Indonesian Journal of Electrical Engineering and Computer Science
Vol. 35, No. 1, July 2024, pp. 397~404
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v35.i1.pp397-404

An efficient controller-based architecture for AES algorithm using FPGA

Reshma Nadaf, Satish S. Bhairannawar
Department of Electronics and Communication Engineering, SDM College of Engineering and Technology, Dharwad, India

Article Info

Article history:

Received Oct 25, 2023
Revised Feb 9, 2024
Accepted Mar 20, 2024

Keywords:

Advanced encryption standard
AES architecture
Cryptography
Field programmable gate array
Symmetric encryption

ABSTRACT

The importance of crucial current technical advancements, particularly those centered on the cryptography process such as Cryptographic advanced encryption standard (AES) hardware architectures are gaining momentum with respect to improving the speed and area optimizations. In this paper, we have proposed a novel architecture to implement AES on a reconfigurable hardware i.e., field programmable gate arrays (FPGA). The controller in AES algorithm is responsible to generate the signals to perform operations to generate the 128 bits ciphertext. The proposed controller uses multiplexer and synchronous register-based approach to obtain area and speed efficient on the FPGA hardware. The entire architecture of AES with proposed controller is implemented on Virtex 5, Virtex 6, and Virtex 7series using XilinxISE 14.7 and tested for critical path delay, frequency, slices, efficiency and throughput. It is observed that all the parameters are improved compared to existing architectures achieving the throughput of 32.29, 40.01, and 43.01 Gbps respectively. The key benefit of this approach is the high level of parallelism it displays in a quick and efficient manner.

Corresponding Author:

Reshma Nadaf
Department of Electronics and Communication Engineering, SDM College of Engineering and Technology
Dharwad, Karnataka, India
Email: reshma.nadaf27@gmail.com

1. INTRODUCTION

In today's digital era, the internet plays a vital role as a communication medium. However, the transmission of sensitive information in a digital format faces various risks from potential adversaries. As a result, protecting sensitive data while transmission over the internet has grown to be a top priority. There is a growing demand for the implementation of cryptographic algorithms on efficient and reconfigurable hardware, particularly in low-power and area-constrained applications. Techniques such as encryption and decryption are used to achieve confidentiality in order to allay these worries. Cryptography, considered both an art and science, involves transformation of confidential data (known as plaintext) into an unreadable data known as ciphertext [1], [2]. The process of transforming plaintext into ciphertext is called encryption, and the process of converting ciphertext back to plaintext is called decryption. This process involves key-based encryption methods, where a key is an essential component that transforms plaintext into ciphertext. The overarching goal of cryptography is to achieve confidentiality, integrity, non-repudiation, and authentication as major objectives [3], [4]. Symmetric cryptographic algorithms, also known as conventional encryption, utilize a single key for both the encryption and decryption processes [5]. The primary focus of symmetric algorithms is on privacy and confidentiality. In reality, a user chooses a key, encrypts confidential information, and creates ciphertext at the transmitter's end. The receiver then receives this ciphertext via a communication channel.

$$Ciphertext = Encryption_K(Plaintext) \quad (1)$$

Receiver uses the same secret key and decrypts the ciphertext:

$$Plaintext = Decryption_K(Ciphertext) \quad (2)$$

No external entities, aside from the sender and receiver, are permitted to engage in communication. Processing capabilities of symmetric algorithms are fast owing to their robust hardware implementations. Widely adopted methods like DES [6] and advanced encryption standard (AES) algorithms are implemented on field programmable gate arrays (FPGA) due its flexibility in parallelism [7], [8]. But key exchange is the main problem in symmetric cryptosystems. This problem is solved by using Asymmetric cryptosystems where a pair of keys are used i.e., public key and private key. RSA, Diffie-Hellman, and elliptic curve cryptography (ECC) [9] are examples of asymmetric algorithms. Key distribution problem of symmetric techniques is solved by asymmetric algorithms. Enhanced security is public-key cryptography's main benefit. Yet, the encryption and decryption processes using these algorithms are slower.

Chellam and Natarajan [10], memory are partitioned into numerous banks to store plaintext and key, allowing access to their contents in a single clock cycle. This guarantees that, after the pipeline is completely engaged, the system will produce output for each clock pulse received as input. The method entails sub-pipelining each AES operation and parallelizing the input reading by putting registers strategically to balance the pipeline. The XC5VLX85, XC6VLX240T, and XC7VX690T devices are used to implement the suggested approach in the study. Their corresponding path delays are 1.42 ns, 1.35 ns, and 1.23 ns. Interestingly, the throughputs of these devices are all very outstanding; they reach 90.4 Gbps, 94.81 Gbps, and 104.06 Gbps, in that order.

AES is implemented in both CTR and electronic code book (ECB) modes in Soltani and Sharifian [11]. A 128-bit encrypted output is obtained at each clock pulse by employing the loop-unrolling technique. In order to address critical path delays, both fully and sub-pipelining are used. By eliminating every loop, the modified critical path is produced. Using fully pipelining and sub-pipelining techniques increases the clock pulse's frequency, which lowers the critical path. The Xilinx Virtex-6 AES implementation in CTR mode achieves a maximum operating frequency of 508.104 MHz and a high throughput of 260.15 Gbps.

Zodpe and Sapkal [12], S-box values are created via a modified AES method using the pseudo number (PN) sequence generator. The PN sequence generator's output determines the beginning key for the first round. Numerous methods are put forth to improve the AES algorithm's quality. These methods are used to XC6SLX150 and XC6VLX240T. Modified AES delivers 60% of the avalanche effect compared to AES algorithm. The strict avalanche criterion is used to test the encrypted results of the improved AES algorithm for 2048 variations. The suggested design has a maximum frequency of 463.42 MHz and 237.45 MHz, respectively, and achieves a throughput of 59.3 Gbps and 30.39 Gbps.

A lightweight AES architecture is created in [13], for internet of things (IoT) devices with limited resources. The design uses designated register banks to store the encryption key, plaintext, and intermediate results. Shift rows is operated within state register in the suggested work. Optimized sub-bytes enable a 15.5% reduction in area on 65-nm technologies. To further cut down on power usage, clock gating is used. As a result, 18.9% less electricity is used. The Virtex-5 is used to achieve this suggested technique. The suggested design was then put into practice and validated using 65-nm technology under various temporal restrictions.

This work [14], suggests using the key expansion strategy to create the AES algorithm with minimal power and good throughput. This high-performance architecture reduces power consumption and crucial path delay. 256 bits are used for encryption and decryption, yielding a throughput of 0.06 Gbps. In comparison to other hardware implementations, the suggested AES architecture performs better in terms of power, throughput, and critical path delay. The operative frequency achieved by the suggested implementation is 277.4 MHz.

The low area, high throughput architecture for the AES algorithm is presented in [15]. A 128-bit key is utilized for both encryption and decryption. Since this encoder uses a 128-bit key, designing a key extension unit is a part of the overall design. To decrease the area of utilization, the folded notion is incorporated into the basic architecture. In order to get a better throughput than the fundamental structure, parallel operation is also used. The intended target device for FPGA implementation is the Virtex-6 XC6VLX75T. This work reaches a maximum frequency of 505.5 MHz and a throughput of 37.1 Gbps.

Priya *et al.* [16] is to create and implement an area-efficient, high-throughput AES encryption method on an FPGA with the intention of improving security measures. The development of an area-efficient and high-throughput structural S-Box design for the AES encryption procedure achieves both high speed and area efficiency. Combinational logic circuitry is used to design a Rijndael Low-area high-speed S-Box, and its performance is evaluated against a traditional S-Box. The developed S-Box is then integrated into the AES encryption process, and the paper provides calculations for its speed and propagation delay.

In order to accelerate the encryption process even more, a productive mix column structure is suggested. The purposeful design of this specific mix column arrangement is to maximize throughput. Next, the suggested mix column and S-Box structures are both added to the AES encryption protocol.

An FPGA Virtex 5 is used to implement the complete suggested AES architecture. The findings acquired are carefully contrasted with the outcomes of a traditional implementation in order to provide light on the efficacy and efficiency of the suggested improvements.

Some of FPGA-based cryptography schemes [17]–[19] are subject to the following restrictions: i) limitations on resources: the amount of logic elements on FPGAs is limited, thus implementing AES can need a lot of resources. More resources might be needed for bigger key sizes and more complex encryption schemes. Restricted block RAM (BRAM): AES frequently requires huge amounts of data storage, and large-scale implementations may encounter difficulties due to the limited on-chip block RAM on FPGAs. ii) latency and throughput: it can be difficult to get high throughput with FPGA-based AES systems. The hardware resources at hand may limit the natural parallelism of AES algorithms. There may be a problem with latency, particularly when handling big data sets. Lower throughput may result from optimizing for low latency, and vice versa. iii) power usage: it's possible that FPGA-based systems use more energy than ASIC implementations. Power efficiency is an important issue to take into account, especially in situations where energy consumption plays a significant role. iv) complexity of design: it can be difficult to create a secure and effective AES implementation on an FPGA; it requires knowledge of both hardware design and cryptography. Although it can be difficult, confirming the design's accuracy and security is essential. v) flexibility: even while FPGAs can be reconfigured, it might be difficult to update the design, particularly in deployed systems, to handle new features or meet evolving cryptographic risks. The paper is organized as follows: Section 2 describes the overview AES operation. In section 3 shows the proposed AES architecture which speeds up encryption process. In section 4 discusses results. Finally, the conclusion is discussed in section 5.

2. OVERVIEW AND METHOD

FPGAs are still a popular option for cryptography applications because of their flexibility, reconfigurability, and appropriateness for low-to-medium volume production, even with these drawbacks. Implementations of FPGA-based AES that are both secure and efficient can be achieved by carefully taking these constraints into account throughout the design phase.

2.1. Overview

The Rijndael algorithm was chosen as the AES in 2001 by the National Institute of Standards and Technology (NIST) because of its easy implementation on hardware and software, secure nature and flexibility. AES is a symmetric algorithm which takes a block size of 128 bits as plaintext, a key size of 128, 192 or 256 bits and generates 128 bits of ciphertext [20], [21]. AES is used for high speed and lightweight device implementations [22]. The structure of AES algorithm is discussed in [23]. Each round consists of sub bytes, shift rows, mix columns and add round key functions. The different key size used in this algorithm defines number of rounds to be performed. Iintermediate results of each round are updated in a state matrix. Each byte in the state matrix is denoted as $S(x, y)$ $(0 \leq x, y \leq 4)$.

2.2. Key expansion

The key expansion algorithm takes four words of initial key matrix arranged column wise which acts as input and generates a linear array of 44 words. Four words of 128 bits acts as a new key for the next round. The function g consists of following sub-operations.

- One-byte circular left shift is performed on the input word. [b0, b1, b2, b3] will be [b1, b2, b3, b4] after transformation. This is referred as RotWord.
- S-box is used to substitute each byte obtained in step 1. This is referred as SubWord.
- New obtained byte is xored with round constant defined in Table 1 and rightmost bytes are zero filled.

Figure 1 describes the working of the key expansion module. Remaining Addroundkeys are generated by using same technique.

Table 1. Rotation constant values [1]

J	1	2	3	4	5	6	7	8	9	10
RC [j]	01	02	04	08	10	20	40	80	1B	36

2.3. Method

In our proposed method the AES works with data blocks, and performance is improved by processing these blocks in parallel. To speed up encryption and decryption, we make use of the parallel processing in which the capabilities of FPGA is exploited in hardware. To minimize loop overhead and capitalize the efficiency, unrolling of for loops are performed while writing the HDL code. The required keys are

precomputed and saved including the round keys in memory rather than calculating them for each round in turn will lessen the computational load.

3. PROPOSED AES ARCHITECTURE

In this section, detailed architecture for each round of AES algorithm is presented. The implementation of each round is optimized in such a way that it consumes less area and hence less power. Figure 2 shows the proposed AES architecture. Pipelining and sub-pipelining are used to speed up the hardware implementation [23], [24]. The plaintext of 128 bits connected as input for the multiplexer 1. The output of the mux is connected to input of the register with clock for synchronous operation. The output of the register is XORed with the subkey which is obtained from the other part of register where the key of 128 bits is passed through the mux 2 and then to register followed by the XOR operation and this operation is called as Addroundkey. After the XOR operation, state array which is in the form of state matrix will be given to sub bytes, where it divides each 8 bits into 4 bits and first 4 bits value will be considered as row of the s-box and next 4 bits value is considered as column of the s-box respectively. Then by intersecting row and column, we will get new value that will be updated in state matrix. Next the new state array will move into shift row block and here the 4 rows are shifted cyclically to left by offset 0, 1, 2, 3 of state array. Finally, the updated state array moves to the mix column and the mix column multiples the updated state array with fixed matrix in GF (2^8). The output of the mix column block is connected to mux 3, and now the output of the mux 3 is feedback as one of the inputs to mux 1. This will complete the 1st round but cipher text is not available and to obtain the cipher text it has to be repeated for 10 rounds. The controller plays a major role in generating rotation constant (RC) used for key schedule round function. For every iteration the controller has to generate new add round key so it becomes necessary to optimize the controller in terms of area and speed to improve the performance. In our work we have proposed an optimized area efficient architecture for controller.

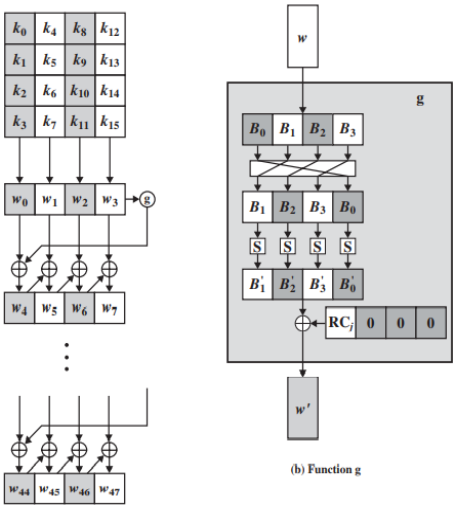

Figure 1. Key expansion [16]

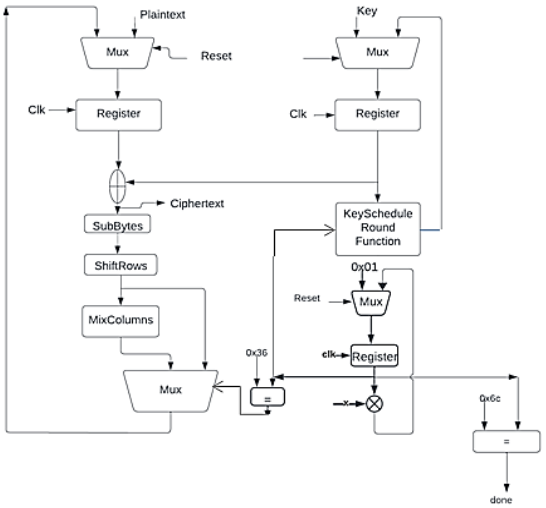

Figure 2. Proposed AES architecture

3.1. Implementation of sub bytes

The sub bytes is a nonlinear transformation [25]. This round is a lookup table substitution referred in [1]. After undergoing initial add round key, the output is updated in 4×4 state matrix. Every individual byte in a state matrix is substituted with a new byte from S-box table. A byte containing 8 bits is segmented into left 4 bits and right 4 bits. Left most 4 bits acts as a row and right most 4 bits acts a column for S-box table. These are now called as indexes of S-box table and they substitute a new value in place of old one. Ex: the value $\{AF\}_{16}$ is now substituted with $\{79\}_{16}$. The values of S-box table are constructed as follows [26].

- First row of S-box is initialized with a byte value from {00} to {0F}, second row is initialized with a value {10} to {1F} and so on, last row is {F0} to {FF}. These will act as indexes while performing sub bytes transformation.
- Each byte in the S-box is now mapped to its multiplicative inverse in the finite field GF (2^8) represented by $S^{-1}{}_{x,\ y}$. The irreducible polynomial used in the AES algorithm for generating multiplicative inverse is$m(x) = x^8 + x^4 + x^3 + x + 1$. {00} value is mapped to itself.

– The final value to be written in S-box is calculated as:

$$S'_{x,\ y=} MS^{-1}{}_{x,\ y} \oplus C \quad (3)$$

where matrix M is 8×8 matrix referred in [1].

3.2. Shift rows

Here the updated state matrix will move into shift rows round. In this round, 4 rows are shifted cyclically to left by offset 0, 1, 2, 3 of state matrix. 1st row shift left by '0' place, 2nd row shifts left by 1 place, 3rd row shifts left by 2 places, and then 4th row shifts left by 3 places. During decryption, the same state matrix is shifted cyclically to right accordingly.

$$\begin{bmatrix} S'_{0,0} & S'_{0,1} & S'_{0,2} & S'_{0,3} \\ S'_{1,0} & S'_{1,1} & S'_{1,2} & S'_{1,3} \\ S'_{2,0} & S'_{2,1} & S'_{2,2} & S'_{2,3} \\ S'_{3,0} & S'_{3,1} & S'_{3,2} & S'_{3,3} \end{bmatrix} = \begin{bmatrix} S_{0,0} & S_{0,1} & S_{0,2} & S_{0,3} \\ S_{1,1} & S_{1,2} & S_{1,3} & S_{1,0} \\ S_{2,2} & S_{2,3} & S_{2,0} & S_{2,1} \\ S_{3,3} & S_{3,0} & S_{3,1} & S_{3,2} \end{bmatrix}$$

3.3. Mix column

Finally state matrix moves to mix column round. Mix column operates on each column individually. Each byte of a column is mapped into a new value that is a function of all four bytes in that column. The transformation of mix column round is defined by the (9).

$$\begin{bmatrix} S'_{0,0} & S'_{0,1} & S'_{0,2} & S'_{0,3} \\ S'_{1,0} & S'_{1,1} & S'_{1,2} & S'_{1,3} \\ S'_{2,0} & S'_{2,1} & S'_{2,2} & S'_{2,3} \\ S'_{3,0} & S'_{3,1} & S'_{3,2} & S'_{3,3} \end{bmatrix} \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} S_{0,0} & S_{0,1} & S_{0,2} & S_{0,3} \\ S_{1,0} & S_{1,1} & S_{1,2} & S_{1,3} \\ S_{2,0} & S_{2,1} & S_{2,2} & S_{2,3} \\ S_{3,0} & S_{3,1} & S_{3,2} & S_{3,3} \end{bmatrix} \quad (4)$$

As a result, four bytes in a column are substituted as follows,

$$\left.\begin{aligned} S'_{0,C} &= (\{02\} \cdot S_{0,C}) \oplus (\{03\} \cdot S_{1,C}) \oplus S_{2,C} \oplus S_{3,C} \\ S'_{1,C} &= S_{0,C} \oplus (\{02\} \cdot S_{1,C}) \oplus (\{03\} \cdot S_{2,C}) \oplus S_{3,C} \\ S'_{2,C} &= S_{0,C} \oplus S_{1,C} \oplus (\{02\} \cdot S_{2,C}) \oplus (\{03\} \cdot S_{3,C}) \\ S'_{3,C} &= (\{03\} \cdot S_{0,C}) \oplus S_{1,C} \oplus S_{2,C} \oplus (\{02\} \cdot S_{3,C}) \end{aligned}\right\} \text{where } 0 \le C \le 4$$

where $\oplus$ is the XOR operation and the $\cdot$ is a multiplication modulo the irreducible polynomial $m(x) = x^8 + x^4 + x^3 + x + 1$. Multiplication of a value by {02} can be implemented as a 1-bit left shift followed by a conditional bitwise XOR with (0001 1011) if the leftmost bit of the original value (prior to the shift) is 1. Multiplication by {03} is performed by repeated addition as shown in (5).

$$S_{x,y} \cdot \{03\} = S_{x,y} \cdot \{02\} \oplus S_{x,y} \quad (5)$$

3.4. Proposed controller

The controller plays an important role in this architecture to generate appropriate control signals at proper timings. Many of the existing work deals with the Finite state machine but in this work, mux based synchronous register is used to improve the area constraint on the hardware. The multiplexer in the controller has two inputs, one which is fixed value 0×01 and other being a feedback signal from the multiplier. The output of the multiplexer is fed to the synchronous register which improves the speed due to its synchronous operation. The multiplier used here is Galois multiplier which generates round constant value (RC) and this is further given to key schedule round function to obtain new updated key function. This iteration continues till 10th round where the multiplier output reaches the value 0×36. The output generated at the end of 10th round is the exact 128 bits cipher text and this is acknowledged in our controller by using the done signal when it reaches value 0×6C.

4. RESULTS AND DISCUSSION

In this section, the implementation results of the proposed AES architecture and its analysis are done on different FPGA boards. The proposed AES architecture is implemented on Virtex 5(XC5VLX30), Virtex

6(XC6VLX240T) and Virtex 7(XC7VX690T) FPGA using VHDL which are widely used. Xilinx ISE 14.7tool is used for the placing and routing, synthesis and timing and power analysis. The hardware parameters of the proposed architecture are tabulated in Table 3.

Table 3. Design summary

Device	Critical path delay ns	Frequency MHz	Slices registers	Slice LUTs	Throughput Gbps	Efficiency Mbps
Virtex 5 (XC5VLX30)	3.964	252.244	264 +1 BRAM	1075	32.29	30.03
Virtex 6 (XC6VLX240T)	3.199	312.5	264 +1 BRAM	1104	40.01	36.24
Virtex 7 (XC7VX690T)	2.970	336.655	264 +1 BRAM	1104	43.09	39.03

Throughput and efficiency are calculated as (6) and (7).

$$Throughput = \frac{Number\ of\ output\ bits}{Critical\ path\ delay} \quad (6)$$

$$Efficiency = \frac{Throughput}{Area(Slices)} \quad (7)$$

The RTL schematic of the proposed architecture is shown in Figure 3 on Virtex 5, where each block is designed and tested separately before integration. The proposed architecture compares well with similar works. The implementation of the proposed architecture is on Virtex 5(XC5VLX30), Virtex 6(XC6VLX240T) and Virtex 7(XC7VX690T) FPGA using VHDL language. It is interesting to note that the proposed architecture achieved favorable results than all three, utilizing less number of slice registers, LUTs and LUT-FF Pairs. Table 4 gives comparison of different works.

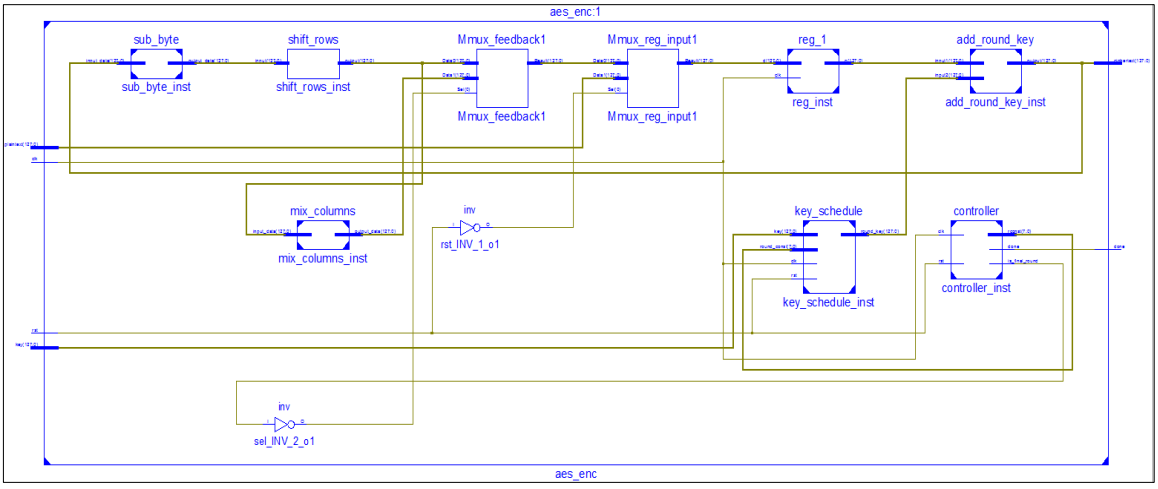

Figure 3. RTL schematic of proposed AES algorithm on Virtex 5

Table 4. Comparison with similar works

Design	Device	Slices	Critical path delay (ns)	Max. frequency (MHz)	Throughput (Gbps)	Efficiency (Mbps)
[10]	XC5VLX85	2940	1.42	704.7	90.4	30.74
[16]	Virtex 7	15680	-	-	-	-
[17]	XC7A35F	7790	-	-	0.35 Mbps	-
[18]	Virtex 6	2688	3.167	315.806	40.42	-
[25]	XC6VLX240T	5759	1.3	732.279	93.73	16.27
[26]	XC5VLX85	7385	1.56	638.162	81.68	11.06
Our	XC5VLX30	1075	3.964	252.244	32.29	30.03
design	XC6VLX240	1104	3.199	312.5	40.01	36.24
	XC7VX690T	1104	2.970	336.655	43.09	39.03

Future scope: the research work is focused on exploring multimodal cryptographic key generation technique from biometrics, implementation and performance analysis of this technique on FPGA to higher efficiency.

5. CONCLUSION

In this work, anunique architecture for AES implementation using a FPGA-based reconfigurable hardware FPGA is proposed. In the AES algorithm, the controller is used for generating the signals needed to carry out the operations necessary to produce the 128-bit encrypted text. The proposed controller achieves area and speed efficiency on the FPGA hardware by utilizing multiplexers and a synchronous register-based approach. Using Xilinx ISE 14.7, the entire AES architecture with the proposed controller is implemented on the Virtex 5, Virtex 6, and Virtex 7 series processors, and its critical path delay, frequency, slices, efficiency, and throughput are evaluated. It is noted that all metrics have improved as compared to existing architectures with the throughput of 32.29, 40.01, and 43.01 Gbps respectively.

REFERENCES

[1] W. Stallings, *Cryptography and network security*. 5th ed. Prentice Hall, 2010.

[2] B. A. Forouzan, *Introduction to cryptography and network security*. McGraw-Hill Higher Education, 2008.

[3] S. H. Murad and K. H. Rahouma, "Implementation and performance analysis of hybrid cryptographic schemes applied in cloud computing environment," *Procedia Computer Science*, vol. 194, pp. 165–172, 2021, doi: 10.1016/j.procs.2021.10.070.

[4] W. A. Al-Musawi, M. A. Ali Al-Ibadi, and W. A. Wali, "Artificial intelligence techniques for encrypt images based on the chaotic system implemented on field-programmable gate array," *IAES International Journal of Artificial Intelligence (IJ-AI)*, vol. 12, no. 1, pp. 347–356, Mar. 2023, doi: 10.11591/ijai.v12.i1.pp347-356.

[5] W. A. Al-Musawi, W. A. Wali, and M. A. Ali Al-Ibadi, "Field-programmable gate array design of image encryption and decryption using Chua's chaotic masking," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 12, no. 3, pp. 2414–2424, Jun. 2022, doi: 10.11591/ijece.v12i3.pp2414-2424.

[6] A. Rabie, K. El Shafie, A. Hammuoda, and M. Rohiem, "Data ecryption based on multi-order FrFT, and FPGA implementation of DES algorith," *International Journal of Reconfigurable and Embedded Systems (IJRES)*, vol. 9, no. 2, pp. 141–152, Jul. 2020, doi: 10.11591/ijres.v9.i2.pp141-152.

[7] T. L. Prasanna, N. Siddaiah, B. M. Krishna, and M. R. Valluri, "Implementation of the advanced encryption standard algorithm on an FPGA for image processing through the universal asynchronous receiver-transmitter protocol," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 12, no. 6, pp. 6114–6122, Dec. 2022, doi: 10.11591/ijece.v12i6.pp6114-6122.

[8] P. Visconti, R. Velazquez, S. Capoccia, and R. De Fazio, "High-performance AES-128 algorithm implementation by FPGA-based SoC for 5G communications," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 11, no. 5, pp. 4221–4232, Oct. 2021, doi: 10.11591/ijece.v11i5.pp4221-4232.

[9] S. Harb and M. Jarrah, "FPGA implementation of the ECC Over GF(2 m) for small embedded applications," *ACM Transactions on Embedded Computing Systems*, vol. 18, no. 2, pp. 1–19, Mar. 2019, doi: 10.1145/3310354.

[10] M. B. Chellam and R. Natarajan, "AES hardware accelerator on FPGA with improved throughput and resource efficiency," *Arabian Journal for Science and Engineering*, vol. 43, no. 12, pp. 6873–6890, Dec. 2018, doi: 10.1007/s13369-017-2925-0.

[11] A. Soltani and S. Sharifian, "An ultra-high throughput and fully pipelined implementation of AES algorithm on FPGA," *Microprocessors and Microsystems*, vol. 39, no. 7, pp. 480–493, Oct. 2015, doi: 10.1016/j.micpro.2015.07.005.

[12] H. Zodpe and A. Sapkal, "An efficient AES implementation using FPGA with enhanced security features," *Journal of King Saud University - Engineering Sciences*, vol. 32, no. 2, pp. 115–122, Feb. 2020, doi: 10.1016/j.jksues.2018.07.002.

[13] K. Shahbazi and S.-B. Ko, "Area-efficient nano-AES implementation for Internet-of-Things devices," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 29, no. 1, pp. 136–148, Jan. 2021, doi: 10.1109/TVLSI.2020.3033928.

[14] K. Kalaiselvi and H. Mangalam, "Power efficient and high performance VLSI architecture for AES algorithm," *Journal of Electrical Systems and Information Technology*, vol. 2, no. 2, pp. 178–183, Sep. 2015, doi: 10.1016/j.jesit.2015.04.002.

[15] K. Rahimunnisa, P. Karthigaikumar, S. Rasheed, J. Jayakumar, and S. S. Kumar, "FPGA implementation of AES algorithm for high throughput using folded parallel architecture," *Security and Communication Networks*, vol. 7, no. 11, pp. 2225–2236, Nov. 2014, doi: 10.1002/sec.651.

[16] S. S. S. Priya, P. Karthigaikumar, and N. R. Teja, "FPGA implementation of AES algorithm for high speed applications," *Analog Integrated Circuits and Signal Processing*, vol. 112, no. 1, pp. 115–125, Jul. 2022, doi: 10.1007/s10470-021-01959-z.

[17] C. Sreevalli, K. Dharavathu, and M. S. Anuradha, "FPGA implementation of AES algorithm using verilog-HDL," *International Journal of Emerging Technologies and Innovative Research*, vol. 5, no. 10, pp. 382–390, 2018.

[18] S. Madhavapandian and P. M. Pandi, "FPGA implementation of highly scalable AES algorithm using modified mix column with gate replacement technique for security application in TCP/IP," *Microprocessors and Microsystems*, vol. 73, Mar. 2020, doi: 10.1016/j.micpro.2019.102972.

[19] A. T. Hashim, A. M. Hasan, and H. M. Abbas, "Design and implementation of proposed 320 bit RC6-cascaded encryption/decryption cores on altera FPGA," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 10, no. 6, pp. 6370–6379, Dec. 2020, doi: 10.11591/ijece.v10i6.pp6370-6379.

[20] L. S. Mezher, "Hamming neural network application with FPGA device," *International Journal of Reconfigurable and Embedded Systems (IJRES)*, vol. 10, no. 1, pp. 37–46, Mar. 2021, doi: 10.11591/ijres.v10.i1.pp37-46.

[21] M. Saber and M. M. Eid, "Low power pseudo-random number generator based on lemniscate chaotic map," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 11, no. 1, pp. 863–871, Feb. 2021, doi: 10.11591/ijece.v11i1.pp863-871.

[22] P. Visconti, R. Velazquez, C. D.-V. Soto, and R. de Fazio, "FPGA based technical solutions for high throughput data processing and encryption for 5G communication: A review," *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, vol. 19, no. 4, pp. 1291–1306, Aug. 2021, doi: 10.12928/telkomnika.v19i4.18400.

[23] X. Zhang and K. K. Parhi, "High-speed VLSI architectures for the AES algorithm," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 12, no. 9, pp. 957–967, Sep. 2004, doi: 10.1109/TVLSI.2004.832943.

[24] N. Ahmad and S. M. R. Hasan, "A new ASIC implementation of an advanced encryption standard (AES) crypto-hardware accelerator," *Microelectronics Journal*, vol. 117, Nov. 2021, doi: 10.1016/j.mejo.2021.105255.
[25] S. Oukili and S. Bri, "Hardware implementation of AES algorithm with Logic S-box," *Journal of Circuits, Systems and Computers*, vol. 26, no. 09, p. 750141, Sep. 2017, doi: 10.1142/S0218126617501419.
[26] C. A. Murugan, P. Karthigaikumar, and S. S. Priya, "FPGA implementation of hardware architecture with AES encryptor using sub-pipelined S-box techniques for compact applications," *Automatika*, vol. 61, no. 4, pp. 682–693, Oct. 2020, doi: 10.1080/00051144.2020.1816388.

BIOGRAPHIES OF AUTHORS

Reshma Nadaf completed B.E. and MTech. from VTU, Belagavi, Karnataka, India. Now she is working as Asst. Professor in the ECE department, at SDM College of Engineering and Technology, Dharwad, Karnataka, from 2007. Her research area includes Cryptography, biometrics. She has attended various conferences and published many papers. She can be contacted email: reshma.nadaf27@gmail.com.

Dr. Satish S. Bhairannawar Professor in the Department of ECE and Dean-CIII at SDM College of Engineering and Technology, Dharwad, Karnataka. He has many funded projects from VGST and DST to his account. He has attended various conferences and presented papers. He has publications indexed in SCIE, SCOPUS, and WoS. He can be contacted at email: satishbhairannwar@gmail.com.