SSRG International Journal of Electronics and Communication Engineering
ISSN: 2348-8549/ https://doi.org/10.14445/23488549/IJECE-V13I4P125

Volume 13 Issue 4, 297-306, April 2026

Original Article

High Throughput Lightweight SPECK Block Cipher Implementation on FPGA Platform for IoT Devices

Satish Kumar B[1,4], Panduranga Rao Malode V[2], G. Ezhilarasan[3]

[1]*Department of Electronics and Communications, JAIN Deemed to be University, Bangalore, India.*
[2]*Department of Computer Science Engineering, FET- JAIN Deemed to be University, Bangalore, India.*
[3]*Department of Electrical and Electronics Engineering, FET- JAIN Deemed to be University, Bangalore, India.*
[4]*Department of ECE, K S Institute of Technology, Bangalore, Affiliated to Visvesvaraya Technological University, Belagavi, India.*

[1]*Corresponding Author : satishkbr1984@gmail.com*

Received: 21 January 2026 Revised: 21 February 2026 Accepted: 23 March 2026 Published: 30 April 2026

Abstract - *IoT devices with limited resources, in which area, power, and throughput must be meticulously balanced, are crucially protected by lightweight cryptography. In this study, a high-throughput FPGA implementation of the lightweight SPECK block cipher that is tailored for Internet of Things applications is presented. On an Artix-7 FPGA platform, the architecture is synthesized using Xilinx ISE Design Suite 14.7 and specified in Verilog-HDL. The SPECK-32/64, SPECK-64/128, SPECK-128/128, and SPECK-128/256 configurations are used to test scalability at different security levels. The suggested design makes use of an ARX-based datapath that has been optimized and employs effective round-key scheduling to get a high operating frequency with minimal hardware overhead. Results from experiments show that throughput can reach 1.064 Gbps and an ultimate frequency of 404.269 MHz, all while using little power (<110 mW). With a hardware efficiency of up to 3.563 Mbps/Slice, the architecture greatly outperforms current SPECK and other lightweight encryption implementations. The findings verify that the suggested architecture offers an efficient trade-off between performance, area usage, and security strength, making it ideal for FPGA cryptographic accelerators with an IoT focus.*

Keywords - *SPECK Block cipher, Lightweight, FPGA, IoT, Throughput.*

1. Introduction

A dispersed network of embedded sensors that work together to collect, process, and send data for various real-time applications is known as the Internet of Things (IoT). Climate inspection, smart automation, healthcare oversight, security facilities, and smart service delivery are a few examples of representative uses. IoT platforms are more vulnerable to security threats such as denial-of-service-induced impairment, compromised authentication, unauthorized usage, and confidentiality leaks as connectivity rises [1]. Therefore, it is essential to guarantee reliable data security in low-power embedded technology. Direct adoption of traditional cryptographic algorithms is, however, hampered by stringent limitations on processing power, on-chip memory, hardware difficulty, and battery life. As a result, efficient protection techniques need to provide dependable operational efficiency, low overhead, and low energy consumption [2]. Block cipher primitives are frequently used for security provision in restricted contexts, including RFID tags, BAN devices, WSN nodes, and IoT endpoints. These primitives are separated into conventional and lightweight versions based on key configuration, block size, and structural difficulty. The incompatibility of traditional approaches with devices with low resources is the main factor driving the development of lightweight cryptography. Approaches for optimization and embedded realization may concurrently increase development overhead and architectural resource use, even though they decrease execution time, switching process, and computational demands [3, 4].

Customized hardware acceleration or software execution can be used to implement cryptographic primitives for limited platforms. Energy efficiency, power consumption, latency, throughput efficiency, area usage, and composite performance measures like FOM are commonly used to evaluate hardware-oriented realizations that use ASICs, FPGAs, or bespoke VLSI architectures [5, 6]. In terms of structure, block cipher techniques use paradigms such as Lai–Massey schemes, Feistel alterations, substitution–permutation mapping, and nonlinear shift-register procedures [7, 8]. Particularly, FPGA technology offers parallel datapath execution and configurability, which speeds up cryptographic processing [9]. Notwithstanding these benefits, encryption focused on the IoTs needs to maintain its lightweight nature. For reliable and efficient IoT applications, this study emphasizes the FPGA implementation of the SPECK encryption family and examines its performance metrics.

1.1. Problem Statement

Secure communication methods that function under stringent area, power, and compute capacity limitations are required by IoT devices. Utilizing FPGA-based embedded platforms, traditional cryptography methods frequently result in inadequate throughput or significant hardware overhead. Existing FPGA implementations of SPECK, despite its lightweight ARX cipher design, sometimes suffer from inadequate throughput–area trade-offs, inefficient usage of resources, or limited scalability across configurations. In order to facilitate scalable and secure Internet of Things deployments, an optimized SPECK FPGA architecture is required, especially on contemporary low-cost FPGA platforms like Artix-7, which provides high throughput while optimizing slice usage and power usage.

1.2. Contributions

A high-throughput and resource-efficient FPGA architecture for the SPECK lightweight block cipher, aimed at Internet of Things applications, is proposed in this paper. To lower combinational latency and enhance timing performance, an improved round-based datapath incorporating effective ARX operations is first created. Second, with just minimal architectural changes, a scalable key extension method is created to accommodate various SPECK configurations (32/64, 64/128, 128/128, and 128/256). Third, the implementation is verified on an Artix-7 FPGA, showing notable gains in hardware efficiency and slice usage over previous SPECK implementations. Lastly, superior area–throughput trade-offs are established through thorough comparison with other lightweight block ciphers like LED, PRESENT, and XTEA. The suggested architecture is appropriate for secure IoT accelerators since it maintains low power usage while achieving gigabit-class speed. The organization of the paper is as follows: Section 2 explains the review of literature concerning the existing ciphers' works. Section 3 provides a detailed description of the hardware architecture for the proposed SPECK Block cipher with pipelining features. Section 4 provides the analysis and performance comparison to make the suggested work results feasible. Finally, Section 5 provides the improvement that brings the overall work to a conclusion.

2. Literature Review

Lightweight cryptographic methods designed for limited platforms have been the subject of extensive research, with a focus on operational scalability, statistical security, and architectural simplicity. Employing AES-derived MCM optimization, Naik et al. [10] introduced a hybrid multi-round LED cipher that supports variable key lengths and allows single-cycle execution to increase throughput while lowering circuit overhead. Mutual-coupled and counter-assisted chaotic maps, which attained strong statistical unpredictability under NIST SP800-22 validation, were used by Lee and Wu [11] to improve chaotic cryptography modeling. Prathap and Sathiyanarayanan [12] looked into FPGA-assisted recognizing attacks in Blowfish, where a bespoke decision-tree equality checker allowed for quick fault discovery with low latency. Serial pipelining and threshold construction were used by Liu and Tang [13] to optimize the uBlock encryption, greatly reducing the amount of hardware resources used in protected designs. Future developments in device innovations, application security, and comparative evaluation are covered in more detail. A lightweight hash structure that is compatible with parallel processing was proposed by Sevin et al. [14], resulting in efficient calculation with the necessary security considerations.

Al-Nofaie et al. [15] reviewed new lightweight block cipher advancements for Internet of Things security, providing a thorough analysis of design vulnerabilities, resilience, and efficiency. While HC-256 is not appropriate for processing huge amounts of data, performance evaluation by Sorescu et al. [16] showed that XChaCha20, Salsa20, and ASCON32 provide superior throughput in limited IoT applications. By introducing a two-stage medical picture encryption and authentication technique, Alsaraireh et al. [17] showed enhanced resilience to attacks in the field of medical facilities security. Penumalli et al. [18] investigated Feistel-structured SLIM ciphers using low-voltage NCFET-based S-box realization, demonstrating benefits over CMOS and SPN-based PRESENT versions. Through algorithmic hybridization, Anton et al. [19] achieved effective ARX-based lightweight encryption by combining SIMECK and TEA with dynamic substitution. Furthermore, a modified SPECK key-schedule with verified statistical randomness was suggested by Mohanapriya and Kumar [20], enhancing protection for lightweight data encryption.

A consistent optimization of throughput, area, and adaptability for FPGA-based IoT systems is not achieved by current lightweight cryptographic designs, which mainly concentrate on either resource reduction or security improvement. While chaotic and hybrid approaches add extra complexity that is inappropriate for effective FPGA realization, approaches like LED optimization and uBlock lower hardware costs but are unable to achieve high throughput. Although stream ciphers have high throughput, their hardware architectures are not scalable or structured. Moreover, previous SPECK-related research mostly focuses on key-schedule enhancements without offering a comprehensive high-performance FPGA design or multi-configuration analysis. Additionally, real implementation insights on contemporary FPGA platforms are lacking in comparative studies. Therefore, a scalable and high-throughput SPECK FPGA design that guarantees effective area usage while satisfying IoT performance requirements is required.

2.1. Research Gaps

A consistent optimization of throughput, area, and adaptability for FPGA-based IoT systems is not achieved by

current lightweight cryptographic designs, which mainly concentrate on either resource reduction or security improvement. While chaotic and hybrid approaches add extra complexity that is inappropriate for effective FPGA realization, approaches like LED optimization and uBlock lower hardware costs but are unable to achieve high throughput. Although stream ciphers have high throughput, their hardware architectures are not scalable or structured.

Moreover, previous SPECK-related research mostly focuses on key-schedule enhancements without offering a comprehensive high-performance FPGA design or multi-configuration analysis. Additionally, real implementation insights on contemporary FPGA platforms are lacking in comparative studies. Therefore, a scalable and high-throughput SPECK FPGA design that guarantees effective area usage while satisfying IoT performance requirements is required.

3. Method

The lightweight ARX (Addition–Rotation–XOR) block ciphers in the SPECK [21] family are made to be highly effective in contexts with limited resources, like embedded CPUs, FPGA platforms, and Internet of Things nodes. Because SPECK only uses modular addition, circular rotations, and bitwise XOR operations—as opposed to substitution–permutation-based ciphers—it is very well suited for hardware implementation because of its low combinational depth and simplified logic. The cipher balances security, area, and throughput by operating on two n-bit words that constitute a $2n$-bit block. It supports various block/key size combinations. Scalable FPGA implementations, from extensively pipelined high-throughput designs to area-optimized iterative cores, are made possible by its structural simplicity.

The ten standardized SPECK versions are listed in Table 1 together with their block size ($2n$), key size (mn), word size (n), number of key words (m), rotation constants (α and β), and total rounds (T). Block sizes between 32 and 128 bits are supported by the cipher, and matching key sizes between 64 and 256 bits are also supported. Rotation constants for $n = 16$ (SPECK-32/64) are $\alpha = 7$ and $\beta = 2$, while they are $\alpha = 8$ and $\beta = 3$ for all other configurations. The number of rounds ranges from 22 to 34, increasing with block/key size. This directly affects pipeline depth in fully unrolled implementations and Latency in iterative systems from the standpoint of FPGA design.

Since only ARX primitives are employed, larger word sizes increase datapath width, although they fail to appreciably increase logic complexity. The SPECK cipher used in this work features a 64-bit key and a 32-bit block size, with possible block and key size combinations of 64/128, 128/128, and 128/256.

Table 1. SPECK cipher configurations

Block Size (2n)	Key Size (mn)	Word Size (n)	Key Words (m)	Rot α	Rot β	Rounds (T)
32	64	16	4	7	2	22
48	72	24	3	8	3	22
48	96	24	4	8	3	23
64	96	32	3	8	3	26
64	128	32	4	8	3	27
96	96	48	2	8	3	28
96	144	48	3	8	3	29
128	128	64	2	8	3	32
128	192	64	3	8	3	33
128	256	64	4	8	3	34

The SPECK block cipher's top-level hardware architecture is shown in Figure 1. The Key Generation Block, the Round Block, and the Control Block are the three main modules that make up the architecture. The key expansion mechanism is used by the Key Generation Block to successively calculate round subkeys. After receiving the master key, it uses the round counter to create a subkey every clock cycle. The Round Block receives the created subkey. The Round Block uses the current subkey to carry out the encryption round transformation. Up until the final ciphertext is generated, it iteratively modifies the internal state for each round after accepting plaintext input via a data register.

The entire operation is coordinated by the Control Block. It has control registers, a round counter, and a Finite State Machine (FSM). The FSM controls round advancement, permits register loading, controls mode selection (encryption/decryption), and indicates completion. This modular partitioning facilitates scalable FPGA generation and enhances timing closure. Because of its predictable datapath layout and low control overhead, the architecture is especially well-suited for IoT applications.

The pipelined SPECK-32/64 implementation is shown in Figure 2. Multiple plaintext blocks can be processed concurrently in this design since each round stage is divided by registers. To ensure synchronization, subkey formation and round operation are synchronized step-by-step. While the right pipeline completes encryption rounds, the left pipeline handles subkey generation. Subkey k_i is created at clock cycle i and is used right away by Round R_i. The architecture produces one ciphertext output per clock cycle following pipeline filling. Throughput is greatly increased by this approach in contrast to the iterative architecture. Logic duplication proportionate to the number of rounds T and an increased register count are the trade-offs. Because of the ARX-based lightweight construction, this pipeline solution maximizes data rate while keeping modest space for FPGA platforms aimed at high-speed IoT gateways.

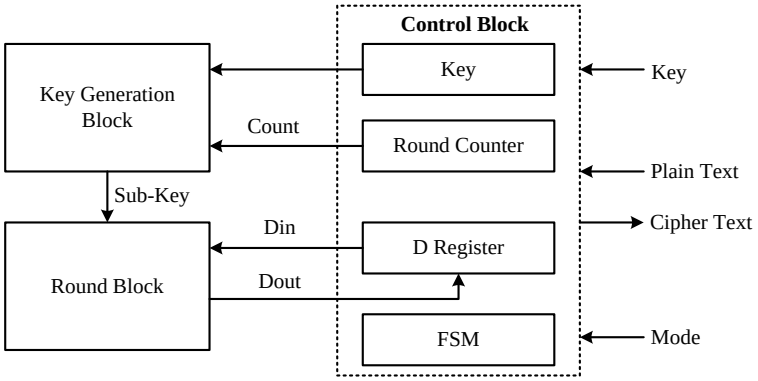


Fig. 1 Hardware architecture of the SPECK block cipher

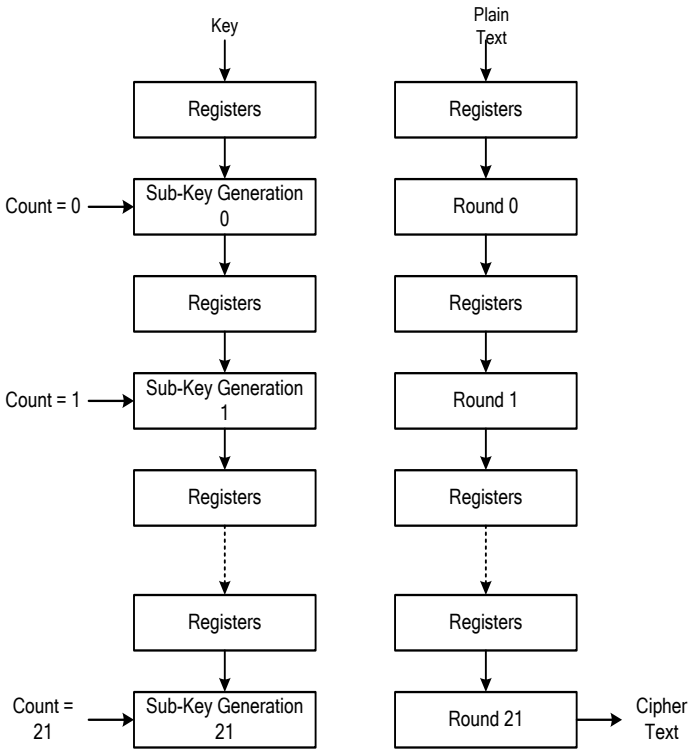
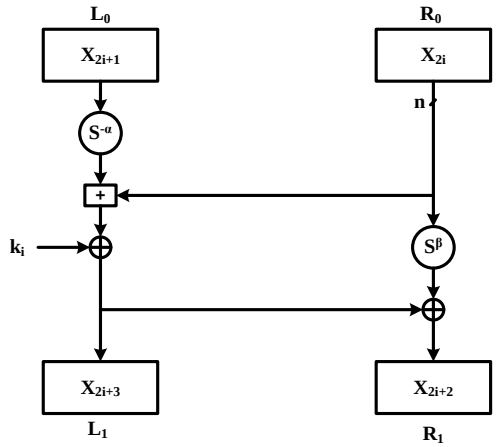


Fig. 2 Pipeline Architecture of SPECK-32/64 Cipher

3.1. Round Function

The SPECK submodules are shown in Figure 3. The internal round function design is shown in Figure 3(a), and its breakdown into Feistel-like operations is shown in Figure 3(b). The two n-bit words X and Y are used by the SPECK $2n$ encryption map. Equation (1) defines the key-dependent round function for $k \in GF(2)^n$.

The round function uses the following operations: Bitwise XOR ($\oplus$), Addition modulo 2^n, Circular right rotation $S^{-\alpha}$, and Circular left rotation S^{β}. X is first rotated to the right by α, then modularly added with Y, and finally XORed with the round key k. The second output XORs the revised first word with Y after rotating it to the left by β.



(a)

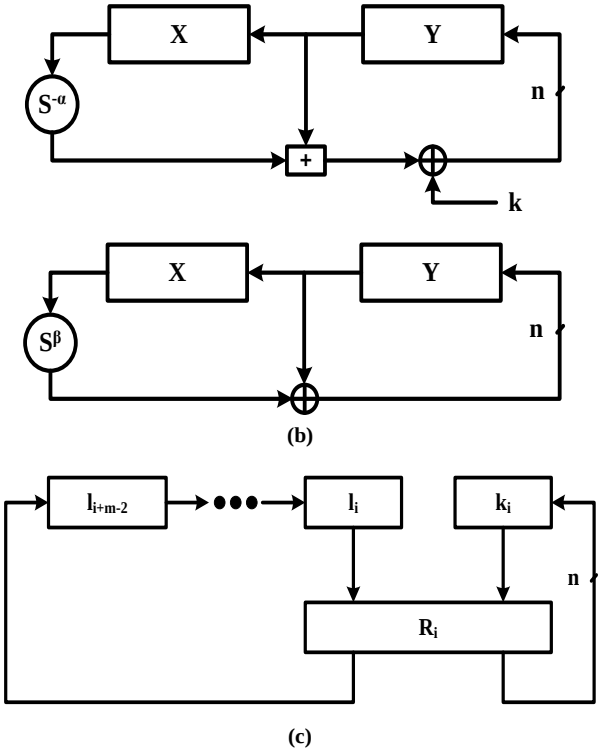


Fig. 3 SPECK Sub modules: (a) Round Function, (b) Feistel Steps Decomposition in Round Function, (c) Key Expansions

Strong diffusion and confusion are guaranteed within a small hardware depth thanks to this dual ARX mixing. The inverse round function required for decryption is given in Equation (2). In this case, modular subtraction takes the place of modular addition, and inverse rotations bring the alignment back. Shared rotation and XOR blocks with little extra subtraction logic make hardware reuse possible. Equation (3) illustrates that the round can be broken down into two Feistel-like mappings, as seen in Figure 3(b). A representation of this breakdown can be found in Figure 3(b). Architectural improvements like resource sharing and balanced pipeline insertion are made possible by this decomposition, which also makes the structural symmetry more understandable.

$$F(X,Y,k) = ((S^{-\alpha}X + Y) \oplus \mathrm{k}, S^{\beta}Y \oplus (S^{-\alpha}X + Y) \oplus \mathrm{k}) \quad (1)$$

$$F^{-1}(X,Y,k) = (S^{\alpha}((X \oplus k) - S^{-\beta}(\mathrm{X} \oplus \mathrm{Y})), S^{-\beta}(X \oplus Y)) \quad (2)$$

$$(x,y) \mapsto (y, (S^{-\alpha}x + y) \oplus k) \text{ and } (x,y) \mapsto (y, S^{\beta}x \oplus y) \quad (3)$$

3.2. Key Expansion

The SPECK key schedule generates T round keys $k_0, k_1, \ldots, k_{T-1}$. The master key is represented as $K = (l_{m-2}, \ldots, l_0, k_0)$. where $l_i, k_0 \in GF(2)^n$ and $m \in \{2,3,4\}$. The recursive key generation is defined in Equation (4). Key schedule nonlinearity and round individuality are guaranteed by injecting the round index via XOR. This growth process is depicted in Figure 3(c). As the round function logic creates new keywords, the l_i registers move progressively. The key schedule for FPGA implementation can be either precomputed and stored (pipeline design) or on-the-fly (iterative architecture). Hardware reuse is extremely effective since the key schedule makes use of the same ARX primitives as the encryption round, which lowers additional area overhead.

$$\ell_{i+m-1} = (k_i + S^{-\alpha}\ell_i) \oplus i \text{ and } k_{i+1} = S^{\beta}k_i \oplus \ell_{i+m-1} \quad (4)$$

4. Results and Discussion

Xilinx ISE Design Suite 14.7 was used to synthesize the suggested high-throughput SPECK lightweight block cipher architecture, which was designed in Verilog-HDL. In order to assess realistic hardware performance under IoT usage constraints, implementation was done on an Xilinx Artix-7 FPGA. To explore scalability over different block and key sizes, many configurations were put into practice: SPECK-32/64, SPECK-64/128, SPECK-128/128, and SPECK-128/256. Hardware resource usage (slices, LUTs, flip-flops), maximum operating frequency (Fmax), power usage, latency (clock cycles), throughput (Mbps), and hardware efficiency (Mbps/Slice) are among the evaluation criteria. This analysis's goal is to measure the trade-off between hardware overhead, operational throughput, and security strength in FPGA-based IoT cryptographic accelerators.

4.1. Synthesis and Performance Results

The suggested SPECK designs' hardware resource consumption on the Artix-7 FPGA is shown in Table 2. Resource usage increases proportionately with block size and number of rounds, as anticipated. With 165 slices, 241 LUTs, and 123 flip-flops, the SPECK-32/64 configuration shows a very compact implementation appropriate for IoT nodes with limited resources. Because of the ARX-based architecture, the maximum operational frequency can reach 404.269 MHz, showing a short combinational critical path.

For SPECK-64/128, the higher round count (27 rounds) and datapath width (32-bit words) result in a slight increase in resource utilization to 276 slices and 383 LUTs. Despite decreasing to 245.966 MHz, Fmax is still high enough for high-speed embedded communication applications. The representation of the Resource Usage of the Proposed SPECK cipher on the Artix-7 FPGA is illustrated in Figure 4.

The hardware cost of the SPECK-128 variations continues to scale. 535 slices and 792 LUTs are consumed by SPECK-128/256 compared to 405 slices and 660 LUTs by SPECK-128/128. The increase is mostly ascribed to longer round counts (up to 34 rounds) and larger word sizes (64-bit datapath). Effective timing closure is confirmed even for large data paths, as the operational frequency stays constant at 282.686 MHz despite the additional complexity.

Table 2. Resource Usage of Proposed SPECK cipher on Artix-7 FPGA

Parameters	SPECK- 32/64	SPECK-64/128	SPECK-128/128	SPECK-128/256
Block size	32	64	128	128
Key size	64	128	128	256
Rounds	22	27	32	34
Slices	165	276	405	535
LUTs	241	383	660	792
FFs	123	211	277	407
Fmax (MHz)	404.269	245.966	282.686	282.686
Dynamic Power (mW)	9	15	22	25
Total Power (mW)	91	97	104	107

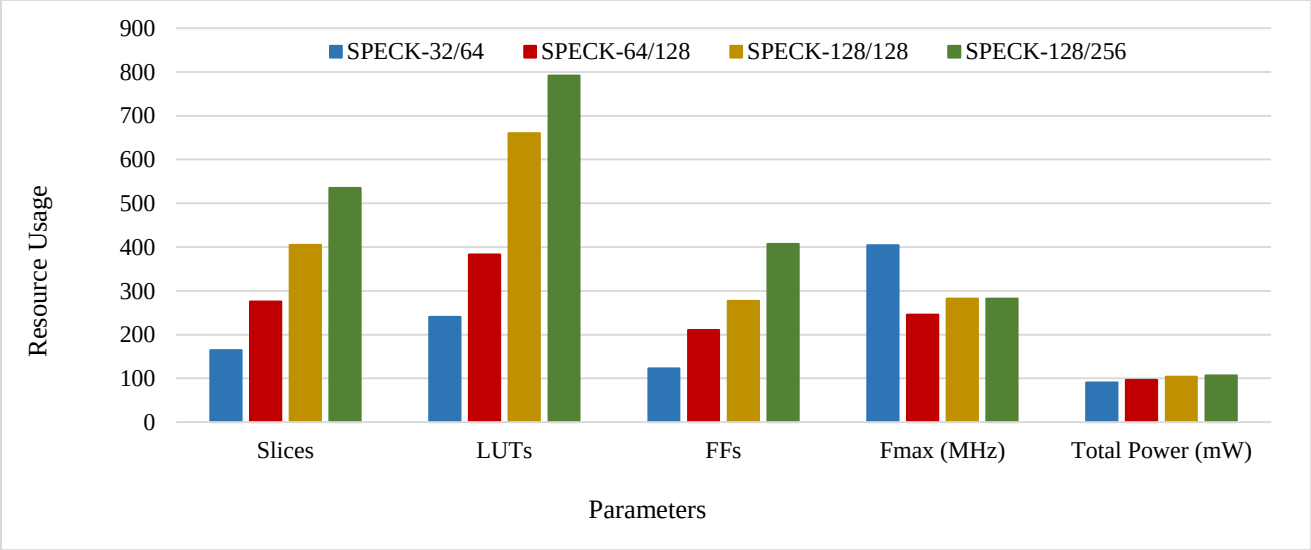


Fig. 4 Representation of Resource Usage of Proposed SPECK cipher on Artix-7 FPGA

Table 3. Performance analysis of the Proposed SPECK cipher on Artix-7 FPGA

Parameters	SPECK-32/64	SPECK-64/128	SPECK-128/128	SPECK-128/256
Latency (CC)	22	27	32	34
Throughput (Mbps)	588.027	583.03	1,130.75	1,064.23
Efficiency (Mbps/Slice)	3.563	2.112	2.791	1.989

The progressive increase in dynamic power usage from 9 mW (32/64) to 25 mW (128/256) is anticipated as a result of increased logic utilization and switching activity. All configurations maintain a total power below 110 mW, confirming their appropriateness for low-power IoT applications. Overall, Table 2 shows that the suggested architecture maintains high clock frequencies because of its ARX-centric layout while exhibiting linear scalability with anticipated hardware increase.

The hardware efficiency, throughput, and latency for the realized configurations are summarized in Table 3. Because the architecture has a round-based pipelined structure, latency is directly correlated with the number of rounds. SPECK-128/256 requires 34 clock cycles, but SPECK-32/64 needs 22. Throughput is still high even when larger variations have higher Latency. SPECK-32/64's extremely high operating frequency allows it to reach 588.027 Mbps. By delivering 583.03 Mbps, SPECK-64/128 shows that a wider datapath makes up for a lower frequency. Most notably, SPECK-128/128 surpasses 1 Gbps speed with 1,130.75 Mbps and SPECK-128/256 with 1,064.23 Mbps. Processing 128-bit blocks in each encryption cycle results in this notable gain.

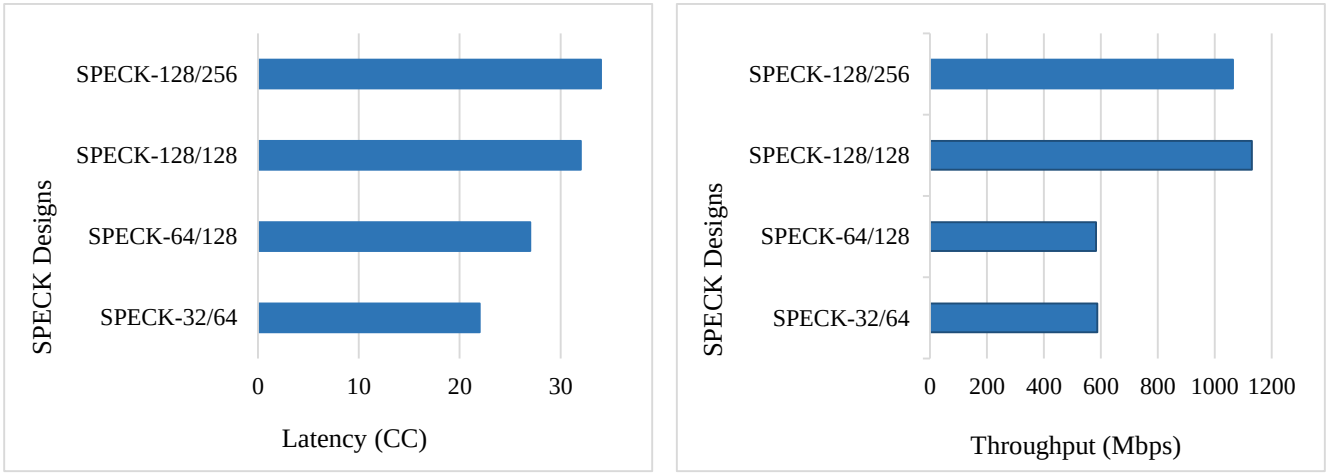
Area-throughput optimization is highlighted by hardware efficiency, which is expressed in Mbps/Slice. With the greatest efficiency of 3.563 Mbps/Slice, SPECK-32/64 demonstrates exceptional compactness. With a balanced trade-off between throughput and resource usage, SPECK-128/128 achieves 2.791 Mbps/Slice. SPECK-128/256 has

somewhat worse efficiency (1.989 Mbps/Slice) as a result of higher round count and key schedule difficulty. These findings validate that while bigger block sizes enhance absolute throughput, smaller combinations maximize area efficiency, making them appropriate for gateway-level IoT applications. The performance metrics representation of the SPECK Cipher is illustrated in Figure 5. The Latency (CCs) and Throughput (Mbps) analysis over various SPECK configurations are shown in Figures 5(a) and (b), respectively.

4.2. Performance Comparison

The suggested design is compared with previous SPECK implementations described in [20-22] in Table 4. The suggested design shows notable gains in throughput and area efficiency across all configurations. The suggested work produces 548.08 Mbps using just 277 slices, increasing efficiency to 1.978 Mbps/Slice, while [21] needs 2014 slices with 266.66 Mbps throughput (0.132 Mbps/Slice) for the 64/128 setup. Similarly, for 128/128, throughput rises from 444.33 Mbps to 810.88 Mbps while slice utilization decreases from 3290 in [21] to 405 in the suggested design. Efficiency increases dramatically from 0.135 to 2.002 Mbps/Slice. The suggested version utilizes only 535 slices and achieves 763.19 Mbps, while [21] uses 5159 slices with 401.777 Mbps throughput in the 128/256 instance. The suggested Artix-7 design outperforms [20, 22] in terms of frequency and efficiency, reaching 588.027 Mbps with 165 slices for 32/64. Overall, the suggested architecture confirms better FPGA mapping and architectural optimization by achieving faster throughput with significantly lower slice use.

The suggested SPECK implementation is compared with LED [23, 25], PRESENT [24, 28], and XTEA [26] in Table 5. The suggested SPECK-64/128 outperforms PRESENT [24] (99.13 Mbps, 0.493 Mbps/Slice) and LED [23] (83.76 Mbps, 0.54 Mbps/Slice) with an efficiency of 2.112 Mbps/Slice and 583.03 Mbps.



(a) **(b)**

Fig. 5 Performance metrics representation of SPECK Cipher, including (a) Latency (CCs) and (b) Throughput (Mbps)

Table 4. Performance comparison of the Proposed SPECK cipher with existing SPECK ciphers

Block Ciphers	Data size	Slices	LUTs	Frequency (MHz)	Power (mW)	Latency (CC)	Throughput (Mbps)	Efficiency (Mbps/ Slice)	FPGAs
SPECK [21]	64/128	2014	NA	125	142	30	266.66	0.132	Spartan-6
This work	64/128	277	383	231.224	60	27	548.08	1.978	Spartan-6
SPECK [21]	128/128	3290	NA	118	162	34	444.23	0.135	Spartan-6
This work	128/128	405	608	202.72	72	32	810.88	2.002	Spartan-6
SPECK [21]	128/256	5159	NA	113	171	36	401.777	0.077	Spartan-6
This work	128/256	535	740	202.724	80	34	763.19	1.426	Spartan-6
SPECK [22]	32/64	427	476	142	121	24	189.33	0.443	Spartan-3
This work	32/64	236	445	143	100	22	208	0.881	Spartan-3
M-SPECK [20]	32/64	NA	240	154	175	32	154	0.64	Artix-7
This work	32/64	165	241	404.269	91	22	588.027	3.563	Artix-7

The suggested architecture maintains improved hardware efficiency when compared to LED [25] and PRESENT [28], which offer higher raw throughput but at much higher slice cost. For instance, the suggested design produces 583.03 Mbps with just 276 slices (2.112 Mbps/Slice), whereas PRESENT [28] obtains 822 Mbps with 711 slices (1.15 Mbps/Slice). The suggested SPECK exhibits significantly higher efficiency when compared to XTEA [26], which reaches 0.34 Mbps/Slice. The suggested design maintains balanced area utilization while surpassing 1 Gbps throughput for the 128/256 configuration. In summary, the suggested SPECK architecture is well-suited for high-performance IoT FPGA scenarios because it provides a better area–throughput trade-off when compared to other lightweight block ciphers and current SPECK implementations.

Table 5. Performance comparison of the Proposed SPECK cipher with existing block ciphers

Block size	Data/key size	Slices	Frequency (MHz)	Latency (CC)	Throughput (Mbps)	Efficiency (Mbps/Slice)	FPGA s
LED [23]	64/128	154	251.28	192	83.76	0.54	Spartan-6
This work	64/128	277	231.224	27	548.08	1.978	Spartan-6
PRESENT [24]	64/128	201	211	136	99.13	0.493	Virtex-6
This work	64/128	535	275.284	27	652.52	1.219	Virtex-6
LED [25]	64/128	404	357.974	27	848.53	2.11	Artix-7
XTEA [26]	64/128	316	264	128	80.43	0.34	Artix-7
LED [27]	64/128	168	333.73	48	444.97	2.65	Artix-7
PRESENT [28]	64/128	455	410	32	822	1.8	Artix-7
This work	64/128	276	245.966	27	583.03	2.112	Artix-7
PRESENT [28]	64/256	711	410	32	822	1.15	Artix-7
This work	128/256	535	282.686	34	1,064.23	1.989	Artix-7

5. Conclusion

A lightweight and high-performance FPGA design of the SPECK block cipher for IoT security applications is featured in the present paper. The suggested architecture provides high operating frequency and effective resource usage on an Artix-7 FPGA by making use of the ARX structure of SPECK and optimizing both the round function and key expansion modules. For 128-bit setups, experimental results show throughput surpassing 1 Gbps while preserving low slice utilization and power usage. Significant gains in hardware efficiency and area–throughput balance are confirmed by comparison with other lightweight block ciphers like LED and PRESENT, as well as with current SPECK implementations. Flexible deployment according to security and performance needs is made possible by the scalability across various block and key sizes. Overall, the suggested design effectively balances hardware overhead, operational performance, and security strength, making it a solid contender for secure FPGA-based edge gateways and IoT nodes.

Conflicts of Interest

We wish to confirm that no known conflicts of interest are associated with this publication.

Funding Statement

There has been no significant financial support for this work that could have influenced its outcome.

References

[1] Mickaël Cazorla, Kevin Marquet, and Marine Minier, "Survey and Benchmark of Lightweight Block Ciphers for Wireless Sensor Networks," *2013 International Conference on Security and Cryptography (SECRYPT)*, Reykjavik, Iceland, pp. 1-6, 2013. [Google Scholar] [Publisher Link]

[2] Bassam J. Mohd, Thaier Hayajneh, and Athanasios V. Vasilakos, "A Survey on Lightweight Block Ciphers for Low-Resource Devices: Comparative Study and Open Issues," *Journal of Network and Computer Applications*, vol. 58, pp. 73-93, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[3] Isha Bhardwaj, Ajay Kumar, and Manu Bansal, "A Review on Lightweight Cryptography Algorithms for Data Security and Authentication in IoTs," *2017 4th International Conference on Signal Processing, Computing and Control (ISPCC)*, Solan, India, pp. 504-509, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[4] Effy Raja Naru, Hemraj Saini, and Mukesh Sharma, "A Recent Review on Lightweight Cryptography in IoT," *2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, Palladam, India, pp. 887-890, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[5] George Hatzivasilis et al., "A Review of Lightweight Block Ciphers," *Journal of Cryptographic Engineering*, vol. 8, pp. 141-184, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[6] Susha Surendran, Amira Nassef, and Babak D. Beheshti, "A Survey of Cryptographic Algorithms for IoT Devices," *2018 IEEE Long Island Systems, Applications and Technology Conference (LISAT)*, Farmingdale, NY, USA, pp. 1-8, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[7] Abdullah Sevin, and Abdu Ahmed Osman Mohammed, "A Survey on Software Implementation of Lightweight Block Ciphers for IoT Devices," *Journal of Ambient Intelligence and Humanized Computing*, vol. 14, pp. 1801-1815, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[8] Nayancy, Sandip Dutta, and Soubhik Chakraborty, "A Survey on Implementation of Lightweight Block Ciphers for Resource Constraints Devices," *Journal of Discrete Mathematical Sciences and Cryptography*, vol. 25, no. 5, pp. 1377-1398, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[9] Hai Tao et al., "Secured Data Collection with Hardware-Based Ciphers for IoT-Based Healthcare," *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 410-420, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[10] Mahendra Shridhar Naik, Desai Karanam Sreekantha, and Kanduri V.S.S.S.S. Sairam, "An Efficient Low-Latency and High Throughput LED Cipher Architecture for IoT Security on a Hardware Platform," *SN Computer Science*, vol. 5, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[11] Tung-Tsun Lee, and Shyi-Tsong Wu, "A Lightweight Keystream Generator Based on Expanded Chaos with a Counter for Secure IoT, *Electronics*, vol. 13, no. 24, pp. 1-22, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[12] Joseph Anthony Prathap, and Mithileysh Sathiyanarayanan," Fault Detection in Blowfish Algorithm Using FPGA-Based Modified Decision Tree Approach," *IEEE Access*, vol. 13, pp. 90591-90600, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[13] Botao Liu, and Ming Tang, "A Very Compact and a Threshold Implementation of uBlock for Internet of Things," *Tsinghua Science and Technology*, vol. 30, no. 5, pp. 2270-2283, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[14] Abdullah Sevin, "Implementation of a Data-Parallel Approach on a Lightweight Hash Function for IoT Devices," *Mathematics*, vol. 13, no. 5, pp. 1-25, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[15] Safia Meteb Al-Nofaie, Sanaa Sharaf, and Rania Molla, "Design Trends and Comparative Analysis of Lightweight Block Ciphers for IoTs," *Applied Sciences*, vol. 15, no. 14, pp. 1-51, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[16] Tiberius-George Sorescu et al., "Comparative Performance Analysis of Lightweight Cryptographic Algorithms on Resource-Constrained IoT Platforms," *Sensors*, vol. 25, no. 18, pp. 1-17, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[17] Saleem Alsaraireh, Ashraf Ahmad, and Yousef AbuHour, "New Step in Lightweight Medical Image Encryption and Authenticity," *Mathematics*, vol. 13, no. 11, pp. 1-17, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[18] Koteswara Rao Penumalli, Venkateswarlu Gonuguntla, and Ramesh Vaddi, "An Energy Efficient and DPA Attack Resilient NCFET-Based S-Box Design for Secure and Lightweight SLIM Ciphers," *Electronics*, vol. 14, no. 6, pp. 1-19, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[19] Alin-Adrian Anton et al., "SIMECK-T: An Ultra-Lightweight Encryption Scheme for Resource-Constrained Devices," *Applied Sciences*, vol. 15, no. 3, pp. 1-34, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[20] R. Mohanapriya, and V. Nithish Kumar, "Modified SPECK (M-SPECK) Lightweight Cipher Architecture for Resource-Constrained Applications," *IEEE Access*, vol. 13, pp. 88993-89002, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[21] Ray Beaulieu et al., "The SIMON and SPECK Lightweight Block Ciphers," *Proceedings of the 52nd Annual Design Automation Conference*, San Francisco California, pp. 1-6, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[22] Ali Nemati et al., "A Low-cost and Flexible FPGA Implementation for SPECK Block Cipher," *2015 12th International Iranian Society of Cryptology Conference on Information Security and Cryptology (ISCISC)*, Rasht, Iran, pp. 42-47, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[23] Wajih El Hadj Youssef et al., "Hardware Implementation of Secure Lightweight Cryptographic Designs for IoT Applications," *Communication*, vol. 2020, pp. 1-13, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[24] Carlos Andres Lara-Nino, Arturo Diaz-Perez, and Miguel Morales-Sandoval, "Lightweight Hardware Architectures for the Present Cipher in FPGA," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 64, no. 9, pp. 2544-2555, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[25] S. P. Guruprasad, and B. S. Chandrasekar, "An Evaluation Framework for Security Algorithms Performance Realization on FPGA," *2018 IEEE International Conference on Current Trends in Advanced Computing (ICCTAC)*, Bangalore, India, pp. 1-6, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[26] R. Anusha, and V. Veena Devi Shastrimath, "LCBC-XTEA: High Throughput Lightweight Cryptographic Block Cipher Model for Low-Cost RFID Systems," *Conference Proceeding 8th Computer Science On-line Conference Cybernetics and Automation Control Theory Methods in Intelligent Algorithms*, pp. 185-196, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[27] Bahram Rashidi, "Flexible Structures of Lightweight Block Ciphers PRESENT, SIMON and LED," *IET Circuits, Devices & System*, vol. 14, no. 3, pp. 369-380, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[28] R. Bharathi, and N. Parvatham, "Light-Weight Present Block Cipher Model for IoT Security on FPGA," *Intelligent Automation & Soft Computing*, vol. 33, no. 1, pp. 35-49, 2022. [CrossRef] [Google Scholar] [Publisher Link]