International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2015): 6.391

# FPGA Implementation of a Image Encryption System using AES Algorithm

**Dr. G. V. R. Sagar[1], G. Ashok Kumar[2]**

G. Pulla Reddy Engineering College

**Abstract:** ***Digital image typically has to be keep and processed in an encrypted format to take care of security and privacy. For the aim of content notation and/or meddling detection, it's necessary to perform data concealment in these encrypted images. During this approach, data concealment in encrypted domain while not decryption preserves the confidentiality of the content. Additionally, it's a lot of efficient while not decryption followed by data concealment and re-encryption. In this paper we tend to implement the image encryption system using AES encryption and decryption algorithmic program. This algorithmic program was implemented using micro blaze Processor on Spartan3EDK (XC3S200) FPGA in Hardware and software co-design environment using Xilinx platform studio and synthesis results show that area consumption is low.***

**Keywords:** AES, Image Encryption, Micro blaze, FPGA, XPS, C

## 1. Introduction

Since the increase of the web one in all the foremost necessary factors of information technology and communication has been the safety of information. Cryptography was created as a way for securing the secrecy of communication and plenty of completely different strategies are developed to encrypt and decrypt information so as to stay the message secret. A system or product that gives encryption and decryption is named cryptosystem. Cryptosystem uses an encryption algorithms that determines however straightforward or advanced the encryption process are going to be, the mandatory software part, and also the key (usually an extended string of bits), that works with the algorithm to encrypt and decrypt the data [2] [3].Research work by S.Sau [4], C. D. Walter[5] A Mazzeo[6] shows FPGA primarily based embedded system became a platform for the implementation of cryptographic algorithms, which require sizable amount of bit-level operations, which may be done expeditiously on FPGA. In this paper implementation of cryptosystem for image using AES algorithm in an FPGA in hardware and software co-design environment is finished using Xilinx platform studio. The paper is organized in the following way section 2 of discusses regarding the proposed diagram wherever the steps in design process are explained. Next in section 3 AES algorithm is introduced and explained varied steps concerned in encryption method, In section 4 AES decryption is introduced that is the inverse method of encryption and in section 5 implementation details are present wherever we tend to used Spartan 3EDK (XCS200) FPGA for implementing our image cryptosystem in Hardware and software system co-design environment using Xilinx platform studio and results conjointly presented during this section, and finally concluded the paper.

## 2. Proposed Block Diagram

Image Encryption system to secure the data from the hackers in the channel and at receiver side retrieves the original image. In order to encrypt the pixel values here we used symmetric encryption technique i.e. AES algorithm. Images are converted to the header File since we are going to implement this on the XPS (Xilinx Plat Form Studio) in software environment. The header file of the image and Source C code file of AES algorithm are applied to XPS results binary file contains the application and the bit stream of hardware system developed both binary and bit stream files were configured and dumped into the Spartan 3EDK (XC3S200) FPGA.

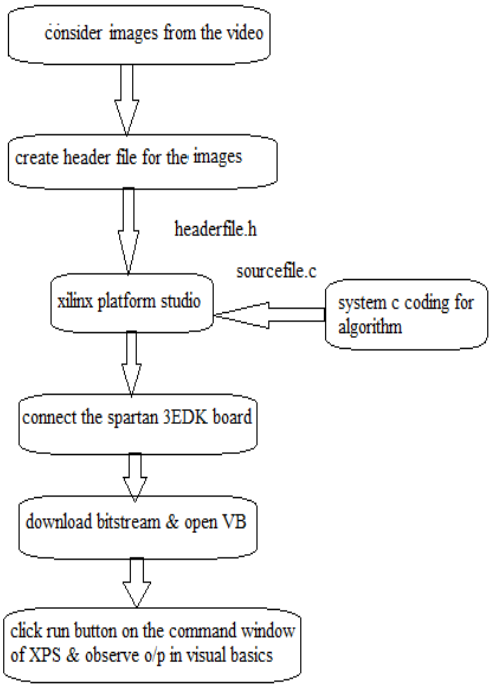


**Figure 1:** Design process block diagram

The proposed algorithm was based on Image encryption based on AES technique where key length is 128,192 and 256.

## 3. AES Algorithm

AES is an interchangeable block cipher with a block size of 128 bits. Key lengths can be 128 bits, 192 bits, or 256 bits [7] called AES-128, AES-192, and AES-256, respectively. AES- 128 uses 10 rounds, AES-192 uses 12 rounds, and AES-256 uses 14 rounds. Fig.2 depicts the AES image encryption and decryption process flow.

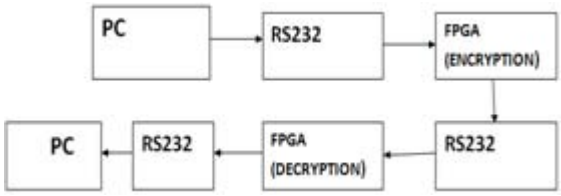


**Figure 2:** AES Image Encryption and Decryption

The main loop of AES performs the subsequent functions:
- **SubBytes ()**
- **ShiftRows ()**
- **MixColumns ()**
- **AddRoundKey()**

The first 3 functions of an AES [8] spherical square measure designed to thwart cryptology via the strategies of "confusion" and "diffusion." The fourth perform truly encrypts the info. Engineer represented the ideas of confusion and diffusion in his seminal 1949 paper, "Communication Theory of Secrecy Systems." "Two strategies recommend themselves for frustrating an applied mathematics analysis. These we tend to could decision the strategies of diffusion and confusion."[9]. Diffusion suggests that patterns inside the plaintext are unfolded inside the cipher text. Confusion suggests that the affiliation between the plaintext and so the cipher text is obscured.

A simpler way to view the AES function order is:
1) Scramble each byte (SubBytes).
2) Scramble each row (Shift Rows).
3) Scramble each column (Mix Columns).
4) Encrypt (Add Round Key).

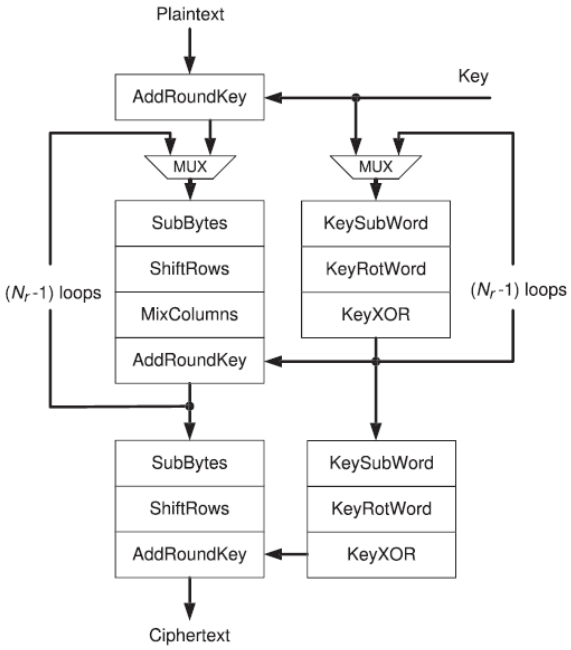


**Figure 3:** Basic Block of AES

### A. Sub Bytes ()
The SubBytes() transformation is a non-linear 8bit substitution that operates independently on every 8bit of the State employing a substitution table (S-box). This S-box as shown in Fig.4 that is invertible is made by composing double transformations:

1) Take the multiplicative inverse in the finite field GF(2^8), described in Sec. 2.2.4.2; the element {00} is mapped to itself.
2) Apply the following affine transformation (over GF(2) ):

| x \ y | 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 |

**Figure 4:** S-Box

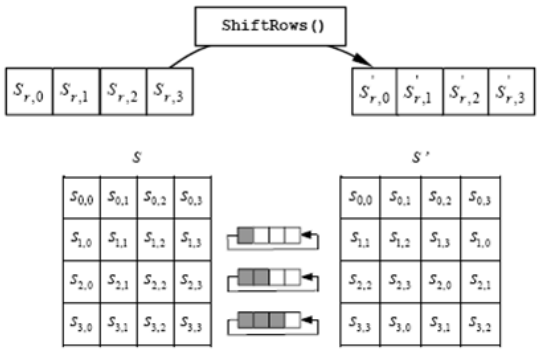


**Figure 5:** Shift Rows

### B. Shift Rows ()
ShiftRows() provides diffusion by mixing data within rows. Row zero of the State is not shifted, row 1 is shifted 1 byte, row 2 is shifted 2 bytes, and row 3 is shifted 3 bytes, as shown in the Federal Information Processing Standards Publications (FIPS) illustration in Fig.5

### C. Mix Columns ()
MixColumns() conjointly provides diffusion by admixture information at intervals columns. The four bytes of every column among the State are treated as a 4-byte vary and reworked to a distinct 4-byte vary via finite field arithmetic, as shown among the FIPS illustration that follows Fig.6.

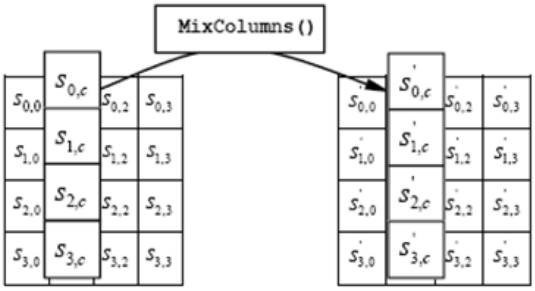


**Figure 6:** Mix Columns

### D. Add_Round_Key ()
The actual 'encryption' is performed within the AddRoundKey() operate, once every 8bit within the State is XORed with the sub key. The subkey comes from the key in line with a key enlargement schedule, as shown within the FIPS illustration that follows as shown in Fig.7.

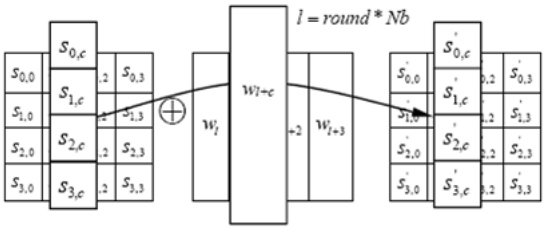


**Figure 7:** Add Round Key

## 4. AES Decryption

Decryption occurs through the function AddRound Key(), plus the inverse AES functions InvShiftRows(), InvSubBytes(), and InvMixColumns(). AddRoundKey() does not require an inverse function, as it simply XORs the state with the sub key (XOR encrypts when applied once, and decrypts when applied again).

## 5. Implementation

The Field Programmable Gate Array is majorly used for generation ASIC IC's to the computations. They offer more speed in execution process. SO, for generation ASIC IC's FPGA's are majorly used.

**Table 1:** Spartan3EDK Configuration

| Property Name | Value |
|---|---|
| Family | Spartan 3 |
| Device | XC3S200 |
| Package | TQG144 |
| Speed Grade | -4 |

### A. Xilinx Platform Studio

The Xilinx Platform Studio (XPS) is that the development atmosphere used for creating the hardware portion of embedded processor system. Xilinx Embedded Development Kit (EDK) is associate integrated computer code tool suite for developing embedded systems with Xilinx MicroBlaze and PowerPC CPUs. EDK includes an expansion of tools associated applications to assist the designer to develop associate embedded system right from the hardware creation to final implementation of the system on an FPGA. System vogue consists of the creation of the hardware and software system parts of the embedded processor system and conjointly the creation of a verification component is elective.

A typical embedded system vogue project involves hardware creation and its verification, soft-ware creation, application creation, and its verification. Base System Builder is that the wizard that's wont to auto generates a hardware platform keep with the user specifications that's defined by the MHS (Microprocessor Hardware Specification) file. The MHS file defines the system design, peripherals and embedded processors. The Platform Generation tool creates the hardware platform uses the MHS file as input.

The software platform is outlined by MSS abbreviated as Microprocessor Software Specification file that defines driver and library customizable parameters for peripherals, processor customizable parameters, custom anyone hundred ten devices, interrupt handler routines, and different software system connected routines. The MSS file is associate input to the Library Generator tool for personalization of drivers, libraries and interrupts handlers.

The creation of the verification platform is facultative and is predicated on the hardware platform. The MHS file is taken as Associate in Nursing input by the Siegen tool to make simulation files for a particular machine. 3varieties of simulation models will be generated by the Siegen tool: behavioral, structural and temporal arrangement models. Another helpful tool on the market in EDK are Platform Studio that provides the GUI for making the MHS and MSS files. Produce / Import IP Wizard that permits the creation of the designer's own peripheral and imports them into EDK come. Platform Generator customizes and generates the processor system within the sort of hardware net lists. Library Generator tool configures libraries, device drivers, file systems and interrupt handlers for embedded processor system. Bit stream Initialize tool initializes the instruction memory of processors on the FPGA shown in figure2.

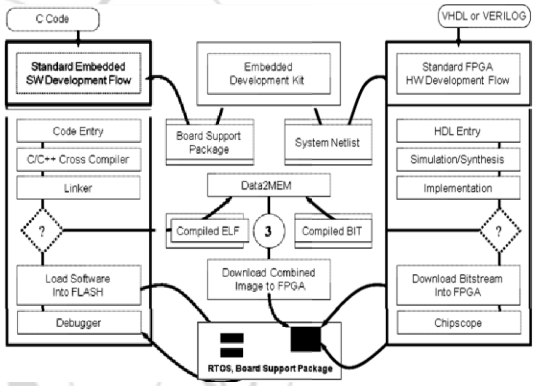


**Figure 8:** Embedded Development Kit Design Flow

Antelope Compiler tools are used for assortment and linking application executable for each processor inside the system. There are a pair of decisions on the marketplace for debugging the appliance created victimization EDK namely Xilinx micro chip debug (XMD) for debugging the appliance package using a small micro chip debug Module (MDM) inside the embedded processor system, and package coder that invokes the package programmer appreciate the compiler obtaining used for the processor. C. Xilinx Platform software Development Kit (SDK) is integrated development atmosphere, complimentary to XPS, that is used for C/C++ embedded package application creation and verification. SDK is formed on the Eclipse open source framework. Soft Development Kit (SDK) is also a set of tools that enables you to vogue a package application for elite Soft IP Cores inside the Xilinx Embedded Development Kit (EDK). The package applications are written throughout in "C or C++" then the complete embedded processor system for user application is completed, else debug & download the bit file into FPGA. Then FPGA behaves like processor implemented on it in a Xilinx Field Programmable Gate Array (FPGA) device.

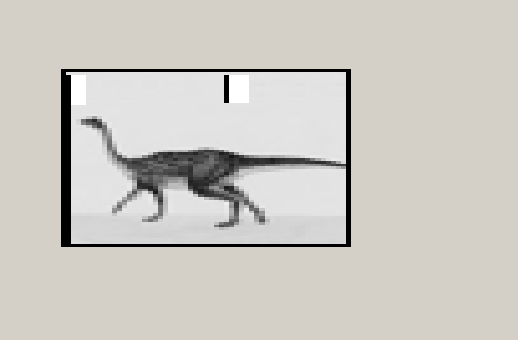

**Figure 9(a):** Original Image

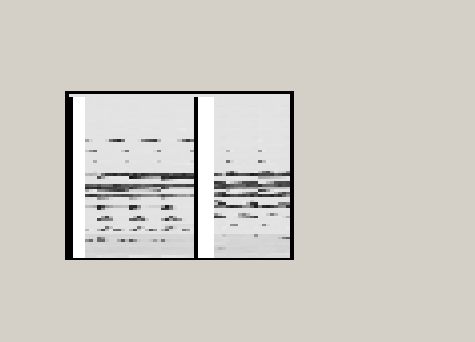

**Figure 9 (b):** Encrypted Image

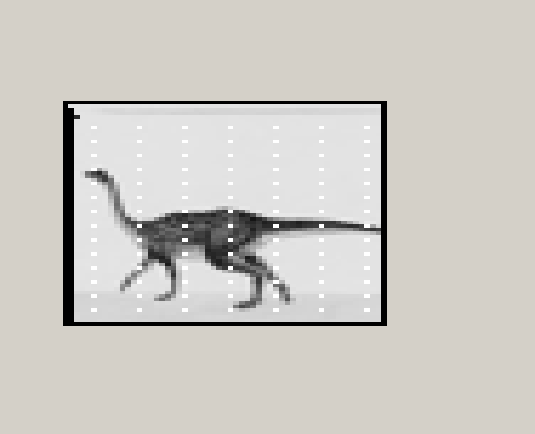

**Figure 9 (c):** Decrypted Image

```
Device utilization summary:
---------------------------

Selected Device : 3s200tq144-4

 Number of Slices:                  1880  out of   1920    97%
 Number of Slice Flip Flops:        2118  out of   3840    55%
 Number of 4 input LUTs:            2971  out of   3840    77%
    Number used as logic:           2418
    Number used as Shift registers:  297
    Number used as RAMs:             256
 Number of IOs:                       62
 Number of bonded IOBs:               62  out of     97    63%
    IOB Flip Flops:                   64
 Number of BRAMs:                      4  out of     12    33%
 Number of MULT18X18s:                 3  out of     12    25%
 Number of GCLKs:                      4  out of      8    50%
 Number of DCMs:                       1  out of      4    25%
```

**Figure 10:** XPS Synthesis report

**Table 2:** Comparison with existing works

| Design | Device | Frequency MHz | Slices | BRAMS |
|---|---|---|---|---|
| Elbirt etal[10] | XCV1000-4 | 31.8 | 10992 | |
| M.McLoone etal[11] | XCV812e-8 | 93.9 | 2000 | 244 |
| K.U.Jarvinen eta[12] | XCV1000e-8 | 129.2 | 11719 | 0 |
| G.P.Saggese[13] | XCV2000e-8 | 158 | 5810 | 0 |
| F.Standaert[14] | XCV3200e-8 | 154 | 15112 | 0 |
| Rourab Paul1[15] | Spartan3e Xc3s500e | 50 | 2495 | 320 |
| Proposed | Spartan 3EDK XC3S200 | 50 | 1880 | 4 |

## 6. Conclusion

In this paper AES algorithm was implemented using FPGA. This system works on micro Blaze architecture of Spartan3 EDK. On the other hand, synthesis results show that area consumption is low, victimization merely 100 percent of logic circuits of FPGA for AES, allowing the implementation of this methodology over cheap FPGAs. The key dimensions are often varied with to a small degree modification within the algorithm. This work can be extended by the encryption and decryption of data wherever inputs are audio, video data returning from completely different multimedia system applications performed over FPGA. Usage of single FPGA with twin processor implementation, wherever one processor can execute the algorithm whereas different one are chargeable for input file acquisition, so the executing processor will handle the algorithm with none interruption, will be a decent step within the world of hardware design.

## References

[1] "Supplemental Streaming SIMD Extensions 3," http://en.wikipedia.org/wiki/SSSE3, 2012.
[2] Kessler, Gary C., (1998). An Overview of Cryptography, available from: http://www.garykessler.net/library/crypto.html#intro. (Accessed December 28, 2008).
[3] B. White, Gregory, (2003). Cisco Security+ Certification: Exam Guide, McGraw-Hill.
[4] S. Sau , C. Pal and A Chakrabarti "Design and Implementation of Real Time Secured RS232Link for Multiple FPGA Communication, Proc. Of International Conference on Communication, Computing & Security,20 1 I,ISBN - 978- 1-4503-0464- 1.
[5] C. D. Walter. August 1999. Montgomery's Multiplication Technique: How to Make It Smaller and Faster. Cryptographic Hardware and Embedded Systems, Lecture Notes in Computer Science, Springer. No. 17 17. pp. 80-93.
[6] A Mazzeo, L. Romano, G. P. Saggese and N. Mazzocca. 2003. FPGABased Implementation of a Serial RSA Processor. Design. Proceedings of the conference on Design, Automation and Test in Europe - Volume I. ISBN:O- 7695- 1870-2 .
[7] M. Matsui and J. Nakajima, "On the Power of Bitslice Implementation on Intel Core 2 Processor.

[8] Ahmad, Hasan, R.; Jubadi, W.M; "Design of AES S-Box using combinational logic optimization", IEEE Symposium on Industrial Electronics & Applications (ISIEA), pp. 696-699, 2010.
[9] Granado-Criado, M. Vega-Rodriguez, J. Sanchez-Perez, and J. Gomez-Pulido, "A New Methodology to Implement the AES Algorithm Using Partial and Dynamic Reconfiguration," Integration,the VLSI J., vol. 43, no. 1, pp. 72-80, 2010.
[10] A. J. Elbirt, W. Yip, B. Chetwynd, and C. Paar. An FPGA implementation and performance evaluation of the AES block cipher candidate algorithm finalist. presented at *Proc. 3rd AES Conf. (AES3)*[Online].Available:http://csrc.nist.gov/encryption/aes/round2/conf3/aes3papers.html.
[11] M. McLoone and J. V. McCanny, "Rijndael FPGA implementation utilizing look-up tables," in *IEEEWorkshop on Signal Processing Systems*, Sept. 2001, pp. 349–360.
[12] K. U. Jarvinen, M. T. Tommiska, and J. O. Skytta, "A fully pipelined memoryless 17.8 Gbps AES-128 encryptor," in *Proc. Int. Symp. Field-Programmable Gate Arrays (FPGA 2003)*, Monterey, CA, Feb. 2003, pp. 207–215.
[13] G. P. Saggese, A. Mazzeo, N. Mazocca, and A. G. M. Strollo, "An FPGAbased performance analysis of the unrolling, tiling and pipelining of the AESalgorithm," in *Proc. FPL 2003*, Portugal, Sept. 2003.
[14] F. Standaert, G. Rouvroy, J. Quisquater, and J. Legat,"Efficientimplementation of Rijndael encryptionin reconfigurable hardware:Improvements & design tradeoffs," in *Proc. CHES 2003*, Cologne, Germany, Sept. 2003.
[15] Rourab Paul Design and implementation of real time aes-128 on real time operating system for multiple fpga communication