A Novel AES Algorithm Based On FPGA

G.CHAMUNDI, K.RAMBABU
[1]*PG-Student, Dept. Of ECE, CMR INSTITUTE OF TECHNOLOGY&SCIENCE, Andhra Pradesh , India*
[2]*Assistant Professor, Dept. Of ECE, CRVITS, SHAMEERPET, RANGAREDDY, Andhra Pradesh, India*
1chamu.gandodhi@gmail.com
2ramakoleti@gmail.com

***Abstract*— Development of Computer Network and Communication Technology involves a great mass of data and information need to be exchanged by public communication networks. High efficiency and high safety of Data transmission become much more important. The appearance of DES (Data Encryption standard) solved many fields of information security problems. But, the security strength of DES is hard to adapt the new security needs by the development of password analysis level In 1997, National Institute of Standards and Technology (NIST) collected advanced encryption standard (AES) through public announcement. Finally, the algorithm of Joan Daemon and Vincent Ragmen who are cryptographers in Belgium was selected as the standard for AES. The AES algorithm can resist any kinds of password attacks that all we have known with a strong practicability in information security and reliability. It has become the main information encryption algorithm now. The following presented an implementation of AES algorithm based on FPGA in order to speed data flow and reduce time for key generating. An implementation of high speed AES algorithm based on FPGA is presented in this paper in order to improve the safety of data in transmission. The mathematic principle, encryption process and logic structure of AES algorithm are introduced. The simulation results show that the high-speed AES encryption algorithm implemented correctly. Using the method of AES encryption the data could be protected effectively.**

Keywords- Advanced Encryption Standard (AES), Cryptography, Data Encryption Standard (DES), Information Security, FPGA.

I. INTRODUCTION

Cryptography is the art & science of keeping messages secure. Cryptography enables you to store sensitive information or transmit it across insecure networks (like the Internet) so that it cannot be read by anyone except the intended recipient. With cryptosystems, we desire perfect secrecy: the probability that the contents of some intercepted data corresponds to some plaintext message is unaltered by knowledge of the ciphertext for that message. Measuring the strength for cryptosystem by what is known as its work factor:The amount of time needed to decipher a message without knowledge of the key. A cryptosystem is considered secure when its workfactor is exponential in the length of the key:2^{Keylen}.

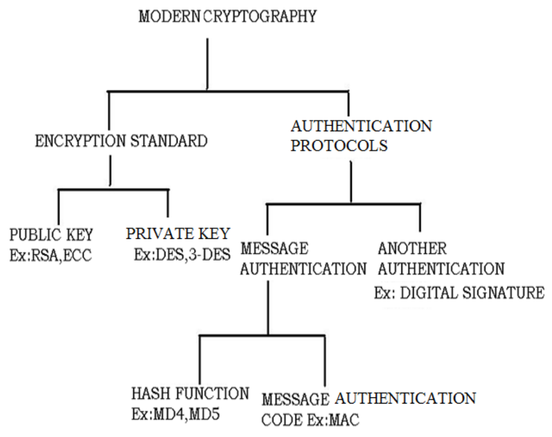


Fig.1 Modern Cryptography Hierarchy

Types of Cryptographic Functions:

- Secret key (symmetric): involves 1 key, known as the secret key
- Public key (asymmetric):involves 2 keys, known as the private & public keys
- hash: involves 0 keys

II. EXISISTING SYSTEM

The Data Encryption Standard (DES) is a previously predominant algorithm for the encryption of electronic data. It was highly influential in the advancement of modern cryptography in the academic world. Developed in the early 1970s at IBM and based on an earlier design by Horst Festal, the algorithm was submitted to the National Bureau of Standards (NBS) following the agency's invitation to propose a candidate for the protection of sensitive, unclassified electronic government data. In 1976, after consultation with the National Security Agency (NSA), the NBS eventually selected a slightly modified version, which was published as an official Federal Information Processing Standard (FIPS) for the United States in 1977. The publication of an NSA-approved encryption standard simultaneously resulted in its quick international adoption and widespread academic scrutiny.

Controversies arose out of classified design elements, a relatively short key length of the symmetric-key block cipher design, and the involvement of the NSA, nourishing suspicions about a backdoor. The intense academic scrutiny the algorithm received over time led to the modern understanding of block ciphers and their cryptanalysis.

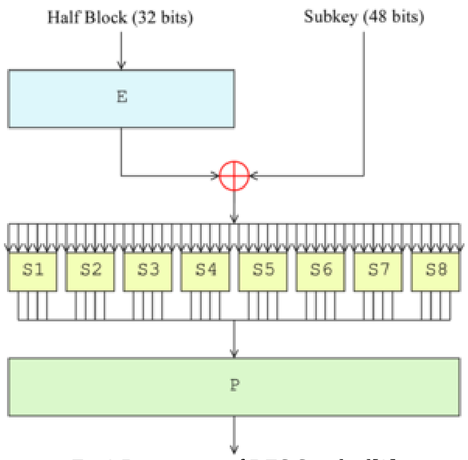


Fig.2 Description of DES Standard[2]

DES is now considered to be insecure for many applications. This is chiefly due to the 56-bit key size being too small; in January, 1999, distributed.net and the Electronic Frontier Foundation collaborated to publicly break a DES key in 22 hours and 15 minutes (see chronology). There are also some analytical results which demonstrate theoretical weaknesses in the cipher, although they are infeasible to mount in practice. The algorithm is believed to be practically secure in the form of Triple DES, although there are theoretical attacks. In recent years, the cipher has been superseded by the Advanced Encryption Standard (AES). Furthermore, DES has been withdrawn as a standard by the National Institute of Standards and Technology (formerly the National Bureau of Standards).

III. PROPOSED SYSTEM

Our paper introduces advanced encryption system (AES) Algorithm that resists any kinds of password attacks that all we have known with a strong practicability in information security and reliability. It has become the main information encryption algorithm now. Here, we implemented AES algorithm based on FPGA.

A. *Working Principle*

The AES algorithm is based on finite field GF (28).The algorithm includes two mathematical operations: INVERSE SUBSTITUTION BOX;

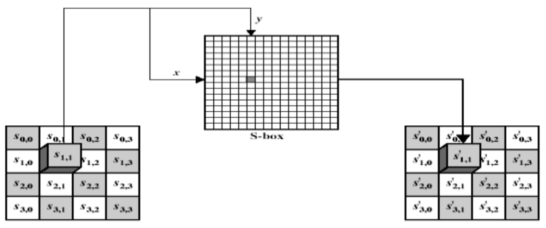
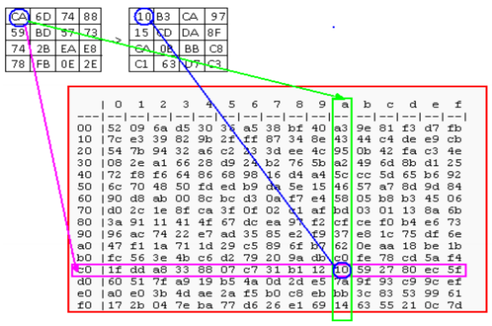


Fig.3 Working Methodology of AES

Each byte at the input of a round undergoes a non-linear byte substitution. A simple substitution of each byte in state array uses one table (S-box) of 16x16 bytes containing a permutation of all 256 8-bit values each byte of state is replaced by byte in row (left 4 bits) & column (right 4-bits) of inverse S-box example: byte {95} is replaced by row 9 column 5 byte in inverse S-box which is the value {AD} S-box is constructed (pre-defined in AES).[4]

1. *Predefined Inverse S-BOX:*



CA	6D	74	88
59	BD	57	73
74	2B	EA	E8
78	FB	0E	2E

10	B3	CA	97
15	CD	DA	8F
CA	0B	BB	C8
C1	63	D7	C3

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
00	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
10	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
20	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
30	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
40	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
50	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
60	90	d8	ab	00	8c	bc	d3	0a	f7	e4	58	05	b8	b3	45	06
70	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
80	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
90	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	df	6e
a0	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
b0	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
c0	1f	dd	a8	33	88	07	c7	31	b1	12	10	59	27	80	ec	5f
d0	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
e0	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
f0	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

Fig.4 Structure of predefined inverse S-BOX

Substitution values for the bytes xy (in hexadecimal format).

2. *Inverse Shift Row:*

A circular byte shift in each row of state

- 1st row is unchanged
- 2nd row does 1 byte circular shift to right
- 3rd row does 2 byte circular shift to right
- 4th row does 3 byte circular shift to right

Since state is processed by columns, this step permutes bytes between the columns.This transformation ensures that 4 bytes of one column are spread out to 4 different columns.

3. *Mix Column:*

Each column is processed separately. Each byte is replaced by a value dependent on all 4 bytes in the column.

Example: s'0,0 = (14*s0,0) + (11*s1,0) + (13*s2,0) +(9*s3,0) But, in GF(28) addition (+) is XOR operation Multiplication (*) is combinations of shift and XOR.

$$\begin{bmatrix} r_0 \\ r_1 \\ r_2 \\ r_3 \end{bmatrix} = \begin{bmatrix} 14 & 11 & 13 & 9 \\ 9 & 14 & 11 & 13 \\ 13 & 9 & 14 & 11 \\ 11 & 13 & 9 & 14 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

Mix Columns + Shift Rows ensure all output bits depend on all input bits after a few rounds.

4. *Inverse ADD Round Key:*

XOR state with 128-bits of the round key. Processed by column (a series of byte operations) Inverse for decryption is identical since XOR is own inverse, just with correct round key designed to be as simple as possible The security is ensured by the complexity of key expansion and the complexity of other stages of AES.

B. System Design Overview

It is incompatible to implement the AES algorithm on hardware between the throughput and hardware resource. Different architecture should be selected according to the fields it is applied to. To make AES algorithm suitable to high-speed rate data application, we need to optimize the architecture. Meanwhile by sharing resource and eliminating common sub expression we can reduce the hardware resource utilization. There are three basic architectures of AES to improve the throughput: Loop unrolled, pipelined, sub-pipelined that could be chosen [4].The top design of AES encryption system is the top design includes three modules: Round module, Key_scheduel module and control module. Round module, this contains four sub modules: Sub Bytes, Shift Rows, Mix Columns and AddRoundKey perform the prime transformation process of AES. Key_scheduel module includes an S-box macro module to realize the nonlinear transformation. Control module in charge of the signals for other modules and the data in Input/output.

c. AES Algorithm Implementation in FPGA

The design uses a synchronous clock in order to make the circuit works with a unified clock and uses pipeline Architecture to improve the working speed.

Round module includes Sub Bytes, Shift Rows, Mix Columns, AddRoundKey and an S-box matrix. Sub Bytes is a substituted operation to execute the operation and the affine transformation on finite field. Shift Rows is a cycle shift with bytes for unit. The most important process in Mix Columns is the multiplication on finite field. AddRoundKey is a process that makes a 128 bits key to exclusive or the data in state one by one. S-box is a matrix That is defined to make a nonlinear replacement for Sub Bytes. The pipeline scheme is utilized for implementations. In the pipeline scheme, the register arrays are as-signed among the operational circuits of Sub Bytes, Shift Rows, Mix Columns and AddRoundKey. Key Schedule module includes Byte-substitution, Byte shifting, round constant, exclusive or operation and an S-box. A key is composed of four words in the design, key_word (0), key_word (1), key_word (2) and key_word (3). The initial input key is 128 bits. The ranges of each word are defined as
Follows:

```
key_word (0) <= key_reg_out (127 down to 96);
key_word (1) <= key_reg_out (95 down to 64);
key_word (2) <= key_reg_out (63 down to 32);
key_word (3) <= key_reg_out (31 down to 0);
```

Control module: The task of Control module is coordinating working signaling for other modules. Control module manages the crypto key memory and the output of the Round constant, and controls the storage of Key Schedule and saves the encryption control signal, to ensure it can be output correctly. In the design, a register is defined to counter the number of the rounds. The sequence is from 1 to 10. Control module controls the 10 round constants in order and makes the transformation in a correct running order.

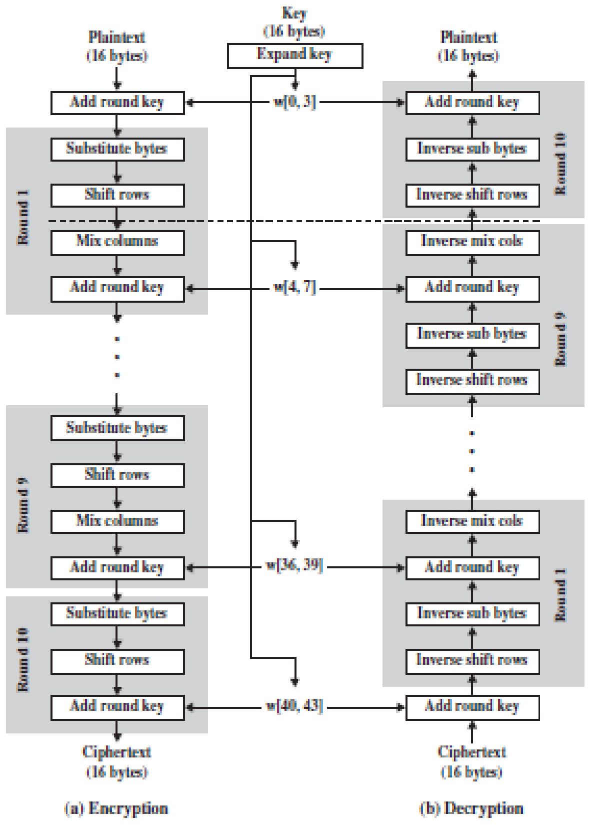


Figure 5.3 AES Encryption and Decryption

```
entity preroundoperaton is

    Port ( a,b : in  STD_LOGIC_VECTOR (127 downto 0);

                c  :  out    STD_LOGIC_VECTOR  (127 downto 0));

end preroundoperaton;


architecture Behavioral of preroundoperaton is

begin

c <= a xor b;

end Behavioral;

entity subbyte is

    Port ( subin : in   STD_LOGIC_VECTOR (127 downto 0);

                      subout : out  STD_LOGIC_VECTOR
       (127 downto 0));

end subbyte;


architecture Behavioral of subbyte is

signal
a0,a1,a2,a3,a4,a5,a6,a7,a8,a9,a10,a11,a12,a13,a14,a15,
p0,p1,p2,p3,p4,p5,p6,p7,p8,p9,p10,p11,p12,p13,p14,p
15:std_logic_vector(7              downto              0);


 component sbox port (a:in std_logic_vector(7 downto 0);

                                             (b:out
       std_logic_vector(7 downto 0));

 end component;
```

VI. SIMULATION AND RESULTS

➢ Software's Tools:

- The project will be implemented using HDL.
- Simulation will be done to verify the functionality and synthesis will be done to get the NETLIST.
- Simulation and synthesis will be done using Xilinx Tools.

➢ Our paper yields the following results when tested in real-time scenario:

- The AES algorithm can resist any kinds of password attacks that all we have known with a strong practicability in information security and reliability.
- It has become the main information encryption algorithm now.
- This implementation of AES algorithm based on FPGA is to speed up data flow and reduce time for key generating.
- The design uses a synchronous clock in order to make the circuit works with a unified clock and uses pipeline architecture to improve the working speed.

Plaintext:

27DABD46F9DA52D79967B7A0B33A492E

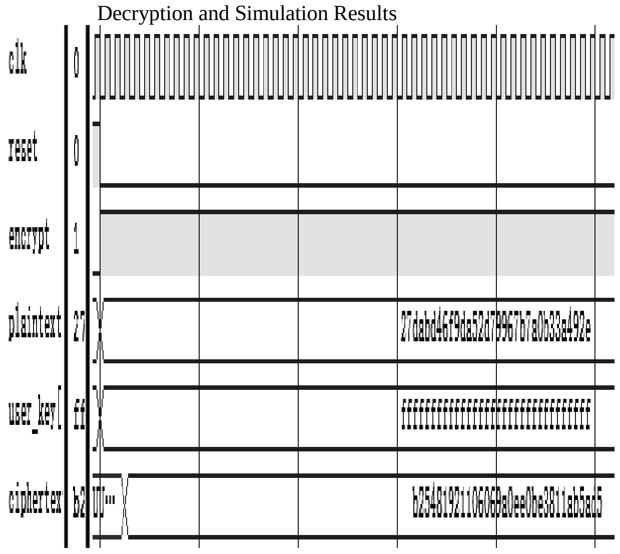
User key:

FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF

Cipher text:

B25481921106069A0EE0BE3811AB5AD5

TABLE I
Decryption and Simulation Results


clk
0
reset
0
encrypt
1
plaintext
27
27dabd46f9da52d78967b7a0b33a492e
user_key[
ff
ffffffffffffffffffffffffffffffff
ciphertex
b2
UU···
b25481921106069a0ee0be3811ab5ad5

VI. CONCLUSION

The design was tested in Xilinx Virtex-5 FPGA. The test Results show that the system could complete the whole process correctly in a 200MHz clock rate. The minimum period is 4.971ns. Based on Virtex-5 FPGA, This paper gives a design of AES encryption algorithm using pipeline structure and parallel processing. All the processes including Synthesis, Implementation and Simulation are finished in ISE13.3 development platform. It adopts a very simple and effective design with characteristics as listed in tabular form shown below:

TABLE II
Characteristics of proposed AES algorithm

Key size (words/bytes/bits)	4/16/128	6/24/192	8/32/256
Plaintext block size (words/bytes/bits)	4/16/128	4/16/128	4/16/128
Number of rounds	10	12	14
Round key size (words/bytes/bits)	4/16/128	4/16/128	4/16/128
Expanded key size (words/bytes)	44/176	52/208	60/240

ACKNOWLEDGMENT

I am extremely thankful to CMRIT Electronics Communication Engineering Department for providing excellent lab facilities which were helpful in successful completion of my project.

REFERENCES

- Zhenzhen Liu. Implementation of AES Encryption based on FPGA.Modern electronic technology, 2007,23(3),pp:103-104.[1]
- http://en.wikipedia.org/wiki/Data_Encryption_Standard[2]
- Symposium DUAN Peng, HE Mingyi, XUE Minbiao, “Design and Simulation of OFDM Synchronization System Based on FPGA,” Meansurment and Control Technology, vol. 28, n. 11, 2009, pp. 63-67 [3].
- Shanxin Qu,Guochu Shou,Yihong Hu,Zhigang Guo,Zongjue Qian. High Throughput Pipelined Implementation of AES on FPGA. International Symposium on Information Engineering and Electronic Commerce.2009[4]