There are a few researchers who have looked into the idea of incorporating both encryption and hashing on a single hardware platform. In some implementations, AES and SHA are implemented as separate hardware units that work concurrently

[15] and in others, the digests generated by SHA are used to enhance the key generation in AES [16], [17]. Key generation using the dynamics of biometrics has also been studied for application-specific applications [18]. The more complex architectures share hardware across cryptographic modules to achieve optimal performance in a reduced area. Further, Kundi et al. showed that it is possible to obtain significant hardware reductions by sharing lookup tables and XOR logic for encryption and hashing modules [19]. However, most integrated designs concentrate on AES and SHA-3, with efficient designs for AES-256 and SHA-256 designs being relatively under-explored.

D. Design Practice and Integration Models

The Verilog or VHDL design language is the most common language allotted to develop the FPGA cryptographic implementations, with the successful designs with good performance being targeted to Virtex devices, while the others providing low resource consumption are targeted to the Spartan or Artix families [5, 15, 19, 13]. Typically functional verification is done through the use of standard FIPS test vectors [1] and [2]. While there have been many works that have suggested stand-alone implementations of hardware cryptographic devices, the few works that have also tried to include both encryption and hashing in a single hardware support have failed to consider the modularity of these operations. This emphasizes the requirement of scalable architectures which are able to perform multiple cryptographic operations efficiently, with minimal hardware overhead.

TABLE I. Comparative Summary of Surveyed Hardware Implementations of AES and the SHA Families

Author	Year	Algorithm	Board	HDL	Principal Contribution	Advantages	Limitations
Arshad et al. [6]	2014	AES-128	Virtex-6	System Generator	Model-based flow with pipelined MixColumns and SubBytes stages	288.19 MHz and 36.864 Gbps reported	Large device; generated RTL limits low-level control; no hashing
Kahri et al. [11]	2015	SHA-256, Blake-256	Virtex-5	VHDL	Reconfigurable hash processor with padding unit separated from the compression datapath	907 slices at 283 MHz for 2.26 Gbps; clean modular partition	Hashing only; runtime algorithm selection adds control overhead
Priya and Karthigaikumar [20]	2017	AES	Virtex-5	VHDL	Eight parallel substitution-box instances with parallel column mixing	58.764 Gbps; demonstrates the ceiling of spatial parallelism	6,568 slices; unusable on low-cost families; no integrity function
Zodpe and Sapkal [10]	2018	AES	Spartan-6	VHDL	Modified substitution box and altered key generation procedure	3.039 Gbps with ten-cycle latency on a mid-range part	Departs from FIPS 197 tables, so interoperability is lost
Kouzehgar and Moghadam [7]	2018	AES	Virtex-5	VHDL	Deep pipeline combined with a memory-based composite-field substitution stage	60 Gbps at 460 MHz for storage-area network traffic	High area and power; benefit disappears on short messages
Al-Shara'a et al. [16]	2019	AES-192 with SHA-1	Spartan-3AN	LabVIEW / System Generator	160-bit digest used directly as cipher key in CBC mode	Shows hash-driven key derivation inside one hardware flow	SHA-1 is deprecated; 22% slice occupancy; non-standard toolchain
Kundi et al. [19]	2020	AES with SHA-3	Virtex	VHDL	Coprocessor sharing lookup tables, six-input logic and	49.37% area saving; 24.12 Gbps AES and 16 Gbps SHA-3	Intricate arbitration logic; costly target device;

					a unified XOR section		SHA-2 not covered
Jasim et al. [15]	2023	AES-128 with SHA-3	Spartan-3AN	VHDL	Two architectures pairing Keccak with AES encryption and with AES decryption	Device utilisation below four percent; parallel operation of both paths	Quoted 64 Gbps is inconsistent with block size and occupancy; AES-128 only
Sravani and Durai [18]	2023	AES with SHA-3	Virtex-7	VHDL	Biometric minutiae hashed and compressed to drive dynamic AES key generation	11.6% area reduction against the authors' baseline; 10.4 Gbps at 415 MHz	0.676–1.42 W dynamic power; architecture tied to one application
Liu et al. [5]	2024	SHA3-512	Virtex-6	VHDL	Two-stage unrolling with unequal sub-pipeline cuts and FIFO decoupling of padding	432.9 MHz and 20.78 Gbps; round constants compressed to seven bits	1,720 slices is large for constrained targets; no cipher present
Simola et al. [21]	2024	AES-256	22 nm ASIC with RISC-V	Chisel	AXI4-Lite mapped accelerator supporting five block cipher modes	82–84% cycle reduction against OpenSSL at about 2% gate overhead	ASIC target; no hashing; not reconfigurable after fabrication
Rajan et al. [13]	2025	SHA-256	Spartan-7, Kintex-7, Artix-7	Verilog	Pipelined and unrolled SHA-256 measured across four seventh-generation devices	LUT usage between 0.93% and 3.02%; useful platform comparison	Hashing only; power figures appear to include board-level assumptions
Saqlain and Ramachandran [17]	2025	AES with SHA-256	Cyclone IV E	Verilog	Session key pre-hashed by SHA-256 before the key expansion stage	12.8 Gbps; strengthens weak passphrases at negligible recurring cost	Cipher and hash remain separate cores; single vendor family evaluated

4 Comparative Analysis

Table I summarizes the characteristics of the surveyed FPGA-based cryptographic implementations using a common set of evaluation parameters. The comparison shows that high-

throughput architectures are generally implemented on Virtex-class FPGAs, while Spartan and Artix devices are preferred for resource-efficient designs [5], [19], [20], [7]. Most low-area implementations focus on either AES or SHA individually,

whereas integrated architectures typically require additional hardware resources. Among the surveyed works, only Kundi et al. [19] demonstrated effective resource sharing between cryptographic modules. Furthermore, AES-256 and SHA-256 integrated architectures remain relatively unexplored, particularly for low-cost FPGA platforms. These observations highlight the need for modular architectures that balance performance, resource utilization, and scalability.

5 Research Gaps

Based on the reviewed literature, the following research gaps are identified:

A. Independent implementation of AES and SHA:

Typically, most integrated designs also have hardwired encryption and hashing modules, wasting a dual-data-path and double-control-logic system because of the duplication of the interface, control and buffering logic in the integrated design. If both functions are combined into a single security system, you may need more interconnections and control resources to achieve high combined performance with separate AES and SHA implementations. This can result in wasting of registers, multiplexers, logic to control the data-path, and wasted memory resources. Moreover, different architectures can lead to some increase in the communication overhead between the encryption module and the hashing module. For this reason, it is crucial to have a single architecture which can effectively manage AES-256 encryption/decryption operations and SHA-256 hashing that will consume the minimum amount of redundant hardware.

B. Limited resource-sharing for AES-256 and SHA-256:

For AES and SHA-3, hardware sharing has been already proved [19] in this case, but it is still largely unexplored to achieve hardware sharing for AES-256 and SHA-256. Although AES-256 and SHA-256 have different computational structures, some support resources may be shared, such as control logic, input/output interfaces, registers, intermediate data storage, etc. Sharing of resources can be used to achieve a reduction in the overall logic utilization without compromising the level of computational throughput significantly. But too many shares will add to the multiplexing and control overhead, leading to higher latency. Therefore, an optimization strategy for resource sharing is needed to decide which pieces of hardware to share and to find an appropriate balance between area, timing and throughput.

C. Area–performance trade-off:

The majority of reported works either address high performance architectures, which require a costly FPGA platform, or compact implementations with reduced function set, leaving little work to be done on a low-cost platform. Large number of lookup tables, flip-flops, block RAMs and other FPGA resources can be used in high throughput architectures through the use of extensive pipelining, loop unrolling and parallel

processing. However, the area of highly resource-constrained architectures can be sacrificed for higher latency and less throughput. This makes demands on an architecture that offers a real-world balance between the use of hardware resources and the performance of the cryptographic algorithms. A balanced application would be ideal for cost-sensitive embedded applications and resource-limited FPGA applications.

D. Limited Modularity:

In most architectures functional modules are tightly coupled and upgrades are challenging, and replacement of algorithms is impossible. For tightly coupled designs, if the control logic, data path, and communication interfaces change for one cryptographic function, then it's likely that changes will need to be made to the others. This lowers the design flexibility, and complicates the maintenance, debugging, and the future enhancement work. A modular architecture can separate the individual cryptographic functions from each other and offer standardized interfaces to those functions to facilitate inter-function communication. This would enable AES-256 and SHA-256 modules to be independently tested, modified, upgraded and replaced without an excessive amount of impact on the rest of the system.

E. Scalability :

Most applications are tested on one FPGA platform and only a few attempts are made to port the application across different FPGA families and/or important configurations. The resource usage, frequency and throughput on another FPGA family may be different for an optimized architecture of a specific FPGA device. Differences in the logic-cell architecture, memory resources, routing, and the types of hardware primitives can have a dramatic impact on implementation results. Futhermore, AES-256's key schedule and storage needs are greater than those of shorter key-length variants of AES, potentially impacting their scalability and storage needs. Thus, the development of a platform independent and parameterized architecture can enhance the portability and make it easier to implement the architecture on different FPGA devices.

F. Limited Verification and Security Evaluation:

The majority of studies validate their functionality by employing actual test vectors, and the deeper formal validation and sidechannel attacks are not discussed so extensively. Functional simulation can validate correct encryption and hashing, but not necessarily help ensure resistance to implementation vulnerabilities. Attacks can exploit the observed manifestations of power use, electromagnetic radiation, timing variations and switching activity of hardware implementations. Moreover, formal verification can be more reliable than simulation in establishing the correctness of the control logic, data-path actions and protocol. So it is recommended that future implementation should combine the systematic verification techniques such as the standard test vectors, simulation-based validation, synthesis level verification and, if applicable, security level evaluation techniques.

6 Proposed Research Direction

Based on the identified research gaps, this survey highlights the need for a modular FPGA architecture integrating AES-256 and SHA-256 under a shared hardware framework. The proposed approach is conceptual and does not include implementation or experimental validation.

As illustrated in Fig. 2, plaintext is first encrypted using AES-256, after which the generated ciphertext is processed by the SHA-256 module to produce a message digest. The ciphertext and hash are transmitted together, allowing the receiver to verify data integrity before decryption.

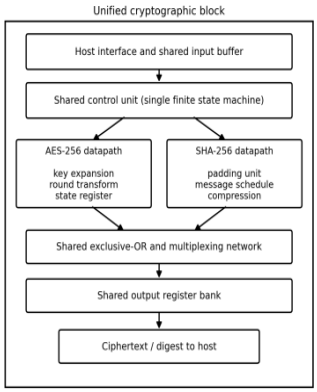

Fig. 3. Conceptual organisation of the proposed unified block, showing the resources shared between the two datapaths.

Figure 3 presents the conceptual hardware organization. The architecture shares common resources such as the host interface, controller, input/output buffers, and XOR logic while maintaining independent AES-256 encryption and SHA-256 hashing modules. This modular organization aims to improve resource utilization without affecting the functional independence of either algorithm.

The proposed architecture primarily focuses on three objectives:

- Efficient resource utilization through shared hardware components.
- Reduced control complexity using a unified controller for sequential operation.
- Improved modularity, enabling independent modification or replacement of the encryption and hashing modules.

An iterative datapath is the natural starting point. It bounds peak throughput at one round per cycle, well below the pipelined figures in Table I, but produces a footprint that is small, predictable and reproducible on a device a product team would actually specify. Where the protocol permits, padding and message-schedule stages can be decoupled from compression by the buffering technique of Liu and co-workers [5]. Four

questions define the intended scope of the work: how much shared logic can be reused before multiplexing overhead cancels the saving; whether the shared exclusive-OR network lies on the critical path of either primitive; how the design behaves when retargeted across device families; and whether resource sharing widens or narrows the side-channel surface relative to two isolated cores.

7 Discussion

In the era of embedded and IoT systems, the need for secure communication has grown, as has the importance of hardware-based cryptography. The use of FPGA technology allows to implement cryptographic algorithms like AES-256 or SHA-256, offering a desirable balance between performance, flexibility, and security. Existing studies mostly consider either increasing throughput or decreasing hardware resources; few consider efficient integration of both encryption and hashing in an integrated architecture. The emphasis in future cryptographic systems is likely to be on modularity, scalability and the ability to adapt to new security needs like authenticated encryption and new cryptographic standards.

8 Conclusion

This survey covered the recent FPGA implementations of AES and SHA-2, focusing on architectural design, optimization strategies and integrated cryptographic systems. Previous research has shown that iterative, pipelined, and parallel are effective architectures to achieve better throughput and resource utilization. In most implementations,

however, optimizations are applied to encryption and hash separately, and only some attempts are made to opt for resource sharing methods that involve both AES-256 and SHA-256 in a single framework of hardware.

As noted in the literature, modular architectures that demonstrate performance, resource usage, and scalability on an affordable FPGA platform are needed. Considering the identified research gaps, it is suggested that the integration of AES-256 and SHA-256 using shared hardware resources is a promising research direction to be explored in future for FPGA based cryptographic systems without much sacrificing flexibility and efficient utilization of hardware resources.

REFERENCES

1. National Institute of Standards and Technology, "Advanced Encryption Standard (AES)," FIPS Publication 197, Nov. 2001.
2. National Institute of Standards and Technology, "Secure Hash Standard (SHS)," FIPS Publication 180-4, Aug. 2015.
3. National Institute of Standards and Technology, "SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions," FIPS Publication 202, Aug. 2015.
4. H. Trang and N. V. Loi, "An efficient FPGA
5. implementation of the Advanced Encryption Standard
6. algorithm," in Proc. IEEE RIVF Int. Conf. Computing and Communication Technologies, 2012, pp. 1–4.
7. X. Liu, X. Zhang, and Y. Yang, "Efficient hardware implementation of hash algorithm SHA-3 for IoT information security," in Proc. 5th Int. Seminar on Artificial Intelligence, Networking and Information Technology (AINIT), 2024, pp. 368–372.
8. A. Arshad, K. Aslam, D. Kundi, and A. Aziz, "FPGA implementation of Advanced Encryption Standard using Xilinx System Generator," Asian J. Applied Sciences, vol. 2, no. 2, Apr. 2014.
9. H. Kouzehgar and M. N. Moghadam, "A high data rate pipelined architecture of AES encryption/decryption in storage area networks," IEEE, 2018.
10. A. Satoh, S. Morioka, K. Takano, and S. Munetoh, "A compact Rijndael hardware architecture with S-box optimization," in Advances in Cryptology – ASIACRYPT 2001, LNCS vol. 2248, Springer, 2001, pp. 239–254.
11. D. Canright, "A very compact S-box for AES," in Cryptographic Hardware and Embedded Systems – CHES 2005, LNCS vol. 3659, Springer, 2005, pp. 441–455.
12. H. Zodpe and A. Sapkal, "An efficient AES implementation using FPGA with enhanced security features," J. King Saud Univ. – Engineering Sciences, 2018.
13. F. Kahri, H. Mestiri, B. Bouallegue, and M. Machhout, "Efficient FPGA hardware implementation of secure hash function SHA-256/Blake-256," in Proc. 12th Int. Multi-Conf. Systems, Signals and Devices (SSD), 2015.
14. R. Chaves, G. Kuzmanov, L. Sousa, and S. Vassiliadis, "Improving SHA-2 hardware implementations," in Cryptographic Hardware and Embedded Systems – CHES 2006, LNCS vol. 4249, Springer, 2006, pp. 298–310.
15. A. Rajan, D. Joseph, et al., "Design and analysis of SHA-256 hash function using FPGA for secure data processing," in Proc. IEEE Int. Conf. Smart Computing and Communication (ICSCC), 2025.
16. H. L. Pham, T. H. Tran, V. T. D. Le, and Y. Nakashima, "A high-efficiency FPGA-based multimode SHA-2 accelerator," IEEE Access, vol. 10, pp. 11830–11845, 2022.
17. A. H. Jasim, D. A. Hammood, and A. Al-Askery, "Design and implementation of AES-SHA security hardware using FPGA," in Proc. 1st Int. Conf. Advances in Electrical, Electronics and Computational Intelligence (ICAEECI), 2023.
18. S. Al-Shara'a, R. K. Ibraheem, and O. Bayat, "Implementation of cryptanalysis based on FPGA hardware using AES with SHA-1," in Proc. Int. Conf. Smart Applications, Communications and Networking (SmartNets), Sharm El Sheikh, Egypt, 2019, pp. 1–7.
19. M. M. Saqlain and S. Ramachandran, "Highly secure AES with key generation using SHA-256 hash function," in Proc. IEEE Int. Conf. Communication and Signal Processing (ICCSP), 2025.
20. M. M. Sravani and S. A. Durai, "Bio-hash secured hardware e-health record system," IEEE Trans. Biomed. Circuits Syst., vol. 17, no. 3, pp. 420–432, Jun. 2023.
21. D.-e.-S. Kundi, A. Khalid, A. Aziz, C. Wang, M. O'Neill, and W. Liu, "Resource-shared crypto-coprocessor of AES Enc/Dec with SHA-3," IEEE Trans. Circuits Syst. I: Reg. Papers, vol. 67, no. 12, pp. 4869–4882, Dec. 2020.
22. S. Sridevi Sathya Priya and P. Karthigaikumar, "High throughput AES algorithm using parallel SubBytes and MixColumn," Wireless Personal Communications, Springer, 2017.

23. O. Simola, A. Korsman, V. Hirvonen, A. Tarkka, J. Helander, K. Järvinen, M. Kosunen, and J. Ryynänen, “RISC-V core with AES-256 accelerator,” in Proc. 31st IEEE Int. Conf. Electronics, Circuits and Systems (ICECS), 2024.
24. N. A. Fauziah, E. H. Rachmawanto, D. R. I. M. Setiadi, and C. A. Sari, “Design and implementation of AES and SHA-256 cryptography for securing multimedia file over Android chat application,” in Proc. Int. Seminar on Research of Information Technology and Intelligent Systems (ISRITI), Yogyakarta, Indonesia, 2018, pp. 146–151.
25. F. Kahri, B. Bouallegue, M. Machhout, and R. Tourki, “An FPGA implementation and comparison of the SHA-256 and Blake-256,” in Proc. 14th Int. Conf. Sciences and Techniques of Automatic Control and Computer Engineering (STA), Sousse, Tunisia, Dec. 2013, pp. 152–157.
26. S. M. Umar Talha, M. Asif, H. Hussain, A. Asghar, and H. Ameen, “Efficient Advanced Encryption Standard (AES) implementation on FPGA using Xilinx System Generator,” IEEE, 2016.
27. P. Kocher, J. Jaffe, and B. Jun, “Differential power analysis,” in Advances in Cryptology – CRYPTO ’99, LNCS vol. 1666, Springer, 1999, pp. 388–397.
28. L. K. Grover, “A fast quantum mechanical algorithm for database search,” in Proc. 28th Annual ACM Symposium on Theory of Computing (STOC), 1996, pp. 212–219.
29. M. Dworkin, “Recommendation for block cipher modes of operation: Galois/Counter Mode (GCM) and GMAC,” NIST Special Publication 800-38D, Nov. 2007.
30. S. Mangard, E. Oswald, and T. Popp, Power Analysis Attacks: Revealing the Secrets of Smart Cards. New York, NY,