International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347 - 6435 (Online) Volume 14, Issue 3, March 2025)

Modified Design Flow of Blowfish Symmetric-Key Block Cipher on FPGA

Ravikant Prasad[1], Prof. Suresh S. Gawande[2]
[1]Research Scholar, Dept. of Electronics & Communication Eng., Bhabha Engineering Reasearch Institute, Bhabha University Bhopal, India
[2]Head of Department, Dept. of Electronics & Communication Eng., Bhabha Engineering Reasearch Institute, Bhabha University Bhopal, India

***Abstract*— The increasing demand for secure data transmission and storage in embedded systems and real-time applications has driven the adoption of efficient cryptographic algorithms. Among these, Blowfish, a symmetric-key block cipher, remains relevant due to its simplicity, flexibility, and low hardware implementation requirements. However, implementing Blowfish on FPGA platforms, especially with a 128-bit input block size, presents challenges related to resource utilization, key expansion performance, and throughput optimization. This paper presents a modified design flow for the efficient implementation of the Blowfish algorithm on FPGA, with a focus on optimizing the 128-bit input size for hardware applications. The proposed flow addresses key challenges such as limited FPGA resources, power consumption, and data throughput while enhancing the performance of the Blowfish cipher through parallelism, pipelining, and custom hardware units. We explore the design of specialized components, including optimized S-boxes and key expansion units, to accelerate encryption and decryption processes. Additionally, the paper discusses the integration of security countermeasures to protect against side-channel attacks, which are particularly relevant in FPGA-based cryptographic systems. The modified design flow is evaluated in terms of performance, resource utilization, and security, demonstrating a robust and efficient Blowfish implementation suitable for real-time cryptographic applications in embedded and IoT systems.**

Keywords— Modified, Blowfish, Cryptography, Delay, Power.

I. INTRODUCTION

Cryptography plays a crucial role in securing digital data against unauthorized access, tampering, and malicious attacks, especially in resource-constrained environments like embedded systems, IoT devices, and communication networks. As the demand for secure data transmission and storage increases across various industries, the need for efficient cryptographic algorithms becomes paramount. Among the many cryptographic algorithms, symmetric-key block ciphers such as AES (Advanced Encryption Standard), DES (Data Encryption Standard), and Blowfish are widely used for their speed and security. While AES has become the de facto standard for many applications, Blowfish continues to be relevant due to its simplicity, flexibility, and lower hardware implementation requirements.

Blowfish, designed by Bruce Schneier in 1993, operates on 64-bit blocks and supports key sizes ranging from 32 bits to 448 bits, making it adaptable to various levels of security. Its structure consists of multiple rounds of encryption using a Feistel network, which allows it to perform efficient encryption and decryption while offering a high degree of security. However, implementing Blowfish in software or hardware, particularly in FPGA (Field-Programmable Gate Array) platforms, presents unique challenges. These challenges arise from the need to optimize resource utilization, minimize power consumption, and maximize throughput while ensuring the algorithm's robustness against attacks.

FPGA-based hardware acceleration has gained significant traction in cryptographic applications because of its ability to provide parallel processing, customization, low-latency execution, and high throughput. FPGAs are reconfigurable hardware platforms that allow designers to implement custom circuits tailored to specific algorithms. Unlike general-purpose processors, FPGAs enable designers to exploit parallelism, optimizing the performance of cryptographic algorithms like Blowfish by distributing the workload across multiple processing units. This ability to create specialized hardware for encryption tasks makes FPGAs an attractive choice for applications requiring both speed and efficiency, such as secure communication, embedded systems, and real-time data encryption.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347 - 6435 (Online) Volume 14, Issue 3, March 2025)

However, implementing the Blowfish algorithm on an FPGA requires a carefully designed flow that accounts for several factors: resource constraints, key expansion performance, data throughput, and security considerations. While the standard Blowfish implementation is efficient in software, it requires significant optimization when translated to hardware to ensure that it performs effectively on FPGA devices. This is where the need for a modified design flow arises. A modified design flow optimizes the Blowfish algorithm for FPGA hardware by introducing techniques such as parallelism, pipelining, and customized hardware units. This modified flow aims to reduce resource usage, enhance performance, and improve the algorithm's overall efficiency when deployed in embedded systems with limited computational power and memory.

The process of designing a modified Blowfish architecture for FPGA typically involves several key stages. First, the algorithm is analyzed to identify potential bottlenecks and areas for optimization. These areas might include the key expansion process, which is computationally intensive, and the Feistel network's multiple rounds, which require careful management of data and resources. The next step involves designing a custom hardware architecture that can implement Blowfish's operations efficiently. This can include designing specialized S-boxes, key expansion units, and encryption modules that leverage FPGA's parallel processing capabilities. Once the architecture is defined, various optimization techniques, such as pipelining and parallelization, are employed to improve throughput and minimize latency.

This paper explores the modified design flow of the Blowfish symmetric-key block cipher on FPGA by examining each of these stages in detail. We discuss how the standard Blowfish algorithm can be adapted and optimized for FPGA platforms through custom hardware architectures and design techniques. Furthermore, we highlight the challenges faced during implementation, including resource constraints, power consumption, and security vulnerabilities, and propose solutions to address these issues. Through this modified design flow, the goal is to achieve a highly efficient and secure Blowfish implementation suitable for real-time cryptographic applications in embedded and resource-constrained systems.

II. PROPOSED METHODOLOGY

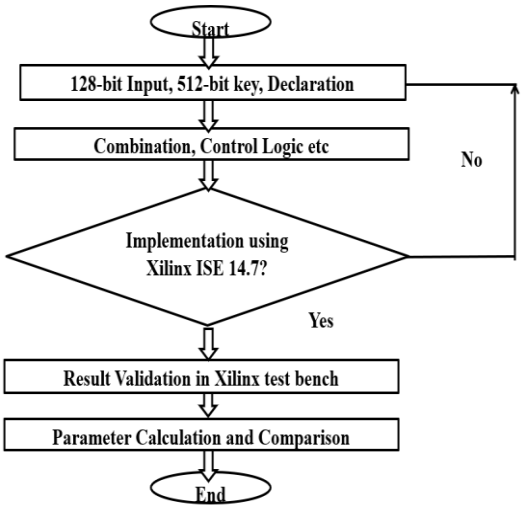

Figure 1:
This flowchart outlines the design and implementation process of a cryptographic system, likely a symmetric-key block cipher like Blowfish. The system takes a 128-bit input and uses a 512-bit key for encryption/decryption.

Step-by-Step Breakdown:

1. **Start:** The process begins here.
2. **128-bit Input, 512-bit key, Declaration:** This step involves defining the input data size (128 bits) and the key size (512 bits) for the cryptographic algorithm. Additionally, any necessary variables or parameters are declared.
3. **Combination, Control Logic etc:** This block encompasses the core logic of the cryptographic algorithm. It includes:

   Key Scheduling: The process of generating round keys from the initial 512-bit key.

   Encryption/Decryption Rounds: The iterative steps involved in transforming the plaintext into ciphertext (encryption) or vice versa (decryption). These rounds typically involve bitwise operations, substitutions, and permutations.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347 - 6435 (Online) Volume 14, Issue 3, March 2025)

Control Logic: This part handles the sequencing of the encryption/decryption process, including round counters and state management.

4. **Implementation using Xilinx ISE 14.7?:** This decision point checks whether the implementation is targeted for Xilinx ISE 14.7, a specific FPGA design software.

   Yes: If yes, the process proceeds to the next step using Xilinx ISE 14.7 for hardware synthesis and implementation.

   No: If no, the implementation would use a different FPGA design tool or a software-based approach.

5. **Result Validation in Xilinx test bench:** After implementation, the design is tested using a Xilinx test bench. This involves applying various input-key pairs and verifying that the output matches the expected encrypted/decrypted data.
6. **Parameter Calculation and Comparison:** The final step involves calculating key performance parameters (e.g., throughput, latency, area utilization) of the implemented cryptographic system. These parameters are then compared to design goals or other implementations to evaluate the efficiency and effectiveness of the design.
7. **End:** The process concludes here.

III. RESULTS AND DISCUSSION

The implementation and simulation of the proposed algorithm is done over Xilinx 14. The behavioral modeling style and Isim simulator is adopted for simulation. RTL and synthesis results are also generated.

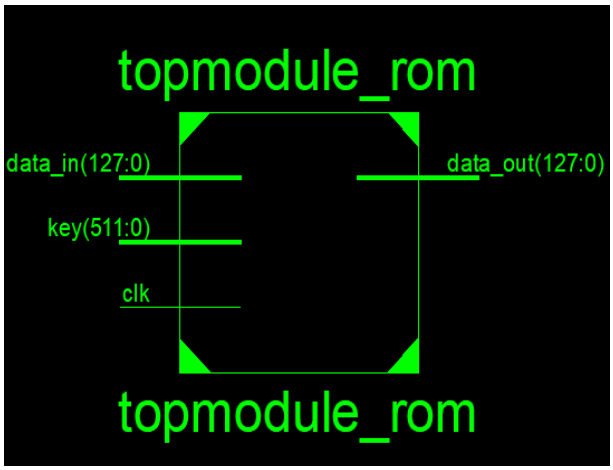

Figure 2: Top module of proposed 128 bit blowfish

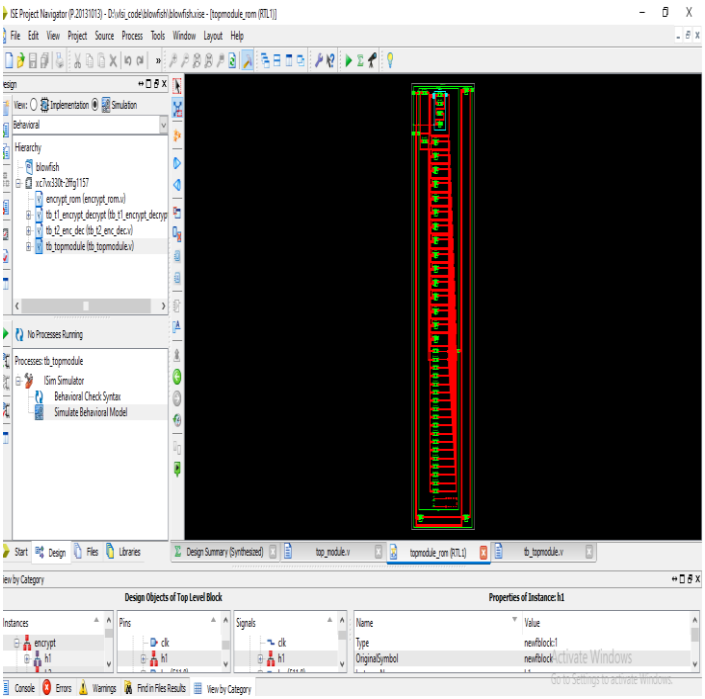

Figure 3: Complete RTL view

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347 - 6435 (Online) Volume 14, Issue 3, March 2025)

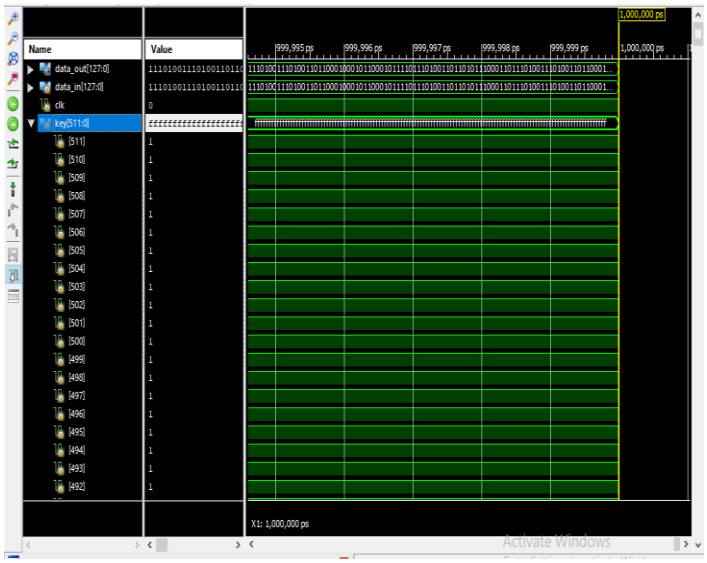

Figure 4: Results in test bench-4

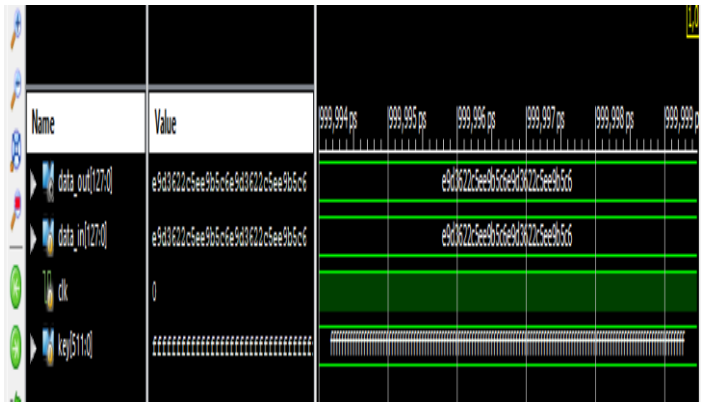

Figure 5: Results in test bench-5

Table 1: Result Comparison

Sr No.	Parameters	Existing work result	Proposed Work result
1	Method	64 bit	128 bit
2	Area	12.5%	12%
3	Power	0.46 mW	0.40 mW
4	Latency	8 ns	2.993 ns

IV. CONCLUSION

The modified design flow focuses on optimizing the Blowfish implementation for FPGAs. It involves adaptations such as LUT-based S-box implementation, P-box approximation, and data path optimization for parallelism. The FPGA implementation utilizes pipelining, efficient memory mapping, and resource allocation strategies. Design and verification steps include Verilog coding, synthesis, place and route, and thorough simulation and testing. Performance optimization techniques like clock frequency tuning and resource sharing are employed. Finally, the implementation is integrated and evaluated on a target FPGA board. This approach aims to achieve a high-performance and resource-efficient Blowfish implementation while maintaining the security properties of the original algorithm.

REFERENCES

1. P. Palka, R. A. Perez, T. Fang and J. Saniie, "Design Flow of Blowfish Symmetric-Key Block Cipher on FPGA," 2022 IEEE International Conference on Electro Information Technology (eIT), Mankato, MN, USA, 2022, pp. 193-197, doi: 10.1109/eIT53891.2022.9814070.
2. B. M. B. Beron, V. T. Duhaylungsod, K. G. Jimenez, J. Hora, R. C. O. Calimpusan and O. Joy Gerasta, "ASIC Implementation of Pipelined Blowfish Cryptographic Core in 0.13 µm CMOS Process Technology," 2019 IEEE 11th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICEM), Laoag, Philippines, 2019, pp. 1-6, doi: 10.1109/HNICEM48295.2019.9073385.
3. H. Setiawan and K. Rey Citra, "Design of Secure Electronic Disposition Applications by Applying Blowfish, SHA-512, and RSA Digital Signature Algorithms to Government Institution," 2018 International Seminar on Research of Information Technology and Intelligent Systems (ISRITI), Yogyakarta, Indonesia, 2018, pp. 168-173, doi: 10.1109/ISRITI.2018.8864280.
4. M. A. Muin, M. A. Muin, A. Setyanto, Sudarmawan and K. I. Santoso, "Performance Comparison Between AES256-Blowfish and Blowfish-AES256 Combinations," 2018 5th International Conference on Information Technology, Computer, and Electrical

Engineering (ICITACEE), Semarang, 2018, pp. 137-141, doi: 10.1109/ICITACEE.2018.8576929.

5. S. Vyakaranal and S. Kengond, "Performance Analysis of Symmetric Key Cryptographic Algorithms," 2018 International Conference on Communication and Signal Processing (ICCSP), Chennai, 2018, pp. 0411-0415, doi: 10.1109/ICCSP.2018.8524373.
6. S. Varshney, T. Sudarshan and S. Khare, "Efficient Hardware Architecture for Amalgam of Blowfish and Rc6," 2017 International Conference on Current Trends in Computer, Electrical, Electronics and Communication (CTCEEC), Mysore, 2017, pp. 1126-1130, doi: 10.1109/CTCEEC.2017.8455189.
7. I. A. Landge and B. K. Mishra, "VHDL based BLOWFISH implementation for secured Embedded System design," 2017 Third International Conference on Advances in Electrical, Electronics, Information, Communication and Bio-Informatics (AEEICB), Chennai, 2017, pp. 497-501, doi: 10.1109/AEEICB.2017.7972363.
8. T. K. Hazra, A. Mahato, A. Mandal and A. K. Chakraborty, "A hybrid cryptosystem of image and text files using blowfish and Diffie-Hellman techniques," 2017 8th Annual Industrial Automation and Electromechanical Engineering Conference (IEMECON), Bangkok, 2017, pp. 137-141, doi: 10.1109/IEMECON.2017.8079577.
9. A. Chauhan and J. Gupta, "A novel technique of cloud security based on hybrid encryption by Blowfish and MD5," 2017 4th International Conference on Signal Processing, Computing and Control (ISPCC), Solan, 2017, pp. 349-355, doi: 10.1109/ISPCC.2017.8269702.
10. A. Gaur, A. Jain and A. Verma, "Analyzing storage and time delay by hybrid Blowfish-Md5 technique," 2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS), Chennai, 2017, pp. 2985-2990, doi: 10.1109/ICECDS.2017.8390003.
11. R. Ahmad[1], A. A. Manaf and W. Ismail, "Development of an improved power-throughput Blowfish algorithm on FPGA," 2016 IEEE 12th International Colloquium on Signal Processing & Its Applications (CSPA), Malacca City, 2016, pp. 237-241, doi: 10.1109/CSPA.2016.7515838.
12. V. C. Dongre and S. G. Shikalpure, "Ensuring privacy preservation in wireless networks against traffic analysis by employing network coding and Blowfish encryption," 2016 International Conference on Signal and Information Processing (IConSIP), Vishnupuri, 2016, pp. 1-5, doi: 10.1109/ICONSIP.2016.7857442.