Communications on Applied Electronics (CAE) – ISSN : 2394-4714
Foundation of Computer Science FCS, New York, USA
Volume 7– No. 22, November – www.caeaccess.org

An Approach towards Design of N-Bit AES to Enhance Security using Reversible Logic

Rohini Hongal
ECE, Dept
KLE, Technological University
India

Jyoti H
ECE, Dept
KLE, Technological University
India

Rajashekar S
ECE, Dept
KLE, Technological University
India

ABSTRACT
The main aim of this paper is to provide higher security for several applications over the Internet by enhancing the overall strength of the existing Advanced Encryption Standard algorithm (AES). The standard AES uses block size of 128-bit and key sizes of 128, 192, 256 bits. AES is the most popular ,highly secure, faster and strong symmetric key, block cipher cryptographic algorithm today . But in nowadays, cyber-attacks are continuously developing. Shannon's theorem states that a one-on-one relationship between each message bit to each key bit (hence both key and message length equal) would give the best security. To improve the security of the original AES, message size is increased from standard block size of 128 bits to 192,256, 512 bits and key size is made as equal as the message block size. We get more security by the use of larger key size and the increased throughput from the larger input block size compared to original AES of 128 bits. The implementation of proposed algorithm require more area but it can be accepted as most of the applications require high level of security and high throughput. Further, paper discusses the use of reversible logic to mitigate the power attacks in conventional AES as reversible gates offer ideally zero internal power dissipation. The proposed work is designed and implemented using Xilinx tool and verified on Virtex-5 FPGA using chipscope. Results are discussed w.r.t power, delay and hardware required in terms of gate count is calculated and implemented using only Feynman gate with quantum cost 1.

General Terms
Security, Algorithms.

Keywords
Chipscope, FPGA, Reversible gates, Symmetric key, Quantum Cost.

1. INTRODUCTION
There were several hardware implementations of AES algorithm with standard block size of 128 and standard key size of 128,192,256 bits, as in [1].These various implementations of AES depend on the applications. Some applications need higher level of security with area restriction, where as some applications need higher throughput and stronger security without caring about area or time limitations. The various AES algorithms can be implemented based on our requirements. Many hardware implementations for AES were found in literature. These implementations were done for the standard AES of 128 block size with 128,192 ,256 bits key size, such as the work proposed in [1][2] [3]. The paper [1] presents the hardware implementation of AES Rijndael. Both encryption and decryption of AES algorithm are implemented using Xilinx Virtex-7 FPGA. The pre-calculated look-up tables (LUT) are used .High throughput and low latency are obtained in this approach. Less complex architecture is obtained by using pre-calculated LUTs. All the three standard key sizes of AES with standard block size of 128 are designed using Verilog - HDL. Advanced encryption standard AES-192 with multiple keys is proposed in the paper [2]. It is implemented using Field Programmable Gate Array (FPGA). The hardware implementation of AES algorithm with key size 128 bits and block size 128 bits is proposed in the paper[3].The algorithm is implemented using FPGA Artix7 Nexys 4 kit by configuring it with the help of Software Development Kit (SDK) in Xilinx ISE design suite. The paper [4] proposes the implementations of all the three standard key sizes of AES (128,192,256) with block size of 128 which will use 10, 12 and 14 iterations respectively. In this approach , the cloud user is provided with all the three key sizes and based on the size ,cloud user will select one of the three encryption techniques and process is done.

The paper [5] explains AES algorithm with hybrid approach of Dynamic Key Generation and Dynamic S-box Generation. In this approach first more complexity is added in data to increase Confusion and Diffusion in Cipher text by using Dynamic Key Generation and then by using Dynamic S-Box Generation. Our proposed paper presents variation of the original AES algorithm with message and key size equal. As more security is needed for applications which need high level of security regardless of area and computational time but due to increase in message size and block size complexity of algorithm is increased. The proposed AES algorithm with equal key and message size provides high level of security and the increased throughput. Using equal key size as that of message results in more security, and more throughput from using two, four times larger block size(i.e. 256,512 block size) than the block size used in the original AES.-128[6]. Our proposed AES algorithm uses 192,256,512 block sizes and same key sizes as that of block size[7][8]. This new algorithm is designed and synthesized using Hardware Description Language (HDL).The reversible logic gates are used to implement the design[9][10].

1.1 Reversible Logic Gates
Reversible logic is widely used nowadays and has got more popularity due to their ability to reduce the power dissipation which is the main requirement in low power VLSI design. A reversible logic gate has an n-input and n-output .It is a logic device with one-to-one mapping which means we can determine the outputs from the inputs and vice versa. We can implement one or more applications using single reversible logic gate.

Feynman gate is a 2*2 one through reversible gate as shown in Fig1. The inputs are A, B and the outputs are X, Y where Y=A, X=A ⊕ B. One of the outputs of Feynman gate is XOR gate operation and the other output is known as garbage which is used to maintain the reversibility [9][10].

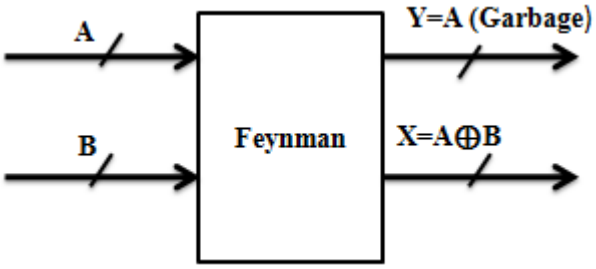

Fig 1: Feynman Gate

2. IMPLEMENTATION DETAILS

The new algorithm consist of the structure which is similar to original AES algorithm but having slight difference that is instead of using 128 bit plain text size and key size 128,192 or 256 bits, uses plaintexts of 192,256 and 512 bits and key size as that of message size 192,256 and 512 respectively that has impact on the whole algorithm structure. There are four transformations in the encryption of AES namely, Byte Substitution, Shift Row, Mix Column and Round Key Addition [11][12].

Analysis of Rijndael Algorithm passes through the 4 layers consists of

2.1 Byte Substitution

The AES algorithm performs operations on a two-dimensional array of bytes called the State. The state matrix consists of four rows of bytes, each containing N bytes, where N is the block length divided by 32. Each byte of data in state matrix is replaced by byte of data from look up table.

2.2 Shift Row Transformation

In shift row transformation, the rows of the state matrix are cyclically shifted by its row number.

- The first row (Row 0) is not shifted.
- The Row 1 is cyclically shifted by 1 byte to the left.
- The Row 2 is cyclically shifted by 2 the left.
- The Row 3 is cyclically shifted by 1 byte to the left or 3 bytes to the right

The shift row transformation of 8 -bit data is shown in Fig 2.It is implemented using reversible logic gate Feynman. In Fig 2.2 N is the amount to be shifted .The same procedure is repeated N times. The same procedure is followed for 48, 64 and 128 bit data transformation in case of 192,256,512 respectively.

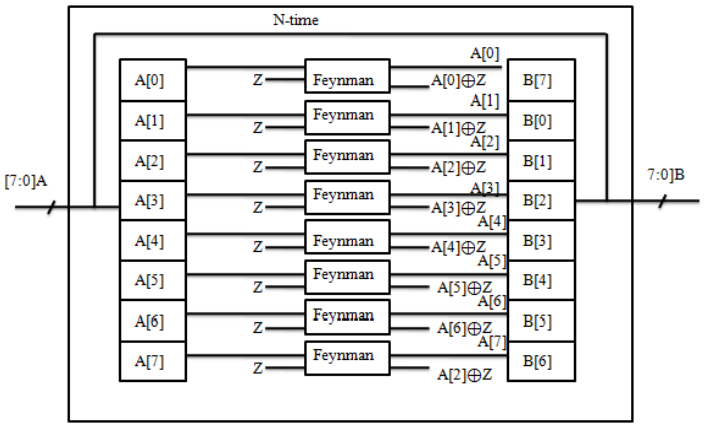

Fig 2: Reversible implementation of shift row transformation

2.3 Mix Column Transformation

The Mix column performs operations on the columns of the state matrix. The data is arranged in the form of two dimensional array of bytes known as state matrix. Each row of bytes in the state matrix is multiplied with the corresponding columns of constant matrix of 4x4.Mix column transformation is simulated using reversible logic gates. The fixed matrix used in our design is given below.

02 03 01 01

01 02 03 01

01 01 02 03

03 01 01 02

Multiplication of byte of data by 0 can be implemented using reversible gate shown in Fig 3.

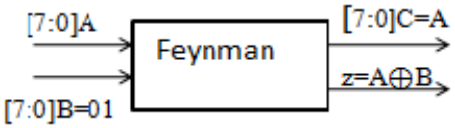

Fig 3: Reversible implementation of multiplication by constant value 01

Multiplication of byte of data by 02 can be implemented as shown in Fig 4. First byte of data is shifted left by 1-bit and then the shifted data is bitwise XOR ed with 00011011(1B) if the leftmost bit(LSB) of the original value before shift operation is 1. If MSB is 0, only 1-bit left shift is performed.

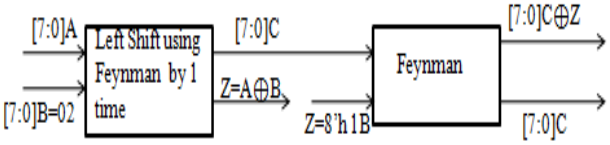

Fig 4: Reversible implementation of multiplication by constant value 02

Multiplication of byte of data by 03 can be implemented using reversible logic gate as shown in Fig 5. The byte of data is multiplied by 2 and then conditional bitwise XOR with the original byte of data before shifting it. The same condition is applied for multiplication by 2 as explained above.

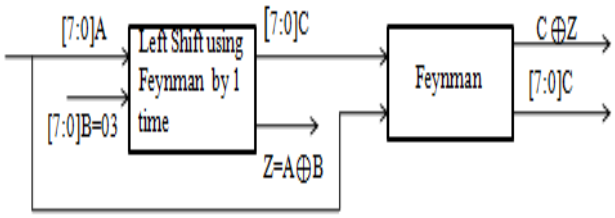

Fig 5: Reversible implementation of multiplication by constant value 03

The multiplication of 2x2 matrix can be written as

$$\begin{bmatrix} A1 & A2 \\ A3 & A4 \end{bmatrix} \begin{bmatrix} B1 & B3 \\ B2 & B4 \end{bmatrix} = \begin{bmatrix} C1 & C2 \\ C3 & C4 \end{bmatrix}$$

The reversible implementation of 2x2 matrix is shown in Fig 6. The multiply is carried out as explained above for different cases (i.e by 01,02,03).The same design is used for multiplication of 4x6,4,8 and 4x16 matrix in case of 192,356 and 512 bits data respectively.

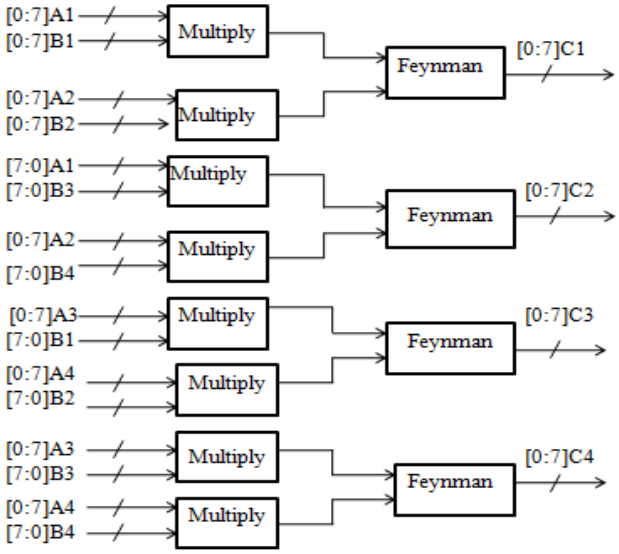

Fig 6: Reversible implementation of mix column matrix for 2x2 matrix

2.4 Add Round Key

There are different number of rounds depending on the key size and block size used. In the first round of encryption, the main key is XOR ed with the plaintext. In the remaining rounds except final round the sub keys obtained for different rounds are XOR ed with the output obtained from the mix column transformation. In the final round the last round sub key is XOR ed with the output of shift row transformation. Reversible logic of addround key is shown in Fig 7.

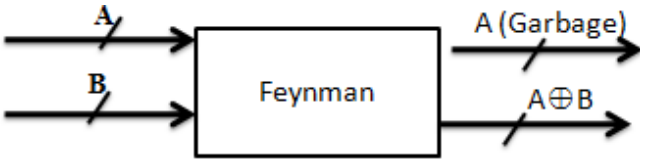

Fig 7: Reversible implementation of addround key

2.5 Example of State Matrix Arrangement

The arrangement of state matrix for all the three plaintexts (i.e,192,256,512) can be written as.

1. AES encryption process takes input of 192-bit plaintext and 192 bit key and produces output of 192 - bit ciphertext. The 192-bit data blocks are divided by 32 to get number of bytes in each row. Dividing 192 by 32 we will get 6.So in each row 6 bytes and therefore there are four rows .So total number of bytes is 6x4 24 bytes. These 24 bytes are arranged as 4×6 State Matrix.

➢ Plaintext in Hex(192 bits) : 45 32 E6 EF 22 4A 67 89 19 42 A6 5D 53 29 A6 20 43 92 25 98 6A 4D 3F 2A

State Matrix can be written as

45 22 19 53 43 6A

32 4A 42 29 A6 20

E6 67 A6 A6 25 3F

EF 89 5D 20 98 2A

2. The 256-bit data blocks are divided by 32 to get number of bytes in each row. Dividing 256 by 32 we will get 8.So in each row 8 bytes and therefore there are four rows .So total number of bytes is 8x4 32 bytes. These 32 bytes are arranged as 4×8 State Matrix.

.

➢ Plaintext in Hex (256 bits): 55 86 67 34 A3 E0 8D 97 45 4E 57 E8 79 0F 62 57 43 97 F6 2B FA 6E 56 02 E4 A9 E6 15 2C 54 37 9F

State Matrix can be written as

55 A3 45 79 43 FA E4 2C

86 E0 4E 0F 97 6E A9 54

67 8D 57 62 F6 56 E6 37

34 97 E8 57 2B 02 15 9F

3.The 512-bit data blocks are divided into groups of 64 bytes each, and arranged as 4×16 State Matrix. The round key function is performed 11 times instead of 22 i.e, according to smith formula we will get 22 i.e, Nb/4 +6 ,where Nb is the number of bytes of key size. As key size and message size is increased, the number of rounds has been reduced.

➢ Plaintext in Hex (512 bits): 01 C8 24 E1 3C F2 BC 18 E7 93 B5 BD 47 29 A2 3E 63 AB 1F 4A EB 03 4E 59 9F AF 22 B8 A0 C7 74 A1 AB 74 4E 7B 48 49 8D 06 0E 8E D7 96 E5 1B 40 78 5C 43 C6 A9 62 0E 57 C0 C8 09 08 EB FE 3D F8 7F 37

State Matrix can be written as

01 3C E7 47 63 EB 9F A0 AB 48 0E E5 43 0E 09 3D

C8 F2 93 29 AB 03 AF C7 74 49 8E 1B C6 57 08 F8

24 BC B5 A2 1F 4E 22 74 4E 8D D7 40 A9 C0 EB 7F

E1 18 BD 3E 4A 59 B8 A1 7B 06 96 78 62 C8 FE 37

2.6 AES -256 Round 0 -14 Cipher Texts

AES encryption process takes as input an 256-bit plaintext and 256 bit key and produces as output an 256 - bit ciphertext.

Plaintext:
54732067544F4E2068204B20776E6954616D75466F656E77
74796E752020656F

Key
4537027645F4E6820204B2077656954616789466F656E76D
4796E752020656F7

Round0:
1144221111bba8a26a24f9270138fc127715e1209933891a33e
f892722263398

Round1:
6f8b40c6ef89ff040f594668c100aa0e6e5f08c9180f33a684fc5c
c8c4d9e192

Round2:
d9704b26c76648c5bef0cb38ef9f2310eb2f0d98595f1d4f43aae
e2231c256d1

Round3:
0284122aeba53e3c8438157c908859f0c837ae36f62ff78396c2
c2e1ad3a4c31

Round4:
3f87b15cb0801100c05b4801054ec5b12b940243c2cfb90e36d
6d949bc4a6504

Round5:
aba332315e369f53fe8f19f613a15d81e1ec6998654ff77ddc9a5
c0eba19872e

Round6:
78bea8ef8f7fe5d86ca310ab016dae9ba6dfe08df84add82417e2
9de85ac3a67

Round7:
1721b141e0c68080837bca4eeae7f8535ce26870ac1720c49dd
925f215ec052e

Round8:
5b931b9c76a240120d6cbd33d8e5d8330683361ce30669ef522
dbec2e8dcc316

Round9:
7e1008493cdabbf2036fa968d43979c882feb32143554eb9cf89
d62f739f3141

Round10:
1ad5ac5b1b2d864cf76aef3864b2d50ccfcee720590334f143f1c
40ef18e3e7f

Round11:
9bd19bc92d700143ef0eaba042a77acdec74c556b2101359139
e9aa22324951c

Round12:
93371181e14b70cb105d82e8002542195a96ec3bd871f5b6c2f
0668167cf4217

Round13:
b20000f839cdc9495814f0280cb7010d3f27efb4a5228dc41b86
66f15a5a851d

Round14:
163f41a97f7f215671179d677b0518a080377512626de422de1
220742c9b6f18

3. RESULTS AND DISCUSSIONS

Verilog coding is done to check functionality of proposed work and verified on Vrtex5 (ML505) FPGA using chipscope. Simulation results of all 3 key size AES (192bit, 256bit and 512bit) are shown in figures 8.9.10. respectively. Also using tool power and delay generated is tabulated in Table 1 and shown in Fig 11,12 & 13. .Hardware required in different phases of design in terms gate count is tabulated in Table 2.

3.1 Simulation Result of 192 Bit Plaintext And Key

plaintext:
45221953436a324a4229a620e667a6a6253fef895d20982a

key:
54732067544f68204b20776e616d75466f6574796e752020

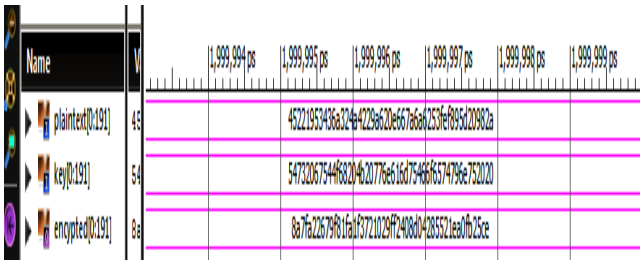

Fig 8: Simulation result of 192 bit

3.2 Simulation Result of 256 Bit Plaintext And Key

Plaintext:
54732067544F4E2068204B20776E6954616D75466F656E77
74796E752020656F

Key:
4537027645F4E6820204B2077656954616789466F656E76D
4796E752020656F7

Encrypted Output:
163f41a97f7f215671179d677b0518a080377512626de422de1
220742c9b6f18

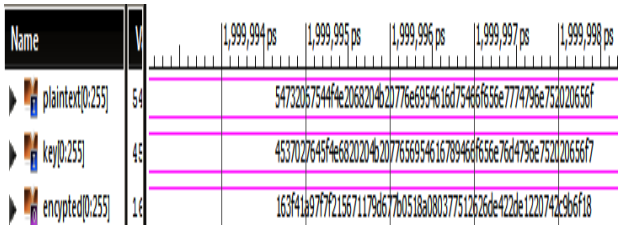

Fig 9: Simulation result of 256 bit

3.3 Simulation Result of 512 Bit Plaintext And Key

Plaintext:
013c6e4763eb9fa0ba84e81b430e093dc02f9392ab30afc775a4
8d40c65708f820cb2ba21f4e2274f48d067da9c0eb7f0e081b31
54590b1a7a320e5d62c8fe37

Key:

7e1008493cdabbf2036fa968d43979c882feb32143554eb9cf89
d62f739f3141Aba332315e369f53fe8f19f613a15d81e1ec6998
654ff77ddc9a5c0eba19872e

Ciphertext:

4a7e43316aec6d7df4f99fdd138503cff6b65614371ba8efed03c c374dfc1b8f5e1786bafd2
59ab85ac8ae485ca1fa3512705366fa5f3faeac447f29d525d03c

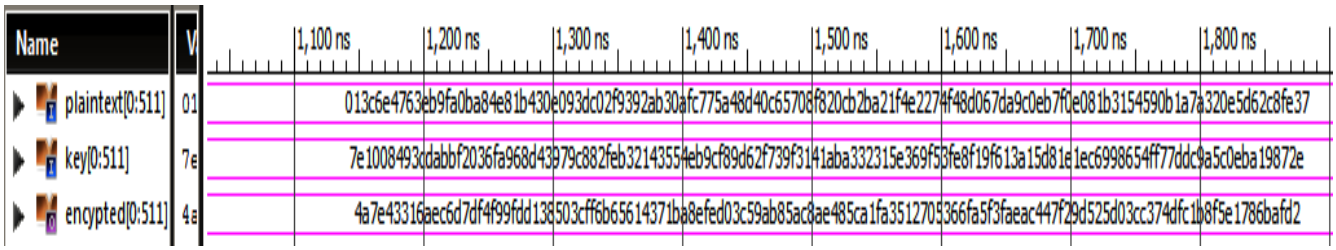

Fig 10: Simulation result of 512 bit

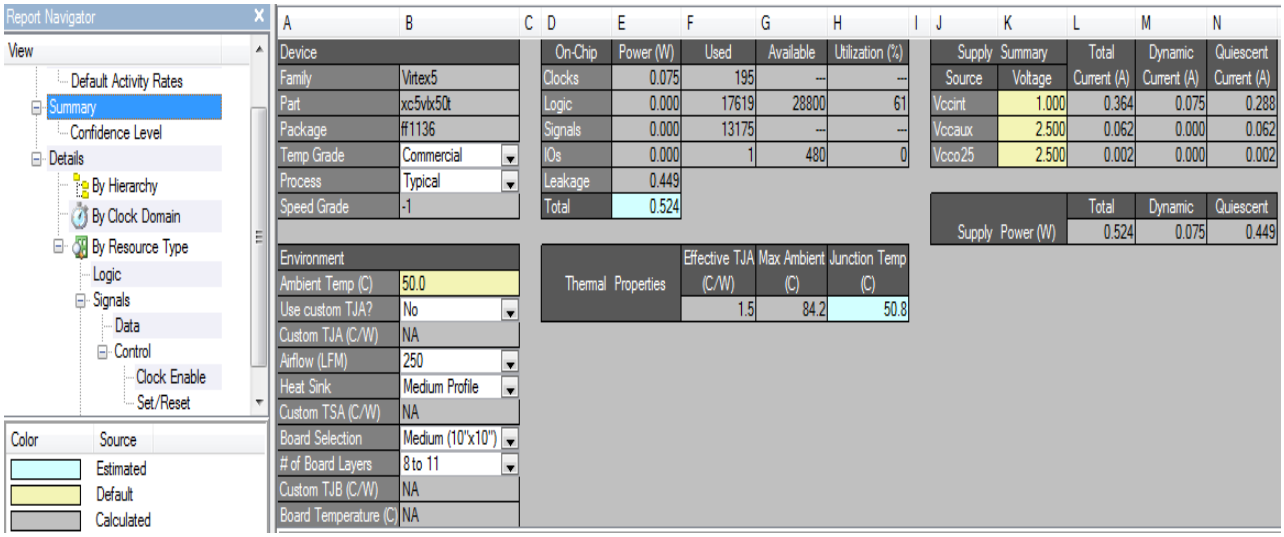

Fig 11: Power Analysis of 192 bit

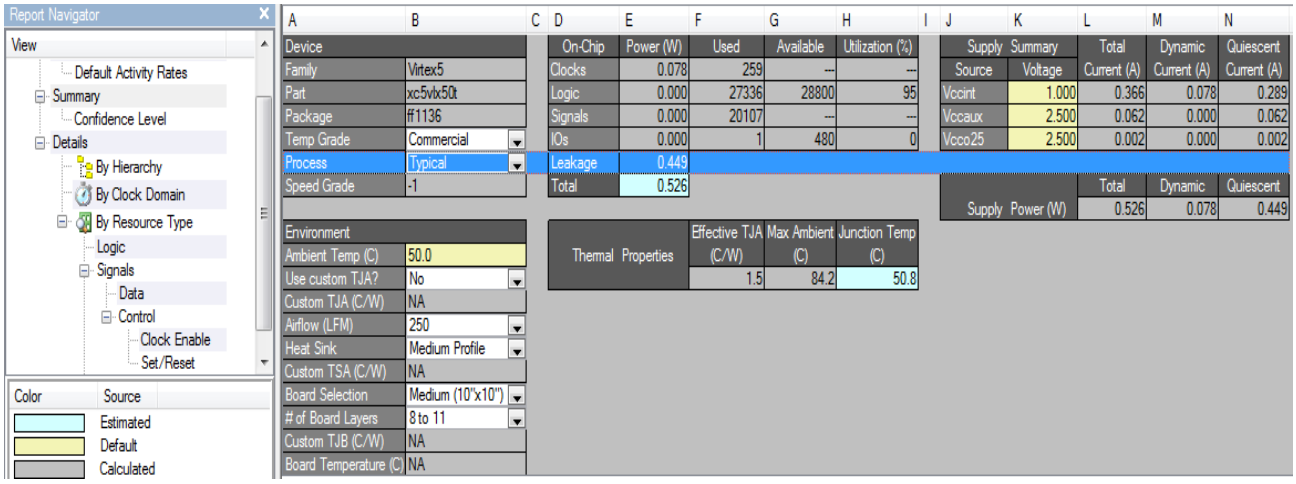

Fig 12: Power Analysis of 256 bit

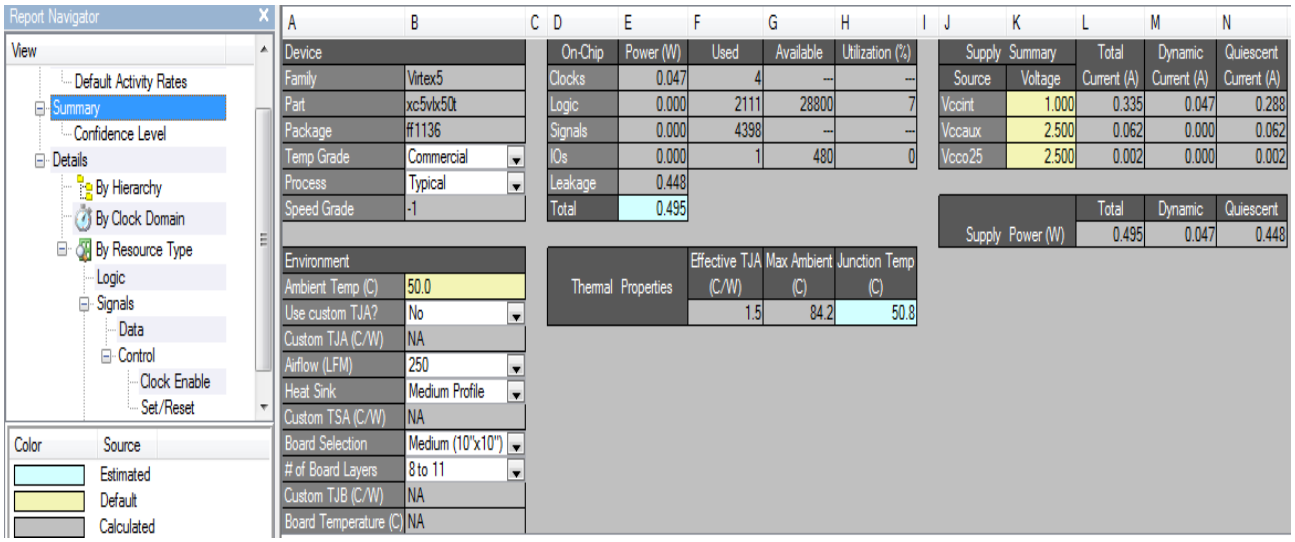

Fig 13: Power Analysis of 512 bit

Table 1 Analysis of hardware required

Operations	Gates 192	Gates 256	Gates 512	Garbage(192)	Garbage(256)	Garbage(512)	Quantum Cost(192)	Quantum Cost(256)	Quantum Cost(512)
Key Expansion	32	32	32	32	32	32	32	32	32
Add Round	192	256	512	196	256	512	192	256	512
Shift Rows	56	72	136	56	72	136	56	72	136
Mixed Column	16	16	16	16	16	16	16	16	16
Total	296	376	696	296	376	696	296	376	696

Table 2 Power and Delay analysis

Key Size	Delay(ns)	Power(watt)
192	34.522	0.524
256	40.145	0.524
512	74.922	0.495

4. CONCLUSION

Attempt is made to increase the security level by using plain text size same as that of key size. And design is implemented using reversible logic gates to address the side channel power attacks in conventional AES. Design is verified using Xilinx tool and implemented on virtex5 board using chipscope. Power and delay is tabulated for 3 key size AES. It is found to be using lesser hardware as it is implemented using only Feynman gate whose quantum cost is 1. Thus the increase in the message length as well as key w.r.t standard AES makes the AES algorithm more stronger against the new attacks and has an acceptable speed of data encryption and decryption.

5. REFERENCES

[1] Srinivas, NS Sai, and Md Akramuddin. "FPGA based hardware implementation of AES Rijndael algorithm for Encryption and Decryption." Electrical, Electronics, and Optimization Techniques (ICEEOT), International Conference on. IEEE, 2016.

[2] Nag, K. Sowmya, H. B. Bhuvaneswari, and A. C. Nuthan. "Implementation of advanced encryption Standard-192 bit using multiple keys." (2013): 2-26.

[3] Jonwal, Sheetal U., and Pratibha P. Shingare. "Advanced Encryption Standard (AES) implementation on FPGA

with hardware in loop." Trends in Electronics and Informatics (ICEI), 2017 International Conference on. IEEE, 2017.

[4] Raj, Gaurav, Ram Charan Kesireddi, and Shruti Gupta. "Enhancement of security mechanism for confidential data using AES-128, 192 and 256bit encryption in cloud." Next Generation Computing Technologies (NGCT), 2015 1st International Conference on. IEEE, 2015.

[5] D'souza, Flevina Jonese, and Dakshata Panchal. "Advanced encryption standard (AES) security enhancement using hybrid approach." Computing, Communication and Automation (ICCCA), 2017 International Conference on. IEEE, 2017.

[6] Saicheur, Vatchara, and Krerk Piromsopa. "An implementation of AES-128 and AES-512 on Apple mobile processor." Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), 2017 14th International Conference on. IEEE, 2017.

[7] Moh'd, Abidalrahman, Yaser Jararweh, and Lo'ai Tawalbeh. "AES-512: 512-bit Advanced Encryption Standard algorithm design and evaluation." Information Assurance and Security (IAS), 2011 7th International Conference on. IEEE, 2011.

[8] Saberi, Iman, Bahareh Shojaie, and Mazleena Salleh. "Enhanced key expansion for AES-256 by using even-odd method." Research and Innovation in Information Systems (ICRIIS), 2011 International Conference on. IEEE, 2011.

[9] H Rohini, Rajashekar,P Kumar: Design of basic sequential circuits using reversible logic. International conference on electrical, electronics, and optimization techniques(ICEEOT),March 2016, pp.2110-2115.

[10] Rohini H, Rajashekar S: Design of Reversible logic based combinational circuits. Communications on Applied Electronics (CAE),Sept 2016, Vol 5, pp. 38-43.

[11] Rohini S H, Jyoti R H, Rajashekar B. S: Reversible Logic Based Modified Design of AES-CBC Mode. Seventh International conference on Advanced electrica Measurement & instrumentation Engineering (EMIE), 13,14 July 2018, pp.171-176.

[12] Rohini S. H, Nikhita M, Pooja A, Rajashekar B. S: Performance Analysis of AES-128bits,192bits & 256bits using reversible logic. Seventh International conference on Advanced electrica Measurement & instrumentation Engineering (EMIE), 13,14 July 2018, pp.165-170.