ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)
Vol 4, Issue 5, May 2017

Implementation of MSEA using 8 bit reversible ALU

[1] Mishal Jasmine Ferrao, [2] Mr. Kiran Kumar. V. G, [3] Mrs. Megha N, [4] Ms. Nisha
[1] M.Tech. in VLSI Design and Embedded Systems, Sahyadri College of Engineering and Management, Adyar, Mangaluru, India ,
[2] Associate Professor, Dept. of Electronics and Communication Engineering, Sahyadri College of Engineering and Management, Adyar, Mangaluru, India ,
[3] Assistant Professor, Dept. of Electronics and Communication Engineering, Sahyadri College of Engineering and Management, Adyar, Mangaluru, India ,
[4] Assistant Professor, Dept. of Electronics and Communication Engineering, Sahyadri College of Engineering and Management, Adyar, Mangaluru, India.

Abstract: **Modified Symmetric Encryption Algorithm (MSEA) is an ARX algorithm. In other words, the encryption procedure includes arithmetic, rotation and XOR operations. MSEA is also a symmetric algorithm. A symmetric algorithm uses the same key for encryption as well as decryption. In this paper, we have a proposed a technique to make MSEA more efficient in terms of power and memory by using an 8 bit reversible arithmetic and logic unit (ALU). Reversible logic is an emerging technology. The fact that there is no loss of data amounts to the advantage of less power dissipation in the reversible logic. The 8 bit ALU to be used in the design will include operations like left rotation, right rotation, increment, XOR operation. These operations are implemented using the basic reversible gates like Feynman gate, Fredkin gate, Peres gate, HNG gate. These gates can also be used to do the basic operations like AND, NAND, OR, NOR gates.**

I. INTRODUCTION

Confidentiality is a very important aspect of secure communication. In an era, where majority of the communication takes place via the Internet, it is essential to maintain the integrity along with the confidentiality of the message. A secure system does not allow the third party to intercept the message even if it does happen; the message will be not be in human readable form. Several techniques are designed with regard to the same. Unpredictability of the encrypted text and the complexity of the algorithm is a measure of how secure the algorithm is. In this paper, a thorough study of the MSEA algorithm is done. Flexibility in the level of security and faster implementation are the key features of MSEA. The data dependent rotations make the algorithm fairly unpredictable. Another key feature is the size of the encrypted data called as the cipher text is always double of the message also called as plain text.

Increasing the complexity of the algorithm means requirement of more power. This is not feasible in devices like mobiles where the resources are constrained. Algorithms having complex operations drain more power from the devices, reducing their life time. Reversible logic is a significant effort towards reduction of power consumption. As there is no loss of data in reversible logic, the power dissipation is almost negligible. In this paper, we have implemented an 8 bit ALU using reversible logic. The ALU uses the basic reversible gates like Feynman gate, Fredkin gate, Peres gate, HNG gate. The implemented ALU can perform 16 logical and arithmetic operations.

In this paper, we have discussed the new version of MSEA which can be implemented using the 8 bit ALU. In Section II,

MSEA algorithm is discussed with the encryption and the key generation process. In Section III, the 8 bit ALU is designed and implemented. Section IV, shows the simulation results of all the modules.

II.ALGORITHM DESCRIPTION

MSEA can use variable plain text but in this paper, the algorithm will be discussed assuming the plain text length to be 128 bits wide. The notability of the algorithm lies in its flexibility for the user. The user can decide the number rounds and the size of the plain text. There is a definite relationship between the sizes of the plain text, size of the cipher text, the number of rounds. Generally assuming that the length of the plain text is s bits, the lengths of the different keys are calculated as follows.
i) Swap Key: The length of the swap key is dependent on the length of the plain text. The swap key length is given as log2 s bits; ii) Master Key: It is used to produce the round keys. The length of the master key is
2s bits; iii) Round key: The key is a unique key produced for each of the rounds. It has the same length as the Master key.

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)
Vol 4, Issue 5, May 2017

Figure 1 shows the encryption procedure of the MSEA algorithm in which the plain text is considered to be of length 128 bits.

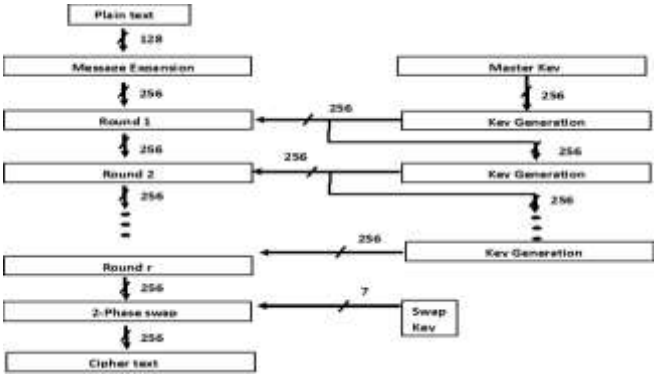

Figure 1: MSEA encryption for 128 bits plain text.

The input plain text of 128 bits is expanded into 256 bits using the message expansion procedure [1]. The expanded message is then passed through the round function where using the round keys, output is generated. For r rounds, r round keys are generated using the round key generation process [1]. The round function of the MSEA for round 1 is described in Figure 2.

In Figure 2, DDR represents data dependent rotations. The 256 bits round key is divided into four equal parts of 64 bits each and each part is named as K1a, K1b, K1c, and K1d. respectively. represents addition while represents XOR operation. The same procedure is repeated for r rounds using the 256 bits round keys. The input is the message block which is the extended version of the plain text and is of 256 bits. The output is 256 bit cipher text.

The swap key which is log2s bits in length is generated using the swap function as described in [1]. Figure 3, shows the cumulative key of MSEA which is shared by the sender and the receiver. The cumulative key consists of four parts. The first part describes the number of round functions and is 6 bits wide. The maximum and the minimum allowable number is 63 and 1 respectively. The second part indicates the size of the plain text whose length ranges from 128 bits to 2048 bits. In case if the plain text chosen has a length lesser than 128 bits then it is padded with zeros to ensure that the result plain text is 128 bits. The remaining two parts are an indication of the length of the swap key and of the master key respectively.

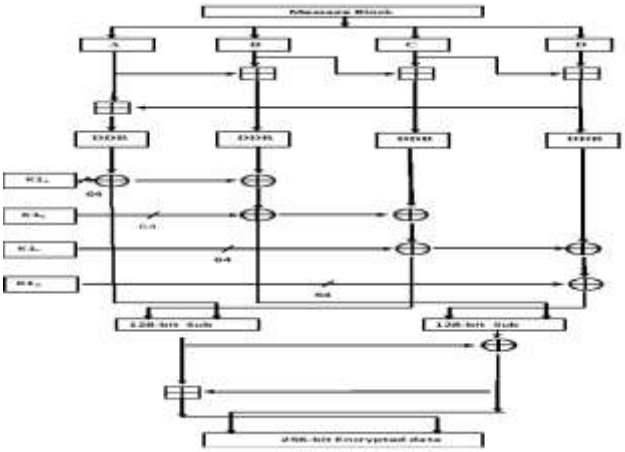

Figure 2: Round Function of the round 1 of MSEA

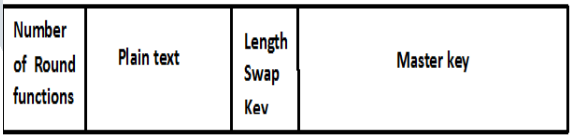

Figure 3: Cumulative key for the MSEA.

III. ALU DESCRIPTION

The ALU implemented uses reversible logic to perform various operations. The list of operations performed by the implemented ALU is listed in Table 1

4-bit Select Input				Operations
S3	S2	S1	S0	
0	0	0	0	Clear
0	0	0	1	Addition
0	0	1	0	Subtraction
0	0	1	1	Multiplication
0	1	0	0	Increment
0	1	0	1	Decrement
0	1	1	0	Left shift
0	1	1	1	Right shift
1	0	0	0	OR
1	0	0	1	AND
1	0	1	0	NOT
1	0	1	1	XOR
1	1	0	0	NOR
1	1	0	1	NAND
1	1	1	0	XNOR
1	1	1	1	Preset

Table 1: Operations performed in the ALU.

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)
Vol 4, Issue 5, May 2017

In this paper, the reversible gates used to implement the ALU are Feynman gate, Fredkin gate, Peres gate, HNG gate and New gate. The symbolic views of these gates are described in [1]. The ALU module includes sub-modules of an adder, multiplier, shifter and a logical unit. The description of each of these modules is given in Figures 4-8.

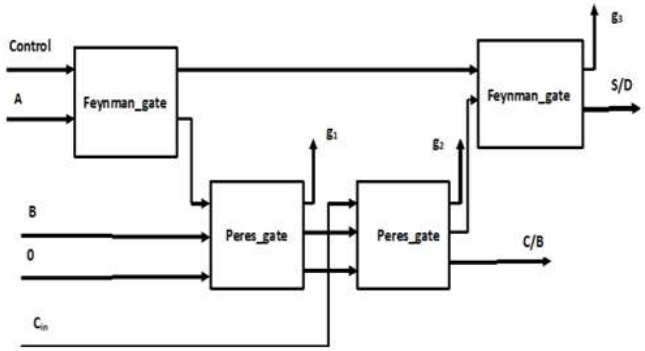

Figure 4: Full adder/subtractor using reversible logic.

Figure 5 is a representation of the ripple adder that has a half adder and full adders cascaded together. The control input determines if the operation to be done is addition or subtraction. If control input is zero, addition is performed else subtraction. To perform addition/subtraction of 8 bits, half adder and a full adder are placed in serial manner. Two inputs each of 8 bits is fed to the adder to produce an 8 bit sum/difference and a carry/borrow.

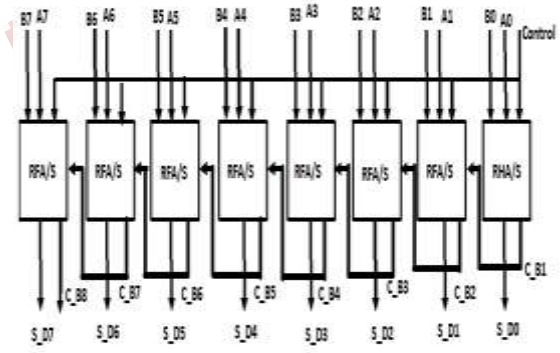

Figure 5: Ripple adder/subtractor using reversible logic.

The multiplier multiplies two 8 bit numbers to produce a 16 bit product. It includes two steps. In the first step, the partial products are generated as shown in [1]. In the second step, the partial products are applied to the Peres and HNG gate to get the final product as shown in Figure 6.

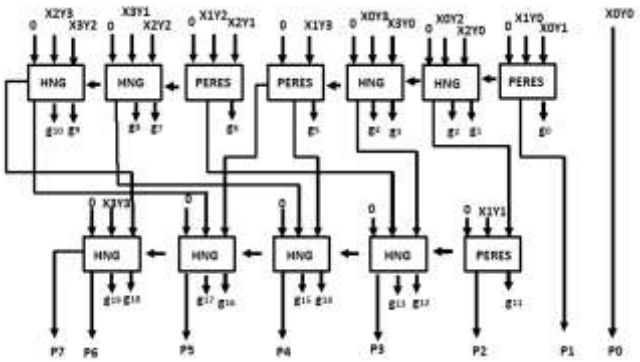

Figure 6: Multiplication using reversible logic.

In Figure 6, X and Y are the 8 bit inputs while P is the 16 bit product. A total of 20 garbage values are generated.

In reversible logic, shifting operation is done using a reversible 2:1 multiplexer that composes of reversible Fredkin gates. The other input, shift which is 3 bits wide in this case, decides the number of shifts. The maximum possible shifts in the designed ALU are 7. Figure 7 describes the right shift operation using 8 bit input, A.

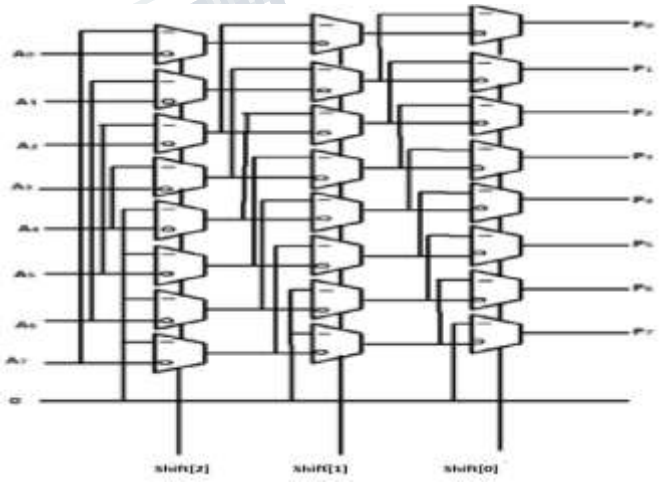

Figure 7: Right shift using reversible logic.

In Figure 7, P is the shifted output. In the first stage, the input is shifted by four bits, in the second stage by two bits and in the last stage by one bit if the values of the shift signal for the

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)
Vol 4, Issue 5, May 2017

stage is one else the outputs of the previous stage as passed to the next stage.

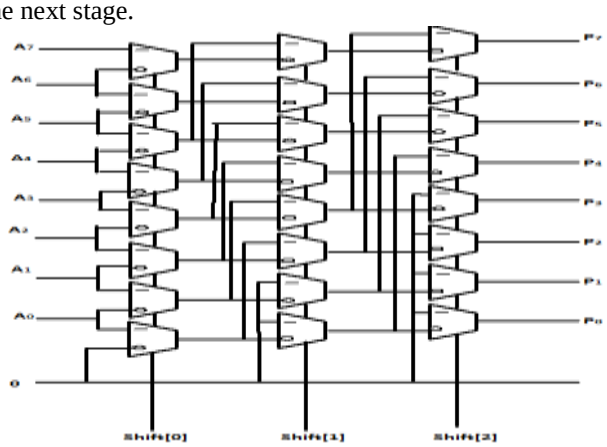

Figure 8: Left shift using reversible logic.

In Figure 8, P is the shifted output. In the first stage, the input is shifted by one bit, in the second stage by two bits and in the last stage by four bits if the value of the shift signal for the stage is zero else the outputs of the previous stage as passed to the next stage.

In this design, the logical operations like AND, OR, NAND, NOR, XOR are done using the reversible gates. The AND and OR gate are implemented using the Fredkin gate, while NAND and NOR are implemented using the New gate. The description of these gates in given in [1].

IV.SIMULATION RESULTS OF ALU

The ALU is implemented using Verilog VDL. The design is synthesized in Xilinx 10.1. Figures 9-18 are the simulation outputs of the arithmetic and logical operations of the ALU. Table 2 lists out the LUT's, IOB's and the total power required for the operations that are performed in the ALU.

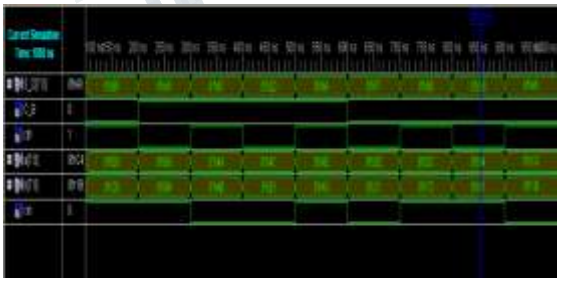

Figure 9: Simulation result of ripple adder.

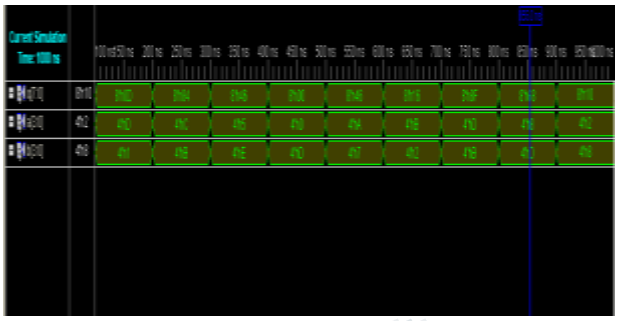

Figure 10: Simulation result of multiplier.

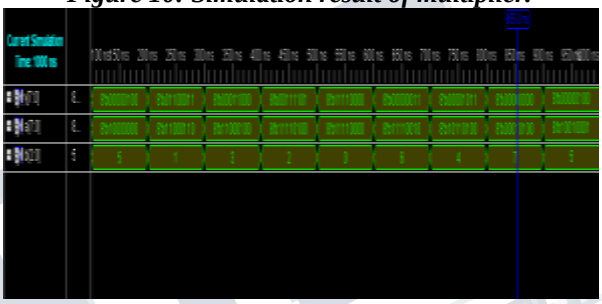

Figure 11: Simulation result of right shift operation.

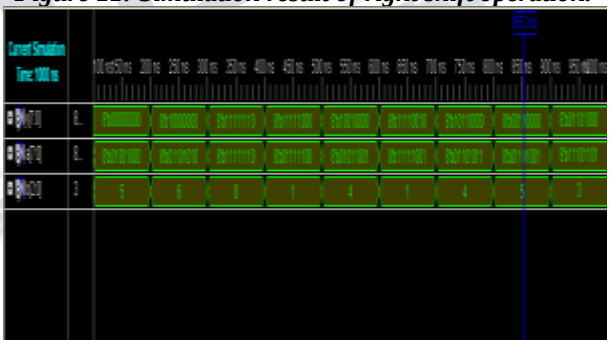

Figure 12: Simulation result of left shift operation.

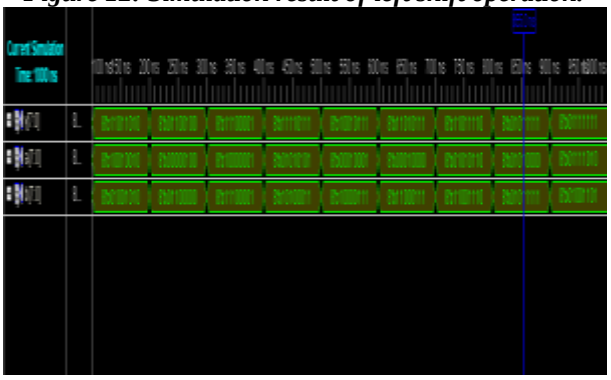

Figure 13: Simulation result of OR operation.

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)
Vol 4, Issue 5, May 2017

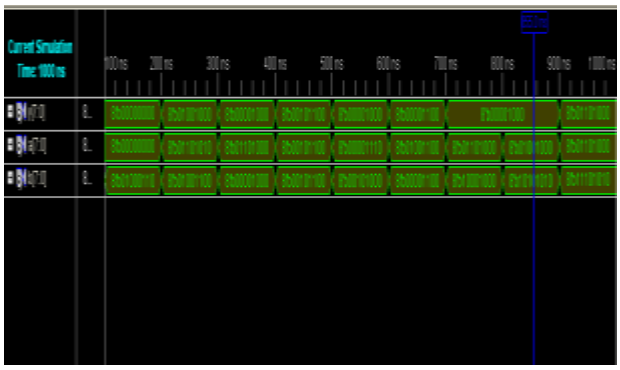

Figure 14: Simulation result of AND operation.

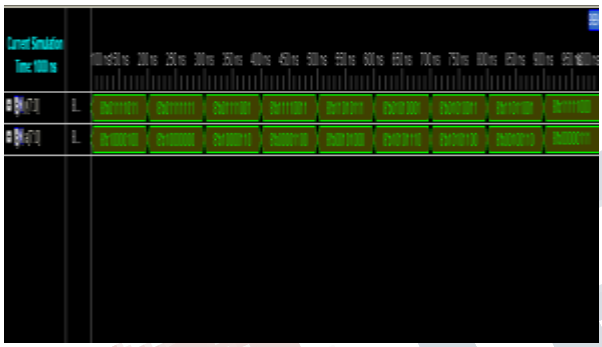

Figure 15: Simulation result of NOT operation.

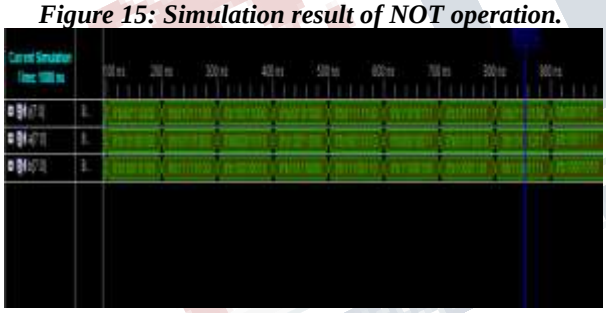

Figure 16: Simulation result of XOR operation.

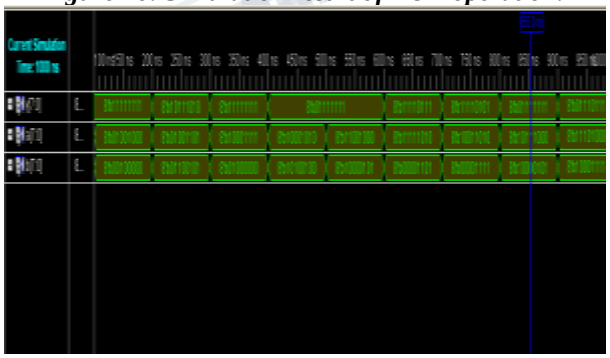

Figure 17: Simulation result of NAND operation.

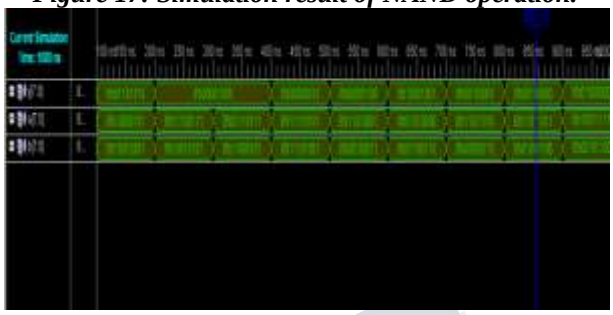

Figure 18: Simulation result of NOR operation.

Logic Utilization	Logic Distribution			Power Analysis				
Number of Input LUT's	Number of Occupied Slices	Number of Slices Containing Only Related Logic	Number of Slices Containing Only Unrelated Logic	Total Number of Input LUT's	Number of Bonded IOB's	Total Quiescent Power	Total Dynamic Power	Total Power
1	1	1	0	1	4	0.056	0.000	0.056
2	1	1	0	2	6	0.056	0.000	0.056
2	1	1	0	2	6	0.056	0.000	0.056
1	1	1	0	1	6	0.056	0.000	0.056
2	1	1	0	2	8	0.056	0.000	0.056
2	1	1	0	2	6	0.056	0.000	0.056
2	1	1	0	2	6	0.056	0.000	0.056
16	12	12	0	16	27	0.056	0.000	0.056
51	27	27	0	51	16	0.056	0.000	0.056
21	11	11	0	21	19	0.056	0.000	0.056
20	10	10	0	20	19	0.056	0.000	0.056
8	8	8	0	8	24	0.056	0.000	0.056
8	8	8	0	8	24	0.056	0.000	0.056
8	8	8	0	8	34	0.056	0.000	0.056
8	8	8	0	8	24	0.056	0.000	0.056
8	8	8	0	8	24	0.056	0.000	0.056
-	-	0	0	-	16	0.056	0.000	0.056

Table 2: Simulation and power details of all modules.

V.CONCLUSION

MSEA being an ARX algorithm is simple in nature. It's key feature being flexibility and not preserving the parity of the input. This prevents any attacks that use parity of the input-output. The data dependent rotations insert lot uncertainties guarding it against any chosen plain text or cipher text attack. According to the results, assigning the number of rounds to 9 achieves strong avalanche effect [1]. For security purpose the number of rounds is taken to be 18 for a 128 bits plain text. Analysis in [1] shows that MSEA is significantly faster than some existing algorithms.

Reversible logic with its advantage in power dissipation makes it beneficial in systems where the resources are

constrained. According to [2], the power dissipation is improved to 39% using reversible logic, hence making it suitable for low power applications. The design can be extended to 16, 32, 64 bits also.

REFERENCES

[1] Rajul Kumar, K.K. Mishra, Ashish Tripathi, Abhinav Tomar, Surendra Singh, "MSEA: Modified Symmetri Encryption Algorithm

[2] Kamaraj Arunachalam, Marichamy Perumalsamy, C. Kalyana Sundaram, and J. Senthil Kumar "Design and Implementation of a Reversible Logic based 8-Bit Arithmetic and Logic Unit" International Journal of Computers and Applications, 2014.

[3] D. Hong, J. K. Lee, D. C. Kim, D. Kwon, K. H. Ryu, and D. G. Lee, "LEA: A 128-Bit Block Cipher for Fast Encryption on Common Processors", WISA 2013, LNCS 8267, Springer International Publishing Switzerland 2014.

[4] Vijay K Panchal, Vimal H Nayak "Analysis of Multiplier Circuit Using Reversible Logic" IJIRST –International Journal for Innovative Research in Science & Technology| 2014.

[5] E. Biham, O. Dunkelman, and N. Keller, "A New Attack on 6-Round IDEA", Proceedings of Fast Software Encryption, Lecture Notes in Computer Science, Springer-Verlag, 2007.

[6] R. Landauer, Irreversibility and heat generation in the computing process, IBM Journal of Research and Development,5(3), 1961, 183–191.