2023 International Conference on Advances in Computing, Communication and Applied Informatics (ACCAI) | 979-8-3503-1590-5/23/$31.00 ©2023 IEEE | DOI: 10.1109/ACCAI58221.2023.10199794

AES Algorithm for the Next Generation of 5G Network Encryption Standards

Manojkumar T[1], Tamilarasi K[2], Mahalakshmi S[3], Vinayagam P[4], Mahanthesha U[5] and Mohanraj S[6]

[1]Department of Electronics and Communication Engineering Karpagam Institute of TechnologyCoimbatore, Tamilnadu, India
[2]Department of Electronics and Communication Engineering Excel Engineering College Namakkal, Tamilnadu, India
[3]Department of Computer Science and Design R.M.K. Engineering College Chennai, Tamilnadu, India
[4]Department of Electronics and Communication Engineering Saveetha Engineering College Chennai, Tamilnadu, India
[5]Department of Artificial Intelligence and Machine Learning BNM Institute of Technology Bengaluru, Karnataka, India
[6]Department of Electronics and Communication Engineering Rajalakshmi Engineering College, Chennai, Tamilnadu, India

E-mail: srimanojkumar@gmail.com tamilarasi.nachimuthu@gmail.com smi.csd@rmkec.ac.in vinayagamp@saveetha.ac.in mahantheshau@bnmit.in mohanraj.s@rajalakshmi.edu.in

Abstract-The Advanced Encryption Standard (AES) symmetric key encryption algorithm is the industry standard for 5G communication encryption. Technology advancements nowadays are mostly aimed at improving the security, speed, and efficiency of processors. The main expansion phase is made easier by the study's pipeline structure. This method simplifies the process of creating subkeys in AES encryption for use in 5G communication networks. There is a 37% increase in throughput between the simulated and implemented versions of the original design on FPGA Virtex4 XC4VLX200, and this difference may be attributed to the reduction in propagation latency while producing subkeys.

Keywords—Encryption; Decryption;5G Communication; Substitution box; FPGA.

I.Introduction

Since 5 G Communication networks process and transport billions of bits of data per second, information has become the most valuable resource. Manage and disseminate this vast volume of data securely. With modern communication networks, information is one of the most valuable assets, making it susceptible to theft or manipulation [1]. Researchers employ cryptography to protect our data over a vulnerable 5G Communication channel. Data storage may necessitate protection. Before transmitting plaintext across an insecure channel, cryptographic methods encrypt and decode the message. A secret key encodes and decodes data. Methods of statistics and analysis that utilize the cryptographic processes key [2–5]. Secure communication methods must be resistant to cryptanalytic attacks, which use the mathematical operations and procedures used to transform plain text to cypher text in order to decipher the secret key.

AES is better than other algorithms because it keeps information private, can be implemented in hardware and software, and runs faster [6,7] for high-speed 5th generation communication systems. Algorithm iterations determine subkey changes. Subkeys are created sequentially. Hence, subkey generation is time-consuming. This article examines a subkey-building framework. Pipelined architecture reduces subkey construction time. This novel idea works in hardware and software.Even with 5G connectivity, this new architecture retains all statistical and mathematical processes, ensuring algorithm security. This design gives a secured efficient AES algorithm for encrypted communication systems.

The paper continues as follows. In Part 2, the literature examines numerous relevant writers on this topic. Part 3 covers the current AES algorithm and encryption/decoding math. Part 4 details the proposed algorithm improvements. Section 5 discusses new design results and impressions. Section 6 concludes this report and offers further investigation.

II.Literature Review

In the 1990s, researchers determined the most efficient way to employ the AES algorithm for safe, high-speed communication. Certain designs are faster, smaller, and more efficient. Zhang et al. [8] presented hardware and software AES algorithm implementation. CFA minimizes the footprint of this innovative construction. Each encryption and decryption cycle makes use of a fully Sub Pipelined encryptor, and the Xilinx XCV1000 achieved 21.56 Gbps. Kamal et al. [9] devised an AES technique that conserves space. This unique AES method uses 2732 slices of the Xilinx Virtex-II XC2V1000bg575 processor to produce 29.32 Mbps throughput and 98.95 MHz clock speed. To speed up the AES cypher, Hammad et al. [10] added CFA. This 39.053 Gbps sub-pipelined multistage system comprises nine 305.1 MHz pipelining stages. Fan and Hwang [11] created a fully pipelined sequential AES solution employing a Xilinx ISE 7.1 synthesizer and a cost-effective AddRoundKey architecture for real-time key creation. Pipelined AES achieves 28.4 Gbps at 222 MHz. Kumar et al. [12] suggested replacing s-boxes with CFA and pipelining to save space. Parallel processing and pipelining boost output. S-Box and MixColumn, which effect throughput, reduce the region.

III.AES Encryption And Decryption

AES has no connection to the Fiedler block cypher. Using a secret key, it encrypts 128-bit blocks of plain text. Table 1 displays the lengths of secret keys for various AES algorithms. Provided the encryption secret key, the same method can be used to convert cypher text to plain text. All implementations of AE employ four techniques to encrypt and decrypt all rounds outside the last one. Before encryption, a 4x4 state matrix with each element representing a word contains 128-bit plain text (2 bytes).

A. Substitution

Substitution changes the value of bytes. The AES algorithm's only nonlinear component is replacement. Every byte of a state matrix is a Galois Field 28 polynomial. Using matrix multiplication and affine translation, polynomials are converted into bytes. The RijndaelS-box can substitute directly. The Inverse S-box is substituted during the decryption process. The step of replacement ambiguizes the original data, so prohibiting unauthorised access.

B. Shift Rows

Using a loop, rows in the state array are relocated in this situation. Nonetheless, the II, III, and IV rows shift to the left by one, two, and three times, respectively, while the Ist row remains constant. The rows are shifted to the right during the decryption process, with the same numbers as during the encryption process. In the first step of the AES algorithm's diffusion process, the row is relocated.This shift-row procedure is also reversible.

C. Mix columns

Similar to shift row step, actions occur at the column level in Mix column step. Many methods have been presented for completing this task. However, matrix multiplication is the more straightforward process.

For MixColumn operation

$$\begin{bmatrix} T'1 \\ T'2 \\ T'3 \\ T'4 \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} T1 \\ T2 \\ T3 \\ T4 \end{bmatrix}$$

For InvMixColumn operation

$$\begin{bmatrix} T'1 \\ T'2 \\ T'3 \\ T'4 \end{bmatrix} = \begin{bmatrix} 0e & 0b & 0d & 09 \\ 09 & 0e & 0b & 0d \\ 0d & 09 & 0e & 0b \\ 0b & 0d & 09 & 0e \end{bmatrix} \begin{bmatrix} T1 \\ T2 \\ T3 \\ T4 \end{bmatrix}$$

T’1, T’2, T’3, T’4 - output of MixColumnTransformation
T1, T2, T3, T4 - input of MixColumnTransformation

D. AddRound key

"Add round key" is the primary function of AES due to the reversibility of the three preceding operations. The AES algorithm fails without this step. AddRoundKey finishes the job of the secret key [12-14]. In this transformation, the first secret main key and all the obtained subkeys are represented as a 4x4 order state matrix. AddRoundKey inserts keys and text into a matrix. The outputs of this add round key operation is ciphertext. The above four transformations are repeated 10 times for AES-128. Each loop utilizes a unique subkey. n rounds necessitate n subkeys. The expansion of a single secret key into several subkeys. Expansion or scheduling of a key should not utilize subkey connections. AES key expansion is a nonlinear mathematical operation that generates several keys from a single key. When researchers create new key expansion methods, the security of keys improves. This proceduremінimizes the total time required to create subkeys from the main secret key.

E. Conventional Key Expansion Architecture

This article details how the AES-128 key can be expanded. The 128-bit main secret key is expanded into four words with this strategy. With AES-128, the master secret key generates 10 more subkeys.These subkeys are used to encrypt texts. Figure 1 demonstrates how crucial scheduling and expansion function. To obtain subkeys from the master key, mathematical operations are used. When subkeys are interdependent, key expansion is effective.

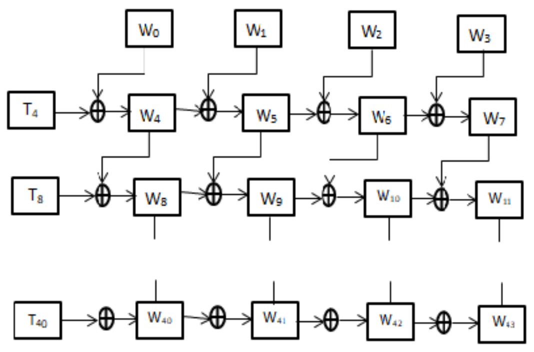


Fig 1.Existing Key Expansion Architecture

Math is performed between the words, and when four consecutive words are merged, they produce a subkey. The first subkey, for instance, is made up of the digits W4, W5, W6, and W7. For a total of 4(Nr+1) words, the key expansion method is responsible.In this instance, the mathematical operation is just an XOR.

A. *Temporary word Generation Procedure*

When it comes to key expansion, the most important piece is the introductory word. Only temporary words deviate from the strict linearity of creating a subkey. Each subkey's initial word is created by XORing the temporary word with a word from the previous subkey.Three steps build a temporary term from a word. Substitution, rotation, and EXOR create a temporary word. Temporary words yield AES-128 keys W4, W_8, W_{12}, W_{16}, W_{20}, W_{24}, W_{28}, W_{36}, and W_{40}. Temporary i4Nr periods are unnecessary. Temporary word can be expressed by equation 1.

$$\text{Temporary word} = \text{Subword}\ (\text{Rotword}(W_{i-1}))\text{Rcon} \quad (1)$$

Table 1 AES – 128 R constant values

Round	Rcon Value	Round	Rcon Value
1	$(01000000)_{16}$	6	$(20000000)_{16}$
2	$(02000000)_{16}$	7	$(40000000)_{16}$
3	$(04000000)_{16}$	8	$(80000000)_{16}$
4	$(08000000)_{16}$	9	$(1B000000)_{16}$
5	$(10000000)_{16}$	10	$(36000000)_{16}$

The Rotwordis same as the Shift row routine, but only changes a single row in this case. Similarly, Subword is equivalent to the Sub byte procedure.

IV. Proposed Key Expansion Structure

Typically, key expansion generates subkeys consecutively. Subkeys must be interconnected in order to be difficult to crack. Subkeys must be linked together. Modern architecture demands Wi-1 and Wi-4 words, but not Wi-1 and Wi-4. This connection assigns the necessary subkeys. Sequential processes are time-consuming because they cannot conclude until all preceding steps are accomplished. The tenth subkey is generated by AES-128 following the ninth. After the ninth subkey, the tenth is generated. As a result, generating the correct amount of subkeys is a time-consuming process. To overcome this delay, our proposal modifies the structure of key expansion in two ways. This decreases the required time. Word-by-word extension of the key. Subkeys are interdependent. The process of confusion and dispersal involves interconnected subkeys. Hence, subkeys should be interdependent to make it more difficult to locate the primary key.

A.*Pipelined key expansion*

In the new structure, an alternate method to the creation of the short-term term is presented. Traditionally, the temporary word is generated by executing a mathematical operation on the final word of the final subkey. Due to this, it is impossible to construct a pipeline. The final digit of the preceding subkey is not necessary for deducing the temporary word. This subkey can be created using the initial letter of the previous subkey. By implementing this modification, we can collect data for the near future significantly faster than before. In lieu of this, a pipelined architecture can be utilized to expedite the production of subkeys.Temporary word can be expressed by equation 2.

$$\text{Temporary word} = \text{Subword}\ (\text{Rotword}(W_{i-4}))\text{Rcon} \quad (2)$$

W8 is dependent on W4 and T8 according to standard design. But what makes up T8 is W7. As a result, W8 cannot be manufactured until W7 has been manufactured. With this update, the term W8 may be used until W4 becomes available. While T8 is entirely dependent on W4, it can be generated concurrently with W5. You need not wait until the completion of the W7 generation. Similarly, W9 may be finished prior to W7. This method expedites the development of subkeys.

B. *Fork-join model of key expansion architecture*

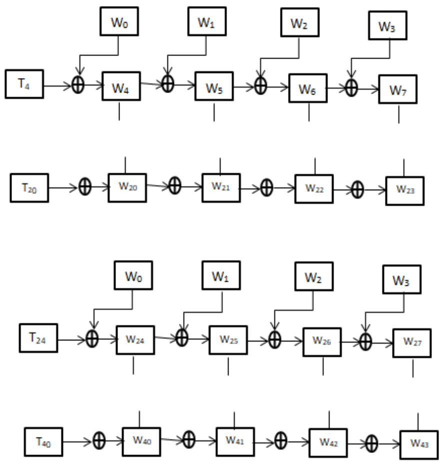
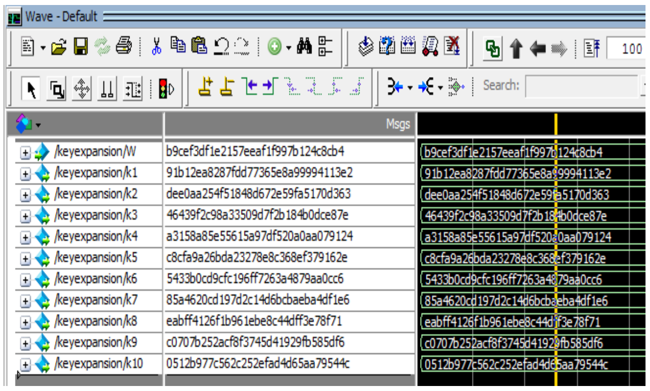


Fig. 2. Block diagram of proposed key expansion

The second concept of the new design is to run two segments simultaneously. AES-128 requires ten subkeys, which are utilized ten times. AES-128 groups the initial five subkeys into one block and the remaining five into another. In order to generate the first subkey, the master secret key must be used. Keys for levels II, III, IV, and V are typically derived from the first four.Figure 2 depicts an updated approach that generates the sixth key from the primary key rather than the fifth subkey. This modification generates the sixth subkey concurrently with the first. For the VII, VIII, and X, the VI, VII, VIII, and IX subkeys are required. Seventh, eighth, IX, and X are comprised of the II, III, IV, and V subkeys. Both sub-blocks generate subkeys from the main secret key, so they can generate subkeys concurrently.

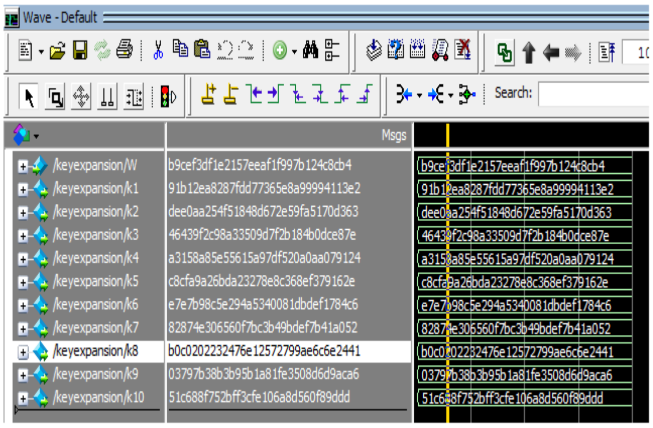
V Experimental Results

The suggested AES Key Expansion structure is backwards-compatible with the original design, meaning it may be used in place of the older version of the algorithm without causing any problems. Subkey generation is tested by simulating a modified key expansion structure in a new AES-128 design. This novel key expansion scheme is capable of producing the AES-128 subkey and AES algorithm after verification. In order to simulate, synthesize, and implement, we use Modalism 10.4d [15] and a Virtex 4XC4VLX200 FPGA. Find the time it took to generate a cipher text using 128 bits.



```
/keyexpansion/W     b9cef3df1e2157eeaf1f997b124c8cb4
/keyexpansion/k1    91b12ea8287fdd77365e8a99994113e2
/keyexpansion/k2    dee0aa254f51848d672e59fa5170d363
/keyexpansion/k3    46439f2c98a33509d7f2b184b0dce87e
/keyexpansion/k4    a3158a85e55615a97df520a0aa079124
/keyexpansion/k5    c8cfa9a26bda23278e8c368ef379162e
/keyexpansion/k6    5433b0cd9cfc196ff7263a4879aa0cc6
/keyexpansion/k7    85a4620cd197d2c14d6bcbaeba4df1e6
/keyexpansion/k8    eabff4126f1b961ebe8c44dff3e78f71
/keyexpansion/k9    c0707b252acf8f3745d41929fb585df6
/keyexpansion/k10   0512b977c562c252efad4d65aa79544c
```

Fig 3 Simulation result of Existing architecture



```
/keyexpansion/W     b9cef3df1e2157eeaf1f997b124c8cb4
/keyexpansion/k1    91b12ea8287fdd77365e8a99994113e2
/keyexpansion/k2    dee0aa254f51848d672e59fa5170d363
/keyexpansion/k3    46439f2c98a33509d7f2b184b0dce87e
/keyexpansion/k4    a3158a85e55615a97df520a0aa079124
/keyexpansion/k5    c8cfa9a26bda23278e8c368ef379162e
/keyexpansion/k6    e7e7b98c5e294a5340081dbdef1784c6
/keyexpansion/k7    82874e306560f7bc3b49bdef7b41a052
/keyexpansion/k8    b0c0202232476e12572799ae6c6e2441
/keyexpansion/k9    03797b38b3b95b1a81fe3508d6d9aca6
/keyexpansion/k10   51c688f752bff3cfe106a8d560f89ddd
```

Fig 4 Simulation result of Proposed architecture

Figures 3 and 4 illustrate simulation results for the current and proposed primary expansion procedures. Simulating the present design and proposed architecture uses the same major key. Simulation findings show that both techniques create identical first five subkeys. It's the same because the initial section hasn't changed. In the second half of the design, only the key's interdependency changes the subkey values.

It means that both designs' second-part subkeys differ significantly. After five rounds, the encrypted data remains the same as the initial five subkeys. Due to the second component's different subkeys, the encryption procedure's outcome will vary by architecture.

Table 2Comparison between Existing and Proposed AES Key Expansion Structures

AES	**No of Slices**	**Propagation Delay (ns)**	**Throughput (Mbps)**
AES	1208	6.0	21,203
4 Stage Pipelinearchitecture	3428	2.7	46,526
6 Stage Pipelinearchitecture	3424	2.3	54,728
6 Stage Sub-pipeline architecture	1405	2.3	54,007
This work	3437	2.2	56,143

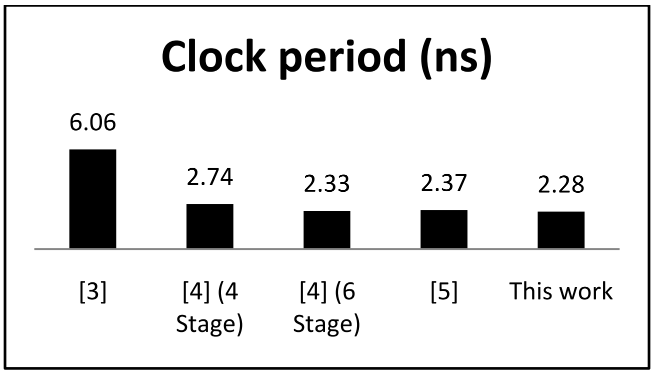


Fig 5 Area comparisons of Proposed and Existing Architecture.

This new key scheduling design replaces the AES algorithm's key expansion process architecture. The pipelined multi-stage AES algorithm's proposed design outperformed the traditional architecture. AES algorithmic synthesis reports are typical [3]. In [4], the typical design becomes a pipelined four-stage architecture. This accelerates AES execution and propagation but requires more storage. Researchers developed an effective 6-step procedure to implement AES in [4]. Utilize sub-pipeline S-boxes to remedy the problem there. AES pipelined architecture uses normal key expansion. Pipeline and parallel block architecture can improve key scheduling throughput in equation 3 and propagation delay.

$$Throughput = \frac{Number of bits processed}{Total clock period} \quad (3)$$

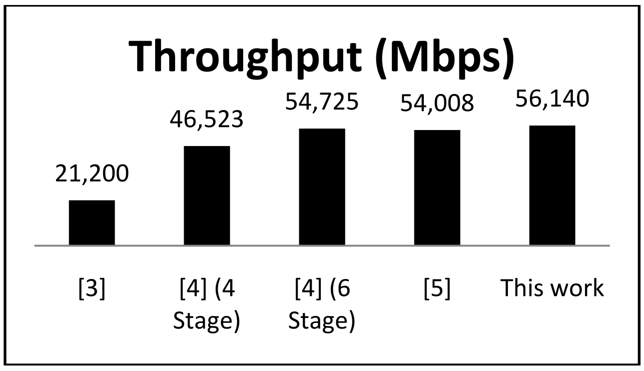


Fig 6 Throughput comparisons of Proposed and Existing Architecture.

After incorporating the new design into the virtex4 FPGA, the algorithm's total encryption time is computed and compared to the current architecture. Table 2 shows the design's propagation latency and throughput performance. Figures 5 and 6 show that this new, improved design has higher throughput. New design throughput is 2.53% higher than [4]. The new design boosts throughput by 3.8% and velocity by [5]. The new design uses 59.09% more slices than [5]. 0.5% lower than [4]. Velocity is measured by cycle time. This construction is space-intensive but has high throughput. While all other phases remain unchanged, the s-non-linearity box's process secures the algorithm.Table 3 gives the throughput efficiency of different pipeline stages.

Table 3 Throughput efficiency improvement on existing architectures comparing with proposed architecture

Throughput efficiency	AES	Pipeline (4 Stage)	Pipeline (6 Stage)	Sub pipeline (6 Stage)
	62.23	9.03	2.53	3.8

The same architecture can be expanded to AES-192 and AES-256. Similar improvement in results can be achieved for AES – 192 and AES – 256 also.

VI Conclusion And Future Work

This research work describes a quick AES algorithm for 5G communication networks. AES is implemented in FPGA. The proposed KeySchedule operation is built on a pipelined CFA architecture. The new architecture enables concurrent subkey production. This design allows the algorithm to run faster without sacrificing security or storage space. It is built with FPGA Vitex-4 chips. According to the study, the proposed design takes up less space than comparable structures. This pipelined structure saves 2.15 percent on essential route time. This design increases throughput by 2.53%. This is explained by pipelined architecture and concurrent block creation. This AES may be used in high-speed 5G communication systems because it has better throughput and critical path delay than standard AES. However, the number of times it runs can be increased without affecting the amount of space it consumes. This is possible by optimising transformations such as the pipeline design in the s-box implementation.

REFERENCES

[1] National Institute of Standards and Technology (NIST) (2001). Federal Information Processing Standard publication 197, the Advanced Encryption Standard (AES).

[2] Karthigaikumar, P., &Baskaran, K. (2010). An ASIC implementation of low power and high throughput blowfish crypto algorithm. Microelectronics Journal, 41, 347–355.

[3] Gielata, A., Russek, P., &Wiatr, K. (2008). AES hardware implementation in FPGA for algorithmacceleration purpose. In Proceedings of the international conference on signals and electronic systems(ICSES) (pp. 137–140), September 14–17, 2008.

[4] Farashahi, R. R., Rashidi, B., &Sayedi, S. M. (2014). FPGA based fast and high-throughput 2-slowreturning 128-bit AES encryption algorithm. Microelectronic Journal, 45, 1014–1025.

[5] SrideviSathyaPriya, PalanivelKarthigaikumar, N. M. Siva Mangai&P. Kirti Gaurav Das (2016). An Efficient Hardware Architecture forHigh Throughput AES EncryptorUsingMUX Based Sub Pipelined S-Box. Wireless Personal Communications, Volume 88, Issue 54.

[6] Harrison, O., & Waldron, J. (2007). AES encryption implementation and analysis on commoditygraphics processing units. In 9th workshop on cryptographic hardware and embedded systems (CHES2007), (Vol. 4727, pp. 209–226).

[7] Tim Good and Mohammed Benaissa, "Very Small FPGA Application-Specific Instruction Processor for AES", IEEE Transactions on Circuits and Systems – I: Regular Papers, Vol. 53, No. 7, July 2006, Page no:1477-1486.

[8] Parhi, K. K.andZhang, X.," High speed VLSI architectures for the AES algorithm", IEEE Transactions on Very Large Scale Integration (VLSI) systems, vol. 12, no. 9, pp. 957–967., 2004.

[9] Youssef, A. M and Kamal, A. A, " An area optimized implementation of the advanced encryption standard", In International conference on microelectronics , vol 6, no. 5, pp 159–162, 2008.

[10] Hammad, I., El-Sankary, K., & El-Masry, E. (2010). High speed AES encryptor with efficient merging techniques. IEEE Embedded Systems Letters, 2(3), 67–71.

[11] Fan, C. P., & Hwang, J. K. (2008). FPGA implementations of high throughput sequential and fully pipelined AES algorithm. International Journal of Electrical Engineering, 15(6), 447–455.

[12] Kumar, S., Sharma, V. K and Mahapatra, K. K. (2013). Low latency VLSI architecture of S-box for AESencryption. In International conference on circuits, power and computing technologies (ICCPCT) (pp.694–698), 20–21 March 2013.

[13] Rahimunnisa, K., Karthigaikumar, P., Rasheed, S., &Jayakumar, J. (2012). FPGA implementation ofAES algorithm for high throughput using folded parallel architecture. Security and CommunicationNetworks, 7(11), 2225–2236

[14] Mathew, S. K., Sheikh, F., Kounavis, M., Gueron, S., Agarwal, A., Hsu, S. K., et al. (2011). 53 GbpsGF(24)2 native composite-field AES-encrypt/decrypt accelerator for content-protection in 45 nm highperformancemicroprocessors. IEEE Journal of Solid-State Circuits, 46(4), 767–776.

[15] Navaneethan, S., and N. Nandhagopal. "RE-PUPIL: resource efficient pupil detection system using the technique of average black pixel density." Sādhanā 46.3 (2021): 114.