Notice of Violation of IEEE Publication Principles

"High Efficient Modified MixColumns in Advanced Encryption Standard Using Vedic Multiplier"
by M. Senthil Kumar, S. Rajalakshmi
in the Proceedings of the 2nd International Conference on Current Trends in Engineering and Technology (ICCTET), July 2014, pp. 462-466

After careful and considered review of the content and authorship of this paper by a duly constituted expert committee, this paper has been found to be in violation of IEEE's Publication Principles.

This paper duplicates text and figures from the papers cited below. The original content was copied without attribution (including appropriate references to the original author(s) and/or paper titles) and without permission.

Due to the nature of this violation, reasonable effort should be made to remove all past references to this paper, and future references should be made to the following articles:

"Novel Architecture for Inverse Mix Columns for AES Using Ancient Vedic Mathematics on FPGA"
by Sushma R. Huddar , Sudhir Rao Rupanagudi, Ramya Ravi, Shikha Yadav, Sanjay Jain
in the Proceedings of the International Conference on Advances in Computing, Communications and Informatics (ICACCI), August 2013, pp. 1924-1929

"Fault-Tolerant Encryption for Space Applications"
by Roohi Banu, Tanya Vladimirova
in the IEEE Transactions on Aerospace and Electronics Systems, Vol 45, No 1, January 2009, pp. 266-279

High Efficient Modified MixColumns in Advanced Encryption Standard using Vedic Multiplier

M.Senthil Kumar,
Research Scholar,
SCSVMV University,
Kanchipuram.

Dr.S.Rajalakshmi
Professor, Dept of CSE,
SCSVMV University,
Kanchipuram.

Abstract—**This paper is designed for the purpose of commercial security algorithms like the Advanced Encryption Standard (AES). We need to protect the sensitive and valuable data transmitted from satellites to ground. It has increased and hence the need to use encryption on board. AES, which is a very popular choice in world communications, is slowly emerging as the preferred option in the aerospace industry includes satellites. In order to meet this requirement, various algorithms have been designed and implemented in the previous, but each of these algorithms have their own shortcomings with respect to an ASIC or an FPGA implementation. In this paper, we propose high efficient architecture for performing the mix columns operation, which is the main function in the Advanced Encryption Standard (AES) method of cryptography. We perform the same using prehistoric Vedic Mathematics techniques. It gives more efficient results in AES. The AES was designed and implemented on a Xilinx Spartan 3 of FPGA. The novel model is designed using Verilog, from which the area and power are measured.**

Keywords **- AES, mix columns, inverse mix columns, Vedic Multiplication, Verilog**

I. INTRODUCTION

Ever since the evolution of wireless communication systems, encryption of data that is to be transmitted, is of a major concern. This is mainly due to the confidentiality of the data at the sender's end and that the risk of a brute force attack by an eavesdropper of such data could prove fatal not only to individuals but also to national security as well. In order to prevent leakage of such potential data, it is necessary to encrypt the same. Cryptography is one such domain which enables data communication with immense security [4]. It involves encrypting the data to be transmitted or shared by means of unique keys, for encryption and decryption, which are known only to the authorized parties, thereby ensuring data security. This serves as a great boon to common man as well the military regime [2].

Earth observation (EO) satellites take images of the Earth with smart and sophisticated imaging sensors. Multispectral images of the Earth captured by optical on-board cameras can be used in monitoring of the environment and disasters, vegetation control, map marking, urban planning, etc. The latest trend now is towards small EO satellites, as they require smaller budgets to build and launch and also involve less maintenance costs. A typical EO small satellite weighs approximately 100 kg and the orbit average power generated by solar panels is 30 to 60 W. The imaging payload units of such satellites comprise imagers, mass memory, and high-rate data transceivers and consume up to 70% of the average orbit power. At present, more and more EO satellites are equipped with on-board encryption to protect the data transmitted to the ground station. However due to confidentiality and security reasons the coverage of this topic in the open literature is very limited. Encryption, by far the most widely adopted security service in terrestrial [1].

Networks, is used to protect data from unauthorized users. Although there are many encryption products and algorithms, the use of these products and algorithms on-board satellites has been overlooked until recently. But now satellite manufacturers are realizing the importance of on-board encryption to protect valuable data, especially after cases, which have proved that intrusion into satellite data is not an impossible task. At present, more and more EO satellites are equipped with on-board encryption to protect the data transmitted to the ground station. Ever since DES was phased out in 2001 and its successor, the Advanced Encryption Standard (also known as Rijndael) took its place, various AES implementations have been proposed both in software and hardware.

The Rijndael algorithm approved as the Advanced Encryption Standard (AES) by the US National Institute of Standards and Technology (NIST) is a block cipher, which encrypts one block of data at a time. To encrypt multiple blocks, modes of operation have been defined by NIST.

AES is being adopted by many organizations across the world. Because of its simplicity, flexibility, easiness of implementation, and high throughput AES is used in many different applications ranging from smart cards to big servers. In fact, hardware implementations of AES are well suited to resource-constrained embedded applications like satellites. There are various hardware implementations of the AES algorithm on platforms like application specific integrated circuits (ASICs) and field programmable gate arrays (FPGAs) that achieve a significant throughput ranging from a few Mbit/s to Gbit/s.Thus the requirements of small EO satellites for high-rate data transmission are met by existing AES implementations. However, in addition to high throughput,

IEEE Conference Number - 33344
July 8, 2014, Coimbatore, India.

immunity of the encryption process against faults is very important in satellites.

In this paper, we introduce a non-laborious, area efficient and computationally less complex methodology to implement the transformations, with the aid of ancient Vedic mathematics. Vedic Mathematics is an archaic style of mathematics which subsisted in India in 1500 B.C., and was later on brought tolimelight by a famous scholar Sri Bharthi Krishna Tirthaji between 1911 and 1918. He systemized it into 16 simple sutras, which are used by most of the researchers and mathematicians due to its ease of use. Out of the 16 formulae available in Vedic Mathematics, the UrdhwaTiryakbhyam Sutra was utilized in order to address the flaws observed in the conventional mix columns architecture utilized in AES [7]. Section IV elaborates the UrdhwaTiryakbhyam Sutra and also deals with the novel implementation of the mix columns transformation using this rule. Section V gives an exhaustive comparison of the novel algorithm with the conventional approaches.

II. ADVANCED ENCRYPTION STANDARD– ALGORITHM AND MODES OF OPERATION

The AES is a symmetric key algorithm, in which both the sender and the receiver use a single key for encryption and decryption. AES defines the data block length to 128 bits, and the key lengths to 128,192, or 256 bits. It is an iterative algorithm and each iteration is called a round. The total number of rounds, Nr, is 10, 12, or 14 when the key length is 128, 192, or 256 bits, respectively [9]. Each round in AES, except the final round, consists of four transformations: Sub Bytes, ShiftRows, MixColumns, and AddRoundKey [3]. The final round does not have the MixColumns transformation the decryption flow is simply the reverse of the encryption flow and each operation is the inverse of the corresponding one in the encryption process. The round transformation of AES and its steps operate on some intermediate results, called state. State can be visualized as a rectangular matrix with four rows. The number of columns in the state is denoted by Nb and is equal to the block length in bits divided by 32. For a 128 bit data block (16bytes) the value of Nb is 4, hence the state is treated as a 4×4 matrix and each element in the matrix represents a byte. For the sake of simplicity, in the rest of the paper, both the data block and the key lengths are considered as 128 bit long. However all the discussions and the results hold true for 192 bit and 256 bit keys as well.

A. Transformations Of AES Algorithm

All the four transformations of the AES round work on the principles of finite fields. The number of the elements in a finite field is called the order of the field. A finite field of order p^n is generally denoted as GF (P^n), where GF stands for Galois field, p is the characteristic of the finite field, and n is the number of bits used to represent the field elements. The basic operations of AES are defined over the elements of the field GF (2^8). The specification of AES has adopted polynomial representation of the byte elements of GF (2^8) with coefficients over the field GF (2). A polynomial representation of byte a(x) with coefficients over the field GF (2) is represented as follows:

$$a(x) = \sum a_i x^i$$
$$= a_7x^7 + a_6x^6 + a_5x^5 + a_4x^4 + a_3x^3 + a_2x^2 + a_1x^1 + a_0$$

Where $a_i = \{0, 1\}$.

The binary representation of the polynomial given by (1) is as follows:

$$a(x) \rightarrow a_7a_6a_5a_4a_3a_2a_1a_0$$

For example, a byte element represented in binaryform as 1100 0001 is written in polynomial form as

$$x^7 + x^6 + 1.$$

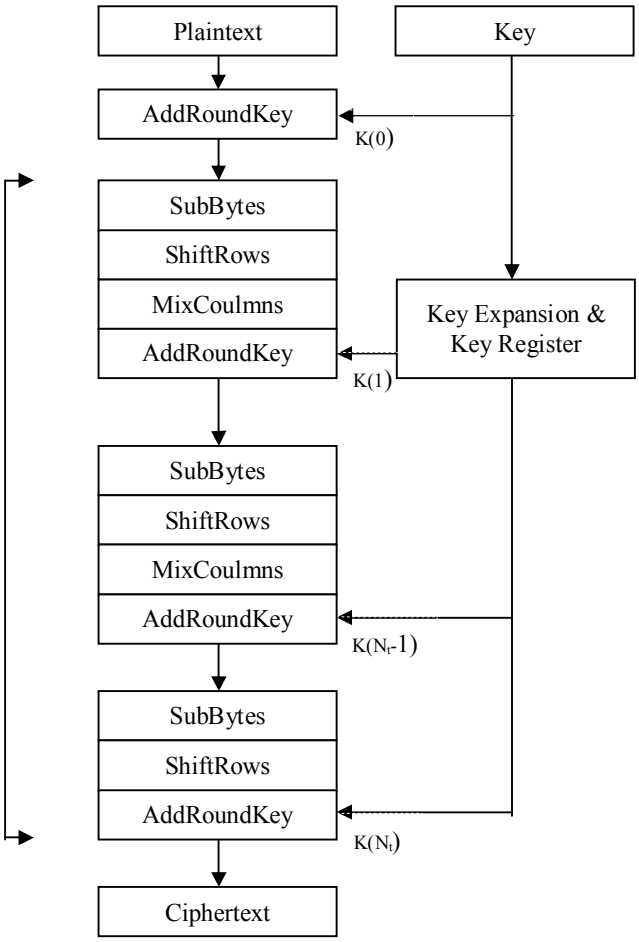

Figure 1. Block Diagram for AES Algorithm

The operation of addition using byte elements is defined as addition of the corresponding polynomials. In case of polynomials over GF (2) addition is just an exclusive OR (XOR) operation. For byte multiplication the following irreducible polynomialis used in AES:

$$m(x) = x^8 + x^4 + x^3 + x + 1$$

Multiplication in GF (2^8) is denoted as $\otimes$. The multiplication of two polynomials a(x) and b(x) is defined as the algebraic product of the polynomials modulo the irreducible polynomial m(x) as below:

$$c(x) = a(x) - b(x) = a(x)b(x) \bmod m(x)$$

SubBytes is a nonlinear transformation in which one byte is substituted for another by means of the affine transformation over the Galois Field GF(2)

$$\begin{bmatrix} b'_0 \\ b'_1 \\ b'_2 \\ b'_3 \\ b'_4 \\ b'_5 \\ b'_6 \\ b'_7 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} - \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}.$$

ShiftRows is a shift operation performed on the last three rows of the state. The last three rows are rotated to the left by 1, 2, or 3 bytes, as seen inFig. 1. MixColumns is finite field matrix multiplication applied every round except the last. Each column is multiplied as a four-term polynomial in GF (28) mod (x4+ 1) using the array

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix} \quad \text{for } 0 \le c < N_b$$

AddRoundKey performs a bitwise XOR operation with the current state and the expanded round key every round including an initial round and the last round. The round key is read from round 0 to (Nr – 1) for encryption and vice versa for decryption. The decryption process is similar to the encryption process, simply executing the inverse of each function. Inverse SubBytes involves taking an inverse affine transformation. Inverse ShiftRows rotates the bytes to the right by: 3, 2, or 1 byte(s). Inverse MixColumnsuses the same operations as MixColumns but usesthe inverse matrix.

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 0e & 0b & 0d & 09 \\ 09 & 0e & 0b & 0d \\ 0d & 09 & 0e & 0b \\ 0b & 0d & 09 & 0e \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix}.$$

Because of the inherent symmetry of addition modulo2 (XOR), AddRoundKey is the same for both encryption and decryption processes [8].

III. PROPOSED MIX COLUMNS CALCULATIONS USING VEDIC MATHEMATICS

A. *Vedic Mathematics for binary numbers*

One of the crucial mathematical operation performed during the mix column step in AES, is the Galois field multiplication [5] and [6]. Multiplication, being a tedious and a power hungry operation, causes the computation of mix columns and its inverse to be an even more arduous task. This is due to the fact that, it involves matrix multiplication. Therefore, there arises a necessity to ease the entire process and also overcome a few of the shortcomings of the previous methods mentioned in the previous section. In order to achieve the same, the UrdhwaTiryakbhyam Sutra of Vedic Mathematics, is incorporated in our proposed architecture for mix columns and its inverse due to its excellence in terms of speed and area. The UrdhwaTiryakbhyam Sutra is one of the significant sutras in ancient Vedic Mathematics. By its definition, UrdhwaTiryakbhyam means "vertically crosswise". This implies that multiplication occurs between extreme bits of the multiplier and multiplicand. The major advantage of this algorithm is the availability of the product of two numbers in a single step. Also, since multiplication of two single bits reduces to a single AND operation, for a VLSI implementation this approach proves to be both area and speed efficient.

$$P_0 = M_0 * N_0 \quad (1)$$

$$C_1P_1 = (M_1 * N_0) + (M_0 * N_1) \quad (2)$$

$$C_3C_2P_2 = (M_2 * N_0) + (M_0 * N_2) + (M_1 * N_2) + C_1 \quad (3)$$

$$C_5C_4P_3 = (M_3 * N_0) + (M_2 * N_1) + (M_1 * N_2) + (M_0 * N_3) + C_1 \quad (4)$$

$$C_7C_6P_4 = (M_4 * N_0) + (M_3 * N_1) + (M_2 * N_2) + (M_1 * N_3) + (M_0 * N_4) + C_3 + C_4 \quad (5)$$

$$C_{10}C_9C_8P_5 = (M_5 * N_0) + (M_4 * N_1) + (M_3 * N_2) + (M_2 * N_3) + (M_1 * N_4) + (M_0 * N_5) + C_5 + C_6 \quad (6)$$

$$C_{13}\,C_{12}C_{11}P_6 = (M_6 * N_0) + (M_5 * N_1) + (M_4 * N_2) + (M_3 * N_3) + (M_2 * N_4) + (M_1 * N_5) + (M_0 * N_6) + C_{7+}\,C_6 \quad (7)$$

$$C_{16}\,C_{15}C_{14}P_7 = (M_7 * N_0) + (M_6 * N_1) + (M_5 * N_2) + (M_4 * N_3) + (M_3 * N_4) + (M_2 * N_5) + (M_1 * N_6) + (M_0 * N_7) + C_{9+}\,C_{11} \quad (8)$$

$$C_{16}\,C_{15}C_{14}P_7 = (M_7 * N_1) + (M_6 * N_2) + (M_5 * N_3) + (M_4 * N_4) + (M_3 * N_5) + (M_2 * N_6) + (M_1 * N_7) + C_{10+}C_{12+}\,C_{14} \quad (9)$$

$$C_{22}C_{21}C_{20}P_9=(M_7*N_2)+(M_6*N_3)+(M_5*N_4)+(M_4*N_5)+(M_3*N_6)+(M_2*N_7)+C_{13}+C_{15}+C_{17} \quad (10)$$

$$C_{25}C_{24}C_{23}P_{10}=(M_7*N_3)+(M_6*N_4)+(M_5*N_5)+(M_4*N_6)+(M_3*N_7)+C_{16}+C_{18}+C_{20} \quad (11)$$

$$C_{27}C_{26}P_{11}=(M_7*N_4)+(M_6*N_5)+(M_5*N_6)+(M_4*N_7)+C_{19}+C_{21}+C_{23} \quad (12)$$

$$C_{29}C_{28}P_{12}=(M_7*N_5)+(M_5*N_6)+(M_5*N_7)+C_{22}+C_{24}+C_{26} \quad (13)$$

$$C_{30}P_{13}=(M_7*N_6)+(M_6*N_7)+C_{25}+C_{27}+C_{28} \quad (14)$$

$$P_{14}=(M_7*N_7)+C_{29}+C_{30} \quad (15)$$

$$P_{15}=(M_7*B_7) \quad (16)$$

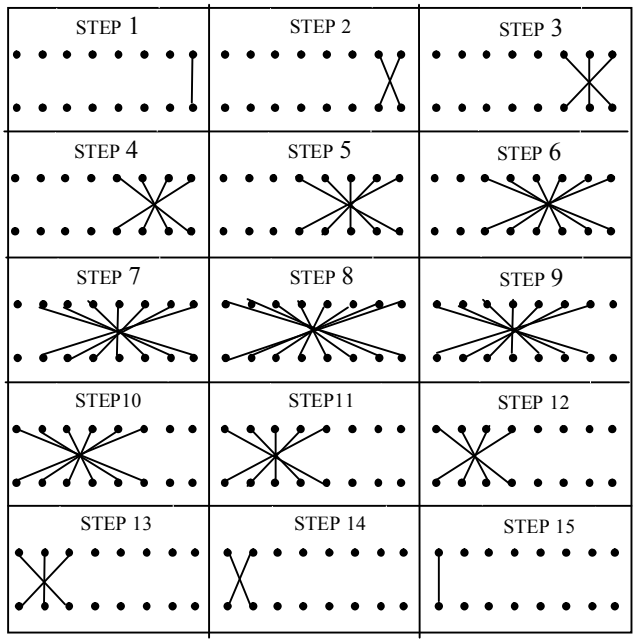

Figure 2. Depiction of UrdhwaTiryakbhyam Sutra for multiplication of 8 bit numbers with the aid of dot diagrams.

B. *Further simplification of the Urdhwa method for Galois field multiplication*

As mentioned above, the UrdhwaTiryakbhyam sutra is a very efficient algorithm for multiplication and hence can be used in the Mix columns/Inverse Mix columns architecture as well. However, since we are required to perform Galois Field multiplication and not the regular multiplication, our proposed architecture slightly deviates from the UrdhwaTiryakbhyam architecture. In our implementation, instead of adding the partial products which were computed in the intermediate stages, we logically XOR the same. Equations (1) to (16) are therefore modified and represented by (17) to (31), based on this architecture.

$$P_0 = M_0 * N_0 \quad (17)$$

$$P_1 = (M_1 * N_0) \oplus (M_0 * N_1) \quad (18)$$

$$P_2 = (M_2 * N_0) \oplus (M_0 * N_2) \oplus (A_1 * B_1) \quad (19)$$

$$P_3 = (M_3 * N_0) \oplus (M_2 * N_1) \oplus (M_1 * N_2) \oplus (M_0 * N_3) \quad (20)$$

$$P_4 = (M_4 * N_0) \oplus (M_3 * N_1) \oplus (M_2 * N_2) \oplus (M_1 * N_3) \oplus (M_0 * N_4) \quad (21)$$

$$P_5 = (M_5 * N_0) \oplus (M_4 * N_1) \oplus (M_3 * N_2) \oplus (M_2 * N_3) \oplus (M_1 * N_4) \oplus (M_0 * N_5) \quad (22)$$

$$P_6=(M_6*N_0)\oplus(M_5*N_1)\oplus(M_4*N_2)\oplus(M_3*N_3)\oplus(M_2*N_4)\oplus(M_1*N_5)\oplus(M_0*N_6) \quad (23)$$

$$P_7=(M_7*N_0)\oplus(M_6*N_1)\oplus(M_5*N_2)\oplus(M_4*N_3)\oplus(M_2*N_5)\oplus(M_1*N_6)\oplus(M_0*N_7) \quad (24)$$

$$P_8=(M_7*N_1)\oplus(M_6*N_2)\oplus(M_5*N_3)\oplus(M_4*N_4)\oplus(M_3*N_5)\oplus(M_2*N_6)\oplus(M_1*N_7) \quad (25)$$

$$P_9=(M_7*N_2)\oplus(M_6*N_3)\oplus(M_5*N_4)\oplus(M_4*N_5)\oplus(M_3*N_6)\oplus(M_2*N_7) \quad (26)$$

$$P_{10}=(M_7*N_3)\oplus(M_6*N_4)\oplus(M_5*N_5)\oplus(M_4*N_6)\oplus(M_3*N_7) \quad (27)$$

$$P_{11} = (M_7 * N_4) \oplus (M_6 * N_5) \oplus (M_5 * N_6) \oplus (M_4 * N_7) \quad (28)$$

$$P_{12} = (M_7 * N_5) \oplus (M_5 * N_6) \oplus (M_5 * N_7) \quad (29)$$

$P_{13} = (M_7 * N_6) \oplus (M_6 * N_7)$ (30)

$P_{14} = (M_7 * N_7)$ (31)

IV. CONCLUSION

It can be clearly noted from Table I, that our proposed architecture which incorporates the Urdhava multiplier occupies an area 1.09 % lesser than the splitting approach for GF(28) multiplication. Encryption is analyzed.

Table.1 Comparison of regular and modified vedic multiplier for multilevel 2d-dwt.

Type	LUT	Slices	Delay(ns)
Conventional Vedic multiplier	767	430	48.637
Modified Vedic multiplier	723	404	47.544

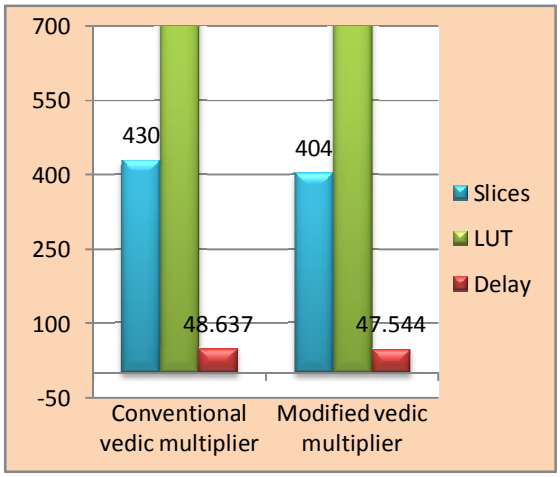

Figure 3. Performance Analysis between Existing and Proposed Vedic Multiplications.

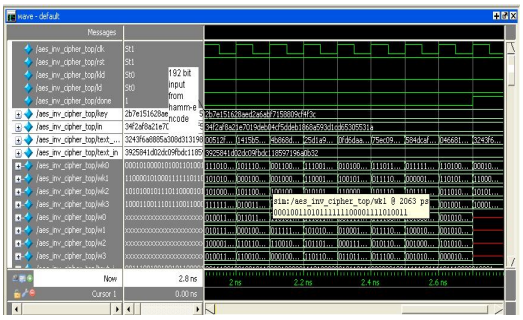

Figure 4. AES encryption using ModelSim.

Also, the proposed architecture proves to be better than the Look-up table approach hands-down, since the area occupied by the Look-up table approach is way beyond the bounds of a Xilinx Spartan 3e series of FPGA. In terms of timing, a comparison in time with the LUT approach is redundant, considering its large area occupancy. In terms of logic gates, our approach consumes a mere 46 XOR gates only.

REFERENCES

[1] Sun, W., Stephens, P., and Sweeting, M. N.Micro-minisatellites for affordable EO constellations–Rapid Eye and DMC. In Proceedings of the IAA Symposium on Small Satellites for Earth Observation, Berlin, Germany, Apr. 2001, IAA-B3-0603.

[2] Surrey Satellite Technology Ltd.www.sstl.co.uk (last accessed on 18th June 2007).

[3] Directory of Earth Observation Resources.http://directory.eoportal.org/pres TopSat.html (last accessed 18th June 2007).

[4] Consultative Committee for Space Data Systems Security threats against space missions. Informational Report CCSDS 350.1-G-1, Green Book, NASA, Washington, D.C., Oct. 2006.

[5] Sweet, K. The increasing threat to satellite communications. Online Journal of Space Communication, 6 (Nov. 2003).

[6] Mariani, R., and Boschi, G.Scrubbing and partitioning for protection of memory systems. In Proceedings of the 11th IEEE International Symposium on On-Line Testing, July 6—8, 2005, 195—196.

[7] Iyer, Nalini C., P. V. Anandmohan, and D. V. Poornaiah."Mix/InvMixColumn decomposition and resource sharing in AES." International Conference on Industrial and Information Systems (ICIIS), on. IEEE, 2010.

[8] Berent, Adam. "Advanced Encryption Standard by Example". Document available at URL http://www.networkdls.com/Articles/AESbyExample.pdf (Apr□l 2007) Accessed: June 2013.

[9] Li, Hua, and Zac Friggstad. "An efficient architecture for the AES MixColumns operation". Circuits and Systems, 2005. ISCAS 2005. IEEE International Symposium on. IEEE, 2005.