Research Journal of Applied Sciences, Engineering and Technology 8(12): 1424-1428, 2014
DOI:10.19026/rjaset.8.1116
ISSN: 2040-7459; e-ISSN: 2040-7467

Submitted: May 31, 2014 Accepted: July 01, 2014 Published: September 25, 2014

Research Article

Incorporation of Composite Field S-box into AES-CBC and AES-CM Modes to Avoid SEUs

[1]K. Sandyarani and [2]P. Nirmal Kumar
[1]Sathyabama University, Chennai, India
[2]Department of ECE, College of Engineering, Guindy, Chennai, India

Abstract: This study is troubled with the use of commercial security algorithms like the Advanced Encryption Standard (AES) in Earth observation minute satellites. Stipulate to secure the sensitive and valuable data transmitted from satellites to ground has increased and hence the need to use encryption aboard. AES, which is well-liked choice in terrestrial memorandums, is gradually budding as the preferred option in the aerospace industry including satellites. This research study initially addresses the encryption of satellite imaging data using one of the AES modes-CBC. An exhaustive analysis of the effect of Single Even Upsets (SEUs) on imaging data at some stage on-board encryption is carried out. Collision of faults in the data occurs during transmission to ground due to noisy channels is also discussed. Though sensor network provides various capabilities, it is not possible to make sure the secure authentication between the sensor nodes, it causes the reduction in reliability of the entire network, The proposed design aims to implement the composite field S-Box into the AES-CBC and AES-CM modes.

Keywords: Advanced Encryption Standard (AES), Composite Field Arithmetic (CFA), S-box, secret key ciphers, security

INTRODUCTION

In our days, the need for secure transport protocols seems to be one of the most important issues in the communication standards. Of course, many encryption algorithms support the defense of private communications. However, the implementation of these algorithms is a complicated and difficult process and sometimes results in intolerant performance and allocated resources in hardware terms (Astarloa *et al*., 2005). Clarification for this truth is because these encryption algorithms were designed some years ago and for general cryptography reasons. In recent years, new flexible algorithms specially designed for the new protocols and applications have been introduced to face the increasing demand for cryptography. In October of 2000, the National Institute of Standards and Technology (NIST) announced the cipher Rijndael as the Advanced Encryption Standard (AES) in order to replace the aging Data Encryption Standard (DES) (Bergamaschi *et al*., 2001). The new algorithm is expected to be a standard by the summer of 2001. In this study, we describe the AES-128 CBC algorithm in the symmetric key encryption which is selected by default in sensor networks. And we measure the encryption and decryption performance on the 8-bit Microcontroller. Then, we analyze the Communication efficiency through the total delay per hop in sensor networks (Castillo *et al*., 2004). The main objective of this research work is to improve the security by incorporation of Composite S-Box into AES CBC and CM modes.

AES (Advanced Encryption Standard): The AES (Advanced Encryption Standard) is an encryption standard as a symmetric block cipher. It was announced by National Institute of Standards and Technology (NIST) as U.S. FIPS PUB 197 (FIPS 197) on November 26, 2001. The central design adoption of symmetry at different platforms and the efficiency of processing. After a 5-year standardization process, the NIST adopted the Rijndael algorithm as the AES. The AES operates on 128-bit blocks of data. The algorithm can encrypt and decrypt building blocks via covert keys. The key size can also be 192, 128 or 256 bit, respectively. The real key range depends on the desired security level. The dissimilar editions are most often represented as AES-128, AES-192 or AES-256. The cipher Rijndael consists of an initial Round Key addition, Nr-1 Rounds, a final round. It shows the pseudo C code of Rijndael algorithm (Hwang *et al*., 2006).

Rijndael (State, Expanded Key):

{Key Expansion (Cipher Key, Expanded Key); Addround Key (State, Expanded Key + Nb); For (i = 1; i<Nr; i++) Round (State, Expanded Key + Nb*i); Final Round (State, Expanded Key + Nb* Nr);}

Corresponding Author: K. Sandyarani, Sathyabama University, Chennai, India

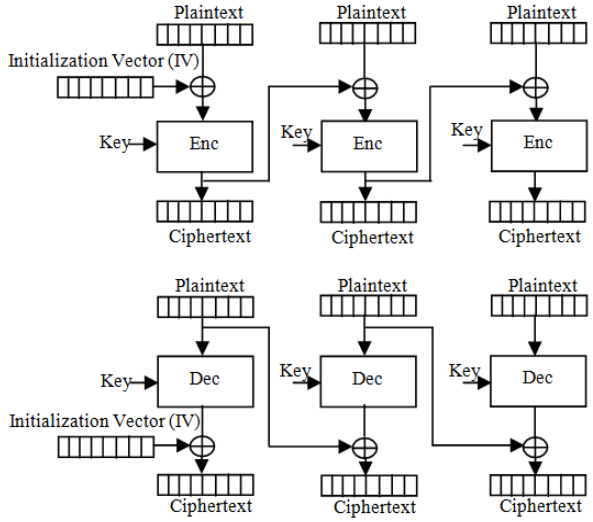

Fig. 1: Cipher block chaining mode encryption and decryption

The key expansion can be done on beforehand and Rijndael can be specified in terms of the Expanded Key. The Expanded Key shall always be derived from the Cipher Key and never be specified straightly.

There is still no constraint on the selection of the Cipher Key itself. It shows the pseudo C code of Rijndael's Expanded Key algorithm (Ichikawa *et al*., 2000):

Rijndael (State, Expanded Key) {Addround Key (State, Expanded Key)
For (i = 1; i<Nr; i++) Round (State, Expanded Key + Nb*i); Final Round (State, Expanded Key + Nb*Nr);}

CBC (Cipher Block Chaining) mode: The CBC (Cipher Block Chaining) uses feed back to feed the result of encryption back into the encryption of the next block (Kampen, 2003). In CBC mode, each block of plaintext is XORed with the previous cipher text architecture. This way, each cipher text block is dependent on all plaintext blocks processed up to that point (Elbirt *et al*., 2000). Also, to make each message unique, an IV (initialization vector) must be used in the first block. The IV does not have to be kept secret. The IV should be a random number to guarantee that all messages are encrypted uniquely (Verbauwhede *et al*., 2003). Cipher Block Chaining mode is shown in Fig. 1.

Counter Mode (CM): AES Counter Mode (CM) mode of operation overcomes those limitations with a different operation way (Astarloa *et al*., 2009). It does not directly use the AES cipher block to encrypt the data like ECB or CBC do; On the contrary, it encrypts an arbitrary value called the counter' and then XORs the result with the plain data to produce the ciphered text. The counter value is typically incremented by 1 for each successive block processed. Counter Mode AES operation is shown in Fig. 2. The message is divided into 128-bit vectors; each of these vectors is XORed with the result of encrypting the counter value correspondent to that block using an AES cipher block (Fu *et al*., 2005). In this case, the counter founds at 1 and increments by one up to 4 and the process crypts 512 bits in parallel. The receiver, which decrypts using the similar circuit, must recognize the initial value of the counter and how it goes forwards.

PROPOSED IMPLEMENTATION OF THE SUBBYTES TRANSFORMATION

The main contributions of this study can be summarized as follows. This study avoids use of LUTs and proposes use of composite field data path for the SubBytes and InvSubBytes transformations. Composite field arithmetic has been employed to design efficient data paths. However, the design in decomposes the inversion in the SubBytes and InvSubBytes. The proposed architecture decomposes the inversion and inversion is implemented by a novel approach, which leads to a more efficient architecture with shorter critical path and smaller area. Implementation of Composite S-Box transformation is shown in Fig. 3.

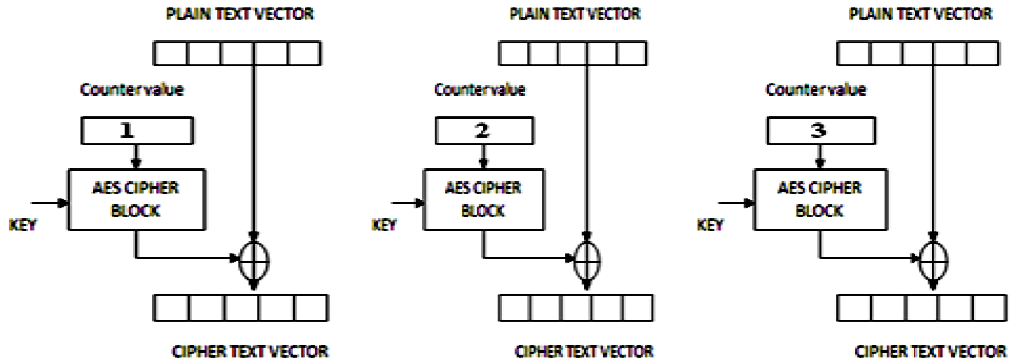

Fig. 2: AES-CM mode of operation block diagram

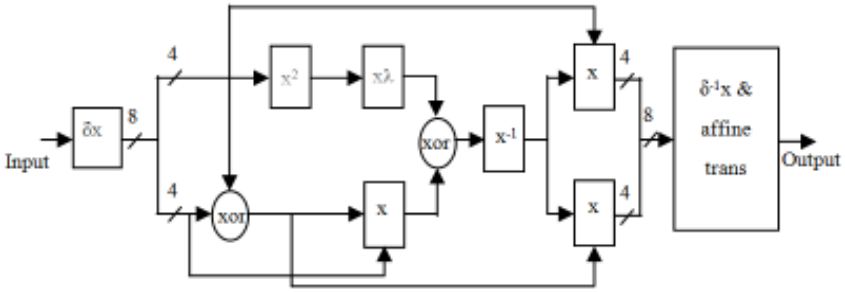

Fig. 3: Implementation of SubBytes transformation

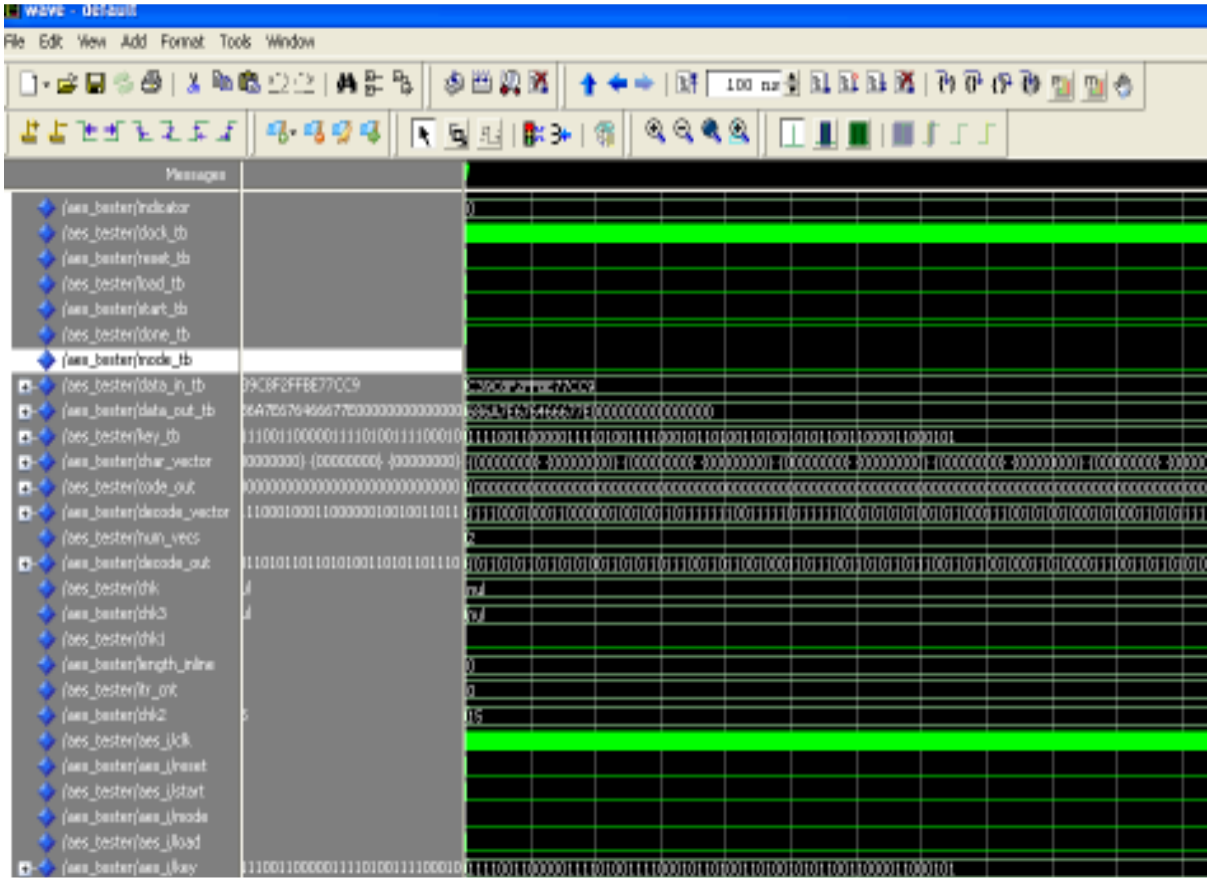

Fig. 4: Simulation result of AES encryption

Composite field arithmetic: The non-LUT-based implementations of the AES algorithm able to exploit the advantage of sub pipelining further. Nevertheless, these approaches may have high hardware complexities. Although two Galois Fields of the same order are isomorphic; the complexity of the field operations may heavily depend on the representations of the field elements. Composite field arithmetic can be employed to reduce the hardware complexity. We have to implement this composite S-Box into the AES-CM and AES-CBC and getting the area and delay.

Detailed hardware implementation architectures: In this section, we present detailed architectures for each of the nontrivial transformations in the AES algorithm. The implementation of each transformation is optimized to reduce area and increase speed. Meanwhile, efficient key expansion architecture suitable for round units is proposed. Based on the analysis on the gate counts in the critical path of the round units and the key expansion, of the AES algorithm are presented.

IMPLEMENTATION RESULTS

The implementation has been developed in VHDL and synthesized to Xilinx FPGA devices. The synthesis process to be done in FPGA with the help of Xilinx

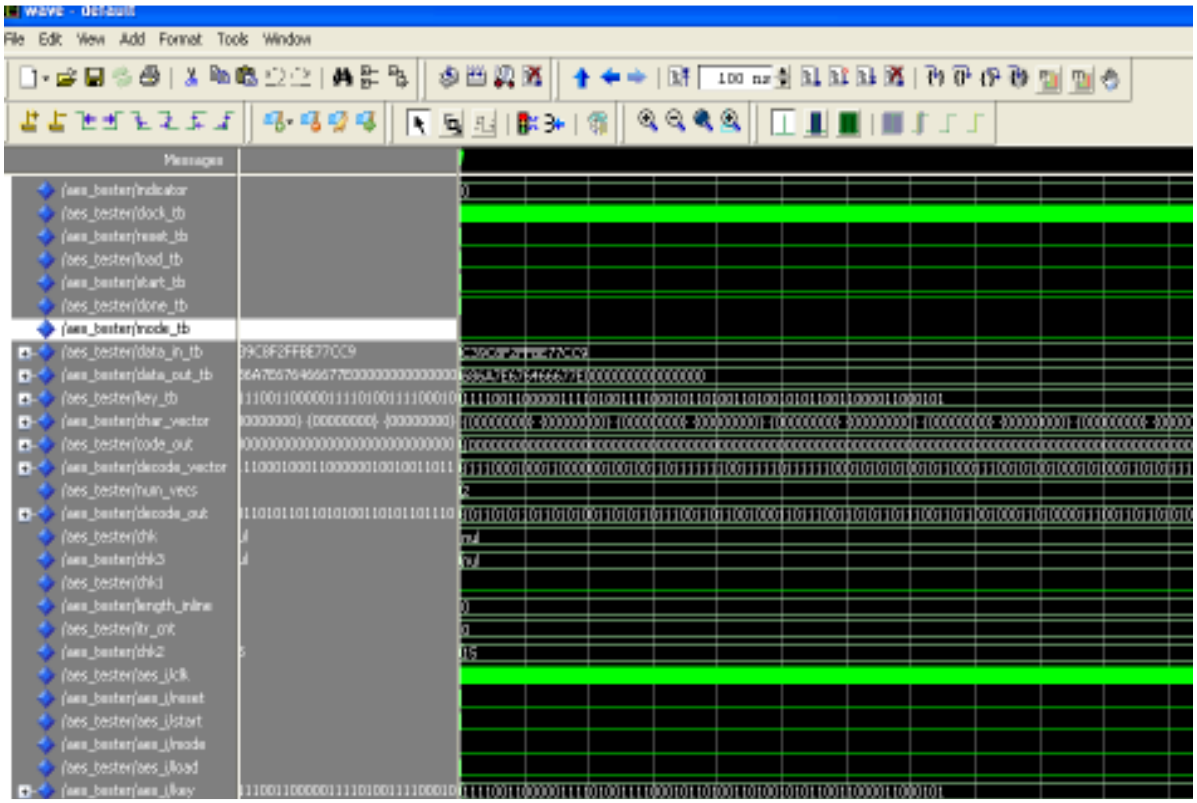

Fig. 5: Simulation result of AES decryption

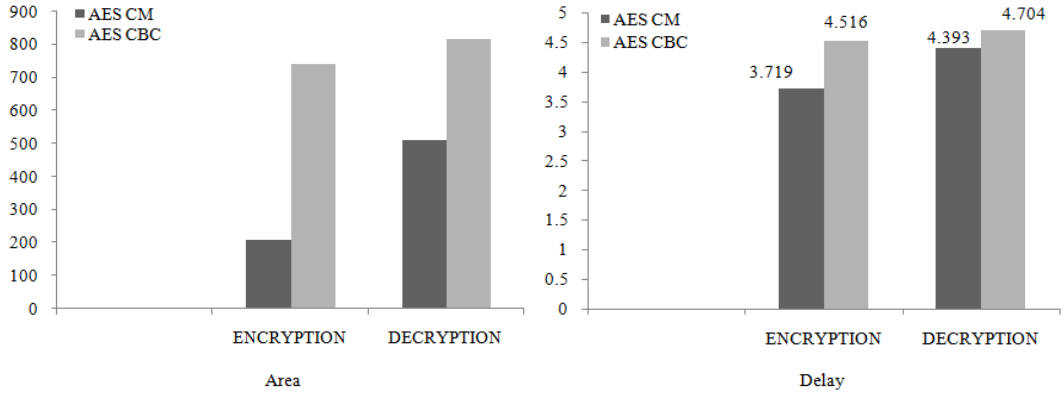

Fig. 6: Graphical representation of simulated results for Table 1 as shown below

Table 1: Comparison of delay and area of AES-modes

Methods	Process	Area	Delay (nsec)
AES CM	ENC/DEC	208/512	3.719/4.393
AES CBC	ENC/DEC	741/819	4.516/4.704

Device: xc5vlx50-3ff1153

software tool 10.1. Simulation is done with the help of ModelSim XE III 6.3.

In this project Advanced Encryption Standard (AES) algorithm was developed using VHDL language. The coding to be tested in both simulation and synthesis. Simulation the ModelSim XE III 6.3 to use for analyzing and for synthesis the Xilinx ISE tool to be used. The Figure 4 shows the simulation result of the encryption module and the Fig. 5 shows the decryption results. Performance analysis between AES CBC and CM mode are given in Fig. 6.

CONCLUSION

The Advanced Encryption Standard algorithm is an iterative private key 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, respectively. The original message is taken to 10 round operations which produces the cipher text. Using inverse transformations the cipher text is decrypted. An efficient

FPGA implementation of 128 bit block and128 bit key AES cryptosystem has been presented. Due to the new mix column architecture (which is nothing accordance with the original concept), this design consumes less power and occupies less chip area. Optimized and Synthesizable Verilog code is developed for the implementation of both 128 bit data encryption and decryption process is verified using ISE 10.1 functional simulator from Xilinx. Compared to previous designs, we achieved significantly higher throughput with corresponding area.

REFERENCES

Astarloa, A., U. Bidarte, J. Lazaro, A. Zuloaga and J. Arias, 2005. Multiprocessor SoPC-Core for FAT volume computation. Microprocess. Microsy., 29(10): 421-434.

Astarloa, A., A. Zuloaga, J. Lázaro, J. Jiménez and C. Cuadrado, 2009. Scalable 128-bit AES-CM crypto-core reconfigurable implementation for secure communications. Proceeding of the Applied Electronics International Conference (AE' 09). Czech Republic, Pilsen, pp: 37-42.

Bergamaschi, R.A., S. Bhattacharya, R. Wagner, C. Fellenz and M. Muhlada, 2001. Automating the design of SOCs using cores. IEEE Des. Test Comput., 18(5): 32-45.

Castillo, J., P. Huerta and J. Martínez, 2004. SystemC design flow for an AES/DES cryptoProcessor. WSEAS T. Inform. Sci. Appl., pp: 193-198.

Elbirt, A., W. Yip, B. Chetwynd and C. Paar, 2000. An FPGA implementation and performance evaluation of the AES block cipher candidate algorithm finalists. Proceeding of the 3rd Advanced Encryption Standard (AES) Candidate Conference, pp: 13-27.

Fu, Y., L. Hao and X. Zhang, 2005. Design of an extremely high performance counter mode AES reconfigurable processor. Proceeding of the 2nd International Conference on Embedded Software and Systems (ICESS'05), pp: 262-268.

Hwang, D.D., P. Schaumont, K. Tiri and I. Verbauwhede, 2006. Securing embedded systems. IEEE Secur. Priv., 4(2): 40-49.

Ichikawa, T., T. Kasuya and M. Matsui, 2000. Hardware evaluation of the AES finalists. Proceeding of the 3rd Advanced Encryption Standard (AES) Candidate Conference, pp: 297-285.

Kampen, H.V., 2003. Pico Blaze Rijndael (AES-128) Block Cipher. Retrieved form: http://www.mediatronix.com.

Verbauwhede, I., P. Schaumont and H. Kuo, 2003. Design and performance testing of a 2.29-GB/s rijndael processor. IEEE J. Solid-St. Circ., 38(3): 569-572.