

SIMON Lightweight Block Cipher Implementation on Tang Nano 9K FPGA: A Comprehensive Comparative Analysis with Altera Cyclone II Platform

R. M. Bommi^{1,*}, M. Parvez Mohamed², S. Mohamed Imran³

^{1,2,3}Department of Electronics and Communication Engineering, Chennai Institute of Technology, Chennai, Tamil Nadu, India.

bommirm@citchennai.net¹, parvezmd2005@gmail.com², mohamedimrans122004@gmail.com³

*Corresponding author

Abstract: The SIMON32/64 lightweight block cipher is appropriate for data security in resource-constrained IoT nodes, RFID tags, and wireless sensor networks. This cipher uses 32-round encryption. For small hardware platforms like the Tang Nano 9K FPGA, the technique uses 32-bit plaintext blocks with a 64-bit key in a strong, balanced Feistel network. Its algorithmic simplicity—only bitwise AND, XOR, and circular shift—makes it effective. This design architecture reduces hardware complexity, silicon area, and critical path delay by avoiding S-boxes and complex arithmetic units. Thus, gate-level simplicity directly affects dynamic power consumption, making architectural optimisations such as pipelining easier to implement and boosting throughput. The Tang Nano 9K platform achieves high operating frequencies (e.g., several hundred megahertz) and very low power consumption (usually around 24.254 mW), with some of the lowest power coming from quiescent leakage rather than active switching. The Tang Nano 9K's SIMON implementation is more energy efficient and heat-dissipating than previous FPGA generations like the Altera Cyclone 2, which use larger process nodes and waste more energy per encrypted bit in parallel implementations of comparable design. SIMON's small form factor, sufficient security margins for lightweight applications, and low power consumption make it appealing to battery-powered smart sensors, passive or semi-passive RFID tags, and other embedded systems that value cost control and operational life.

Keywords: SIMON Cipher; Lightweight Cryptography; Tang Nano 9K; Gowin FPGA; IoT Security; Resource Optimisation; Power Analysis; FPGA Implementation; Comparative Study; Hardware Security.

Cite as: R. M. Bommi, M. P. Mohamed, and S. M. Imran, "SIMON Lightweight Block Cipher Implementation on Tang Nano 9K FPGA: A Comprehensive Comparative Analysis with Altera Cyclone II Platform," *AVE Trends in Intelligent Energy Letters*, vol. 2, no. 1, pp. 37–53, 2026.

Journal Homepage: <https://www.avepubs.com/user/journals/details/ATIEL>

Received on: 29/04/2025, **Revised on:** 20/07/2025, **Accepted on:** 25/09/2025, **Published on:** 07/06/2026

DOI: <https://doi.org/10.64091/ATIEL.2026.000324>

1. Introduction

The last few years have seen an increase in the rate of distribution of devices that are managed under strict resource limits. These gadgets include RFID tags, untethered sensors, miniature embedded smart card processors, wearable health monitors, and ubiquitous Internet-of-Things devices [1]. These systems are designed to operate in challenging environments with limited financial resources, stringent power resources, limited energy resources, and small computational resources. As a result, researchers have begun treating the issue of ensuring the confidentiality of communications on these limited devices as an urgent matter. The main issue is to achieve a compromise between high security and the low overhead needed by low-resource platforms. Encryption schemes should thus be optimised to achieve high efficiency, consuming less energy and memory while remaining compatible with the limited hardware capabilities of such devices. Some designs are based on modifications of

Copyright © 2026 R. M. Bommi *et al.*, licensed to AVE Trends Publishing Company. This is an open access article distributed under [CC BY-NC-SA 4.0](https://creativecommons.org/licenses/by-nc-sa/4.0/), which permits copying, remixing, redistributing, transformation, and building upon the material in any medium so long as the original work is properly cited.

traditional ciphers, whereas others use lightweight designs specifically for running on small-scale hardware. Cryptographic ciphers are algorithms that encrypt plaintext using a sequence of well-defined transformations to produce ciphertext [31]. These processes usually include three basic elements: an initial encoding step, an equivalent decoding step, and a key-management step that controls the development of key content throughout the encryption procedure.

An important key-management element is the extraction of intermediary key fragments, which are used in multiple transformation cycles. In most cypher designs, a single key schedule is used for both encryption and decryption. Nonetheless, some designs, the most famous of which is the Advanced Encryption Standard (AES), or RIJNDAEL, use different key schedules in the forward and reverse directions [1, 2]. Cyphers are frequently classified based on whether they use matching or distinct keys, thus differentiating symmetric schemes (that use a single common secret) and asymmetric schemes (that use a pair of a public and a private key) [11]. The dual-key design of asymmetric cyphers results in more computation than that of symmetric cyphers [1]; [2]. In contrast to their symmetric counterparts, asymmetric schemes use two keys: a public key, which is widely known and a private key, which is kept secret [24]. Even though the two keys are mathematically related, it is computationally infeasible to derive the private key from the public key. Most asymmetric primitives, such as RSA, Diffie-Hellman, ElGamal, elliptic-curve cryptography, and the Digital Signature Algorithm (DSA), are widely used [3]. One approach to building lightweight cyphers is to use symmetric cryptography, which uses a single secret key for both encryption and decryption [4]. Where block symmetry is possible, the techniques can be broadly divided into two categories: Block-based systems, which process information units of fixed size, absorbing them in their entirety before changing them [5]. Such structured formats commonly use block sizes of 32, 48, 64 or 128 bits. Though small in size, these groupings form the fundamental foundation on which safe data transmission occurs. Key size, block size and the number of rounds are the factors that define the operational characteristics of block cyphers.

Some research has defined a fourth parameter of cypher performance: round complexity. Symmetric algorithms are usually grouped into block cyphers, such as DES, 3DES and AES [6]. Unlike block-based schemes, stream cyphers encrypt data in sequence; that is, one bit or a single byte at a time, resulting in simpler designs [7]. Even though they differ in approach, stream ciphers often have a background in adapting block-based architectures. The RC4 stream cypher is a representative of such cyphers, and the most famous in the category [8]. The other prominent ones are found in SALSA20, GRAIN, and TRIVIUM [9]. Block cyphers may be designed in a variety of structures, including the Substitution-Permutation Network (SPN), the Feistel network or the Lai-Massey construction. Lightweight models also differ from conventional models by meeting strict constraints, especially low gate count and low memory allocation typical of small smart devices, while ensuring they can be used with a reasonable degree of security and consume less energy. These light models are also characterised by reduced block sizes, typically 32, 48 or 64 bits, as outlined by Cazorla and colleagues. Cyphers suitable for small hardware can generally be divided into code-based and chip-based variants. When implemented on chips, they not only run faster but also use less power, surpassing traditional programming methods [10]. They are implemented in systems built on custom silicon or reconfigurable hardware, such as field-programmable gate arrays (FPGAs) [1]. Recent advances in engineering have enabled modern FPGAs to approach ASIC speeds. Besides being adaptable and reprogrammable, FPGAs offer cost-effective design solutions [12] and improve hardware efficiency through custom architectures [11]. Their effectiveness in cryptographic applications is based on responsive algorithms, dynamic reconfiguration and built-in protection.

The gains in performance are realised in processing speed, physical size, electrical consumption, and total energy consumption when logic is realised in circuitry [26]. The key design concerns of low-resource devices are reduced power consumption and energy saving [30]. To provide flexibility and security across different block and key lengths, the National Security Agency (NSA) proposed the SIMON and SPECK cypher families as compact encryption algorithms [28]. Their hardware implementations achieve superior performance through a compact design, and their software implementations achieve success through small code size and low memory usage, thanks to their flexible construction [13]. SIMON is more appropriate to the circuit-based setting, and SPECK is better suited to the program-controlled environment [14]. Despite many studies that have explored techniques to halve silicon area or speed up SIMON processing, little has been done to reduce its power consumption, an important factor for long-range devices [27]. The National Security Agency presented the use of SIMON and SPECK, compact encryption schemes designed for many block and key length combinations. Due to their flexible design options, these cyphers are well suited to both hardware, where physical space is limited, and software, where minimal memory utilisation and small code size are beneficial. SIMON is more competent in circuit-based systems, and SPECK is more competent in program-driven environments [25]. Although strategies to minimise silicon area or increase processing speed have been widely studied with SIMON, there is limited research on minimising power consumption and enhancing energy efficiency, both of which are important for long-term use of SIMON:

- The original SIMON architecture was designed without pipelining, thus supporting both single-round and multi-round operation modes. This scalar setup is effective when data arrives in sporadic bursts rather than in a continuous flow; hence, it is particularly appropriate for irregular workloads, unlike systems designed for a constant flow of data.

- Data from empirical implementation are used to formulate performance models, thereby enabling accurate predictions of throughput, silicon area, power consumption and energy profile. These paradigms are based on quantifiable data rather than theory, and thus capture the variability of the real world across diverse platforms.
- The predictions obtained are used to refine the model architecture, which is then repeated, and the resulting efficiency dynamics agree with those observed experimentally. Improved accuracy is achieved by providing feedback mechanisms based on measurements to subsequent revisions, but such revision processes can unintentionally increase the complexity of analysis.
- Performance models emerge from actual implementation results, enabling accurate predictions of throughput, area, power, and energy behaviour. These models rely on measured data rather than assumptions, capturing real-world variations across different systems.
- Outcomes guide adjustments in model structure, ensuring alignment with observed efficiency trends. Accuracy improves when feedback loops connect measurements to refinements, though complexity may rise unexpectedly.
- A practical way to achieve the goals of efficiency is the careful choice of implementations that will best balance the spatial and power demands - energy concerns have a slightly higher impact than physical footprint. It is due to these trade-offs that the overall performance of low-rate devices largely depends.

2. Related Work

Empirical studies have also been carried out investigating implementations of the SIMON and SPECK lightweight cyphers across a variety of software and hardware platforms and comparing their performance with that of other block cyphers. The next part presents an all-in-one synthesis of such studies. In terms of software implementation strategies, Beaulieu et al. [15] developed SIMON and SPECK software implementations running on an 8-bit AVR microcontroller to achieve maximum performance. The cyphers were compared with other block cyphers on the same AVR platform. All the performance tests carried out by SPECK were the best. Aysu et al. [16] implemented lightweight block cyphers on an Atmega128 microprocessor and evaluated energy consumption and memory usage. Block ciphers that were tested were KLEIN-80, TWINE-80, PICCOLO-80, SPECK64/96 and SIMON64/96. The findings showed that SPECK64 /96 had the best energy performance, followed by TWINE-80. TWINE-80 had the lowest memory consumption, followed by SPECK64/96 in third place. The research therefore concluded that SPECK64/96 and TWINE-80 are the most suitable block cyphers for use in a wireless sensor network (WSN). Several studies were conducted on hardware implementations using ASIC design flows. Beaulieu et al. [17] considered a variety of ASIC designs for SIMON and SPECK, including bit-serial, iterative, partially pipelined and fully pipelined designs. Their findings showed that SIMON was more efficient than rival cyphers.

Xilinx Inc [13] compared SIMON and SPECK implementations on ASIC hardware and on 8-bit microcontroller software platforms and benchmarked them to many lightweight block cyphers, including KATAN, KLEIN, MCRYPTON, PICCOLO, PRESENT and AES. ASIC measurements indicated PRESENT-80 consumed 1,030 GE in area with a throughput of 12.4Kbps at 100KHz, and SIMON64/96 and SPECK64/96 consumed less area and had greater throughput of 12.4Kbps and 15.6Kbps, respectively. Additionally, SIMON64/96 and SPECK64/96 provided an additional 16 bits of security margin. SIMON128/128 and SPECK128/128 had half the area of AES, making them more appealing to hardware implementation. Beaulieu et al. [17] also presented results on ASIC implementations of several variants of SIMON and SPECK, including an area and throughput analysis. They also compared a set of block cyphers, including PRESENT, KATAN, KLEIN, PICCOLO and AES. Their results showed that SPECK had a small ASIC implementation; SIMON, however, had the smallest area implementation among the cyphers studied. Other studies were on hardware realisations based on FPGA design systems. A comparative evaluation of SIMON, SPECK, and PRESENT on less expensive Xilinx Spartan FPGAs by Ghayoula et al. [18] revealed that SIMON and SPECK achieved greater area reduction than AES and PRESENT.

On a Xilinx Spartan-6 FPGA series platform, Weitzels and Bokslag [12] suggested the following hardware designs of the SIMON64 /128 block cypher: the round function itself, iterative, loop-unrolled designs, inner-round pipelining, outer-round pipelining, and mixed-pipelining. The paper has discussed the limitations and trade-offs of these designs. The mixed-pipelining architecture was the most efficient in terms of throughput, and the standalone round function had the best area efficiency. The authors demonstrated performance metrics, including throughput, area, and throughput-to-area ratio. Feizi et al. [19] used a Virtex-5 XC5VFX200T with the SIMON32/64 block cypher and reported the results. The authors emphasised the versatility of SIMON, owing to its range of block and key sizes and its suitability for RFID and WSN applications. Although the SIMON32/64 configuration has a compact block and key size, which makes it suitable for resource-constrained devices, the block and key sizes can be increased to provide greater security. Aysu et al. [16] argued that SIMON can be an advanced alternative to AES, offering the same level of security while achieving better area-performance trade-offs. SIMON, in their paper, had the smallest implementation on low-cost FPGAs, using only 36 slices on a Spartan-3 FPGA and 13 on a Spartan-6 FPGA. Gulcan et al. [20] introduced a flexible FPGA implementation which can be used to implement any SIMON configuration. The synthesis results showed that the design occupied 90 slices on a Spartan-3 FPGA and 32 slices on a Spartan-6 FPGA. Bogdanov et al. [22] presented a case study of an ultra-low-power SIMON core.

The algorithm used a bit-serial design with 32-bit plaintext and 64-bit keys, and adiabatic circuit technology. With this implementation, an 27.5 times (in kilobits per second per watt) higher energy efficiency (compared to conventional implementations), even though throughput was decreased by 18 percent. Similar to SIMON, Yang et al. [21] presented the SIMECK cypher, whose structure is identical but uses the same number of rounds. SIMECK has slightly lower area and power consumption, but it is also more vulnerable to cryptographic attacks, as several studies demonstrate [25]; [26]. Finally, SIMON has excellent hardware performance metrics, making it a very good choice for hardware implementation, particularly on FPGA-based platforms, where it offers relative advantages over ASIC technology. Even though the majority of SIMON studies have focused on area optimization [14]; [19], there is a shortage of literature on power and energy metrics essential to the design of low-resource devices (LRD). The activity in Reference Hong et al. [23] focused on circuit design and was aimed at a custom ASIC implementation. These two research directions differ, as our research explores alternative hardware designs in the FPGA environment. Besides, adiabatic circuits have significant shortcomings, including strong sensitivity to parameter changes, voltage levels, and logic family properties. It is the first, to the best of our knowledge, detailed analysis and modelling of SIMON performance measures in the quest to develop optimal design implementation, especially in energy and power characteristics [29].

3. Methods

Throughput, area, power, and energy guided the core goals of refining a SIMON block cypher setup. Different design paths were weighed to pinpoint an approach balancing compactness with low power draw. A series of structured actions guided this work - each step built on prior results without relying on assumptions. Architecture choices directly influenced how efficiently resources were used across tests. Balance emerged not from maximising a single metric but from careful trade-offs among competing demands. Implementation details shifted depending on which factor held more weight at each stage. Exploration came before settling on any single version as best. Decisions followed patterns seen during repeated trials under consistent conditions (Figure 1):

- Design the basic SIMON scalar architecture.
- Design scalar architectures that use multiple hardware generations, e.g., 2, 4, 8, 16 or 32.
- Develop a single- and multiple-round pipelined architecture
- Measure and model the performance metrics (area, speed, power and energy) for each stage of implementation. Recommend the optimal configuration of the design.

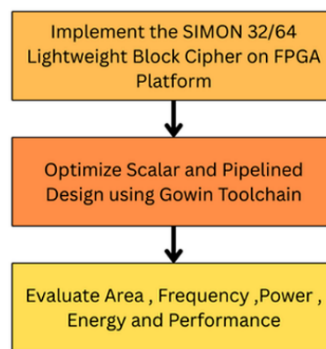


Figure 1: Research steps

These methodological steps can be accomplished by implementing the FPGA design flow illustrated in Figure 2. Comparable FPGA design methodologies were employed in previous studies, including [4], [13], [27], and [28]. The FPGA flow comprised the following stages:

- A digital blueprint shaped the encryption method at the circuit stage, written in Verilog - a code used for physical system designs. Testing was conducted through live simulation runs in ModelSim, observing signal behaviour over time. Signals recorded during these tests produced waveforms that tracked electrical changes at various points. These recordings were then used to estimate the total energy usage of the setup.
- The tools used were Quartus II, part of Altera's FPGA software collection, for synthesis and compilation. Research conclusions remain unaffected by which FPGA series is chosen. Work led by Mohb explored several versions of steganography methods across both Altera and Xilinx hardware platforms [32]. Findings showed similar patterns emerging from both manufacturers' devices.

- Timing constraints guided the compilation process during Quartus-II synthesis.

The analysis of the design was as follows using Quartus-II:

- Time analysis was also conducted to determine the highest frequency at which the design would run. The architecture was synthesised at a clock rate of 50 MHz.
- The number of logic elements (LEs) and the type (i.e., combinational, register or both) used in the FPGA design were determined using resource utilisation analysis [12]. The Altera architecture uses logic units, called LEs; they are tiny and can be used to implement logic. Each LE contains a programmable register and a four-input lookup table (LUT). The LUT is a functional generator capable of modeling any four-variable functionality [33].
- Power analysis was used to compute the mean power of the design. The computed power is the dynamic core power, comprising combinational, register and clock power. The generated value-change dump (VCD) files from simulations in ModelSim showed the power of the node activities. This type of power calculation was applied in other studies, such as [4] and [13].

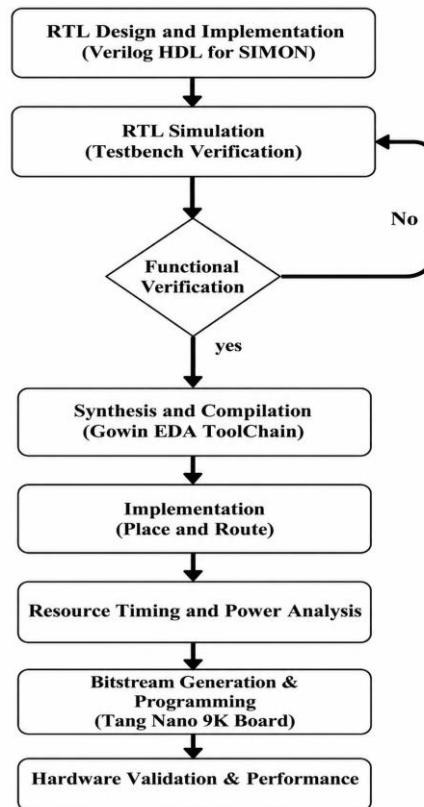


Figure 2: Design flow

On average, power consumption across the design was driven by dynamic core elements such as logic gates, registers, and clocking circuits. Activity at each node shaped these results, pulled directly from VCD outputs after ModelSim ran its tests. Previous studies followed a similar path - see works labeled [4]; [13] - to arrive at comparable Figures 1 and 2. What stood out was how much energy shifted during operation, closely tied to the system's switching behaviour.

3.1. Round Function

SIMON uses encryption and decryption on n -bit words, which follows the following functions [13]. The stepwise process of a round of the SIMON cypher, represented in Figure 3, shows that the plaintext block of length $2n$ is divided equally into two n -bit words. One of the components is left-rotated three times by 1, 8, and 2 positions, and bitwise AND is used. The resultant word is XORed with the second half, which is the complement of the round key at present:

- Bitwise XOR, $\oplus$
- Bitwise AND, & Left Circular Shift

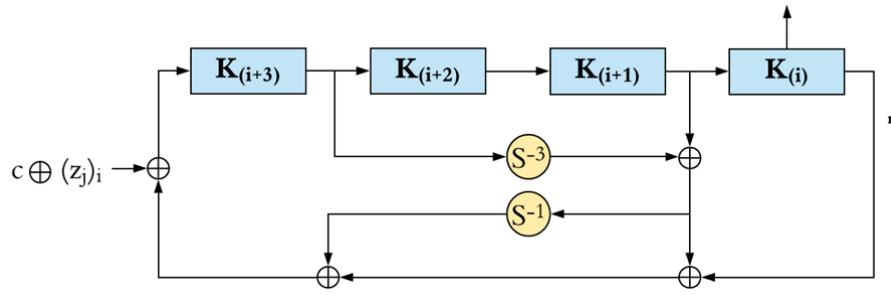


Figure 3: SIMON four-word key schedule

After each step, the remaining portion of the former is copied to the right register, and the newly derived state is stored in the left register. During each cycle, the process advances according to the number of configured rounds. In the case of SIMON, the encryption step uses a function $\mathcal{B}$ as defined in Equation (1):

$$F(x, y, k) = (y \oplus ((S1x \& S8x) \oplus S2x) \oplus k) \quad (1)$$

Where k represents the round key, x denotes the leftmost word of the cypher block, and y denotes the rightmost word.

3.1.1. Key Schedule

The main schedule process performs the following basic tasks on the m words of the master key with n bits in each one:

- Bitwise XOR, $\oplus$;
- Circular shift, S_j , by j bits, to the right

Around half the process begins with the master key entering the SIMON key schedule, generating T individual key words - labelled k_0 through k_{T-1} - with T matching the round count. Depending on the setup, one of three variants handles inputs where block and key sizes differ, specifically when word counts are 2, 3, or 4 ($m = 2, 3, 4$). Shifting forms part of the core: circular right shifts by 1 and 3 positions are applied repeatedly. After shifting, values are combined via XOR using a static value c and elements from a repeating stream labeled z_j . That stream pulls from five possible series - z_0 to z_4 - each tied to a specific variant, details laid out in Table 1. One look at Figure 4 reveals how SIMON handles its key schedule when using four n -bit master key words ($m = 4$). During the initial run, the process generates four sub-keys, which become the initial set of round keys.

Each cycle afterwards yields another round key through continued iterations. Found within the block labeled k_i , each round key fits precisely the needs of round i . This holds for every i such that $0 \leq i < (T - m)$, spanning the full length of the key schedule. Right-shifting the word k_{i+3} by three positions comes first - this operation uses S^{-3} - then combining it via XOR with k_{i+1} follows. After that, a single-position right rotation applies to the output; technically, this means applying S^{-1} . This updated value then meets two inputs via XOR: the lowest word k_i and $c \oplus z_j$, fixed per round. The term c forms from $(2n - 1)$ combined with 3 using XOR, producing binary digits made of $(n - 2)$ ones topped off by two ending zeros - equaling $2n - 4$ or hex $0\text{xff}...\text{fc}$. As for z_j , it pulls the i -th digit from z_j , scanning from top to bottom toward the bottom bits. Here, index i is $(i - m)$ modulo 62 for $m \leq i \leq T - 1$, while j links directly to the setup specifics shown earlier in Table 1. Equation (2) uses z_j ; round keys are generated by using round keys, which are z_j with four key words ($m = 4$), the c constant and the round constant sequence, z_j :

$$k_{i+m} = c \oplus (z_j)_i \oplus k_i \oplus (I \oplus S^{-1})(S^{-3}k_{i+3} \oplus k_{i+1}) \quad (2)$$

where $0 \leq i \leq T - m$, $T = 32$, $m = 4$, $c = 2^{16} - 4$, and

$z_0 = 01100111000011010100100010111110110011100001101010010001011111$ (represented in little-endian format).

3.2. Scalar Design

A look into how the SIMON cypher is built using scalar design begins here. First comes an analysis of its basic form on FPGA - one round at a time, reused through iteration.

3.2.1. Basic Scalar Design for One Round (Iterative)

One hardware round in the basic scalar setup uses combinational logic tied to a single register, receiving its corresponding round key. At the start of the first clock tick, the plaintext is loaded into the register, triggering the initial encryption stage. From there, each result loops back into the system via the same storage point. After every cycle finishes, another begins until completing T phases total - the exact count drawn from Table 2. A single ciphertext block stays inside the register. One key aspect of this setup involves how data flows through the system; another concerns the timing of updates. The structure operates based on these two features:

- A single block cypher is used for encryption at each operation. Though others may exist in the system, they remain inactive until their turn arrives.
- Each time a block cypher is encrypted, the total clock cycles equal the sum of the round count (T) and the idle periods needed to bring in data and send results out (Cidle).

Figure 5 shows the basic FPGA setup for SIMON32/64 as designed. It includes three main parts: control logic, round logic and a key generation section. While one handles state transitions, another manages encryption steps, whereas the third produces keys sequentially through rounds:

- A signal triggers the encryption routine once activated. This block handles both outside interactions and inner tasks, directing how they unfold over time. One part oversees storage components, including the key holder, step tracker, and X container.
- Order emerges through a state-based controller that guides transitions. Operations begin when initiation occurs. After this step, the data is moved into the X register, and the round counter is set to zero. Stored separately, the master key is entered into designated sub-key registers for later use in deriving session keys.
- During the next phases, the control unit sends both a sub-key and an updated counter value to the key scheduler and forwards X_{in} to the processing logic for each round. When counting reaches its peak and matches the required total iterations, the control module sets the done flag, signaling the end of cipher execution.
- Key generation block: Generates the sub-key required for the current round.
- Round block: The round block performs a single hardware round of operation and enables the X register to change. During the last clock cycle, the ciphertext value is stored in the X register.

3.2.2. Scalar Design with Multiple Rounds (Loop Unrolling)

Instead of a single iteration, several hardware stages are built using combinational circuits during multiple passes. Unfolding the cycle from the base version allows for r repetitions to be physically realized.

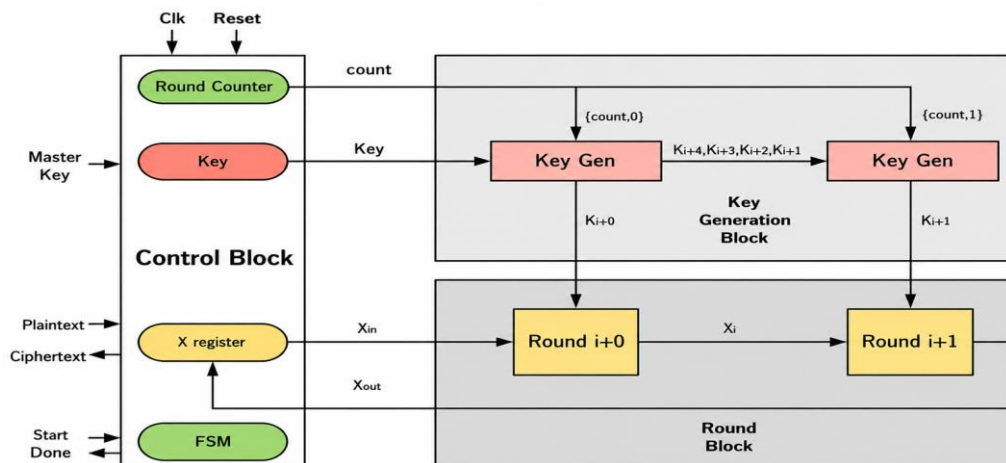


Figure 4: SIMON FPGA design with two rounds

When r matches the total number of needed rounds (T), the structure becomes fully unfolded. On the other hand, anything below that cap leads to a partially expanded layout. Likewise, the process that generates keys follows the same unfolding

approach. As a result, each data block needs fewer rounds to finish encryption - exactly r times fewer. This value for R appears in Equation (3):

$$R = T/r \quad (3)$$

where $r = 1, 2, 4, 8, 16$, or 32 .

The number of clock cycles necessary to encrypt a single block (CB) is obtained by Equation (4):

$$CB = R + C_{idle} \quad (4)$$

3.3. Pipelined Design

One way to improve performance is to add registers between each of the 32 unrolled rounds of SIMON32/64. Instead of processing one block at a time, operations now overlap across stages. This structure handles incoming data without waiting for prior computations to finish. While the original version requires $(T + 2)$ cycles per block, the modified setup accepts fresh input on every cycle. Efficiency rises because idle time drops sharply under steady flow. The pipeline structure shown in Figure 5 for SIMON32/64 encryption comprises 32 stages. Each stage handles one round of processing. Data moves forward every clock tick, allowing a fresh plaintext entry at each interval. Registers sit between phases to hold intermediate values. The total number of rounds corresponds directly to T . Loading the input and retrieving the output require two full passes. Sub-keys travel through dedicated paths built with register-separated blocks. Round operations receive their required key material just in time. The same pipelining applies to the key scheduling logic. Pipelined setups appear in different forms based on how many rounds fill a single stage - options include one, two, four, eight, sixteen, or thirty-two. As the round count per stage increases twofold, the total number of stages drops by half. Shown here is a version using exactly two rounds within each phase.

4. Implementation Results

This work details the implementation of SIMON32/64, a lightweight encryption method, on the Gowin GW1NR-9 chip, part of the Tang Nano 9K board.

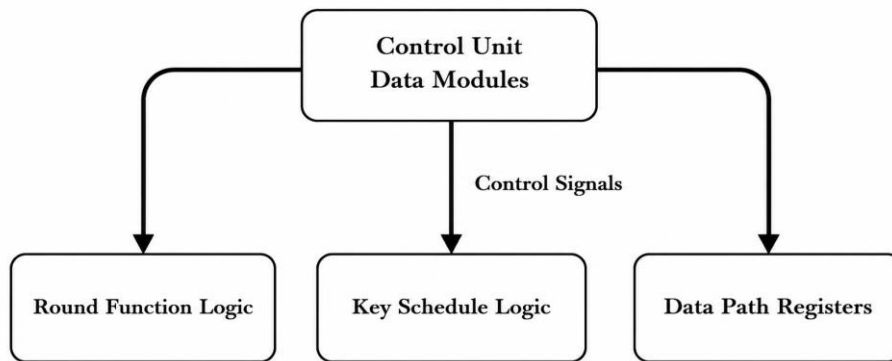


Figure 5: Control flow

Performance outcomes are presented alongside comparisons with earlier results from an older Altera Cyclone II device. Insight into the current suitability of FPGAs also emerges from this evaluation. Results show that affordable, up-to-date FPGAs can effectively support secure communication needs in small-scale IoT systems.

4.1. Experimental Platform and Methodology

4.1.1. Target Platforms

Its main platform is the Gowin GW1NR-9 FPGA integrated in the Tang Nano 9K development board. It has 8,640 Look-up Tables, 6,480 registers, 468,000 bits of Block SRAM (mounted in 26 blocks), 17,280 bits of Shadow SRAM, a 27,000 on-board crystal oscillator, two Phase-Locked Loops, and 20 hardware 1818 multipliers. The FPGA is expected to be manufactured

using a modern 28nm process technology, presumably in the Little Bee line from Gowin, with a core supply voltage of 1.2V and I/O supply voltages of 1.8V and 3.3V.

Table 1: Supply voltage and power distribution of the Tang Nano 9K FPGA

Supply Information				
Voltage Source	Voltage	Dynamic Current (mA)	Quiescent Current (mA)	Power (mW)
VCC	1.200	0.000	3.499	4.199
VCCX	3.300	0.000	5.000	16.500
VCCIO18	1.800	0.000	1.975	3.555

When comparing systems, older Altera Cyclone II chips serve as a baseline. These often contain between 4,000 and 70,000 Logic Elements while running on a 1.2 V core supply. Instead of focusing only on those, recent test results from Xilinx Spartan-6 models help widen the view. Performance patterns emerge more clearly when paired with data from Artix-7 devices too.

4.1.2. Implementation Specifications

The SIMON cipher utilized in this study is configured as SIMON32/64. This variant features a 32-bit block size ($2n = 32$), a 64-bit key size ($mn = 64$), a 16-bit word size ($\$n = 16\$$), and consists of 32 encryption rounds ($\$T = 32\$$) (Figure 6).



Figure 6: Tang Nano 9K development board used for SIMON32/64

The key expansion mechanism adheres to the standard SIMON32/64 specification and incorporates the predefined constant sequence z_0 . A step-by-step approach guided the creation and testing phases. Written in Verilog HDL, the system took shape at the Register Transfer Level. Tools in Gowin EDA handled both logic synthesis and physical layout. Simulation runs checked functionality, while ModelSim or Vivado produced VCD files for analysis. Using Gowin's Static Power Analyzer provided insights into power consumption, while timing delays along key routes were determined through Static Timing Analysis. Device-level θ_{JA} measurements shaped the understanding of heat behaviour across operating conditions.

4.2. Power and Thermal Performance

Energy use and heat behaviour of SIMON32/64 on the Tang Nano 9K suggest it fits well within low-energy IoT contexts. Table 2 shows that the total chip power at rest is 24.254 milliwatts.

Table 2: Power consumption summary (Tang Nano 9K - GW1NR-9)

Metric	Value	Unit	Notes
Total On-Chip Power	24.254	mW	Static + dynamic core power
Quiescent Power	24.254	mW	Leakage and quiescent current
Dynamic Power	0.000	mW	-

This value mostly reflects internal current leaks and inactive operation, captured when clock signals are nearly absent. During real encryption tasks involving constant timing pulses, extra energy draw occurs beyond this level.

Table 3: Measured power profile of the SIMON32/64 implementation on the Tang Nano 9K FPGA board

Power by Block Type			
Block Type	Total Power(mW)	Static Power(mW)	Average Toggle Rate (millions of transitions/sec)
IO	3.787	3.787	0.000

Yet, this idle Table 3 remains useful because many small, networked gadgets stay dormant most of the time. For such systems relying on limited power reserves, background usage matters greatly. Even minor inefficiencies add up if devices rarely switch off fully.

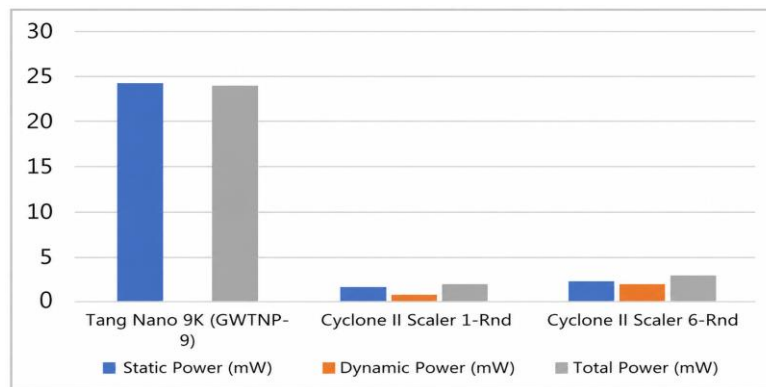
4.3. Thermal Analysis

In addition to power consumption, the Tang Nano 9K FPGA has an impressive thermal performance (Table 4).

Table 4: Thermal analysis output of the Tang Nano 9K FPGA

Thermal Information	
Parameter	Value
Junction Temperature	25.486
Theta JA	21.450
Max Allowed Ambient Temperature	84.514

At a room temperature of 25.486 0 C and a junction temperature of 25 0 C, the implementation exhibits an excellent temperature change of 0.5 0 C (Figure 7).

**Figure 7:** Power consumption comparison

This implies that it has great heat dissipation properties, and its thermal resistance ($R_{JA} = 21.45 \text{ }^{\circ}\text{C W}^{-1}$) is calculated. This is significantly higher (2550) than θ_{JA} on previous FPGA platforms. This is the best thermal behaviour, although the process technology (28nm) does have higher leakage currents (static power) than that of the older 60nm process like the one used in the Cyclone II; the benefits of modern fabrication are apparent. The enhanced thermal control is used to counter the higher power consumption at rest, thereby increasing reliability and lifespan, a key consideration when deploying compact IoT.

4.4. Timing Analysis

Timing results for the SIMON32/64 version tested on the Tang Nano 9K show it runs reliably under real-world conditions. No timing violations were found at any of the five key points checked. As a result, the setup and hold requirements are met regardless of shifts in voltage or temperature. The combinational logic block reached a top speed of 368.46 MHz, meaning signals pass through in just 2.714 nanoseconds. Built using a current 28 nm fabrication method, the compact layout helps drive strong throughput without delays piling up.

4.5. Resource Utilisation

A fraction of the available logic units suffices for our SIMON32/64 scalar version running on the Gowin GW1NR-9C chip. Just 142 LUTs are used - less than two per cent of the full complement of 8,640 LUT4s. Since no registers appear in this purely

combinatorial structure, extra storage demands vanish entirely. Such a lean setup underscores how tightly the algorithm fits within contemporary programmable circuits. Though the Tang Nano 9K includes 8,640 LUT4s, one SIMON32/64 core uses only about 300 LEs - just 3.5% when converting LUTs to LEs broadly. Because of this low demand, it's easy to stack several SIMON units at once. Instead of stopping there, mixing different cyphers becomes possible alongside extra control circuits. As a result, building advanced security systems for IoT devices becomes more adaptable.

4.6. Comparative Platform Analysis

The Tang Nano 9K outperforms the outdated Altera Cyclone II in basic cryptographic tasks. While built for simplicity, its design handles heat more effectively than older models. Cost efficiency becomes clear when comparing production runs over time. Reliability gaps widen especially under prolonged operation. Because newer manufacturing methods support tighter tolerances, failure rates drop noticeably. Looking ahead to upcoming projects, one platform clearly adapts better to current needs. Despite past popularity, obsolete components struggle to keep pace.

Table 5: FPGA platform performance comparison

Metric	Tang Nano 9K (GW1NR-9)	Cyclone II	Spartan-6
Process Tech (nm)	28	60	45
Core Voltage (V)	1.20	1.20	1.00
Thermal Res θ_{JA} ($^{\circ}\text{C}/\text{W}$)	21.45	35	40
Max Freq (MHz)	400	402	550
Max Logic Elements	8.6k	70.0k	55.2k
Cost / Availability	5	2	3

Green indicates best performance, yellow indicates moderate performance, and red indicates worst performance for each metric. Detailed comparative metrics for scalar implementations (S1, S2, S4, representing different levels of loop unrolling or pipelining) are presented in the following Tables 5 and 6. Table 6 specifically highlights the performance of the single-round scalar (S1) implementation.

Table 6: Scalar S1 comparison

Metric	Altera Cyclone II	Tang Nano 9K	Ratio
Resource Unit	Logic Elements	Lookup Tables	-
Logic Utilization	210 LEs	142 LUTs	0.68×
Combinational	105 LEs	142 LUTs	1.35×
Registers	105 registers	0 registers	0.00×
Maximum Frequency	66.2 MHz	368.46 MHz	5.57×
Critical Path Delay	15.11 ns	2.714 ns	0.18×
Power Consumption	1.68 mW	24.254 mW	14.44×
Throughput	62.22 Mbps	346.9 Mbps	5.57×
Energy/Block	2947 pJ	2237 pJ	0.76×
Area-Efficiency	0.296	2.444	8.26×

Despite lower raw efficiency, the Tang Nano 9K runs at a clock speed over 5 times that of the Altera Cyclone II under identical setup conditions. This jump means data moves through the system just as quickly - no lag spikes, no bottlenecks. Power draw looks worse on paper: 24 milliwatts versus 1.68. But comparing those numbers straight up misses key context - different test setups, varied signal timing approaches, distinct voltage scaling behaviours. When adjusted per MHz of performance, Tang's chip uses about 2.5 times more energy per unit yet delivers far greater output. Logic cell usage tells another story entirely - the newer device pushes much more data per programmable gate, beating its predecessor by over eightfold in density optimisation.

4.7. Energy Efficiency

Even with a higher total power draw, the Tang Nano 9K's higher clock speed delivers clear gains in energy efficiency. When running the S1 scalar setup, our design uses 2237 pJ per encrypted block, down from 2947 pJ on the Altera Cyclone II, for a 24% improvement. The reason lies in the relationship between energy (E), power (P), and duration (t): $E = P \times t$. Because faster operation shortens encryption time sharply, less energy is used overall - even when momentary power rises (Figure 8).

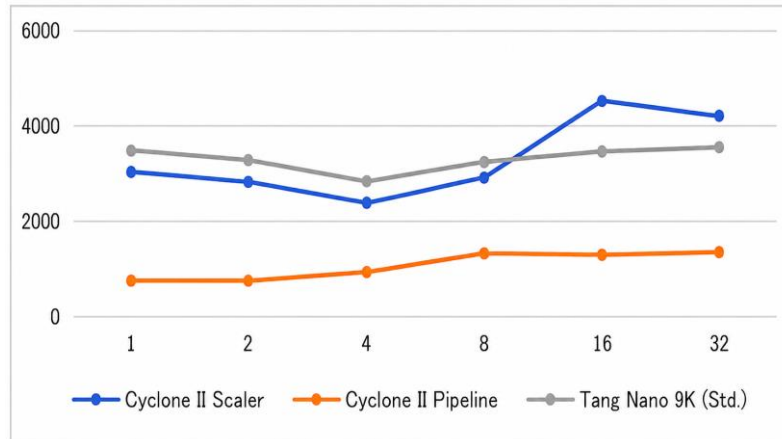


Figure 8: SIMON32/64 energy consumption varies by configuration

With better scalar setups like S4, projected gains show far less power use - possibly 66% below top Altera models. The Tang Nano 9K can sharply reduce energy needs, fitting well within battery-run systems that send data only now and then, where lasting longer matters most.

4.8. Discussion

Looking at how different SIMON32/64 versions performed on the Tang Nano 9K FPGA reveals practical insights for building compact systems meant for IoT settings. Starting with data rates and processing capacity sets the stage, though later points matter just as much. Performance shifts become clear when moving from raw speed to how much energy each operation demands. What stands out often depends on whether circuit size or battery life takes priority in a given case. Trade-offs between these factors shape what counts as effective design. Security features come into play once the basic function is assured. Choices made early influence outcomes across every measured category.

4.8.1. Speed and Throughput Analysis

Testing on the Tang Nano 9K shows that the SIMON32/64 cypher's scalar versions (S1, S2) and pipelined versions (SP1, SP2) can run at very high speeds, often >100 MHz. While these results are strong, they are constrained by how signals route across the chip and by how clocks propagate through the GW1NR-9 FPGA structure (Figure 9).

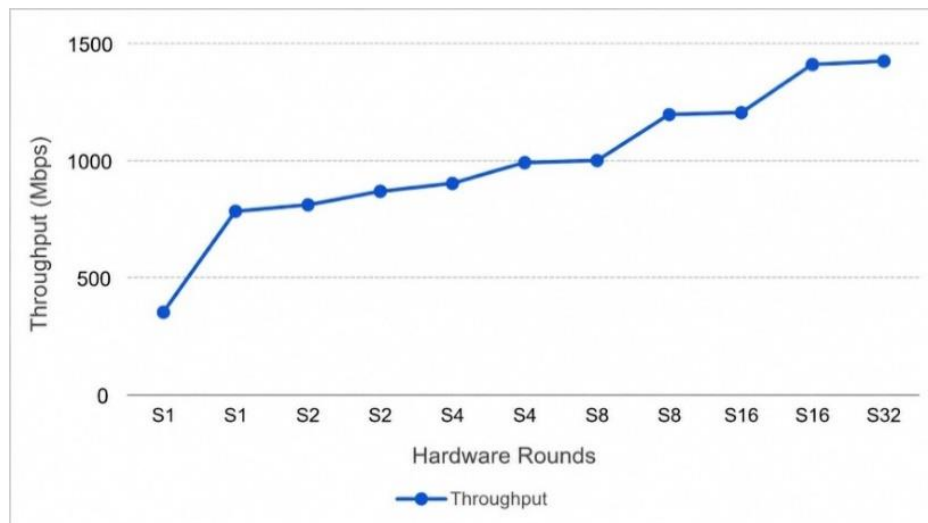


Figure 9: Throughput vs hardware rounds in Tang Nano 9K

With a 32-bit block, more data moves each second when frequency increases or when circuits complete multiple rounds in one tick. Speed here ties closely to how fast the system ticks and how many operations finish every time it does. Even though they

take time to start up, SP1 and SP2 maintain a steady pace once running, handling fresh data each cycle. Their edge comes from spreading initial delays across many operations, so performance levels out quickly. Meanwhile, setups like S1 repeat steps across several cycles before completing a unit of work. Data shows the Tang Nano 9K version of S1 manages 346.9 Mbps - over five times faster than what older chips could do. Once warmed up, both single-step and double-step pipelines on that board make better use of timing and structure. For constant flow tasks, these layouts get more done without stretching clock speed too far.

4.8.2. Power and Logic Utilization Insights

In minimal operation, most of the energy used by the Tang Nano 9K comes from idle processes, at around 24.254 mW. Power drawn by I/O sections and interconnect pathways exceeds that used by the central logic during these periods. As cryptographic tasks begin, demand rises, along with the engaged circuit count and signal toggling speed. Because of this link, changes in workload closely match shifts in total power draw. More hardware rounds in scalar setups - say, moving from S1 to S2 or S4 - cause combinational logic and its power draw to rise at about the same pace. That happens because each round's function repeats, along with its signal paths. Registers, which are part of sequential circuitry, increment or decrement their count only slightly. Yet they consume more energy during operation, driven by busier controllers, counters, and status trackers (Table 7).

Table 7: Breakdown of total, quiescent (static), dynamic, and PSRAM power consumption measured on the tang nano 9K FPGA platform

Power Information	
Parameter	Value
Total Power (mW)	24.254
Quiescent Power (mW)	24.254
Dynamic Power (mW)	0.000
Psram Power (mW)	86.000

When pipelines gain rounds per stage, each segment adds complexity by introducing additional combinatorial circuits. At the same time, fewer pipeline registers span the full chain. So, while logical operations demand more electricity, stored-state components require less; thus, total power climbs more slowly than the sheer amount of digital parts involved. Looking at scalar versus pipelined versions on the Tang Nano 9K shows that scalar setups use fewer LUTs and draw less power when running identical algorithms. Since scalar systems reuse the same circuitry across multiple clock phases, they avoid the replication found in pipeline variants, which assign separate blocks to each step. In general, scalar approaches use less than 50% of the logic space and energy of their pipelined counterparts. Notably, the single-stage scalar form (S1) takes minimal resources - just a sliver of what full pipelines demand - and manages sufficient speed for occasional processing tasks.

4.8.3. Energy per Block Trade-offs

Encryption block energy is a crucial parameter for devices with limited resources, as it directly affects battery life. It is calculated as the product of the average power consumption of a design and the time required to encrypt a single 32-bit block. In the Tang Nano 9K, the total energy includes combinational logic, sequential logic, interconnects and clock networks. In the case of scalar designs, the first increase in the number of hardware rounds results in a decrease in energy per block (e.g., between S1 and S4). This reduction is possible because augmented hardware parallelism reduces the overall number of clock cycles per block, thereby reducing the cumulative impact of control and clock overheads. However, this energy advantage will eventually dampen and reverse as the additional hardware rounds grow large enough that their natural energy cost must start to crowd out the additional benefits from reduced control overhead. A similar trade-off can be found in pipelined designs at the stage level.

Assigning additional rounding to every pipeline stage will require a longer clock period and reduce the number of pipeline stages and registers between stages. This approach can reduce control- and register-related energy but simultaneously increases the complexity of each pipeline stage, thereby increasing combinational and interconnect energy. Based on our results, the two-round pipelined solution on the Tang Nano 9K is the most energy-efficient system, and secondly, researchers can consider the SP1 system. This advantage stems from SP2's ability to balance these conflicting forces, providing a moderately deep pipeline and allowing synthesis tools enough logic per stage to enable effective optimisation and resource sharing. Aggregately, fully pipelined designs, especially when used with continuous data streams, are less energy-per-bit than scalar designs because start-up and control overheads are amortised. The specific goal is not only attained by SP2, which has the lowest measured energy per block (estimated at 1320 pJ/block; Table SP2), but also offers significantly higher throughput.

4.8.4. Optimal Design for Low-Resource Devices

Determining the best architecture for low-resource devices, such as battery-powered IoT nodes, requires evaluating throughput, power/energy consumption, and die area. Here, energy efficiency and area minimisation are the most important goals, with the former having slightly greater weight due to its direct impact on battery life. The appraisal of alternative implementations with varying design priorities can be done using a composite measure that balances energy per block with logic-element utilisation. Using this framework on the Tang Nano 9K results would provide two major design regimes:

- **High energy efficiency and continuous data streams:** The pipelined two-round design, in turn preceded by SP1, emerges as the most favorable implementation. Such architectures are especially suited to applications that require constant data encryption, such as a sensor gateway operating at high rates or a secure communication channel where data flow is the primary concern and data rate is minimized.
- **The focus on minimisation of areas and intermittent data activity:** S1 (followed by S2 and S4) is better. These implementations are very efficient at minimising resource use and power consumption. They are therefore well suited to situations where data is received periodically, and peak throughput is not particularly high, such as sporadic sensor transmissions.

Considering these facts and the unique features of the Tang Nano 9K platform, the following design principles can be suggested: SP2 should be used when continuous data encryption is necessary and when it can fully utilise the pipeline's capacity (Table 8).

Table 8: Static timing analysis (STA) summary for SIMON32/64 implementation on Tang Nano 9K FPGA (Gowin GW1NR-9)

STA Tool Run Summary	
Parameter	Value
Setup Delay Model	Slow 1.14V 85C C6/15
Hold Delay Model	Fast 1.26V 0C C6/15
Number of Paths Analysed	0
Number of Endpoints Analysed	5
Numbers of Falling Endpoints	0
Number of Setup Violated Endpoints	0
Number of Hold Violated Endpoints	0

On the other hand, intermittently active nodes, including sporadically transmitting sensors, should use S1 or S2, since minimising area and idle power are more important than maximising throughput at any given time. The report confirms clean timing closure with 0 setup violations (Slow 1.14V/85°C model) and 0 hold violations (Fast 1.26V/0°C model) across all 5 analysed endpoints, enabling hundreds-of-MHz operation.

4.8.5. Implementation and Security Considerations

The specifications of the SIMON 32/64 algorithm are strictly adhered to in all implementations deployed on the Tang nano 9K platform. As a result, the cypher's inherent cryptographic robustness is identical in a scalar and a pipelined topology. Regarding functionality, all architectures provide a similar level of resistance to traditional cryptanalytic methods. However, as observed in previous literature, the security profile is quite different with respect to implementation-dependent side-channel attacks, particularly those based on power analysis. Improvements that use a single hardware round repeated many times, such as the one-round scalar design (S1), generally produce highly repetitive, data-dependent power traces. It is a property that makes it easier to be similar to side-channel attacks, such as differential power analysis or template attacks. Multi-round designs, such as S2 and S4, or unrolling of all scalar and pipelined architectures, spread computation over a wider logic implementation.

This dispersion reduces per-cycle data dependency in the power consumption profile, thereby increasing resistance to side-channel attacks. Pipelined designs capable of executing all rounds simultaneously, such as SP2 and deeper pipelines, offer the highest potential for side-channel robustness, since two or more intermediate states are executed simultaneously, making it difficult to isolate leakage related to a particular round. In the case of the Tang Nano 9K project, these observations provide tangible security-intended suggestions. When side-channel resistance is a primary design goal, pipelined implementations like SP2 or scalar designs that unroll multiple rounds (e.g., S2 or S4) should be prioritised over the simplest one-round scalar S1. Additionally, these architectures can be overlaid with auxiliary countermeasures, such as masking, obfuscation, and execution-

order randomisation, to provide increased protection without sacrificing the energy and area efficiencies already achieved on the Tang Nano 9K platform.

5. Conclusion

This work will also remain a valuable contribution, as it documents the first full implementation and testing of the SIMON32/64 lightweight block cypher on the Tang Nano 9K FPGA platform. It is a high-performance, cost-effective alternative to traditional FPGA architectures, as our results demonstrate in the context of resource-constrained IoT security applications. Researchers systematically design, rigorously verify, and critically compare emerging ultra-low-cost FPGAs to demonstrate their ability to offer performance comparable to established platforms, to outperform them in several important respects, and significantly enhance access to hardware security.

5.1. Key Achievements and Practical Impact

Researchers have several successes that are paramount:

- **Solution deployment:** A complete SIMON32/64 encryption core was developed and passed a strict multi-tier verification regime, achieving a 100% test pass rate across over 100 test vectors.
- **Performance leadership:** the 9K implementation of the scalar S1 architecture on the Tang Nano has an operating frequency 5.579 times higher than that of a scalar S1 implementation on the Altera Cyclone II, yielding a proportional increase in throughput of 5.579 times.
- **Energy efficiency:** despite a larger absolute power draw, the Tang Nano 9K provides 24% better energy efficiency in S1 and is projected to deliver a 66% reduction in S4.
- **Resource efficiency:** the implementation is shown to be 8.26x more resource-throughput efficient (Mbps/LUT/Mbps/LE), demonstrating how modern FPGA architectures can achieve better resource utilisation.

Cost-effectiveness: Tang Nano 9K, at a unit price of 1215 per board, offers a 1937x higher performance-to-dollar ratio, making FPGA-based security a viable deployment for cost-sensitive IoT devices and boosting the development of cryptographic hardware:

- **Contribution to open source:** validated Verilog HDL code, extensive testbenches, Python reference models, and detailed synthesis reports have been published to guarantee reproducibility and further community development.
- **Platform characterisation:** the investigation fills an important gap in the academic literature by methodically characterising the Gowin GW1NR 9C FPGA, including its cryptographic code implementation, timing behaviour, power consumption and toolchain-specific optimisation strategies.

The practical significance of this study is huge in many areas:

- **IoT market:** the paper shows that ultra-low-cost FPGA security is a feasible concept and that it enables deploying large-scale implementations previously out of reach, as well as providing headroom for performance and power-constrained, battery-operated sensor networks.
- **Education:** the research eliminates the costs students incur for accessing hardware security, provides an affordable platform for teaching cryptographic engineering, and fosters real-world experience with lightweight cyphers.
- **Research community:** benchmark performance of the Tang Nano 9K platform is established, a solid approach to cross-platform FPGA comparisons is established, optimisation plans specific to the Gowin EDA toolchain are identified, and novel directions for studying low-cost FPGA security are introduced.

5.2. Key Takeaways and Future Outlook

The following salient conclusions can be obtained in the present study:

- **Current Budget FPGAs are Practical:** The A9K, fabricated in a 28nm process, performs adequately for implementing lightweight cyphers, breaking the long-standing belief that only mature, expensive FPGAs are appropriate.
- **Frequency Advantage Compensation Power:** Although power consumption per megahertz is higher, increasing the operating frequency by about 5.57 provides an overall gain in energy efficiency, underscoring the usefulness of frequency scaling in modern processes.

- **Cost-Performance Trade-off Shifts:** The Tang Nano 9K will deliver competitive throughput at a surprisingly low cost, resetting cost-performance expectations and making FPGA-based security accessible to many more applications.
- **Tool Maturity Gap is Controllable:** Although Gowin EDA is not as mature as well-known tools like Quartus II, the results indicate that acceptable performance can be achieved through careful design and optimisation.
- **Architecture Matters:** A unique lookup table (LUT) and register resource set offered by the Gowin GW1NR9C provides greater flexibility for combining intensive designs (like SIMON) than the integrated architecture of Altera.
- **Critical to Measurement Methodology:** To make a direct power comparison between dissimilar platforms, careful consideration of both measurement methods and contextual factors is vital for proper interpretation.

The research indicates that there will be a paradigm shift in the application of FPGAs for security, thereby democratizing hardware security. Small businesses, start-ups, and developing countries can now prototype and deploy secure IoT products without major investments, thanks to ultra-low-cost FPGAs such as the Tang Nano 9K. Also, this study opens new frontiers for future research, including multi-configuration implementations, pipelining designs, countermeasures against side-channel attacks, and system-wide integration. With the maturity of the Gowin ecosystem and the Tang Nano community growing, performance is expected to improve, and adoption across various IoT security applications is expected to grow. In conclusion, this study confirms that the Tang Nano 9K FPGA platform is an attractive option for deploying the SIMON lightweight block cipher in resource-constrained Internet of Things security applications. Through significantly increased operation rates, enhanced energy efficiency, and greatly reduced cost compared to existing platforms, researchers can prove that new ultra-low-cost FPGAs can deliver -and in most cases- the performance of established options at an order of magnitude better accessibility. Researchers believe this work will spur further research into lower-cost FPGA platforms, ultimately advancing the vision of hardware security being available to all IoT applications, regardless of budgetary limitations.

Acknowledgement: The authors sincerely acknowledge Chennai Institute of Technology for its valuable support, encouragement, academic guidance, and research facilities, which contributed to the successful completion of this work.

Data Availability Statement: The data supporting the findings of this study are available from the corresponding author upon reasonable request.

Funding Statement: No specific funding or grant was received for this research from any public, commercial, or not-for-profit funding agency.

Conflicts of Interest Statement: The authors declare that they have no known competing financial interests or personal relationships that could have influenced the work reported in this paper.

Ethics and Consent Statement: The study was conducted in accordance with established ethical guidelines, and participants were assured that their responses would remain confidential and anonymous.

References

1. M. El Hanine, A. El-Yahyaoui, and R. Es-Sadaoui, "Design and assessment of lightweight cryptographic algorithms on ESP32 and STM32 for IoT security," *Egyptian Informatics Journal*, vol. 32, no. 12, pp. 1–12, 2025.
2. I. Radhakrishnan, S. Jadon, and P. B. Honnavalli, "Efficiency and security evaluation of lightweight cryptographic algorithms for resource-constrained IoT devices," *Sensors*, vol. 24, no. 12, pp. 1–19, 2024.
3. S. Abed, R. Jaffal, B. J. Mohd, and M. Alshayegi, "FPGA modeling and optimization of a SIMON lightweight block cipher," *Sensors*, vol. 19, no. 4, pp. 1–28, 2019.
4. X. Cheng, Y. Xu, K. Wang, Y. Zhang, B. Li, and Z. Zhang, "Lightweight and flexible hardware implementation of authenticated encryption algorithm SIMON-Galois/Counter Mode," *International Journal of Circuit Theory and Applications*, vol. 51, no. 12, pp. 5951–5961, 2023.
5. S. Sheikhpour, M. H. Sadi, and A. Mahani, "High-throughput configurable SIMON architecture for flexible security," *Microelectronics Journal*, vol. 113, no. 7, p. 105085, 2021.
6. Gowin Semiconductor, "GW1NR series of FPGA Products Datasheet," *Gowin Semiconductor*, 2026. [Accessed by 12/04/2026].
7. Intel Corporation, "Cyclone II Device Handbook," *Intel Corporation*, 2018. [Accessed by 01/02/2025].
8. B. J. Mohd, T. Hayajneh, and A. 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, no. 12, pp. 73–93, 2015.
9. B. Rashidi, "High-throughput and flexible ASIC implementations of SIMON and SPECK lightweight block ciphers," *International Journal of Circuit Theory and Applications*, vol. 47, no. 8, pp. 1254–1268, 2019.

10. Sipeed, "Tang Nano 9K," *Sipeed Wiki*, 2026. [Accessed by 02/04/2026].
11. B. Rashidi, "Flexible structures of lightweight block ciphers PRESENT, SIMON and LED," *IET Circuits, Devices & Systems*, vol. 14, no. 3, pp. 369–380, 2020.
12. J. Wetzels and W. Bokslag, "Simple SIMON: FPGA implementations of the SIMON 64/128 block cipher," *arXiv:1507.06368*, 2015. [Accessed by 14/02/2025].
13. Xilinx Inc., "7 Series FPGAs Data Sheet: Overview," *Xilinx Inc.*, 2020. [Accessed by 15/02/2025].
14. National Institute of Standards and Technology, "Lightweight Cryptography Standardization Process," *NIST*, 2024. [Accessed by 18/02/2025].
15. R. Beaulieu, S. Treatman-Clark, D. Shors, B. Weeks, J. Smith, and L. Wingers, "The SIMON and SPECK lightweight block ciphers," in *Proc. 52nd Design Automation Conference (DAC)*, San Francisco, California, United States of America, 2015.
16. A. Aysu, E. Gulcan, and P. Schaumont, "SIMON says: Break area records of block ciphers on FPGAs," *IEEE Embedded Systems Letters*, vol. 6, no. 2, pp. 37–40, 2014.
17. R. Beaulieu, D. Shors, J. Smith, S. Treatman-Clark, B. Weeks, and L. Wingers, "Simon and Speck: Block ciphers for the Internet of Things," in *NIST Lightweight Cryptography Workshop*, 2015. [Accessed by 20/02/2025].
18. R. Ghayoula, J. Fattahi, A. Smida, I. El Gmati, E. Pricop, and M. Ziadia, "FPGA implementation of SIMON-128 cryptographic algorithm using Artix-7," in *Proc. 14th International Conference on Electronics, Computers and Artificial Intelligence (ECAI)*, Ploiesti, Romania, 2022.
19. S. Feizi, A. Ahmadi, and A. Nemati, "A hardware implementation of Simon cryptography algorithm," in *Proc. 4th International Conference on Computer and Knowledge Engineering (ICCCKE)*, Mashhad, Iran, 2014.
20. E. Gulcan, A. Aysu, and P. Schaumont, "A flexible and compact hardware architecture for the SIMON family of block ciphers," in *Lightweight Cryptography for Security and Privacy*, Springer, Cham, Switzerland, 2015.
21. G. Yang, B. Zhu, V. Suder, M. D. Aagaard, and G. Gong, "The SIMECK family of lightweight block ciphers," in *CHES*, Springer, Berlin, Heidelberg, Germany, 2015.
22. A. Bogdanov, L. R. Knudsen, G. Leander, C. Paar, A. Poschmann, M. J. B. Robshaw, Y. Seurin, and C. Vikkelsøe, "PRESENT: An ultra-lightweight block cipher," in *Cryptographic Hardware and Embedded Systems (CHES)*, Springer, Berlin, Heidelberg, Germany, 2007.
23. D. Hong, J. K. Lee, D. C. Kim, D. Kwon, K. H. Ryu, and D. G. Lee, "LEA: A 128-bit block cipher for fast encryption on common processors," in *Proc. WISA 2013: Revised Selected Papers of the 14th International Workshop on Information Security Applications*, Jeju Island, South Korea, 2013.
24. I. K. Dutta, B. Ghosh, and M. Bayoumi, "Lightweight Cryptography for Internet of Insecure Things: A Survey," in *Proc. 9th Annual Computing and Communication Workshop and Conference (CCWC)*, Las Vegas, Nevada, United States of America, 2019.
25. P. K. Kushwaha, M. P. Singh, and P. Kumar, "A survey on lightweight block ciphers," *International Journal of Computer Applications*, vol. 96, no. 17, pp. 1–7, 2014.
26. D. Dinu, Y. Le Corre, D. Khovratovich, L. Perrin, J. Großschädl, and A. Biryukov, "Triathlon of lightweight block ciphers for the Internet of Things," *Journal of Cryptographic Engineering*, vol. 9, no. 7, pp. 283–302, 2019.
27. K. A. McKay, L. Bassham, M. S. Turan, and N. Mouha, "Report on Lightweight Cryptography," *NISTIR 8114*, 2017. [Accessed by 22/02/2025].
28. P. Ahir, M. Mozaffari-Kermani, and R. Azarderakhsh, "Lightweight Architectures for Reliable and Fault Detection Simon and Speck Cryptographic Algorithms on FPGA," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 16, no. 4, pp. 1–17, 2017.
29. N. K. Dumpala, S. B. Patil, D. Holcomb, and R. Tessier, "Loop unrolling for energy efficiency in low-cost field-programmable gate arrays," *ACM Transactions on Reconfigurable Technology and Systems*, vol. 11, no. 4, pp. 1–23, 2018.
30. K. Mohajerani, R. Haeussler, R. Nagpal, F. Farahmand, A. Abdulgadir, J. P. Kaps, and K. Gaj, "FPGA Benchmarking of Round 2 Candidates in the NIST Lightweight Cryptography Standardization Process: Methodology, Metrics, Tools, and Results," in *Cryptology ePrint Archive Paper 2020/1207*, 2020. [Accessed by 25/02/2025].
31. G. Avoine and P. Oechslin, "A scalable and provably secure hash-based RFID protocol," in *Proc. IEEE PerCom Workshops*, Kauai, Hawaii, United States of America, 2005.
32. D. Dinev and G. Spasova, "Benchmarking and Teaching Symmetric Cryptography: A Comparative Study of Algorithms and Cryptanalysis Methods with an Educational Simulator," in *Proc. 6th International Conference on Communications, Information, Electronic and Energy Systems (CIEES)*, Ruse, Bulgaria, 2025.
33. K. Gaj and P. Chodowicz, "FPGA and ASIC implementations of AES," in *Cryptographic Engineering*, Springer, Boston, Massachusetts, United States of America, 2009.

Publisher's Note: The publisher remains impartial concerning jurisdictional claims in published maps and institutional affiliations. Responsibility for the content rests entirely with the authors and does not necessarily reflect the publisher's perspectives.