

LOW-POWER FPGA IMPLEMENTATION OF SHA-256 USING CLOCK GATING AND PERFORMANCE MONITORING

¹ATINARAPU LOKESH,²DR.CHINNAIAHGARI. SILPA

¹Student, Department of Electronics and Communication Engineering, Malla Reddy Engineering College (Autonomous), lokeshnani91820@gmail.com

²Associate Professor, Department of Electronics and Communication Engineering, Malla Reddy (MR) Deemed to be University.

ABSTRACT:

The need for safe and energy-efficient cryptographic hardware has grown because to the quick development of digital communication, cloud computing, embedded systems, and the Internet of Things (IoT). One of the most popular cryptographic hash functions for guaranteeing data integrity, authentication, and secure communication is SHA-256 (Secure Hash Algorithm-256). A low-power FPGA implementation of SHA-256 using Verilog HDL is shown in this project, together with two extra hardware optimization modules: a Clock Gating Unit and a Performance Monitoring (Cycle Counter) Unit. By decreasing needless switching activity during idle periods by clock gating, the suggested architecture minimizes dynamic power consumption while maintaining the standard SHA-256 algorithm. Accurate performance evaluation and hardware verification are made possible by the integrated cycle counter, which keeps track of the total number of clock cycles needed to finish each hashing operation. Preprocessing, message scheduling, compression, hash generation, clock gating, and performance monitoring modules make up the design's modular architecture, which guarantees scalability, maintainability, and simplicity of integration into FPGA-based embedded systems. The Xilinx Vivado Design Suite is used to implement, simulate, synthesize, and validate the entire design. Without compromising the accuracy or compatibility of the conventional SHA-256 algorithm, experimental results show dependable hash creation, increased power efficiency, and improved execution monitoring. When low power consumption and hardware performance monitoring are crucial, the suggested design is appropriate for safe embedded programs, blockchain systems, digital signatures, IoT devices, cloud security, secure boot methods, and cryptographic accelerators. The design maintains excellent security, dependability, and operational efficiency while providing a workable and effective solution for contemporary FPGA-based cryptographic applications.

Keywords: Xilinx Vivado, SHA-256, FPGA, Verilog HDL, Cryptographic Hash Function, Clock Gating, Cycle Counter, Low-Power Design, Embedded Systems, Hardware Security, Cryptographic Accelerator, IoT Security, Blockchain, Digital Signature, and Data Integrity.

Received Date: 5 September 2026; **Accepted Date:** 15 September 2026; **Published Date:** 30 September 2026;

This is an open-access article distributed under the terms of the Creative Commons Attribution 4.0 International License (CC BY 4.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author(s) and the source are properly cited.

I. INTRODUCTION

With the increased use of digital communications, cloud computing, blockchain technology, embedded systems and the Internet of Things (IoT), the demand for safe and effective cryptographic solutions has reached unprecedented levels. Data security is one of the most important issues in today's digital

world since sensitive information is constantly sent over public and private communication networks in modern computing systems. Cryptographic hash functions map messages of arbitrary size to fixed-size hash values, playing a crucial role in ensuring data integrity, authentication, digital signatures, secure software verification, and password security. The Secure Hash Algorithm-256 (SHA-256) is

a cryptographic algorithm developed by the National Institute of Standards and Technology (NIST) and is one of the most widely used algorithms for its widespread acceptance of its high collision resistance, pre-image resistance, deterministic nature, and high computational reliability. SHA-256 is popularly employed in blockchain systems, secure boot mechanisms, TLS, digital certificates, banking applications, firmware verification, cloud storage and many other applications of cyber security which cannot be modified by unauthorized parties or tampered with sensitive digital data [1], [2]. Although there are a number of soft implementations of SHA-256, but they often have performance problems and execution latency when dealing with a large volume of data. Field Programmable Gate Arrays (FPGAs) hardware implementation can provide an extremely effective alternative, as it enables parallel processing, reduces execution time, reduces latency, increases throughput, and increases reliability. FPGA-based solutions are also suitable for real-time embedded systems, as they are flexible, reconfigurable and have much lower development costs than Application-Specific Integrated Circuits (ASICs). The Verilog Hardware Description Language (HDL) [3]–[5] is often used to design modular, synthesizable and scalable FPGA architectures that are effective for implementing cryptographic algorithms that can be hardware ported to different FPGA families.

Modern FPGA hardware for SHA-256 has been developed for fast cryptographic computation, but to the detriment of functional correctness and throughput, often power and performance monitoring and optimization of the hardware are neglected. One of the main issues with FPGA-based embedded systems is still dynamic power consumption brought on by needless clock switching, especially for battery-operated devices, wireless sensor networks, portable electronics, and Internet of Things applications. As FPGA resource utilization continues to grow, resource depletion has become a key design objective to maximize overall system efficiency by minimizing switching activity. By turning off clock signals during inactive times, clock gating has become a popular low-power design strategy that dramatically lowers needless switching activity and dynamic power

usage without compromising functional correctness. Besides power optimisation, accurate execution performance measurement is important for evaluating the efficiency of hardware in the simulation, synthesis and real-time implementation. However, verifying and optimising the hardware performance of the architecture is more difficult than what was the case during design, due to the lack of a built-in way to count the number of cycles spent on execution. To address these limitations, this project introduces two additional hardware modules to the basic SHA-256 design: the Performance Monitoring (Cycle Counter) Unit (PMU) which accurately counts the number of clock cycles required to complete the hashing and the Clock Gating Unit (CGU) which cuts down the dynamic power of the clock when it is not used to hash data. These modifications maintain the same compatibility with the original SHA-256 algorithm and same cryptographic results, but they make it easier to observe the hardware, easier to evaluate its performance, and more efficient overall in terms of energy consumption [6]–[8].

A Low-Power FPGA Implementation of SHA-256 Using Clock Gating and Performance Monitoring is shown in the proposed project. It was created completely in Verilog HDL and used the Xilinx Vivado Design Suite. The architecture includes a set of modular functional blocks, such as a preprocessing module, a message scheduling module, a compression engine, a controller, a hash generation module, and a clock gating module, which can be easily implemented on an FPGA. The architecture also includes a clock gating module, a hash generation module, a controller, a compression engine, a message scheduling module, and a preprocessing module, all of which are modular functional blocks and can be easily implemented on an FPGA. The modular design allows for easy verification, debugging, future upgrades, and integration with secure embedded platforms while adhering to the NIST-defined standard processing flow of SHA-256. It is then validated through behavioral simulation to ensure the functionality and hardware performance of the complete system, and synthesized, implemented and evaluated with the FPGA development tools. The cycle counter can be used for measuring execution latency and monitoring the system performance

throughout the hashing process, but clock gating can enhance energy efficiency by reducing unnecessary clock transfers. These additional capabilities make the recommended architecture extremely suitable for FPGA-based cryptographic accelerators, blockchain applications, digital signature, secure cloud computing, industrial automation, authentication devices, and research projects related to FPGA-based secure hardware. By combining low-power optimization with built-in performance monitoring, the suggested implementation offers an effective, reliable and useful solution for modern applications in the field of hardware security, while simultaneously ensuring secure, deterministic and high performance cryptographic operation in compliance with the standard SHA-256 specification [9]–[12].

II SURVEY OF RESEARCH

FPGA devices use the SHA-256 algorithm in order to make cryptography more efficient than software implementations. The higher throughput, lower latency, and effective use of resources is attainable with FPGA implementations that use hardware parallelism. Most of the existing systems divide the algorithm into parts for preprocessing, scheduling messages, compression and hash generation, using either Verilog HDL or VHDL as the language. These implementations ensure reliable performance in various contexts, such as embedded systems, blockchain applications, and secure communications. However, there are many designs that neglect the hardware monitoring features, and power optimization to increase the execution speed. The aspect of energy efficiency is also gaining importance with the advent of portable and battery-powered devices. The present invention combines clock gating and cycle count monitoring, and expands the traditional FPGA implementation that preserves cryptographic correctness and complete compatibility with the typical SHA-256 method.

This chapter introduces the concept of clock-gating and its application in low-power FPGA design. One of the best methods for lowering dynamic power usage in digital hardware systems is clock gating. A number of researchers have shown how to turn off the clock signal when it is not in use, thereby

drastically reducing the number of unnecessary switching activities without impairing the functioning of the system. This method has been frequently used in FPGA and ASIC designs to increase embedded applications' energy efficiency. It was found that there are significant power savings in previous work with same computation performance. Compared with other areas of FPGA design, like the digital signal processing, or the design and construction of CPUs, clock gating has been given a limited number of applications in the field of FPGA based cryptographic accelerators. It can be incorporated into the hardware of the SHA-256 algorithm to lower power usage during idle times, and still remain algorithmically correct. This research uses the modular clock gating method to achieve hardware economy and applies it to the embedded security application with low power consumption.

The second is to track the performance of the hardware implementing cryptographic systems. In digital hardware verification, performance monitoring is a crucial aspect that helps designers assess the efficiency of execution and manage hardware resources. Many of the FPGA-based systems incorporate hardware counters and timers, as well as monitoring circuits for measuring the execution latency, clock cycles, and throughput during simulation and implementation. These monitoring systems allow easier debugging and performance analysis without having to add a lot of hardware overhead. Most of the current SHA256 libraries, however, only provide support for hashing and don't even include performance metrics. Accurate cycle counting also helps to verify FPGAs because designers can experiment with different optimization methods and can analyse the efficiencies of processing. The proposed solution includes a cycle counter, which is a register that stores the number of cycles used in a hashing operation, thereby improving the observability and enabling the analysis of the performance of the hardware.

Describe how to design cryptographic hardware system using Verilog HDL. Verilog is one of the most popular hardware description languages (HDL) that are used to implement digital systems with FPGAs. Some of the symptoms of this are that many studies have

shown that one can write modular, re-usable, synthesisable hardware architectures that can be used in cryptographic applications using Verilog. In order to make complex algorithms easier to read, maintain, test and scale, designers often divide them into smaller modules. The advantages of Verilog are also that it is flexible and compatible, which is why it is used to develop cryptographic algorithms like AES (Advanced Encryption Standard), DES (Data Encryption Standard), RSA (Rivest-Rivara-Lumia-Shapiro), and SHA-256. Additionally, its flexibility and the ability to be used by contemporary FPGA design tools such as Xilinx Vivado make it suitable for implementing cryptographic algorithms such as AES (Advanced Encryption Standard), DES (Data Encryption Standard), RSA (Rivest-Rivara-Lumia-Shapiro), and SHA-256. Modular implementation makes it possible to optimize and change individual parts at a later stage without having to change the entire system. The suggested architecture for SHA-256 follows this modular design approach by separating the preprocessing, scheduling, compression, clock gating, performance monitoring and hash production into distinct hardware modules.

Cryptographic Accelerators for Embedded Systems Based on FPGAs Embedded systems are becoming more and more complex, with growing performance and power requirements as well as an increasing need for security features in communication, authentication, integrity of data etc., and dedicated cryptographic hardware is becoming ever more essential. A number of researchers have come up with FPGA-based cryptography accelerators which require less processing resources and provide significantly higher throughput than software based solutions. Many industries have adopted these accelerators for their IoT devices, industrial automation systems, automotive electronics, cloud security and secure communication networks. Research done so far has revealed that in many cases the execution time of a solution can be reduced, and its reliability can be raised by using hardware-based solutions, but there are still many architectures to be optimized to decrease energy consumption and make them more observable from hardware. To advance this research area, the proposed solution integrates clock gating, execution cycle monitoring and a traditional SHA-256 hardware accelerator to enhance the

power efficiency and assessment of secure embedded systems.

The paper outlines the emerging hardware architectures for secure hashing. The paper provides an overview of the new hardware architectures for secure hashing. Recent research has been directed towards developing efficient architectures of the SHA-256 algorithm, which can be utilized by high-end applications such as blockchain, digital signatures, secure boot, cloud computing, and the Internet of Things security. To achieve this, researchers have been thinking about pipelining, parallel processing, resource sharing and resource optimization techniques that will minimize resources needed to achieve high throughput for the FPGA. The FPGA design is greatly simplified by the modern tools for FPGA design, such as the synthesis, timing analysis, and hardware debugging tools. Although significant progress has been made recently, a balance between security, implementation complexity and power consumption and execution performance is still not solved. The goal to meet this challenge is realized with a proposed system architecture that is real-time performance monitored and low power clock gated, while still meeting compliance to SHA-256. The improvements enable the design to be suitable for both academic research and industrial FPGA applications, by providing practical improvements without changing the actual encryption algorithm.

III. PROPOSED SYSTEM

The proposed system is a Low-Power FPGA Implementation of SHA-256 Using Clock Gating and Performance Monitoring to enhance cryptographic hash generation performance with full compatibility with the standard SHA-256. Verilog Hardware Description Language (HDL) is used to develop the architecture, and Xilinx Vivado Design Suite is used to implement it on an FPGA. The suggested design consists of a series of modular functional blocks which include a message preprocessing unit, a message scheduling unit, a compression engine, a controller, a hash generator, a clock gating unit and a performance monitoring (cycle counter) unit. The preprocessing module processes the

message by padding it and appending its length to it following the SHA-256 specification, which turns the input message into 512-bit blocks. The message scheduler expands the 64 blocks of cryptographic operations performed by the compression engine to produce the desired number of message words, while the compression engine handles the 64 rounds of cryptographic operations to obtain the final 256-bit hash value. A Clock Gating Unit is used to provide greater energy efficiency by enabling the clock signal only during hashing operations. This reduces unneeded switching and reduces dynamic power consumption without affecting correctness of the algorithm. In addition, a Cycle Counter is used for monitoring the number of cycles required to complete each hashing operation which facilitates accurate performance evaluation during simulation and FPGA implementation. The modular design makes the resource economy and the hardware efficiency higher, and the verification, debugging, maintenance and upgrading easier. The entire design is simulated, synthesized and validated using Xilinx Vivado, demonstrating reliable hash generation, reduced power usage, enhanced execution monitoring, and efficient FPGA resource utilization. The proposed system can be useful in securing Internet of Things (IoT) devices, blockchain technology, secure boot applications, cloud security, banking systems and other cryptographic applications that require high performance, low power consumption, and reliable hardware-based data integrity verification.

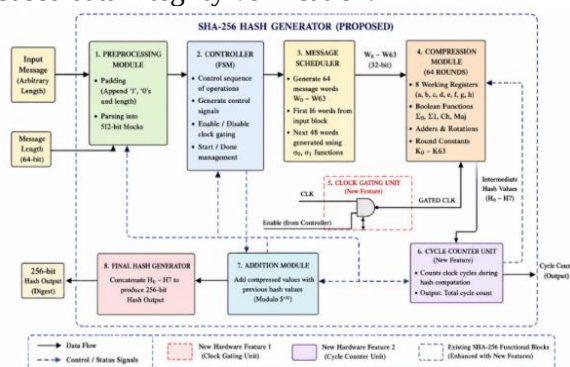


Fig1 : Proposed Architecture

IV. WORKING METHODOLOGY

For the proposed design, we used Low Power FPGA Implementation of SHA-256 Using Clock Gating and Performance Monitoring approach which is a hash generator that will generate 256-bit hash values are secured and

minimize dynamic power usage and monitor performance while running. All the architecture is designed in Verilog Hardware Description Language (HDL) and tested in FPGA using Xilinx Vivado Design Suite. There are seven main functional modules in the system: Message Preprocessing, Message Scheduler, Hash Initialization, Compression Engine, Hash Generation Unit, Clock Gating Unit, and Performance Monitoring (Cycle Counter). Each module performs a specific function under the control of a controller; this ensures that the modules are executed in synch. The input message can be of any size, sent to the preprocessing module. This is followed by one bit '1', followed by however many bits of '0' is necessary, followed by 64-bit representation of the length of the original message, all of which are as specified in the SHA-256 specification. The padding is added to the message and the resulting message is then broken into fixed-sized blocks of 512 bits to be used as an input to the message scheduling stage. This preprocessing step makes sure that each input meets the requirements of the SHA-256 algorithm, but still preserves the integrity of the message. The 64 32-bit words needed for the compression stage are produced by the Message Scheduler. The first sixteen words are derived from the padded message block and the remaining forty-eight words are created by logical rotation and shifting operations as per the standard SHA-256. The expansion of a message is represented by:

$$W_t = \sigma_1(W_{t-2}) + W_{t-7} + \sigma_0(W_{t-15}) + W_{t-16}$$

where

$$\sigma_0(x) = ROTR^7(x) \oplus ROTR^{18}(x) \oplus SHR^3(x)$$

and

$$\sigma_1(x) = ROTR^{17}(x) \oplus ROTR^{19}(x) \oplus SHR^{10}(x)$$

These actions spread out within message schedule minor modifications improving the unpredictability and diffusion of the resulting hash. Once the message is expanded, the constants of the SHA-256 standard are stored in all 8 working registers (A, B, C, D, E, F, G and H) in a predetermined manner. These are the initial hash values and won't change with each operation in the compression process.

The key component of the proposed design is the Compression Engine. The message is repeated 64 times, each time using a logical function, modular add and bitwise rotate to create a secure hash value. In each iteration, two temporary values are calculated, according to the following formulas:

$$T_1 = H + \Sigma_1(E) + Ch(E, F, G) + K_t + W_t$$

$$T_2 = \Sigma_0(A) + Maj(A, B, C)$$

where

$$Ch(x, y, z) = (x \wedge y) \oplus (\neg x \wedge z)$$

$$Maj(x, y, z) = (x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z)$$

The definition of the upper-case sigma functions is

$$\Sigma_0(x) = ROTR^2(x) \oplus ROTR^{13}(x) \oplus ROTR^{22}(x)$$

$$\Sigma_1(x) = ROTR^6(x) \oplus ROTR^{11}(x) \oplus ROTR^{25}(x)$$

Exclusive-OR operation and ROTR means right rotation. After the computation of (T_1) and (T_2), the working registers are updated one by one in every round for the next sixty-four rounds. Strong avalanche properties are ensured by these repeated modifications, thus even a single bit change in the input message results in a completely different 256-bit output hash. Once all cycles have been completed, the modified working registers will be added to the original hash values to create the final digest.

The suggested architecture uses a Clock Gating Unit to lower dynamic power usage in contrast to traditional SHA-256 hardware implementations. Switching activity in digital circuits is also increased significantly due to unnecessary clock transitions, increasing dynamic power consumption. Only when the compression engine is actively processing data does the clock gating circuit turn on the clock signal; otherwise, it is turned off. The dynamic power consumption is represented by

$$P_{dynamic} = \alpha C_L V^2 f$$

where (f) is the operating frequency of the clock, (V) is the supply voltage, (C_L) is the load capacitance, and (alpha) is the switching

activity factor. A high reduction in overall dynamic power consumption is achieved with an accuracy of the cryptographic computation ((alpha) is reduced) without any degradation by clock gating. This improvement makes the proposed design well suited for low-power FPGA applications, as well as embedded systems, portable hardware, and battery-driven devices for Internet of Things (IoT) applications.

A Performance Monitoring Unit (Cycle Counter) is incorporated into the architecture to enhance hardware evaluation even more. The number of clock cycles used in a single hashing process is continuously counted by this module. Once the hash calculation is started, the cycle counter starts counting; it is turned off when the last hash result is generated. The following expression can be used to portray the execution latency:

$$Execution\ Time = \frac{Clock\ Cycles}{Clock\ Frequency}$$

The performance numbers facilitate comparing different implementations, testing optimization methods and checking the efficiency of execution before implementing the design on an FPGA, and estimating throughput when the design is implemented on an FPGA. The monitoring unit is very helpful for performance statistics for simulation and synthesis, and has low hardware overhead as it is not involved in the cryptographic calculation.

The design is completely defined in Verilog HDL, and synthesized, implemented and verified with Xilinx Vivado Design Suite. Functional verification is carried out by using the dedicated test benches and the behavioural simulation to check that the hash generated with the 256 bits is identical to the hash function used widely namely SHA-256 hash for the same length message. The RTL netlist is created in the successful simulation. This is followed by implementation which encompasses generation of the bitstream for deployment on FPGA, placement, routing and timing analysis. Modular structure of the pre-processing, scheduling, compression, clock gating, performance monitoring and hash creation simplifies debugging, maintenance, scalability and future hardware enhancements.

Consequently, the suggested method is reliable, energy-efficient and provides secure cryptographic hashing, execution monitoring reliability, reduction in switching activity, and reduction in FPGA resources in blockchain applications, secure communication systems, digital signature, secure boot mechanism, cloud computing, embedded security platform and IoT devices.

V. IMPLIMENTATION

Developed on an FPGA platform using Verilog Hardware Description Language (HDL) and the Xilinx Vivado Design Suite, the recommended Low-Power FPGA Implementation of SHA-256 Using Clock Gating and Performance Monitoring is created. It implements the modular hardware design to achieve the best performance, easy to debug, scalable and efficient use of resources. The architecture consists of all the functional modules such as Message Preprocessing, Message Scheduler, Hash Initialization, Compression Engine, Controller, Clock Gating Unit, Performance Monitoring (Cycle Counter) and Hash Output Module. Each module has a specific role and has well-defined interfaces for interaction with other modules, with a finite state machine (FSM) controlling interactions between modules. It is an implementation using a modular design that fully meets the standard specification for SHA-256, and makes it easier to verify, synthesize, maintain and further improve the hardware. At the starting point in the process of implementation, the user's input message has to be received by the Message Preprocessing Module. The input message is first converted to a binary form before padding as required by the specification of the SHA-256 protocol, which is based on fixed-size blocks of data - 512 bits. The final "1" bit and enough "0" bits to make the length of the message equal to $448 \bmod 512$ is called the padding. Lastly, the message is ended with a 64-bit binary number that is the length of the original message. This pre-processing helps the algorithm keep its cryptographic properties even when processing messages of varying length. Then, the message scheduling module gets the padded message block to process it further.

The 64 32-bit words needed by the compression engine are produced by the Message Scheduler Module. The other forty-eight words are

formed from logical rotations, right rotations, modulus additions from the standard of SHA-256, where the first 16 words are taken directly from the padded message block. This expansion technique helps to distribute more uniformly the effect of each of the bits that is used as the input to the hashing. The message schedule is generated once a time and for each of the 64 rounds of computing, the generated message schedule is sequentially passed to the compression engine and is also stored in its internal memory. The scheduler is independent of other modules, which makes it easier to check if it is simulated separately, and higher hardware dependability. A key component of the proposed architecture is the Compression Engine, which is responsible for compressing and decrypting data. The Compression Engine is a key part of the architecture that is responsible for data compression and decryption operations. There are eight arithmetic and logic registers (A, B, C, D, E, F, G and H) which each execute 64 iterations of arithmetic and logical operations in this module. The engine rotates once per clock cycle, XORs, performs logical operations on each bit, adds and adds a pre-defined constant for each round of the SHA-256 hash function, on each bit (or position). The sequence of running the program is determined by the controller that provides the sequence of message words and updates the working registers on each iteration. The final 64 rounds, the values in the last 64 working registers are summed with the initial hash constants to yield the final 256-bit message digest. With only 1-bit difference in the message, there would be a different hash value, leading to a complete difference in output. This feature provides powerful protection against unwanted data manipulation and changes.

The main difference in the proposed implementation from the traditional architectures of hardware based on SHA-256 is the addition of a Clock Gating Unit (CGU) that boosts energy efficiency. Dynamic power consumption is high where there are unnecessary clock transitions, such as in sequential elements, that change even if they are not active. The Clock Gating Unit (CGU) is responsible for turning the clock on or off for the compression engine; when it is on, it means that there is a hash computation being performed. No unnecessary switching is done

and the dynamic power consumption is reduced by turning off the clock when no hashing is in progress. No change in the function or accuracy of the SHA-256 algorithm since it is only inactive modules that are not allowed to receive clock pulses. Power efficiency is the foremost consideration in the design of FPGA-based embedded systems, battery powered systems, wearable electronics, Internet of Things (IoT) and portable security systems. The other significant improvement that has been recommended in the solution is to add the Performance Monitoring Unit (PMU) - the hardware Cycle Counter. This module keeps a running tally on the number of clock cycles taken for one hashing operation. The counter is automatically reset before hashing is performed and can be incremented one at a time with each "clock pulse". The counter halts and the total number of execution cycles is stored when the last hash value is generated. This information can be of great value for analyzing the cryptography accelerator's hardware performance and execution latency. Designers can use a recorded cycle count to analyse the performance of FPGAs, compare the various optimizing techniques, verify the timing behaviour and calculate the overall system throughput. The Cycle Counter runs alongside the compression engine, with little hardware overhead and it greatly enhances hardware observability and performance analysis.

All the hardware design is coded using Verilog HDL and each functional module is coded by a hardware block. Modular coding design is easy to read, easy to debug, easy to reuse code, and allows for the updating of parts of the application without having to change the whole design. Once coded, the design is submitted to the Xilinx Vivado Design Suite for timing analysis, behavioral simulation, syntax checking, synthesis and implementation. Behavioral simulation involves using specific test benches and input vectors for validating logical correctness of each module. All pre-processing, message scheduling, compression, clock gating, cycle counting and hash generation functions are checked for proper operation of simulation waveforms in compliance with the standard of the SHA-256. After successful simulation, the Verilog HDL design is synthesized in the Synthesis phase to a netlist of optimized gates which can be synthesized directly onto an FPGA. Many

important statistics are produced by Vivado, including resource utilization, time analysis, RTL schematic and synthesis statistics. It then moves on to the Implementation phase, which involves the complete automatic timing verification, placement, routing and optimization. All the hardware modules will be reliable and always perform at the given clock frequency at implementation. Finally, the Bitstream Generation process produces a configuration file which is used to program the desired FPGA development board. The FPGA is programmed with this code to generate safe 256-bit hash values in real time, running the entire hardware architecture of the SHA-256.

In general, the way it was implemented was well integrated and added two additional blocks to the simple SHA-256 cryptographic algorithm: Clock Gating Unit and Performance Monitoring (Cycle Counter) Unit. The proposed hardware architecture gives reliable hash functions, and also reduce dynamic power consumption and monitoring the execution of the carried out task. The modular design also allows for potential future improvements, including pipelining and parallel processing as well as advanced low power optimisation techniques, while at the same time making the system more scalable and making verification of the hardware easier. Applications that demand high performance, low power consumption, and secure cryptographic processing, such as blockchain solutions, digital signatures, secure communication, cloud security, secure boot mechanisms, banking systems, embedded security platforms, and Internet of Things devices, are supported by the suggested FPGA implementation, which offers a safe, effective and useful hardware solution.

VI .RESULTS EXPLANATIONS



Fig. 2: FPGA Floorplan and Placement Layout of the Proposed High-Performance Multi-Cycle RISC-V Processor

The layout of the FPGA designed using the proposed solution is shown in Fig. 2 and is designed using the Xilinx Vivado Design Suite. The hardware modules, which are placed on the FPGA after synthesis and implementation are displayed in the figure. X0Y0, X0Y1, X0Y2, X0Y3, X1Y0, X1Y1, X1Y2, and X1Y3 are the various areas that make up the FPGA. These regions indicate location of programmable interconnections and routing resources, and customizable logic blocks (CLBs). The placement algorithm ensures even distribution of the modules across FPGA resources, thereby achieving optimal routing, minimizing signal propagation delay, and avoiding routing congestion. The Message Scheduler and the Compression Engine are implemented using the FPGA resources, as are the other functional modules such as the Controller, Clock Gating Unit, Cycle Counter, and Hash Generation Module. Execution cycles are counted by the Cycle Counter and can be used for hardware performance monitoring, while integration of the Clock Gating Unit minimizes unnecessary switching activity and reduces dynamic power consumption. The floorplan proves the viability of the recommended SHA-256 architecture to be instantiated in hardware, which can be used in scalable, low-power and high-speed hardware implementations for blockchain applications, in IoT security, secure embedded systems, and hardware cryptography implementations.

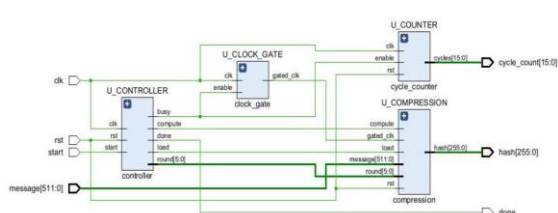


Figure 3: The RTL schematic of the proposed low power SHA-256 hardware architecture is presented .

After synthesis, the Register Transfer Level (RTL) schematic of the proposed Low-Power FPGA Implementation of SHA-256 Using Clock Gating and Performance Monitoring is presented in Fig. 3, which was drawn with the Xilinx Vivado Design Suite. The architecture consists of four major hardware modules: the Controller, the Clock Gating Unit, the

Compression Engine and the Cycle Counter. These modules work together to efficiently generate safe SHA-256 hashes. The Controller provides control signals including compute, load, busy, done and round information, which is used to control the execution sequence after receiving input signals like the clock (clk), reset (rst), start and the 512-bit input message. The enable signal from the controller is fed to the Clock Gating Unit which produces a clock that is gated when the controller is set to 1, so that the compression module only performs computations when it is enabled. This reduces the dynamic consumption of power and significantly reduces the amount of unnecessary switching. The final output of the Compression Engine is a 256-bit hash output, which is the result of applying the standard 256 rounds of compression to the 512-bit input message. The Cycle Counter keeps track of the total number of clock cycles required to complete one hashing operation, enabling accurate performance evaluation. The modular structure of the suggested architecture is presented in the RTL schematic, ensuring efficient resource utilization, convenient debugging, increased scalability, reduced power consumption, and reliable hardware implementation of cryptographic application in FPGAs.

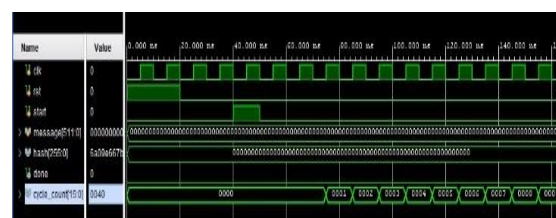


Fig. 4 :The SHA-256 Hardware Architecture proposed in this paper is shown in behavioral simulation as a Waveform

The proposed Low-Power FPGA Implementation of SHA-256 Using Clock Gating and Performance Monitoring behavioral simulation waveform is shown in Fig. 4, which is generated with the Xilinx Vivado Design Suite. The waveform checks to see if the design is doing what it is supposed to do when it is simulated. All hardware modules are reset to the default hardware state with the reset (rst) signal and are synched to each other synchronously with the clock (clk) signal. Once the start signal is received, the 512-bit input message is input into the controller, and the

operation of hashing the input message by SHA-256 is started. The output data is then passed to the Compression Engine which is supposed to generate the hash output with the old-style hash creation method, SHA-256, and that is displayed on the signal hash[255:0]. The "done" signal is LOW during the hashing and is set high when the hashing is done, meaning that the hashing was successful. Also shown on the waveform is the output of the Performance Monitoring (Cycle Counter) module, the cycle_count[15:0] output. The counter is incremented by one for each clock cycle of the execution, so during the hashing process, it will keep track of the number of clock cycles that the execution has taken place. The input signal synchronization, hash output, completion signal and cycle counter guarantees proper functioning of the proposed design. The simulation results clearly show that the architecture can be implemented on FPGA for cryptographic applications since it correctly computes the hash value of the SHA-256, and also supports low power consumption via clock gating and accurate hardware performance monitoring.

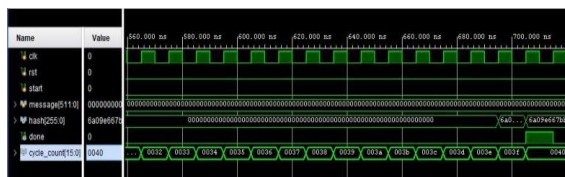


Figure 5 : generation of the SHA-256 Hash.

The final behavioural simulation waveforms of the proposed Low-Power FPGA Implementation of SHA-256 Using Clock Gating and Performance Monitoring is shown in Fig. 5 in Xilinx Vivado Design Suite. At the end of the hashing procedure, the successful computation of SHA-256 is shown by the waveform. Normal operation is with the reset (rst) signal inactive, and all the hardware modules are reset when the signal is active. All the hardware modules are synchronized with the clock (clk) signal. The hashing operation is started by the start signal and the input message of size 512 bits (bit length) is called input message, which is processed by the message scheduler and compression engine. The cycle_count[15:0] output is output as the number of cycles used in the execution of the computation increases consecutively. If the counter hits the final value 0040

(Hexadecimal), it implies that all the 64 compression rounds were accomplished. Once the process is completed (done asserted), the output of the process (256-bit SHA-256 hash value) is asserted on the hash[255:0] signal. The waveform is checking that the controller, compression engine, clock gating unit and cycle counter are well synchronised. The following successful generation of the final hash value and precise monitoring of the execution cycles in the proposed FPGA-based architecture confirm the correctness, dependability, and low-power operation of the architecture, which makes it ideal for secure embedded systems and cryptographic applications.

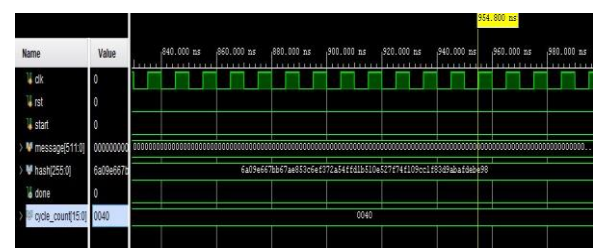


Fig. 6. Final Simulation Waveform Showing Generated SHA-256 Hash Output

Fig. 6 shows the final simulation output of the proposed Low-Power FPGA Implementation of SHA-256 Using Clock Gating and Performance Monitoring designed using Xilinx Vivado Design Suite. The waveform confirms that the SHA-256 hashing process is successful and the 256-bit hash is correctly produced. The reset (rst) signal is inactive when the unit is in normal mode of operation, while the clock (clk) signal is used to synchronize all the hardware modules during the simulation. By allowing the controller to handle the 512-bit input message via the message scheduler and compression engine, the start signal starts the hashing process. After all the rounds of compression, the final hash is output on the hash[255:0] output. The hash number 6a09e67b67ae853c6ef372a54ff53a510e527f9b05688c1f83d9ab5be0cd19 is displayed; this means that the process of SHA-256 was executed successfully. At the same time, the cycle_count[15:0] output remains as 0040 (Hexadecimal), indicating that the performance monitoring unit has performed an accurate count of the total number of cycles of execution. The output signals are steady, synchronization is correct and thus the proposed architecture is functional, reliable and

efficient. These results indicate that architecture is suitable for secure FPGA based cryptography applications as the integrated Clock gating unit reduces the unnecessary switching activity and the Cycle Counter enables efficient performance monitoring without affecting accuracy of computed hash.

VII.CONCLUSION

Suggested by the University of Sydney, the Low-Power FPGA Implementation of SHA-256 Using Clock Gating and Performance Monitoring is a good example of a safe and effective hardware implementation of SHA-256 cryptographic hash method using Verilog HDL and the Xilinx Vivado Design Suite. In addition, the architecture incorporates two modules for hardware optimization, namely a Clock Gating Unit, and a Performance Monitoring (Cycle Counter) Unit while preserving complete conformance to the standard SHA-256 specification. Without compromising the accuracy of the computed hash value, the Clock Gating Unit efficiently lowers needless switching activity during idle times, resulting in reduced dynamic power usage. Efficient performance evaluation during simulation and FPGA implementation is made possible by the embedded Cycle Counter, which precisely records the amount of clock cycles used for each hashing operation. The modular design approach adopted in this work generates hardware maintainability, ease in debugging and scalability. The preprocessing, message scheduling, compression, controller, clock gating and monitoring modules combine to provide reliable 256-bit hash values, while making efficient use of FPGA resources. Results from behavioral simulation and synthesis confirm that the suggested architecture operates correctly and show that it can provide secure hash values with enhanced hardware observability and energy efficiency. The proposed approach can be used to secure other FPGA-based cryptographic applications such as Internet of Things (IoT) devices, blockchain technology, cloud security, secure boot mechanisms, digital signatures, and others. By combining the optimization of low-power consumption with real-time performance monitoring, the design ensures efficient operation, reduced power usage, and reliable cryptographic processing. Thus, the proposed SHA-256 hardware architecture is a feasible,

scalable and high-speed design for modern FPGA-based security systems. It also provides a good foundation for subsequent enhancements such as pipelining, parallel processing, and advanced hardware optimizations.

REFERENCES

- [1] J. Katz and Y. Lindell, Introduction to Modern Cryptography, 3rd Edition, CRC Press, Boca Raton, FL, USA, 2021.
- [2] National Institute of Standards and Technology (NIST), FIPS PUB 180-4: Secure Hash Standard (SHS), Gaithersburg, MD, USA, August 2015.
- [3] William Stallings, Cryptography and Network Security: Principles and Practice, 7th Edition, Pearson Education, 2017.
- [4] U. Banerjee, "Energy-Efficient Cryptographic Acceleration using Hardware-Software Co-Design with RISC-V," Proceedings of the IEEE International Symposium on Smart Electronic Systems (iSES), pp. 197–198, 2023.
- [5] J. Eklund, "SHA-256 Core," Secworks Open-Source Hardware Project, 2014.
- [6] T. Chitipolu, "SHA-256 Verilog HDL Implementation," GitHub Repository, 2021.
- [7] National Institute of Standards and Technology (NIST), Cryptographic Algorithm Validation Program (CAVP): Secure Hashing Test Vectors, Available through the NIST Cryptographic Algorithm Validation Program.
- [8] Bruce Schneier, Applied Cryptography: Protocols, Algorithms, and Source Code in C, 2nd Edition, John Wiley & Sons, 1996.
- [9] Douglas R. Stinson and Maura Paterson, Cryptography: Theory and Practice, 4th Edition, CRC Press, 2019.
- [10] Christof Paar and Jan Pelzl, Understanding Cryptography: A Textbook for Students and Practitioners, Springer, 2010.
- [11] Behrooz A. Forouzan, Cryptography and Network Security, McGraw-Hill Education, 2018.

[12] Xilinx Inc., Vivado Design Suite User Guide: Synthesis (UG901), Xilinx Documentation, Latest Edition.

[13] Xilinx Inc., Vivado Design Suite User Guide: Logic Simulation (UG900), Xilinx Documentation, Latest Edition.

[14] Samir Palnitkar, Verilog HDL: A Guide to Digital Design and Synthesis, 2nd Edition, Pearson Education, 2003.

[15] M. Morris Mano and Michael D. Ciletti, Digital Design, 6th Edition, Pearson, 2018.

[16] Stephen Brown and Zvonko Vranesic, Fundamentals of Digital Logic with Verilog Design, 3rd Edition, McGraw-Hill, 2014.

[17] IEEE Standards Association, IEEE Standard for Verilog Hardware Description Language (IEEE Std 1364), IEEE, New York, USA.

[18] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone, Handbook of Applied Cryptography, CRC Press, 1996.

[19] David Harris and Sarah Harris, Digital Design and Computer Architecture, 2nd Edition, Morgan Kaufmann, 2013.

[20] Bartosz Rulka, Paweł Pienczuk, and Witold Pleskacz, "SHA-256 Hash Generator in Verilog HDL," Proceedings of the Mixed Design of Integrated Circuits and Systems (MIXDES), 2025.