Proc. of the Intl. Conf. on Recent Trends In Computing and Communication Engineering -- RTCCE 2013

ISBN: 978-981-07-6184-4

VHDL Implementation of AES-128

Richa Sharma, Purnima Gehlot, S. R. Biradar

***Abstract*-Security has become an increasingly important feature with the growth of electronic communication. The Symmetric in which the same key value is used in both the encryption and decryption calculations are becoming more popular. The AES algorithm is capable of using cryptographic keys of 128, 192, and 256 bits to encrypt and decrypt data in blocks of 128 bits. This standard is based on the Rijndael algorithm. In this project our main concern is to implement all modules of this algorithm on hardware.This methodology uses VHDL implementation of all the modules in terms of Delay and Frequency.**

***Key Words*- Cryptography , Secret key AES, Rijndael, FPGA, VHDL.**

I. Introduction

Nowadays cryptography has a main role in embedded systems design. As the number of devices and applications which send and receive data are increasing rapidly, the data transfer rates are becoming higher. In many applications, this data requires a secured connection which is usually achieved by cryptography. Cryptography is divided in two categories first is symmetric key cryptography (sender and receiver shares the same key) and the second one is asymmetric key cryptography (sender and receiver shares different keys). Here we are concerned about symmetric key cryptography due to its use in military application, embedded system design, financial and legal files, medical reports, and bank services via Internet, telephone conversations, and e-commerce transactions etc. Many symmetric key cryptographic algorithms were proposed, such as the Data Encryption Standard (DES), the Elliptic Curve Cryptography (ECC), the Advanced Encryption Standard (AES) and other algorithms[1]. Here the hardware implementation of AES algorithm is presented to increase the data transfer speed.

Richa Sharma, Purnima Gehlot *(M.Tech VLSI Design)*
MITS University, Lakshmangarh
INDIA

Dr. S.R. Biradar *(Professsor)*
MITS University, Lakshmangarh
INDIA

II. The AES Algorithm

A. *Background of Algorithm*

The National Institute of Standards and Technology (NIST) has initiated a process to develop a Federal information Processing Standard (FIPS) for the Advanced Encryption Standard (AES),specifying an Advanced Encryption Algorithm to replace the Data Encryption standard (DES) the Expired in 1998. The Rijndael Algorithm was chosen since it had the best overall scores in security, performance, efficiency, implementation ability and flexibility[2].

B. *Basic of Algorithm*

The Rijndael algorithm is a symmetric block cipher that can process data blocks of 128 bits through the use of cipher keys with lengths of 128, 192, and 256 bits..The AES algorithm as Rijndael is also a symmetric block cipher that can encrypt (encipher) and decrypt (decipher) information [1]. Encryption converts data to an unintelligible form called cipher-text. Decryption of the cipher-text converts the data back into its original form, which is called plaintext. The number of rounds is depends upon the key length as described in TableI [3].

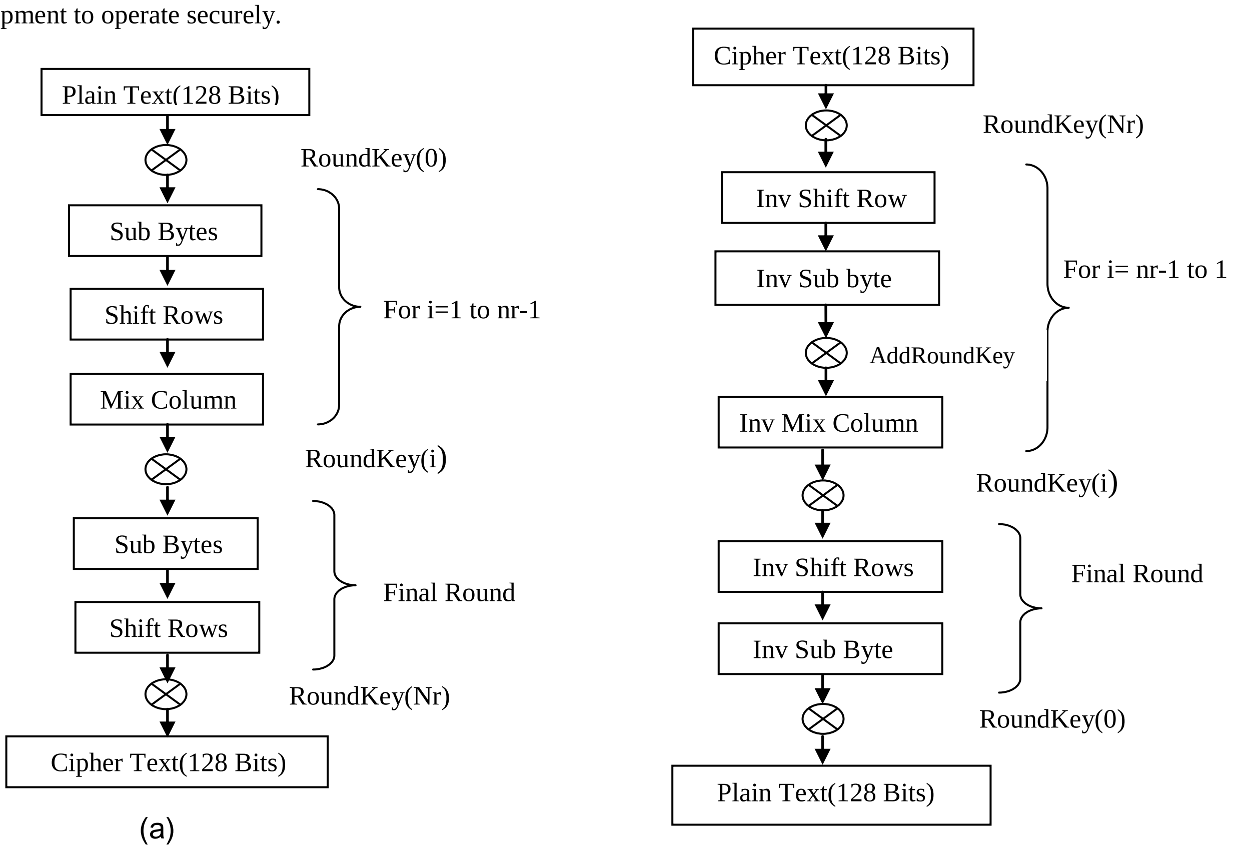
C. *Encryption and Decryption in AES*

The Basic AES Encryption and Decryption structure with various steps is Shown in FIGURE I.This block diagram is generic for AES specifications. It consists of a number of different transformations applied consecutively over the data block bits, in a fixed number of iterations, called rounds [2]. The number of rounds depends on the length of the key used for the encryption process. The Advanced Encryption Standard can be programmed in software or built with pure hardware[5].

TABLEI. KEY BLOCK ROUND COMBINATION

Block Size (Nb words) = 4		
Bit Mode	**Key Length (Nk words)**	**Number of Rounds (Nr)**
128	4	10
192	6	12
256	8	14

The hardware implementation of the Rijndael algorithm can provide either high performance or low cost for specific applications. At backbone communication channels or heavily loaded servers it is not possible to lose processing speed, which drops the efficiency of the overall system while running cryptography algorithms in software [5]. On the other hand, in the performance comparison between software and hardware implementation the priority is to evaluate which system provides higher security. Hardware's inflexibility eliminates possibility for the external changes to the system, and this result in a high quality physical security when compared with software implementations. . A low cost and small design can be used in smart card applications, which allows a wide range of equipment to operate securely.



(b)

FIGURE I: AES Algorithm

(a) Encryption structure.

(b) Equivalent Decryption structure.

III. Transformations in AES

A. *Sub Byte Transformation*

The bytes substitution transformation is a non-linear substitution of bytes that operates independently on each byte of the State using a substitution table (S-box)[2].The sub byte transform is shown in FIGURE II. In AES hardware implementation, S-box design contributes a major role in optimization. There are two approaches for S-box design. Design a multiplicative inversion and affine transformation separately or Construct a logic circuit defining the input and output of the S-box function [8].

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

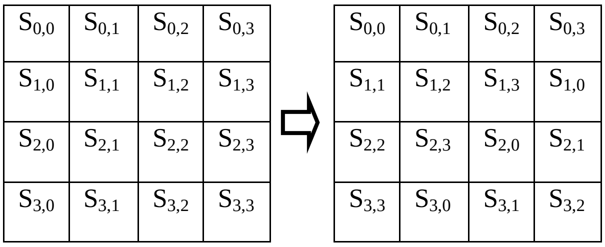
⇒

$S'_{0,0}$	$S'_{0,1}$	$S'_{0,2}$	$S'_{0,3}$
$S'_{1,0}$	$S'_{1,1}$	$S'_{1,2}$	$S'_{1,3}$
$S'_{2,0}$	$S'_{2,1}$	$S'_{2,2}$	$S'_{2,3}$
$S'_{3,0}$	$S'_{3,1}$	$S'_{3,2}$	$S'_{3,3}$

FIGURE II: Application of S-box to the Each Byte of the State.

B. *Shift Rows Transformation*

In the Shift Rows transformation the bytes in the last three rows of the State are cyclically shifted over different numbers of bytes (offsets). The first row, *r* =0, is not shifted, while the second, third and fourth rows cyclically shift one byte, two bytes and three bytes to the left, respectively[3].The Shift row transform is shown in FIGURE III.



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

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

FIGURE III: Shift Row transformation

To implement the Shift Row Function on hardware we have basically two main approaches. Designing it by using barrel Shifter or it can be implemented by using Look up Table [8].

C. *Mix Columns Transformation*

This transformation is based on Galois Field multiplication. Each byte of a column is replaced with another value that is a function of all four bytes in the given column. The Mix Columns transformation is performed on the State column-by-column [5].The mix column implementation is shown in FIGURE IV. Each column is considered as a four-term polynomial over GF(2^8) and multiplied by $a(x)$ modulo $x^4 + 1$, Where

$$a(x) = \{03\}x^3 + \{01\}x^2 + \{01\}x + \{02\}.$$

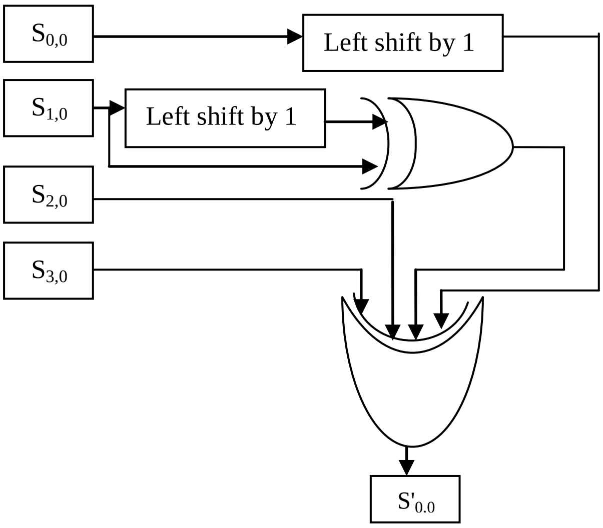


FIGURE IV: Mix Column Transformation in Matrix Form

D. *Add Round Key and Key Expansion Unit*

1) *Add Round Key*

Add Round Key step is applied one extra time comparing to the other encryption and decryption steps. The first Add Round Key step is applied before starting the encryption and decryption iterations, where in the encryption process the first 128 bits of the input key the whole key in case of using key size of 128 bits are added to the original data block as shown in FIGURE V. This round key is called the initial round key [4]. It is implemented in hardware as a simple exclusive-or operation of the 128 bit data and key.

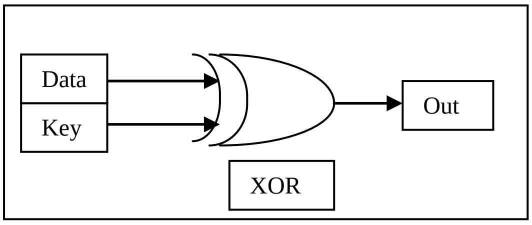


FIGURE V: Hardware Implementation of Add Round Key

2) *Key Expansion*

The key expansion term is used to describe the operation of generating all Round Keys from the original input key. The initial round key will be the original key in case of encryption and the last group of the generated key expansion keys in case of decryption[6]. The whole operation is shown in FIGURE VI.

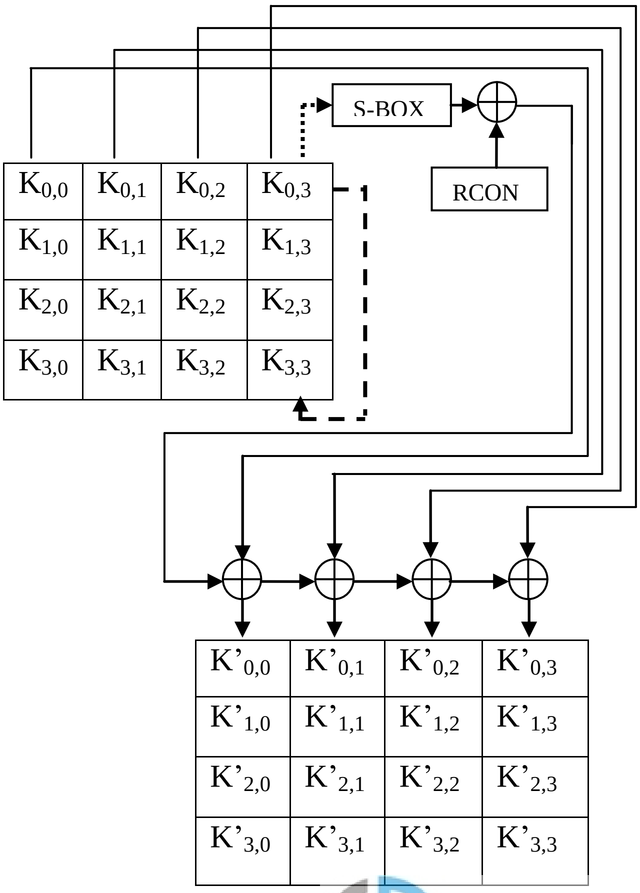


FIGURE VI: Implementation of Key Expansion

IV. Simulations and Results

VHDL is used as the hardware description language because of the flexibility to exchange among environments. The software used for this work is Xilinx 6.1i and the waveforms are simulated with the help of model sim simulator. This is used for writing, debugging, simulating and checking the performance results using the simulation tools available on Xilinx 6.1i. The delay is calculated with three different Device families [3]. The delay have been generated as result is shown in TABLE II. As different Delay calculations by virtex 2. The interface and RTL for Encryption and Decryption is shown in FIGURE VII.

TABLE II. Delay and Frequency of Different Modules

Modules	Delay (ns)	Frequency (MHz)
Sub Byte	10.69	93.54
Shift Rows	7.158	139.70
Mix Column	8.168	122.428
Key Expansion	13.475	74.211

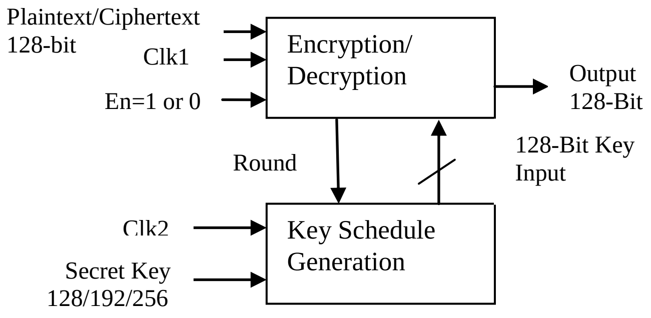


FIGURE VII: RTL for Encryption and Decryption

Conclusion

As the cryptography is playing the major role in today's world. So the frequency is the main concern so that the time period can be minimized. Here in this report we have explained about the basics of AES algorithm and the implementation of its modules by using VHDL. Here the simulations are performed with different device families. The software we have used is Xilinx6.1i and the waveforms are simulated with model sim simulator.

Future Work

As in this whole work we are trying to reduce the delay by applying different techniques so need to work to minimize more delay as well as to implement the algorithm in Different applications as in Bluetooth, cloud computing etc.

References

[1] Xinmiao Zhang and Keshab K. Parhi "Implementation Approaches for the Advanced Encryption Standard Algorithm" *IEEE* 2002.

[2] Hui QIN, Tsutomu SASAO, Yukihiro IGUCHI "An FPGA Design of AES Encryption Circuit with 128-bit Keys" *GLSVLSI'05, ACM* 2005

[3]Chih-Peng Fanand and Jun-Kui Hwang "FPGA Implementations Of High Throughput Sequential And Fully Pipelined AES Algorithm" *International journal of Electrical Engineering,* vol.15, no.6, pp. 447-455, 2008.

[4] Mehran Mozaffari-Kermani and Arash Reyhani-Masoleh "Efficient and High Performance Parallel Hardware Architecture for the AES-GCM" *IEEE Transactions On Computers,* vol.61, no. 8, August 2012.

[5] Saambhavi Baskaran and Pachamuthu Rajalakshmi "Hardware Software Co-Design of AES on FPGA" *ICACCI '12,ACM* August 2012.

[6] Ashwini M. Deshpande, Mangesh S. Deshpande and Devendra N. Kayatanavar "FPGA Implementation of AES Encryption and Decryption" *International Conference on Control, Automation, Communication and Energy conservation* -2009

[7] X. Zhang and K. K. Parhi, "High-speed VLSI architectures for the AES algorithm,"*IEEE Transactions on Very Large Scale Integration Systems*, vol.12, issue 9, pp.95 967, Sep. 2004.

[8] Jin Gong ,Wenyi Liu, Huixin Zhang "Multiple Lookup Table-Based AES Encryption Algorithm Implementation" *Elseveir-*2012 vol.25 pg no.842 – 847.

Ms. Richa Sharma has completed her B.E in Electronics and communication from RCEW, Jaipur (Raj) and Pursuing her M.Tech in VLSI Design from MITS, Lakshmangarh,India.Her research interest includes network security and privacy.

Ms. Purnima Gehlot has completed her B.E. in Electronics and communication from MIT, Ujjain (M.P), and Pursuing her M.Tech in VLSI Design from MITS, Lakshmangarh,India. Her research interest includes network security and privacy.

Mr. S.R. Biradar is a Professor in the department of Computer Science and Engineering, MITS, Lakshmangarh, India. He received his B.E,M.Tech and Ph.D degrees in Computer Science and Engineering from Karnataka University, MAHE Manipal and Jadavpur Universityrespectively. His research interest includes Mobile Ad-hoc networking, advanced wireless communication.