S-Box using AES Technique

H. Anusuya Baby[1], Christo Ananth[2]
[1](PG Scholar/ECE, Francis Xavier Engineering College/ Anna University , India)
[2](Assistant professor/ECE, Francis Xavier Engineering College/ Anna University , India)

***Abstract*— To design a substitution box (S-BOX) using both encryption and decryption. From that, the proposed system can achieve a higher throughput and higher energy efficiency. The S-BOX is designed by using Advanced Encryption Standard (AES). The AES is a symmetric key standard for encryption and decryption of blocks of data. In encryption, the AES accepts a plaintext input, which is limited to 128 bits, and a key that can be specified to be 128 bits to generate the Cipher text. In decryption, the cipher text is converted to original one. By using this AES technique the original text is highly secured and the information is not broken by the intruder. From that, the design of S-BOX is used to protect the message and also achieve a high throughput , high energy efficiency and occupy less area.**

Keywords— ***Advanced Encryption Standard (AES), Substitution Byte, Cryptography***.

I.INTRODUCTION

Cryptography is a technique which is used to protect the information. In 1997 the National Institute of Standards and Technology (NIST), a branch of the US government, started a process to identify a replacement for the Data Encryption Standard (DES). It was generally recognized that DES was not secure because of advances in computer processing power. The goal of NIST was to define a replacement for DES that could be used for non-military information security applications by US government agencies. Of course, it was recognized that commercial and other non-government users would benefit from the work of NIST and that the work would be generally adopted as a commercial standard. The NIST invited cryptography and data security specialists from around the world to participate in the discussion and selection process. Five encryption algorithms were adopted for study. Through a process of consensus the encryption algorithm proposed by the Belgium cryptographers Joan Daeman and Vincent Rijmen was selected. Prior to selection Daeman and Rijmen used the name Rijndael(derived from their names) for the algorithm. After adoption the encryption algorithm was given the name Advanced Encryption Standard (AES) which is in common use today.

In 2001, the National Institute of standards and Technology (NIST) found the Advanced Encryption Standard (AES) Technique. It can be implemented in hardware. There are a lot of disadvantages in DES Technique. It is insecure and the message is easily broken by the intruder. AES Technique has been widely used in a variety of applications such as secure communication systems and high throughput data servers.

.

The AES encryption algorithm is a block cipher that uses an encryption key and a several rounds of encryption. A cipher key is an encryption algorithm that works on a single block of data at a time. In the case of standard encryption technique the data is 128 bits, or 16 bytes, in length. The term “rounds” refers to the way in which the encryption algorithm mixes the data re-encrypting it ten to fourteen times depending on the length of the key.

AES encryption uses a single key as a part of the encryption process. The key can be 128 bits (16 bytes), 192 bits (24 bytes) or 256 bits (32 bytes) in length. The term 128-bit encryption refers to the use of a 128-bit encryption key. With AES both the encryption and the decryption are performed using the same key. This is called a symmetric encryption algorithm. Encryption algorithm uses two different keys that is public and a private key. Both are called asymmetric encryption algorithm key technique. An encryption key is simply a binary string of data used in the encryption process. Because the same encryption key is used to encrypt and decrypt data, it is important to keep the encryption key as a secret and to use the keys that are hard to guess. Some keys are generated by software used for this specific task. Another method is to derive a key from a pass phrase. Good encryption systems never use a pass phrase alone as an encryption key.

The previous techniques used in the encryption are parallel mix column and one target one process. The parallel mix column occupies more area and delay. The one term process also occupies more area and delay. So the complex parallelism is introduced.. By using this technique , a higher energy efficiency is achieved and also delay reduction is possible. This technique is applied for so many applications like military purpose, computer password and so on.

The reminder of this paper is organized as follows: section 2 explains the basic types of encryption. The encryption types involve the brief explanation about four techniques which we have given in the abstract. Section 3 presents the complex parallelism. The section also explains the cyclic loop of this mechanism. Section 4 discuss the simulation results of one target one process, parallel mix column and Complex parallelism.

II.TECHNIQUES INVOLVE IN ENCRYPTION

AES is a symmetric encryption algorithm, and it takes a 128-bit data as a input and performs several rounds of transformations to generate output cipher text. It is a computer security standard issued by NIST for protecting the electronic data. The basic processing unit used in this AES algorithm is byte. AES is used to encrypt/decrypt data blocks of 128-bits and it can be implemented in both hardware and software. AES acts as a block cipher which operates on fixed length group of bits of data. AES is a stream cipher which means the plain text bits are encrypted one and set of transformations have been applied to the bits. It may vary during encryption process. The plain text input and cipher output are the blocks of 128 bits. The number of rounds depend on key size. Each 128-bit is processed in a permutation and rotation operation. There are different techniques involved in this encryption.

A. Substitution Byte:

It is a non- linear substitution byte. Each Byte is replaced by another byte. This substitution Byte uses S-BOX for generating the cipher text. This S-box involves two process. First one is used to take the multiplicative inverse of finite field of the matrix (i.e input data). Secondly, the Affine Transformation is applied to the output of multiplicative inverse. Area reduction is possible in this finite field and finite field is used to create a compact field AES implementation. In new technology, the S-Box can be obtained from its truth table by using two level logic such as sum of products and product of sum. If the above mentioned technology is used , the primitive logic cells can be reduced and also cell size can be optimized using synthesis tool. The S- Box is computed from inverse of input to the original input. The example of Affine Transformation is given by

$$\begin{bmatrix} T1 \\ T2 \\ T3 \\ T4 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} F1 \\ F2 \\ F3 \\ F4 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

Consider an example, 4X4 matrix is a input text and [s1 s2 s3 s4] is the inverse of the input . the remaining one [0 1 1 0] is a cipher key. The output is a [z1 z2 z3 z4]. The input is multiplied with inverse of input with a cipher key and the output is obtained. The inverse input is the multiplicative inverse of the given input matrix. The example of Substitution Byte is given below.

1	2	3	4
5	6	7	8
9	$\boxed{10}$	11	12
13	14	15	16

Example of Substitution Byte input

1	2	3	4
5	6	7	8
9	$\boxed{22}$	11	12
13	14	15	16

Operation of Substitution Byte

B. Shift Row:

The technique used in this model is the transformation of the row. Consider a 4x4 matrix, the first row of the matrix remains unchanged. The second row , first bit is shifted to the last one. Then the last one is shifted to the third place. Finally the third row and forth row is finally rotated. The message is shuffled. In otherwords, The row transformation can be expressed as a reconstruction of the matrix using an key expression for each element. The row expressions calculate circular transformation .The example of shift row is given below.

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

Example of Shift Row input

1	2	3	4
6	7	8	5
11	12	9	10
16	13	14	15

Operation of Shift Row

C. Mix Column:

During this process, the matrix of the input column is shuffled. From that, the message is unbroken. It is similar to Substitution Byte. It uses the polynomial function. It is also based on finite field multiplication. The Mix column is based on the multiplication of two matrices and xor operation of both input and cipher key.

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

Example of Mix Column input

1	3	2	4
5	7	6	8
9	11	10	12
13	15	14	16

Operation of Mix Column

D. Add Round Key:

The sender sends a message to the receiver using a password (i.e key). The key is known by both sender and receiver. The key is added to the input (which is in the form of cipher text). The message is not hackable by any other intruders and also the information is more shuffled and secure.

III. ANALYSIS OF SYSTEM TECHNIQUES

The different types of technique involve in this encryption and decryption. There are three techniques

A. One Target and One Process:

The input is fed to the add round key. so the key is mixed with input (i.e cipher text). Then the output of the add round key is shuffled with sub byte, shift row, mix column and add round key. This process is repeated upto nine times. Then the ouput is processed with key elongation process. The output of key elongation is send to the final stage add round key. Finally the cipher text is generated from the plain text by using this OTOP technique.

B. Parallel Mix Columns:

The OTOP model is easily hackable by intruder. So the efficiency of OTOP model is small. This process is similar to the OTOP model. The input is fed to the add round key (i.e cipher text). The output is fed to the sub byte and shift row. The output of the shift row is added to the parallelizing mix column for shuffling the message. Then the output is added to the add round key. The process is repeated for nine times. The key elongation process is applied to the final stage output. The efficiency of parallel mix column is much higher than OTOP model. The area reduction is possible in this parallel mix column.

C. Complex Parallelism:

The input is fed to the four main blocks that is replacement bye, row transformation, shuffle the column and xor operation with key. The process is simulated upto nine times. The process is optimized with complex parallelism and the message is secure with cipher keys.

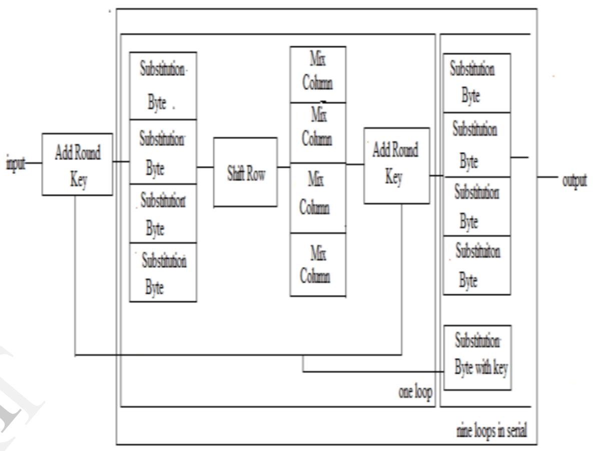


Fig 1: Complex Parallelism

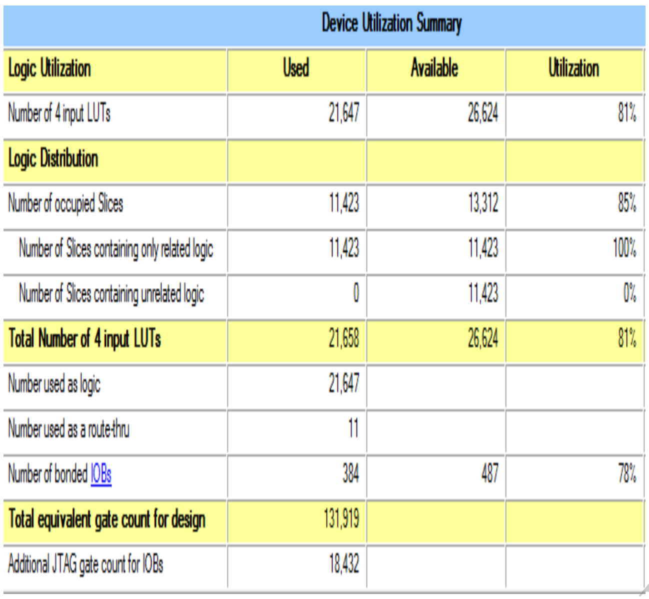
First the input is fed to the xor operation with key. The process involves in this stage is inserting a key to the input data. Then we have to send the data to replacement byte. Parallelizing the replacement byte is used to secure the message. The message is much more shuffled by combining the replacement byte. Then the next one is row transformation. This is used to transfer or shift the data. Then the next step is shuffle the column. It is used to shuffle the input with key. this is done by polynomial function. Then the last one is xor operation with key. The input is xored with key. The process is repeated upto nine times for shuffling the message. Finally the original text is covered by cipher key and the output of the data is cipher text(only with cipher keys). The cipher text information is unbroken by any other intruder. Finally the cipher text text input is given to the reverse process of complex parallelism. The original message is received by the S-box. Thus The information is secured by using complex parallelism.

IV RESULTS AND DISCUSSION

The information is encrypted by using complex parallelism. The simulation results of OTOP model and Parallel Mix Column are discussed below. Finally the encrypted output of complex parallelism is also given below.

A. One Target One Process:

The input is a 128-bit. The plain text is given to the OTOP encryption key. The cipher text is generated by using cipher keys. The OTOP model involves the process of SubBytes, Shift Row, Mix Column and Add Round key. All the above process is used to perform the message shuffling purpose. The permutation and rotation process are done by using the key elongation process. The message is secure and the information is shuffled for security purposes. The number of LUTs are reduced by 0.8%. Then the number of occupied slices are decreased by 0.9%. The gate count is increased in this OTOP model.



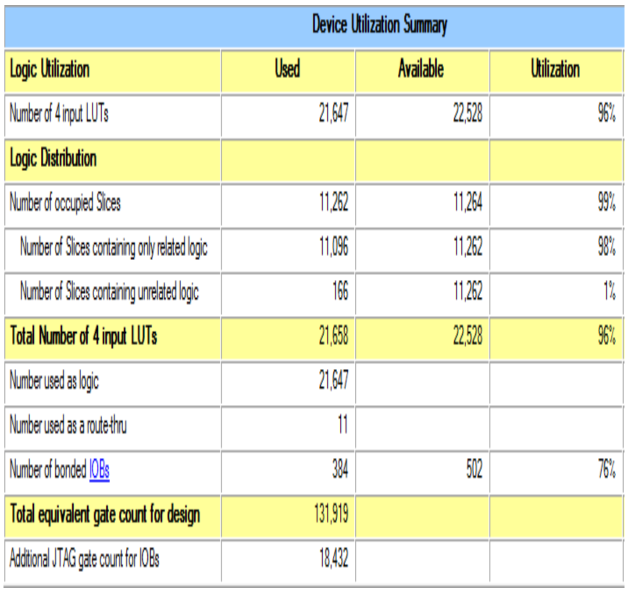
Device Utilization Summary			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of 4 input LUTs	21,647	26,624	81%
Logic Distribution			
Number of occupied Slices	11,423	13,312	85%
Number of Slices containing only related logic	11,423	11,423	100%
Number of Slices containing unrelated logic	0	11,423	0%
Total Number of 4 input LUTs	21,658	26,624	81%
Number used as logic	21,647		
Number used as a route-thru	11		
Number of bonded IOBs	384	487	78%
Total equivalent gate count for design	131,919		
Additional JTAG gate count for IOBs	18,432		

Fig 2: Compilation of OTOP model

The figure 2 shows the compilation of OTOP model. The area utilization is 81% in this OTOP model. The delay is calculated in the comparison table for area efficiency.

B.Parallel Mix Column:

The input is a 128-bit. The plain text is given to the Parallel Mix Column encryption key. The cipher text is generated by using cipher keys. The Parallel Mix Column model involves the process of SubBytes, Shift Row, Mix Column and Add Round key. All the above process is used to perform the cyclic rotation for rotating the input keys . The key elongation process is done by using key rotation. The message is safe and the information is shuffled for security purposes. The number of LUTs are reduced by 0.9%. Then the number of occupied slices are decreased by 0.9%. The gate count is increased in this Parallel Mix Column model.

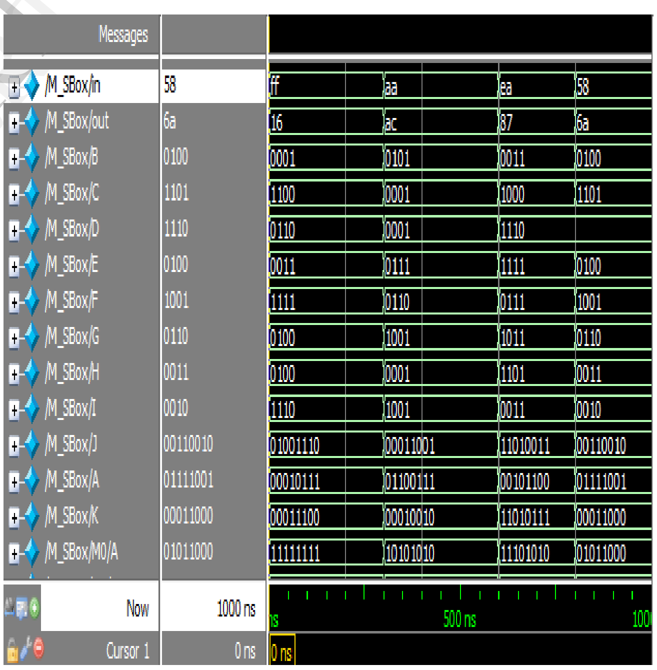


Device Utilization Summary			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of 4 input LUTs	21,647	22,528	96%
Logic Distribution			
Number of occupied Slices	11,262	11,264	99%
Number of Slices containing only related logic	11,096	11,262	98%
Number of Slices containing unrelated logic	166	11,262	1%
Total Number of 4 input LUTs	21,658	22,528	96%
Number used as logic	21,647		
Number used as a route-thru	11		
Number of bonded IOBs	384	502	76%
Total equivalent gate count for design	131,919		
Additional JTAG gate count for IOBs	18,432		

Fig 3: Compilation of Parallel Mix Column

The figure 3 shows the compilation of Mix Column. The area utilization is 96% in this Parallel Mix Column. The path route delay is calculated

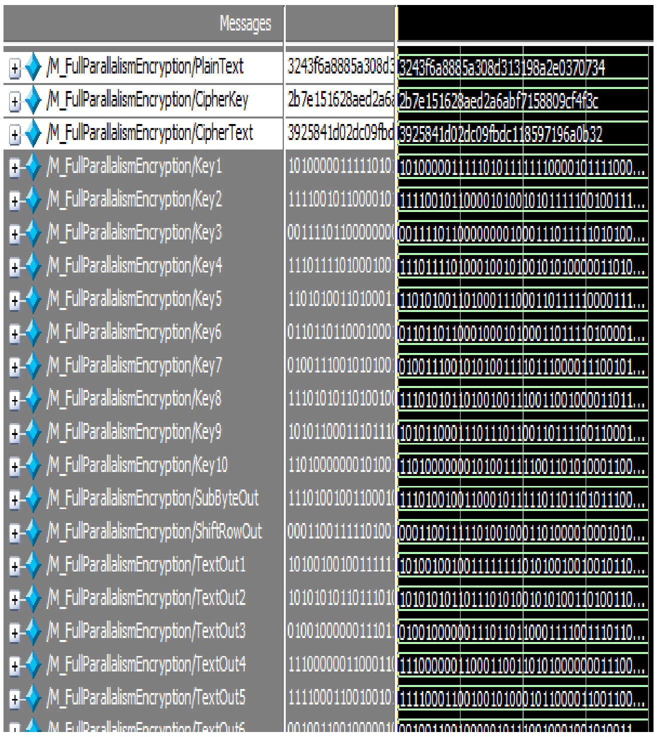
F.Complex Parallelism:



```
Messages
/M_SBox/in       58         ff        aa        ea        58
/M_SBox/out      6a         16        ac        87        6a
/M_SBox/B        0100       0001      0101      0011      0100
/M_SBox/C        1101       1100      0001      1000      1101
/M_SBox/D        1110       0110      0001      1110
/M_SBox/E        0100       0011      0111      1111      0100
/M_SBox/F        1001       1111      0110      0111      1001
/M_SBox/G        0110       0100      1001      1011      0110
/M_SBox/H        0011       0100      0001      1101      0011
/M_SBox/I        0010       1110      1001      0011      0010
/M_SBox/J        00110010   01001110  00011001  11010011  00110010
/M_SBox/A        01111001   00010111  01100111  00101100  01111001
/M_SBox/K        00011000   00011100  00010010  11010111  00011000
/M_SBox/M0/A     01011000   11111111  10101010  11101010  01011000
Now              1000 ns    ns        500 ns    100
Cursor 1         0 ns       0 ns
```

Fig 4: Compilation of S-Box

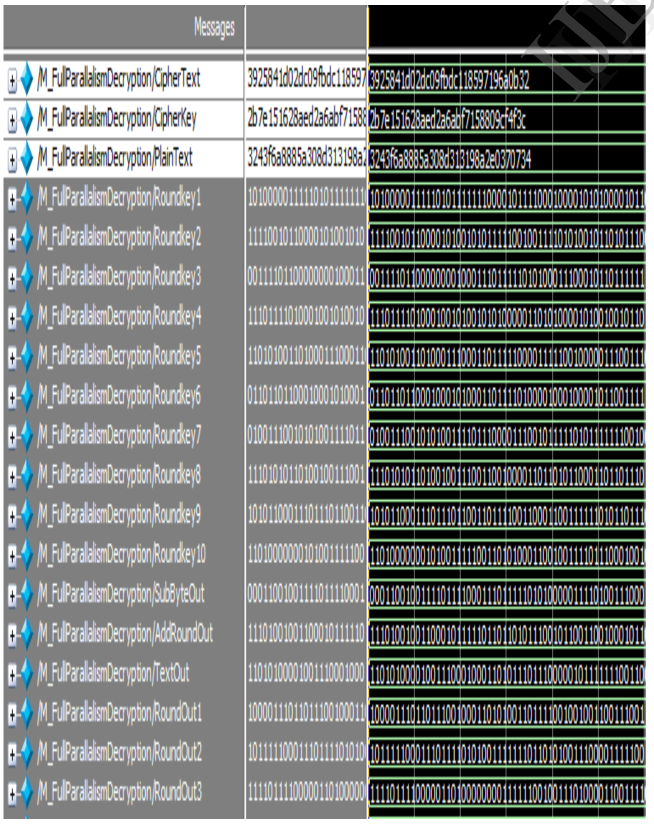
The design of S-Box is used to the protect the message. The figure 4 shows the compilation of substitution byte. The above figure shows the shuffling of message and also elongating the key.



```
Messages
/M_FullParallalismEncryption/PlainText    3243f6a8885a308d3   3243f6a8885a308d313198a2e0370734
/M_FullParallalismEncryption/CipherKey    2b7e151628aed2a6a   2b7e151628aed2a6abf7158809cf4f3c
/M_FullParallalismEncryption/CipherText   3925841d02dc09fbd   3925841d02dc09fbdc118597196a0b32
/M_FullParallalismEncryption/Key1         1010000011111010    10100000111110101111110000101111000...
/M_FullParallalismEncryption/Key2         1111001011000010    11110010110000101001010111110010011 1...
/M_FullParallalismEncryption/Key3         0011110110000000    00111101100000000100011101111 1010100...
/M_FullParallalismEncryption/Key4         1110111101000100    11101111010001001010010101000011010...
/M_FullParallalismEncryption/Key5         1101010011010001    11010100110100011100011011111 0000111...
/M_FullParallalismEncryption/Key6         0110110110001000    01101101100010001010001101111 0100001...
/M_FullParallalismEncryption/Key7         0100111001010100    01001110010101001111011100001 1100101...
/M_FullParallalismEncryption/Key8         1110101011010010    11101010110100100111001100100001 1011...
/M_FullParallalismEncryption/Key9         1010110001110111    101011000111011101100110111100110001...
/M_FullParallalismEncryption/Key10        1101000000010100    11010000000101001111100110101 0001100...
/M_FullParallalismEncryption/SubByteOut   1110100100110001    11101001001100010111110110110 1011100...
/M_FullParallalismEncryption/ShiftRowOut  0001100111110100    000110011111010010001101000010001010...
/M_FullParallalismEncryption/TextOut1     1010010010011111    101001001001111111101010010010010110...
/M_FullParallalismEncryption/TextOut2     1010101011011101    101010101101110101001010100110100110...
/M_FullParallalismEncryption/TextOut3     0100100000011101    01001000000111011011000111100111 0110...
/M_FullParallalismEncryption/TextOut4     1110000000110001    11100000001100011001101010000000 11100...
/M_FullParallalismEncryption/TextOut5     1111000110010010    1111000110010010100010110000110011 00...
/M_FullParallalismEncryption/TextOut6     0010011001000001    00100110010000010111001000100101001 1...
```

Fig 5: Encrypted data of Complex Parallelism

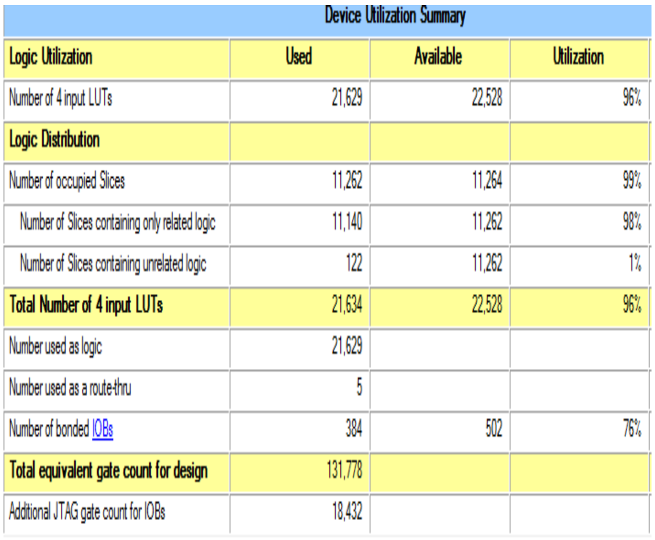
The figure 5 shows the proces of complex parallelism encryption. It will show the complex parallelism process. The process involves the operation of one task one processor, Parallel Mix columns and complex parallelism.



```
Messages
/M_FullParallalismDecryption/CipherText   3925841d02dc09fbdc118597   3925841d02dc09fbdc118597196a0b32
/M_FullParallalismDecryption/CipherKey    2b7e151628aed2a6abf71588   2b7e151628aed2a6abf7158809cf4f3c
/M_FullParallalismDecryption/PlainText    3243f6a8885a308d313198a2   3243f6a8885a308d313198a2e0370734
/M_FullParallalismDecryption/Roundkey1    10100000111110101111111    10100000111110101111110000101111000100001010100001011
/M_FullParallalismDecryption/Roundkey2    11110010110000101001010    111100101100001010010101111100100111101010010110101110
/M_FullParallalismDecryption/Roundkey3    00111101100000000100011    001111011000000001000111011111010100011100010110111111
/M_FullParallalismDecryption/Roundkey4    11101111010001001010010    111011110100010010100101010000110101000010100100101 10
/M_FullParallalismDecryption/Roundkey5    11010100110100011100011    110101001101000111000110111110000111110010000011100111
/M_FullParallalismDecryption/Roundkey6    01101101100010001010001    011011011000100010100011011110100001000100001011001111
/M_FullParallalismDecryption/Roundkey7    01001110010101001111011    01001110010101001110111000011100101111101011111110010
/M_FullParallalismDecryption/Roundkey8    11101010110100100111001    111010101101001001110011001000011011010110001101101110
/M_FullParallalismDecryption/Roundkey9    10101100011101110110011    101011000111011101100110111100110001100111111010110111
/M_FullParallalismDecryption/Roundkey10   11010000000101001111100    11010000000101001111100110101000110010011101110001001
/M_FullParallalismDecryption/SubByteOut   00011001001111011110001    000110010011110111100011101111101010000011110100111000
/M_FullParallalismDecryption/AddRoundOut  11101001001100010111110    111010010011000101111101101101011100101100110010001011
/M_FullParallalismDecryption/TextOut      11010100001001110001000    110101000010011100010001101011101110000010111111100110
/M_FullParallalismDecryption/RoundOut1    10000111011011100100011    10000111011011100100011010100110111100100100110011100 1
/M_FullParallalismDecryption/RoundOut2    10111110001110111101010    10111110001110111010100111111101101010011100001111100
/M_FullParallalismDecryption/RoundOut3    11110111100000110100000    111101111000001101000000001111110010011101000011001111
```

Fig 7: Decrypted data of Complex Parallelism

It implements in 167 processor using complex parallelism technique. The process is the combination of various techniques like Substitution Byte, Shift Row Mix Column and Add Round Key. The message is secure and the delay is reduced by other methods. The figure 6 and 7 show the compilation of complex parallelism. The delay of complex parallelism is small compared to other techniques.



Device Utilization Summary			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of 4 input LUTs	21,629	22,528	96%
Logic Distribution			
Number of occupied Slices	11,262	11,264	99%
Number of Slices containing only related logic	11,140	11,262	98%
Number of Slices containing unrelated logic	122	11,262	1%
Total Number of 4 input LUTs	21,634	22,528	96%
Number used as logic	21,629		
Number used as a route-thru	5		
Number of bonded IOBs	384	502	76%
Total equivalent gate count for design	131,778		
Additional JTAG gate count for IOBs	18,432		

Fig 8: Compilation of Complex Parallelism

The above simulation results discuss the detailed description of three models. The comparison table of area and delay are discussed in the below section. The table 1 discusses the comparison of area with LUTs and route path. The table 2 discusses the comparison of delay with path and route delay.

TABLE 1: Comparison of Area

Encryption Name	LUT	Route Path
OTOP model	21658	11
Parallel Mix Column	21658	11
Complex Parallelism	21634	5

The OTOP model occupies more LUT in the hardware implementation. The number of gates in the OTOP model is very high .The parallel mix column occupies less LUT compare to OTOP model and high compare to complex parallelism. The number of gates occupied in the hardware is same as the OTOP model. The complex parallelism technique occupies less LUTs for hardware implementation. The number of paths in the complex parallelism are less compared to the both previous model. The path delay and route delay is efficient in the complex parallelism compared to the OTOP model and Parallel Mix Column.

TABLE 2: Comparison of Delay

Encryption Name	Delay	Gate Delay	Path Delay
OTOP model	307.909ns	105.862ns	202.047ns
Parallel Mix Column	232.742ns	116.830ns	115.912ns
Complex Parallelism	232.245ns	116.683ns	115.562ns

VI. CONCLUSION

In this brief, cryptography AES technique is presented to protect the information. To increase the efficiency, the complex parallelism technique is used to involve the processing of Substitution Byte, Shift Row, Mix Column and Add Round Key. Using S- Box complex parallelism, the original text is converted into cipher text. From that, we have achieved a 96% energy efficiency in Complex Parallelism Encryption technique and recovering the delay 232 ns. The complex parallelism that merge with parallel mix column and the one task one processor techniques are used. In future, Complex Parallelism single loop technique is used for recovering the original message.

VII. REFERENCES

[1] E. Kasper and P. Schwabe, “Faster and Timing-Attack Resistant AES-GCM,” Proc. 11th Int’l Workshop Cryptographic Hardware and Embedded Systems (CHES ’09), pp. 1-17, 2009
[2]Agarwal.A, Ander M.A, Gueron .S, Hsu. S.K, Kaul. H, Kounavis .M, S.K. Mathew, F. Sheikh, R.K. Krishnamurthy, “53 gbps Native GF(2^4) Composite-Field AES-Encrypt/Decrypt Accelerator for Content-Protection in 45 nm High-Performance Microprocessors,” IEEE J. Solid-State Circuits, vol. 46, no. 4, pp. 767-776, Apr. 2011.
[3].Bernstein.D and P. Schwabe, “New AES Software Speed Records,” Proc. INDOCRYPT ’08: Ninth Int’l Conf. Cryptology in India: Progress in Cryptology, pp. 322-336, 2008.
[4].Biham.E, “A Fast New DES Implementation in Software,” Proc. Fourth Int’l Workshop Fast Software Encryption, pp. 260-272, 1997.
[4].Borkar.S, “Thousand Core Chips: A Technology Perspective,”Proc. 44th Ann. Design Automation Conf., pp. 746-749, 2007.
[5].Cheng .W, D. Truong, T. Mohsenin, Z. Yu, T. Jacobson, G. Landge, M. Meeuwsen, C. Watnik, P. Mejia, A. Tran, J. Webb, E. Work, Z. Xiao, and B. Baas, “A 167-Processor 65 nm Computational Platform with Per-Processor Dynamic Supply Voltage and Dynamic Clock Frequency Scaling,” Proc. IEEE Symp. VLSI Circuits, June 2008.
[6].Cheng W.H, D.N. Truong, , T. Mohsenin, Z. Yu, A.T. Jacobson, G. Landge, M.J.Meeuwsen, A.T. Tran, Z. Xiao, E.W. Work, J.W. Webb, P. Mejia, and B.M. Baas, “A 167-Processor ComputationalPlatform in 65 nm CMOS,” IEEE J. Solid-State Circuits, vol. 44, no. 4, pp. 1130-1144, Apr. 2009.
[7].Chen Y.C, C.-J. Chang, C.-W. Huang, K.-H. Chang, and C.-C. Hsieh, “High Throughput 32-Bit AES Implementation in FPGA,” Proc. IEEE Asia Pacific Conf. Circuits and Systems, pp. 1806-1809, Nov. 2008.
[8].Gomez –Pulido. J, Granado-Criado.J, M. Vega-Rodriguez and J. Sanchez-Perez, “A New Methodology to Implement the AES Algorithm Using Partial and Dynamic Reconfiguration,” Integration, the VLSI J., vol. 43, no. 1, pp. 72-80, 2010.
[9].Guo.Z, S. Qu, G. Shou, Y. Hu and Z. Qian, “High Throughput, Pipelined Implementation of AES on FPGA,” Proc.Int’l Symp. Information Eng. and Electronic Commerce, pp. 542-545, May 2009.
[10].Hodjat.A and Verbauwhede.I, “Area-Throughput Trade-Offs for Fully Pipelined 30 to 70 Gbits/s AES Processors,” IEEE Trans. Computers, vol. 55, no. 4, pp. 366-372, Apr. 2006.
[11].Hodjat.A and Verbauwhede.I, “A 21.54 gbits/s Fully Pipelined AES Processor on FPGA,” Proc. IEEE 12th Ann. Symp. Field-Programmable Custom Computing Machines, pp. 308- 309, Apr. 2004.
[12].Kuo.H, I. Verbauwhede and P. Schaumont, “Design and Performance Testing of a 2.29 gb/s Rijndael Processor,” IEEE J. Solid-State Circuits, vol. 38, no. 3, pp. 569-572, Mar. 2003.