Implementation of Triple AES Encryption & Decryption Design for High Security Application

[1] Harsha A [2] Santhosh Kumar G
[1] M.Tech in LDE, [2] Assistance professor Dept of ECE
[1][2] East West Institute of Technology Bengaluru, India
[1]hash3192@gmail.com [2] skg2185@gmail.com

Abstract: **This paper presents a combinational logic based Rijndael S-Box implementation for the Sub Byte transformation in the Advanced Encryption Standard (AES) algorithm. Recent publications on AES implementation have shown that the combinational logic based S-Box is proven for its small area occupancy and high throughput, given the fact that pipelining can be applied to this S-Box implementation as compared to the typical based lookup table implementation which access time is fixed and unbreakable. In this paper, the construction procedure for implementing a 2 stage pipeline combinational logic based S-Box is presented and illustrated in a step-by-step manner. Implementation of shift row, mixed column, ad round key with DNA keys are done.Finally, for the purpose of practicality, the depth of the mathematics involved has been reduced in order to allow the reader to better understand the internal operations within the S-Box. A worked example by hand is also provided to help the reader better understand the functionality of the internal operations.**

I. INTRODUCTION

The National Institute of Standards and Technology (NIST) has been working with the international cryptographic community to develop an Advanced Encryption Standard (AES). The overall goal is to develop a Federal Information Processing Standard (FIPS) that specifies an encryption algorithm capable of protecting sensitive government information well into the twenty-first century. NIST expects that the algorithm will be used by the U.S. Government and, on a voluntary basis, by the private sector. The competition among the finalists was very intense, and NIST selected Rijndael as the proposed AES algorithm at the end of a very long and complex evaluation process. This report describes that process and summarizes many of the characteristics of the algorithms that were identified during the public evaluation periods. The following sections provide an overview of the AES development followed by a discussion of specific analysis details

On August 20, 1998, NIST announced fifteen AES candidate algorithms at the First AES Candidate Conference (AES1) and solicited public comments on the candidates. Industry and academia sub mitters from twelve countries proposed the fifteen algorithms. A Second AES Candidate Conference (AES2) was held in March 1999 to discuss the results of the analysis that was conducted by the international cryptographic community on the candidate algorithms. In August 1999, NIST announced its selection of five finalist algorithms from the fifteen candidates. The selected algorithms were MARS, RC6TM, Rijndael, Serpent and Two fish.

The main contributions of this paper can be summarized as follows. This paper avoids use of LUTs and proposes use of composite field data path for the Sub Bytes and Inv Sub Bytes transformations. The use of such data paths is the key for the design of high-speed sub pipelined AES architectures. Composite field arithmetic has been employed in to design efficient data paths. However, the design in decomposes the inversion in in the Sub Bytes and Inv Sub Bytes to . Instead, the proposed architecture in this paper decomposes the inversion to , and the inversion in is implemented by a novel approach, which leads to a more efficient architecture with shorter critical path and smaller area Nevertheless, it is not efficient to use composite field arithmetic in all the transformations of the AES algorithm as done. Another main contribution of this paper is the key expansion architecture. This paper, for the first time, presents a key expansion architecture which is well suited for sub pipelined designs. In addition, this architecture can operate in an on-the-fly manner. Using the proposed data path and key expansion architecture, post-placement timing report shows a fully sub pipelined encryptor of 128-bit key with 7 sub stages in each round unit can operate at a throughput of 21.56 Gbps on a Xilinx XCV 1000 e-8bg 560 device in nonfeedback. modes. Architectures utilizing multiple sub stage sub pipelining have been published recently. However, their designs are less efficient. The architecture presented in this paper can achieve higher speed than the prior fastest FPGA implementation , and is 79% more efficient in terms of equivalent throughput slice

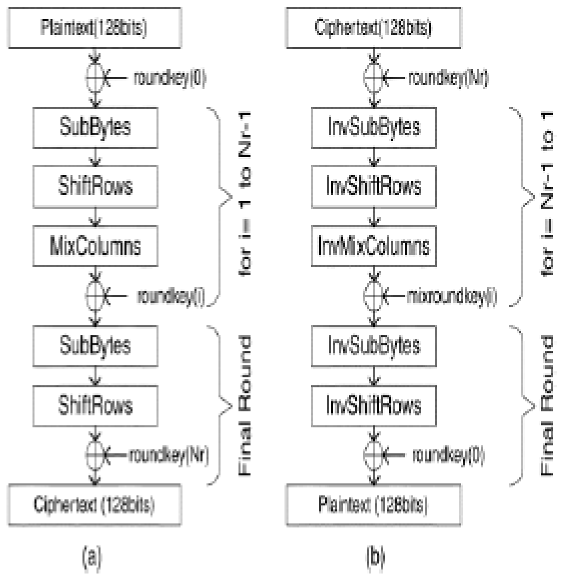


Fig.1.TheAESalgorithm.(a)Encryption structure.(b)Equivalent decryption structure.

II. STEPS INVOLVED IN AES

The encryption process uses a set of specially derived keys called round keys. These are applied, along with other operations, on an array of data that holds exactly one block of data? The data to be encrypted. This array we call the state array. The following AES steps of encryption for a 128-bit block:

1. Derive the set of round keys from the cipher key.

2. Initialize the state array with the block data (plaintext).

3. Add the initial round key to the starting state array.

4. Perform nine rounds of state manipulation.

5. Perform the tenth and final round of state manipulation.

6. Copy the final state array out as the encrypted data (ciphertext).

The reason that the rounds have been listed as "nine followed by a final tenth round" is because the tenth round involves a slightly different manipulation from the others.
The block to be encrypted is just a sequence of 128 bits. AES works with byte quantities so we first convert the 128 bits into 16 bytes. We say "convert," but, in reality, it is almost certainly stored this way already. Operations in RSN/AES are performed on a two-dimensional byte array of four rows and four columns. At the start of the encryption, the 16 bytes of data, numbered D0 ? D15, are loaded into the array.
Each round of the encryption process requires a series of steps to alter the state array.

III SYSTEM IMPLEMENTATION

AES Encryption

The AES algorithm is used for encrypting and decrypting the sequence. Hence the security of the overall cryptographic process is increased. The Advanced Encryption Standard (AES) is a specification for the encryption of electronic data established by U.S National Institute of Standards and Technology (NIST) in 2001. This standard was accepted and used by U.S government and now it finds its use worldwide. Fig 2 discusses the flow diagram.

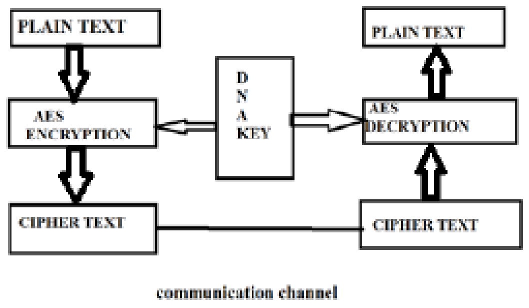
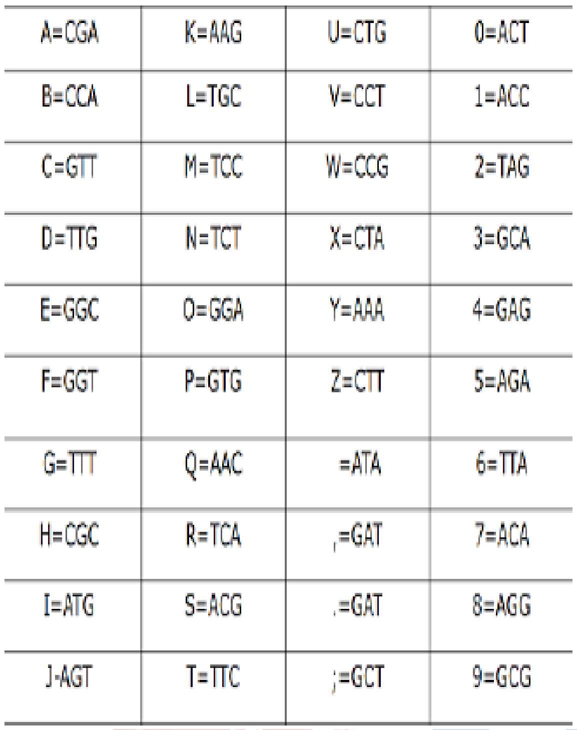


Fig. 2 flow diagram.

One of the most essential components required for the functioning of all living organisms is DNA. DNA stands for Deoxyribonucleic acid and it has many properties like vast parallelism, exceptional energy storage capability. There are four classes of nucleotides, Adenine, Guanine, Cytosine, Thymine (A,C,G,T). These nucleotides are strung into polymer chains (DNA strands). DNA is basically used to store genetic information. This information cannot be duplicated or copied. Fig 2 shows the basic structure of a DNA molecule. The concept of biotechnology, where DNA strands are used as a carrier for conveying message from the sender to receiver are used in DNA cryptography.
The existing cryptographic schemes like RSA and DES is prone to many attacks in the near future and has been broken. This is one of the main reason for using DNA concept. This concept is still in the early stages and further developments are going on in this area. One end orientation of DNA is 5' and the other end is 3'.

DNA which usually exists in the form of double helix structure and these structures are formed with the help of hydrogen bonds between them. This structure is called as Watson – Crick. An amplification method which is used to amplify DNA strands which are generally very small in size is called PC Amplification. PCR stands for Polymer Chain Reaction and it is a fast DNA amplification technology based on Watson-Crick complementarily. This being the most fragile method of amplification, after just 20 cycles the DNA molecule can beamplified to 106.



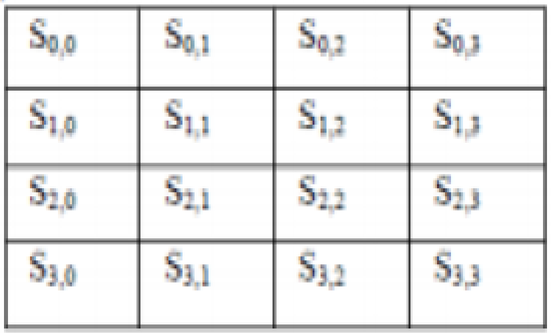
A=CGA	K=AAG	U=CTG	0=ACT
B=CCA	L=TGC	V=CCT	1=ACC
C=GTT	M=TCC	W=CCG	2=TAG
D=TTG	N=TCT	X=CTA	3=GCA
E=GGC	O=GGA	Y=AAA	4=GAG
F=GGT	P=GTG	Z=CTT	5=AGA
G=TTT	Q=AAC	=ATA	6=TTA
H=CGC	R=TCA	,=GAT	7=ACA
I=ATG	S=ACG	.=GAT	8=AGG
J-AGT	T=TTC	;=GCT	9=GCG

TABLE 1. DNA code set

IV AES ALGORITHM OVERVIEW

[1] The AES Algorithm

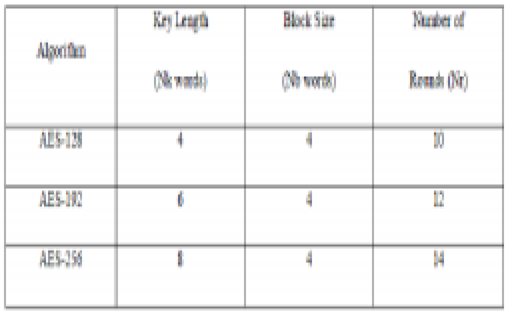
The AES accepts a 128-bit plain text, and produces a 128-bit cipher text under the control of a 128 ,192, or 256-bit secret key. It is a Sub stitution-Permutation Network design with a single collection of steps called a round For the AES algorithm, the number of rounds to be performed during the execution of the algorithm is dependent on the key length VLSI Based Implementation Of Single Round



$S_{0,0}$	$S_{0,1}$	$S_{0,2}$	$S_{0,3}$
$S_{1,0}$	$S_{1,1}$	$S_{1,2}$	$S_{1,3}$
$S_{2,0}$	$S_{2,1}$	$S_{2,2}$	$S_{2,3}$
$S_{3,0}$	$S_{3,1}$	$S_{3,2}$	$S_{3,3}$

Figure 3 Block- Round Combinations

That are repeated 10, 12, or 14 times (depending on the key length) to map the plain text to cipher text each round uses its own 128-bit round key, which is derived from the supplied secret key through a process known as a key schedule. It distributes the entropy of the key across each of the round keys. If that entropy is not spread properly, it causes all kinds of trouble such as equivalent keys, related keys, and other similar distinguishing attacks. AES treats the 128-bit input as a vector of 16 bytes organized in a column major (big endian) 4 x 4 matrix called the state. That is, the first byte maps to S0,0, the third byte to S2,0the fourth byte to S3,0, and the 16th byte maps to S3,3shown in Fig(4)



Algorithm	Key Length (Nk words)	Block Size (Nb words)	Number of Rounds (Nr)
AES-128	4	4	10
AES-192	6	4	12
AES-256	8	4	14

Figure 4:State diagram single round of AES consists of four Transformations namely Sub Bytes, Shift Rows, Mix Columns and Add Round Key shown in Fig (5).

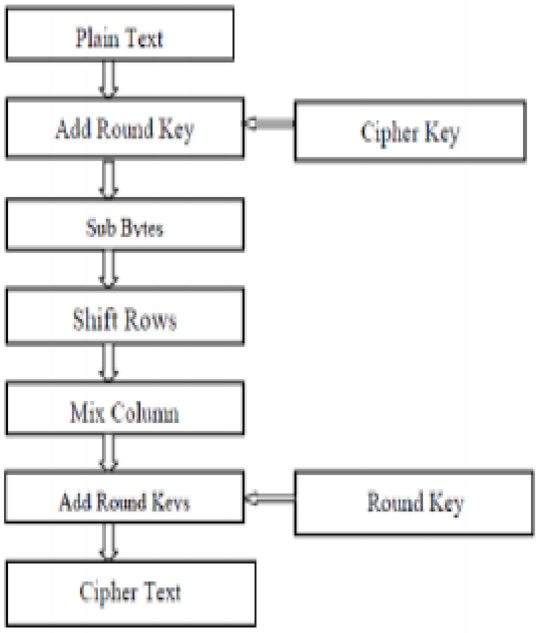


Figure 5: Single Round AES Algorithm

[2] Add Round Key

This step of the round function adds in GF(28). the round key to the state. It performs 16 parallel additions of key material to state material. The addition is performed with the XOR operation Fig (6).

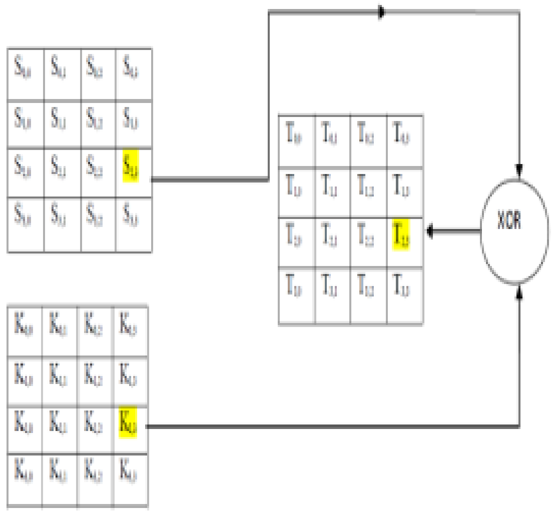


Figure 6: Add Round Key Function

The k matrix is a round key and there is a unique key for each round. Since the key addition is a simple XOR, it is often implemented as a 32-bit XOR across rows in 32-bit keys.

[3] Sub Bytes

It maps each of the 16 bytes in parallel to a new byte by performing two-step sub stitutions Fig (4). The sub stitution is composed of a multiplicative inversion in GF(28) followed by an affine transformation (Fig 7)in GF(28)[9].The multiplicative inverse of a unit S is another unit T, such that ST modulo the AES polynomial is congruent .

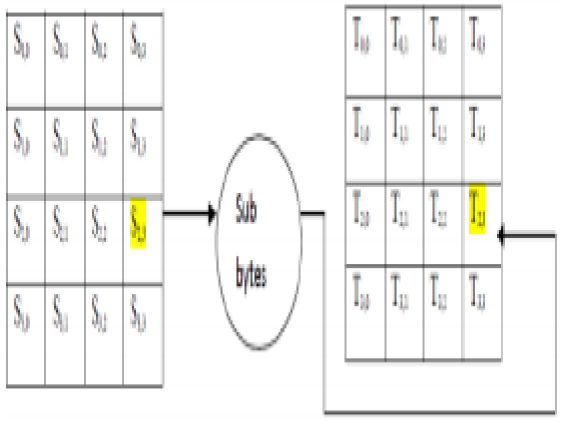


Figure 7: Sub Bytes Function

$$\begin{pmatrix} 1&0&0&0&1&1&1&1\\ 1&1&0&0&0&1&1&1\\ 1&1&1&0&0&0&1&1\\ 1&1&1&1&0&0&0&1\\ 1&1&1&1&1&0&0&0\\ 0&1&1&1&1&1&0&0\\ 0&0&1&1&1&1&1&0\\ 0&0&0&1&1&1&1&1 \end{pmatrix} \begin{pmatrix} x_0\\ x_1\\ x_2\\ x_3\\ x_4\\ x_5\\ x_6\\ x_7 \end{pmatrix} + \begin{pmatrix} 1\\ 1\\ 0\\ 0\\ 0\\ 1\\ 1\\ 0 \end{pmatrix}$$

Figure 8 Affine Transformation

[4] Shift Rows

The Shift Rows operation only changes the byte position in the state. It rotates each row with different offsets to obtain a new state as follows:

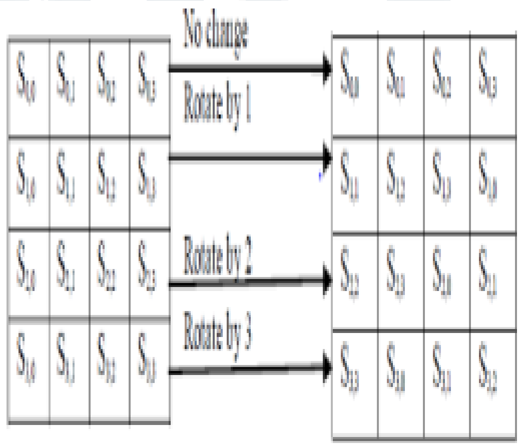


Figure 9 Shift Row Transformations

The first row is unchanged, the second row is left circular shifted by one, the third row is by two, and the last row is by three.

[5] Mix Columns

The Mix Columns operation mixes every consecutive four bytes of the state to obtain four new bytes as follows:

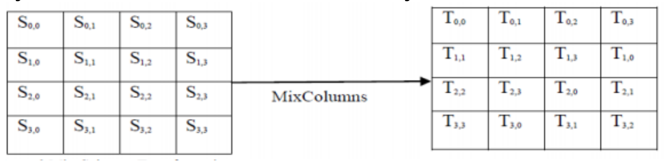


Figure 10 Mix Columns Transformations

Let Si,Si+1, Si+2 and Si+3 represent every consecutive four bytes, where i belongs to {0,1,2,3}Then,

$$\begin{pmatrix} t_i \\ t_{i+1} \\ t_{i+2} \\ t_{i+3} \end{pmatrix} = \begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 02 & 03 & 01 \end{pmatrix} \begin{pmatrix} s_i \\ s_{i+1} \\ s_{i+2} \\ s_{i+3} \end{pmatrix}$$

Each entry of the constant matrix in (1) belongs to GF(28)hence Equation (1) is a matrix-vector multiplication over GF(28)[9].

[6]Add Round Key and Key Expansion

Each round has a 128-bit round key which is segmented into 16 bytes ki Add Round Key is simply addition (2) The key expansion expands a unique private key as a key stream of (4r + 4) 32-bit words, where r is 10, 12, or 14. The private key is segmented into Nk words according to the key length, where Nk is 4, 6, or 8 for a 128- bit, 192-bit, or 256-bit cipher key, respectively. then, it generates the ith word (32 bits) by EXORing the (iNk)th word with either the (i -1)th word or the conditionally transformed (i- 1)th word.
where Nk ≤ i ≤ (4r + 3). The (i- 1)th word is conditionally transformed by RotWord, Sub Bytes and EXORing with Rcon[i/ Nk]={02 [I / Nk],00, 00, 00} where the polynomial presentation of 02[i/ Nk] is x[i/ Nk] over GF(28). Finally, the key stream is segmented into several round keys which are involved in the Add Round Key operation and last we get the output as cipher text.

III RESULT

VHDL is used as the hardware description language because of the flexibility to exchange among environments. The software used for this work is Xilinx ISE Design Suite 13.2 The tools primarily used are the Xilinx ISE and ISim for simulation, Synthesis and implementation. Round one encryption AE

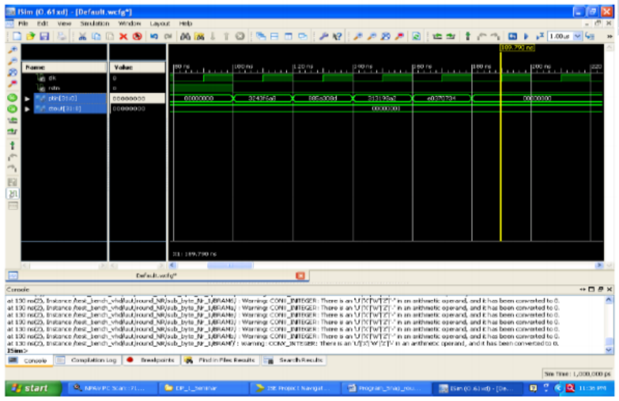


Figure 11. Cipher Text input

Test bench output shown in Fig (10) for the input plaint text of 128 bit in four clock cycle default binary input plain text converted in the hexadecimal form. Input State: 3243 f6 a8 88 5a 30 8d 31 31 98 a2 e0 37 07 34

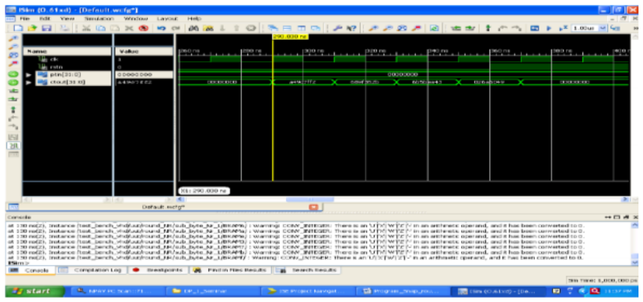


Figure 12. Cipher Text Output

Test bench output shown in Fig(12) for the output Cipher Text of 128 bit in four clock cycle default binary output Cipher text converted in the hexadecimal form. Round one Cipher Text : a4 9c 7f f2 68 9f 35 2b 6b 5b ea 43 02 6a 50 49

IV CONCLUSION

In this chapter, new hardware architectures for the Advanced Encryption Standard (AES) algorithm were presented. FPGA Xilinx technology was used to synthesise the designs and provide post placement results using Xilinx ISE 10.1 for AES pipeline architecture and Xilinx ISE 13.4 has used for the AES iterative algorithm. The maximum throughput of the design is 185.815 Mbits/s. Medium resolution video (640x480) of true colour depth (24 bits per pixel) has a bit rate of 184.3 Mbits/s. Therefore, the proposed architecture implementation in spartan6 FPGA has enough throughputs to encrypt the video resolution mentioned above in real time. Because, the proposed iterative design has low area, it is suitable for the implementation in small devices like, smart cards, cellular phones, etc.

REFERENCE

[1] *Advanced Encryption Standard (AES)*, Nov. 26, 2001.

[2] A. J. Elbirt, W. Yip, B. Chetwynd, and C. Paar. An FPGA implementation andperformance evaluationof the AESblock ciphercandidate algorithm finalist. presented at *Proc. 3rd AES Conf. (AES3)*. [Online]. Available: http://csrc.nist.gov/encryption/aes/round2/conf3/aes3papers.html

[3] V. Fischer and M. Drutarovsky, “Two methods of Rijndael implementation in reconfigurable hardware,” in *Proc. CHES 2001*, Paris, France, May 2001, pp. 77–92

[4] K. Gaj and P. Chodowiec. Comparison of the hardware performance of the AES candidates using reconfigurable hardware. presented at *Proc. 3rd AES Conf. (AES3)*.

[Online]. Available: http://csrc.nist.gov /encryption/aes/round2/conf3/aes3papers.html

[5] H. Kuo and I. Verbauwhede, “Architectural optimization for a 1.82 Gbits/sec VLSI implementation of the AES Rijndael algorithm,” in *Proc. CHES 2001*, Paris, France, May 2001, pp. 51–64

[6] M. McLoone and J. V. McCanny, “Rijndael FPGA implementation utilizing look-up tables,” in *IEEE Workshop on Signal Processing Systems*, Sept. 2001, pp. 349–360

[7] A. Satoh, S. Morioka, K. Takano, and S. Munetoh, “A compact Rijndael hardware architecture with S-Box optimization,” in *Proc. ASIACRYPT 2001*, Gold Coast, Australia, Dec. 2000, pp. 239–254

[8] A. Rudra, P. K. Dubey, C. S. Jutla, V. Kumar, J. R. Rao, and P. Rohatgi, “Efficient implementation of Rijndael encryption with composite field arithmetic,” in *Proc. CHES 2001*, Paris, France, May 2001, pp. 171–184

[9] K. U. Jarvinen, M. T. Tommiska, and J. O. Skytta, “A fully pipelined memoryless 17.8 Gbps AES-128 encryptor,” in *Proc. Int. Symp. FieldProgrammable Gate Arrays (FPGA 2003)*, Monterey, CA, Feb. 2003, pp. 207–215

[10] G. P. Saggese, A. Mazzeo, N. Mazocca, and A. G. M. Strollo, “An FPGA based performance analysis of the unrolling, tiling and pipelining of the AES algorithm,” in *Proc. FPL 2003*, Portugal, Sept. 2003

[11] F. Standaert, G. Rouvroy, J. Quisquater, and J. Legat, “Efficient implementation of Rijndael encryption in reconfigurable hardware: Improvements & design tradeoffs,” in *Proc. CHES 2003*, Cologne, Germany, Sept. 2003

[12] C.Paar, “EfficientVLSI architecturefor bit-parallelcomputationsin Galois field,” Ph.D. dissertation, Institute for Experimental Mathematics, University of Essen, Essen, Germany, 1994

[13] M. H. Jing, Y. H. Chen, Y. T. Chang, and C. H. Hsu, “The design of a fast inverse module in AES,” in *Proc. Int. Conf. Info-Tech and Info-Net*, vol. 3, Beijing, China, Nov. 2001, pp. 298–303.

[14] C. C. Lu and S. Y. Tseng, “Integrated design of AES (advanced encryption standard) encrypter and decrypter,” in *Proc. IEEE Int. Conf. Application Specific Systems, Architectures Processors*, 2002, pp. 277–285

[15] X. Zhang and K. K. Parhi, “Implementation approaches for the advanced encryption standard algorithm,” *IEEE Circuits Syst. Mag.*, vol. 2, no. 4, pp. 24–46, 2002