International Journal for Research in Engineering Application & Management (IJREAM)
ISSN : 2454-9150 Vol-04, Issue-03, June 2018

Encryption of Real Time Image –Implementation in FPGA

Ditta Saju, MTech-VLSI & Embedded Systems, Mar Athanasius College of Engineering Kothamangalam, India, dittasaju20@gmail.com

Nithin James, Department of Electronics and Communication, Mar Athanasius College of Engineering Kothamangalam, India, nithinjames@mace.ac.in

Abstract- Nowadays, confidential images are widely communicated over the networks. Sometimes, image may fall into the wrong hands, if the sender doesn't use any image securing techniques. The security of image is a vital and challenging task. Various methods are used to secure the image, such as encryption, steganography, and watermarking. Among these, encryption is the widely accepted technique. If you send an encrypted image, only the person with the encryption key can retrieve the image. This project proposes a method for encrypting an image. For encryption Advanced Encryption Standard (AES) is used. In this project, SPARTAN6 FPGA (Field Programmable Gate Array) is used for implementing the algorithm.

Keywords—AES,Encryption,Spartan6,FPGA,Steganography,Watermarking

I. INTRODUCTION

Transfer of confidential image is an essential thing in many fields, such as hospitals, military, picture messages on phones and so on. Some private or personal image was not well protected and that may maliciously use by other parties. For securing such confidential images, encryption technique can be used. Encryption can be performed by many methods like Advanced Encryption Standard (AES), Data Encryption Standard (DES), Triple DES algorithm, RSA and Blowfish. But many of these methods are susceptible to various kinds of attacks. Therefore, the need of flexible and most secure and algorithm arises. The secure and widely accepted algorithm is AES. AES has a fixed block size of 128bits, and a key size of 128, 192 or 256 bits [1]. For this work, a key length of 128-bit and 10 number of iterations is used.

Image captured by a camera module is encrypted in SPARTAN 6 FPGA using Verilog as the Hardware Description Language (HDL). Captured and the encrypted image can be displayed on monitor via VGA (Video Graphics Array). Encrypting an image simply means encrypting the pixel data of image. Since AES is a 128 bit block cipher, each 128 bit pixel data of image is taken out and encrypted.

II. ENCRYPTION ALGORITHM (AES)

The more popular and widely adopted symmetric encryption algorithm is the Advanced Encryption Algorithm (AES). AES is iterative, since it consists of a series of linked operations. Some of linked operations were replacing inputs by entries in a lookup table (substitution) and shifting of bits. All operations of AES are performed on bytes rather than bits. AES treats 128 bit data as 16 bytes. AES uses 10 rounds for 128-bit keys, 12 rounds for192-bit keys and 14 rounds for 256-bit keys. Each of these rounds uses different 128-bit round key. Round keys are calculated from the original AES key, which is known only to sender and receiver. Each round consists of four sub processes. Fig.1 shows these sub processes [1].

AES is a safest and trusted algorithm, which is used by numerous organizations. Since AES is a symmetric key algorithm, same key is used for both encryption and decryption [2]. In earlier days, Data Encryption Standard (DES) had a high priority, but now AES is the more preferred symmetric encryption standard. Even a supercomputer can't hack AES key, since it uses 128-bit key.

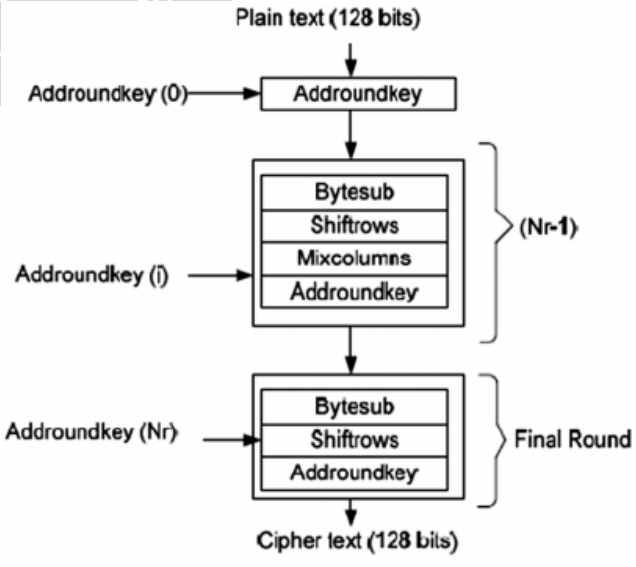

Fig.1 Encryption Steps

 DOI : 10.18231/2454-9150.2018.0365

A. ADD ROUND KEY

Output from previous round (p) is EXOR with successive key (k) in each round.

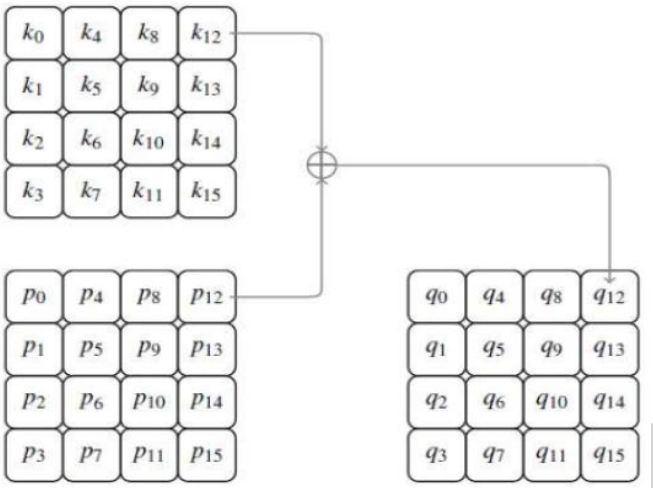

Fig.2 AES Add Round Key

B. KEY EXPANSION

Round keys are created using key expansion.

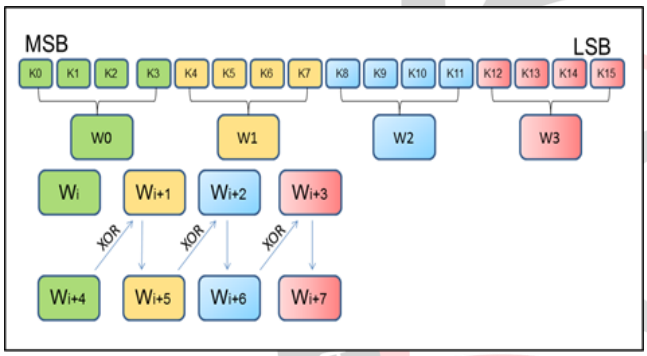

Fig.3 AES Key Expansion

In Fig.3, to obtain Wi_{+4} to W_{i+7} from W_i to W_{i+3}

W_{i+5} = W_{i+4} xor W_{i+1}

W_{i+6} = W_{i+5} xor W_{i+2}

W_{i+7} = W_{i+6} xor W_{i+3}

To obtain W_{i+4}, the following 3 steps are performed:

- X = One byte left circular rotation of W_{i+3}
- Y = BYTE substitution for each byte of X (use S-BOX as shown in TABLE 2 for encryption and decryption).

W_{i+4} = Y xor R_{con} xor Wi

R_{con} = Round constant (shown in TABLE I [1])

Round	Round Constant (hex)
1	01 00 00 00
2	02 00 00 00
3	04 00 00 00
4	08 00 00 00
5	10 00 00 00
6	20 00 00 00
7	40 00 00 00
8	80 00 00 00
9	1b 00 00 00
Final	36 00 00 00

TABLE I. AES Round Constant

C. BYTE SUBSTITUTION

Each byte of current 128-bit is substituted with corresponding entries in fixed table (S-BOX). AES-SOX consist of 256 elements (each entry is 8 bit). Rows and columns of S-BOX are represented using hexadecimal (0 to F) numbers. The least significant nibble of the number to be substituted determines the column, and the highest nibble determines the row of the S-BOX. For example, the number 0x9A is substituted with 0xB8 in S-BOX.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08
C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

TABLE II. AES SBOX [2]

D. SHIFTROWS

In this step, four rows of matrix are shifted to the left. Those entries that 'fall-off' are re-inserted on the right side of row.

Shifting is carried out in a way as follows-

- First row is not shifted
- Second row is shifted one (byte) positions to the left
- Third row is shifted two (byte) positions to the left
- Fourth row is shifted three (byte) positions to the left

E. MIXCOLUMNS

Special mathematical function is used for transforming each column of 128-bit. Mix column step is not performed in the last round.

$$\begin{bmatrix} 0x02 & 0x03 & 0x01 & 0x01 \\ 0x01 & 0x02 & 0x03 & 0x01 \\ 0x01 & 0x01 & 0x02 & 0x03 \\ 0x03 & 0x01 & 0x01 & 0x02 \end{bmatrix}$$

The above matrix is multiplied with the current 128-bit (output after shift row operation). Here the multiplication is carried out in Galois field. Suppose the current 128-bit is represented a matrix S,

$$\begin{bmatrix} S[127:120] & S[95:88] & S[63:56] & S[31:24] \\ S[119:112] & S[87:80] & S[55:48] & S[23:16] \\ S[111:104] & S[79:72] & S[47:40] & S[15:8] \\ S[103:96] & S[71:64] & S[39:32] & S[7:0] \end{bmatrix}$$

 DOI : 10.18231/2454-9150.2018.0365

In Galois field the multiplication of above two matrix look like as follows,

([(0x02)*(S [127:120])]%283) ⊕ ([(0x03)*(S[119:112])]% 283) ⊕ .....

III. PROPOSED METHOD

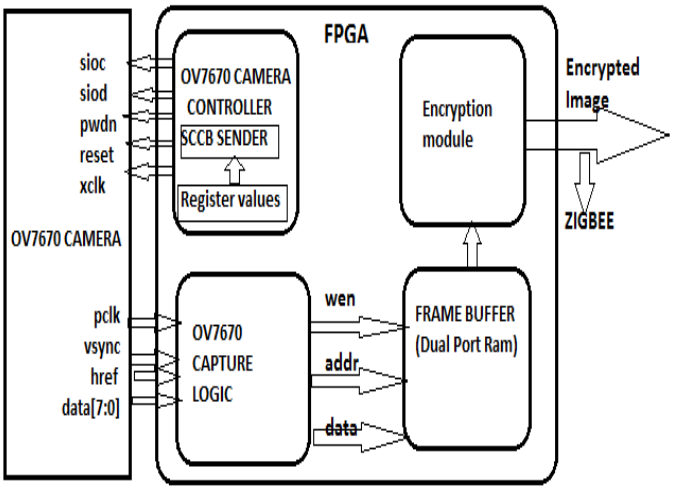

Fig.4 Real time encryption module

FPGA platform is used for implementation of this project. When OV7670 camera is correctly configured, it outputs the pixel data of image that captured. This pixel data is stored in dual port RAM of FPGA, which is then encrypted using AES algorithm. And the encrypted image thus obtained is transmitted via ZIGBEE.

A. REAL TIME IMAGE CAPTURING

In this work, image that is taken in real time is encrypted, which is then transmitted through wireless media. OV7670 CMOS camera is compatible with FPGA, and it captures the real time image. OV7670 provides wide range of image formats, controlled through Serial Camera Control Bus (SCCB) interface.

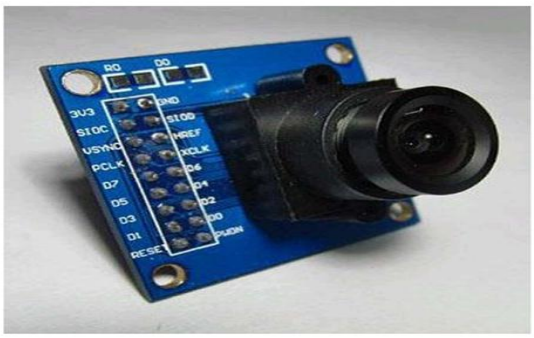

Fig.5 OV7670 camera

The camera module is powered from a single +3.3V power supply and it will only operate as slave device [6]. With proper configuration of camera module via SCCB bus, camera outputs pixel clock (PCLK), and camera data (8 bits) back to master (FPGA). Image format used in this project is RGB565 i.e., 1 pixel corresponds to two consecutive bytes (First byte- R [4:0] G [5:3], Second byte- G [2:0] B [4:0]

Pin	Type	Description
VDD	Supply	Power supply
GND	Supply	Ground level
SDIOC	Input	SCCB clock
SDIOD	Input/output	SCCB data
VSYNC	Output	Vertical synchronization
HREF	Output	Horizontal synchronization
PCLK	Output	Pixel clock
XCLK	Input	System clock
D0-D7	Output	Video parallel output
RESET	Input	Reset (Active low)
PWDN	Input	Power down (Active high)

TABLE III. OV7670 Camera module pins

B. SPARTAN6 FPGA

FPGAs are field programmable, it can configure and reconfigure by the customer using any HDL. The vital requirement of most of the image processing system is high speed of processing. This work meets this requirement through the use of SPARTAN6 FPGA, since it has an advantage of parallel processing.

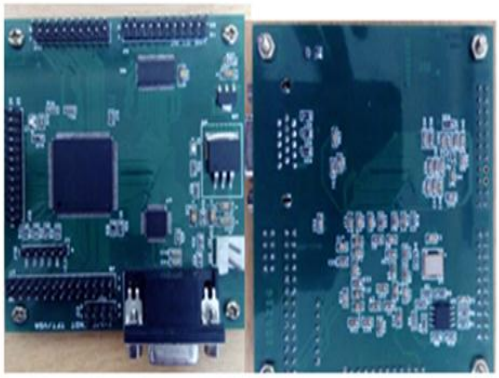

Fig.6 Spartan6 FPGA board

PCB board is designed as four layer board (2 signal layers, power layer and ground layer) for meeting EMC criteria.

C. ZIGBEE

In this work ZIGBEE transceiver is used for wireless transmission. This module is power and cost effective. It operates at 9600 baud rate. Encrypted pixel data is send via ZIGBEE at a baud rate of 9600.

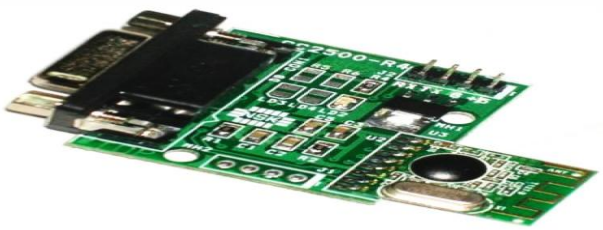

Fig.7 ZIGBEE transceiver

 DOI : 10.18231/2454-9150.2018.0365

IV. EXPERIMENTAL SETUP

Fig.8 shows the experimental set up for the proposed system, depicting camera module, encryption module, ZIGBEE transceiver module, and VGA display.

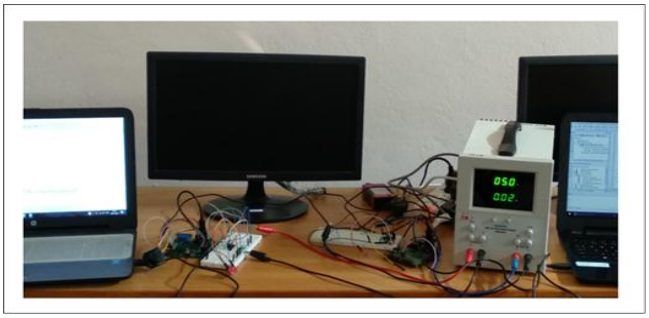

Fig.8 Experimental Setup

V. RESULTS AND FINDINGS

Comparison between AES and other techniques including the DES and 3DES are summarized in TABLE IV.

PARAMETER	DES	3DES	AES
Key length (bits)	64	168,112	128,192,256
Block size (bits)	64	64	128
Level of security	Adequate security	Adequate security	Excellent
Encryption speed	Very slow	Very slow	Faster
Rounds	16	48	10,12,14

TABLE IV. Comparison

A. SIMULATION RESULT FOR 128-BIT PLAIN TEXT

Plain Text = 12aabb22334455667788990aabbccee

Cipher Key = aabbccddeeff12345678901234567890

Encryption output = 3ac215d1f6d1f25e2e8ac48508f73072 (Cipher Text)

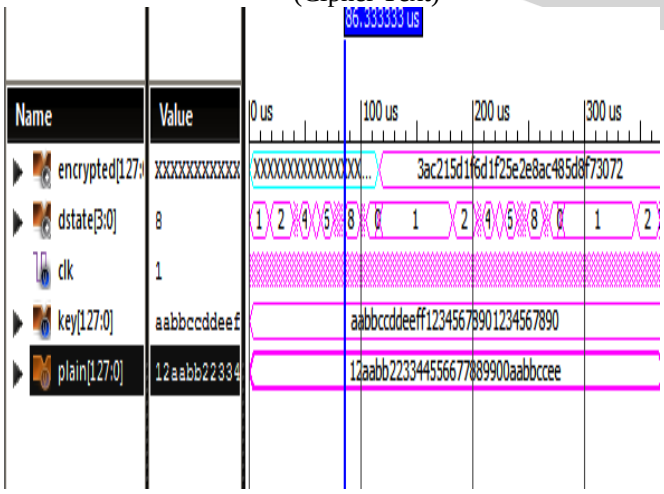

Fig.9 Simulation result

Simulation results for AES encryption is shown in Fig.9. 10 rounds of encryption are shown using the variable 'dstate' in the simulation.

Fig.10 Final output

Final output of the proposed system is shown in Fig.10. For encrypting an image, AES is done in each pixel data. In this work, the original image (taken using OV7670 camera) is a 32 x 32 pixel image. Each pixel representing 16 bits (RGB565 format) i.e., 5 bits for Red, 6 bits for Green and 5 bits for Blue. Therefore 8 pixels (8*16=128 bits) form 128 bit AES input.

VI. CONCLUSION

The Image Encryption using AES algorithm was successfully implemented in Spartan6 FPGA using Verilog coding. Thereby image data can be secured from an unauthorized access during communication, data storage and transmission. Symmetric key AES algorithm is one of the best encryption standard available in market. It is observed that this algorithm have extremely large security and can withstand most common attacks such as the brute force attack, cipher attacks and plaintext attacks.

REFERENCES

[1] William Stallings, "Advance Encryption Standard, in Cryptography and Network Security", 4th Ed., India: PEARSON,pp. 134–165.

[2] Yewale Minal J, M. A. Sayyad, "Implementation of AES on FPGA", IOSR Journal of VLSI and Signal Processing, 2016.

[3] Sayali S. Kshirsagar, Savita Pawar, "Encryption and Decryption by AES algorithm using FPGA", IJETEMR, volume 2, issue 2-June 2016.

[4] Jaydeep Rusia "RF based wireless data transmission between two FPGA's" IEEE Conference paper, November 2016.

[5] Priya Deshmukh "An image encryption and decryption using AES Algorithm" International Journal of Scientific & Engineering Research, Volume 7, Issue 2, February 2016.

[6] OmniVision, "OV7670 camera implementation guide," OV7670 datasheet, Sept.2005.

[7] Priya Deshmukh "An image encryption and decryption using AES Algorithm" International Journal of Scientific & Engineering Research, Volume 7, Issue 2, February 2016.

[8] Jaydeep Rusia "RF based wireless data transmission between two FPGA's" IEEE Conference paper, November 2016.

 DOI : 10.18231/2454-9150.2018.0365