Design and simulation of AES algorithm- Encryption using VHDL

Mital Maheta
Student
Wireless Communication Technology (E.C), Gujarat Technological University, Visnagar
Gujarat-384315, India

Abstract: The Cryptographic Algorithm is most widely used throughout the world for protecting information. Cryptography is the art of secret writing, followed by the guarantee to authenticate data and important messages and protect the systems from valid attacks. It comprises of encryption and decryption process each associated with a key which is supposed to be kept secret. We have implement RC6 Algorithm. This is considered as a secured and elegant choice for AES due to its simplicity, security, performance and efficiency. RC6 supports 32 bit and 64 bit processing. An eight step algorithm is used to encipher the 64 bit plain text block. The encrypted data is then decrypted by performing the reverse algorithm on the same. This paper introduces encryption of RC6 using Xilinx 8.1i.

Index Terms— AES Algorithm, RC6, Encryption, Cryptography, Cipher text.

I. INTRODUCTION

In the recent years, we have witnessed the increasing deployment of applications with a crucial need for security functions such as confidentiality, authentication, non-repudiation and time-stamping. These include for example e-commerce, secure e-mail, e-banking and other security functions. A cryptographic algorithm works with a key — a word, number, or phrase — to encrypt the plain text. The same plaintext encrypts to different cipher text using different keys. The security of encrypted data is mainly depends on two things: the strength of the cryptographic algorithm and the secrecy of the key. A cryptographic algorithm is all possible keys and all the protocols that make it work comprise a cryptosystem. In conventional cryptography, also called *secret-key* or *symmetric-key* encryption, single key is used both for encryption and decryption. In asymmetric cryptography, the encryption and decryption keys are different on both the sides. We have implement RC6 Algorithm.

Why RC6?

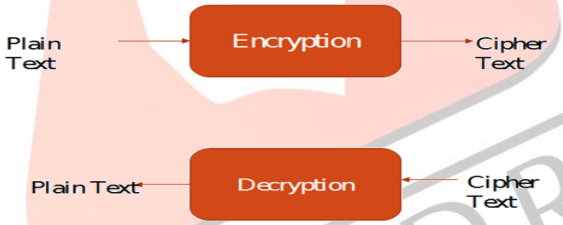


Fig 1 General Structure of Cryptography

Cryptography is used in our routine life. It becomes important for setting up safe connections over insecure networks like the Internet. It permits to exchange authenticated data and to generate digital signatures. The applied mathematical operations are very time-consuming. RC6 is considered as a secured and elegant choice due to its simplicity, security, performance and efficiency. It appears that RC6 is well suited for implementation in the targeted Xilinx FPGA (Spartan-3). Our studies reveal that multiplication and addition are the major bottlenecks as far as speed of encryption in the RC6 cipher is concerned. However up to a great extent this shortcoming was handled using pipelining in our design. Consequently, because of RC6 works best in non-feedback mode, the highest Speed/Area ratio can be achieved in the same RC6 is a symmetric key block cipher derived from RC5 algorithm. It was designed by Ron Rivest, Matt Robshaw, Ray Sidney, and Yiqun Lisa Yin to meet the requirements of the Advanced Encryption Standard (AES) competition.

RC6 has a block size of 64 or 128 bits and supports key sizes of 128, 192 and 256 bits, but, like RC5, it can be parameterized to support a wide range of word-lengths, key sizes and number of rounds. RC6 is similar to RC5 in structure, using data-dependent rotations, and modular addition and XOR operations; in fact, RC6 could be viewed as interweaving two parallel RC5 encryption processes. However, RC6 does use an extra multiplication operation which is not present in RC5 in order to make the rotation dependent on every bit in a word, and not just the least significant few bits.

II. ARCHITECTURE OF AES ALGORITHM.

AES is a non-Feistel cipher that encrypts and decrypts a data block of 128 bits. It uses 10, 12, or 14 rounds in algorithm. The key size, which can be 128, 192, or 256 bits depends on the number of rounds. AES uses four types of transformations to provide security.

1. Substitution
2. Permutation
3. Mixing
4. Key Adding

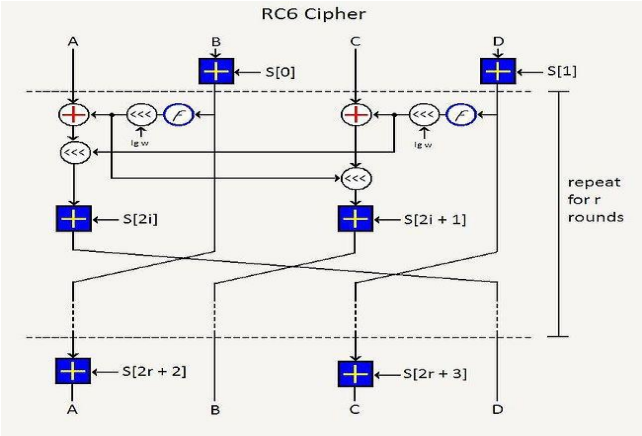


Fig 2 Basic block diagram of RC6 algorithm

III. ENCRYPTION OF RC6

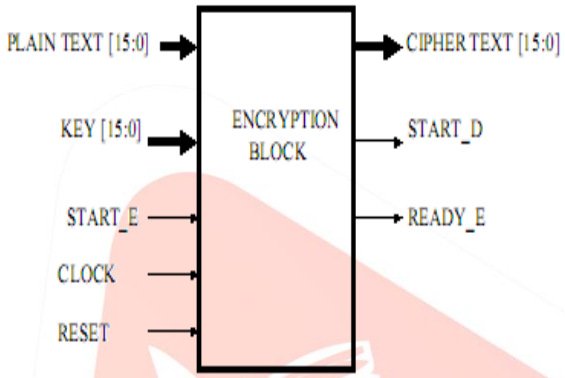
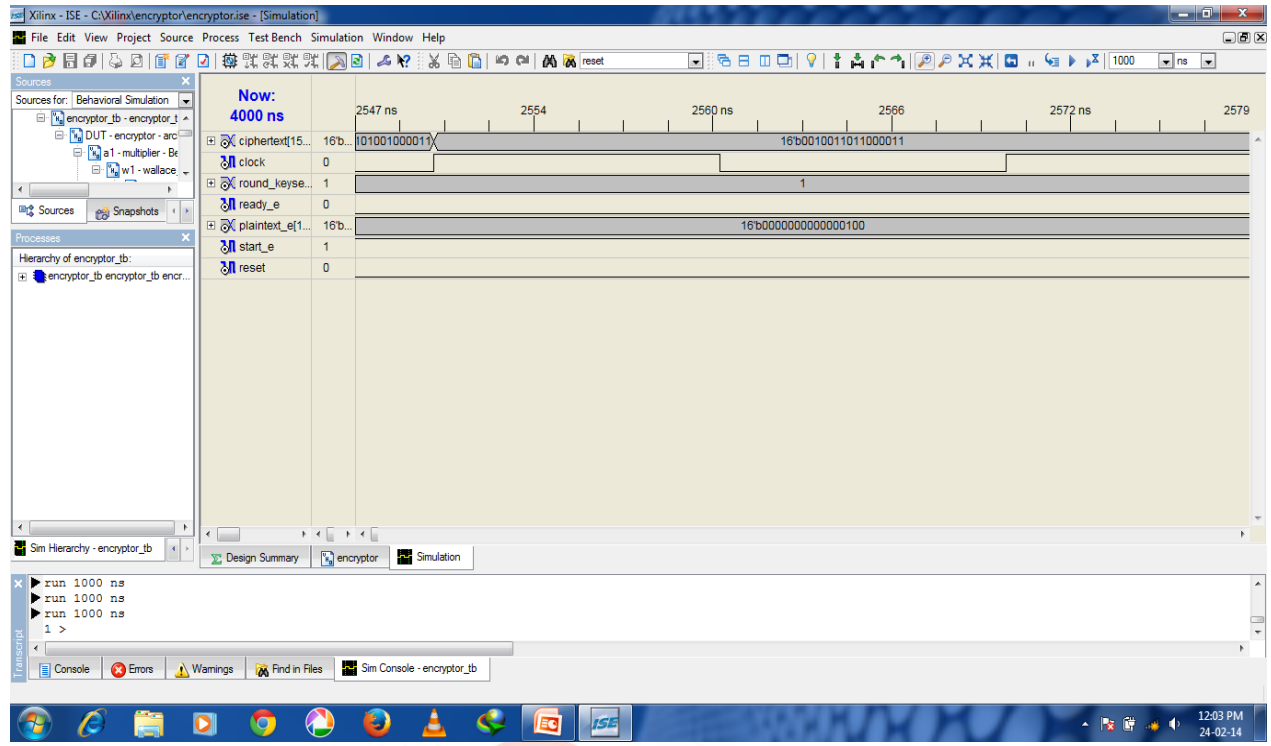


Fig 3 Encryption block diagram

Signal Specifications

1. Plaintext: This 16-bit data input signal. This corresponds to a word of plaintext data which is to be encrypted. Four consecutive 16-bit plaintext values form a 64-bit block which is to be encrypted.
2. Round_keys_e: This 16-bit data input signal. Which corresponds to one encryption round key. A total of 44 round keys are used to encrypt and decrypt the input data. Although the encryptor and decryptor use identical round keys for a given block of data, separate encryptor and decryptor round keys are provided, since they receive the round keys at different times and processor the round keys in reverse order. This also allows the decryptor to be deciphering one block, while the encryptor is ciphering the next.
3. Start_e : This 1-bit control input signal. This tells the encryptor that it will start receiving plaintext_e during the next cycle. This signal should only go high for one cycle.
4. Reset: This 1-bit control input signal, which resets both the encryptor and decryptor. When reset occurs, all registers are cleared and the controllers go to a known state.
5. Clock: This 1-bit input signal, which corresponds to the system clock for the encryptor and decryptor. Values are to be latched into registers at the positive edge of the clock.
6. Ciphertext: This 16-bit data output signal, which corresponds to a word of cipher text data that has been encrypted and needs to be decrypted. Four consecutive 16-bit cipher text values form a 64-bit block that has been encrypted. The same signal is used at input side to the decryptor.
7. Ready_e: This 1-bit control output signal, which indicates that the encryptor is ready to receive new plaintext. It goes high following a reset signal and stays high until the start_e goes high. Once the encryptor has finished encrypting the data ready_e goes high, until the next time start_e goes high.
8. Start_d : This 1-bit control output signal, Which tells the decryptor that the encryptor will start sending cipher text during the next cycle. This signal should only go high for only one cycle. This same signal is used at input side to the decryptor.

IV. SIMULATION RESULTS



Simulation result in binary

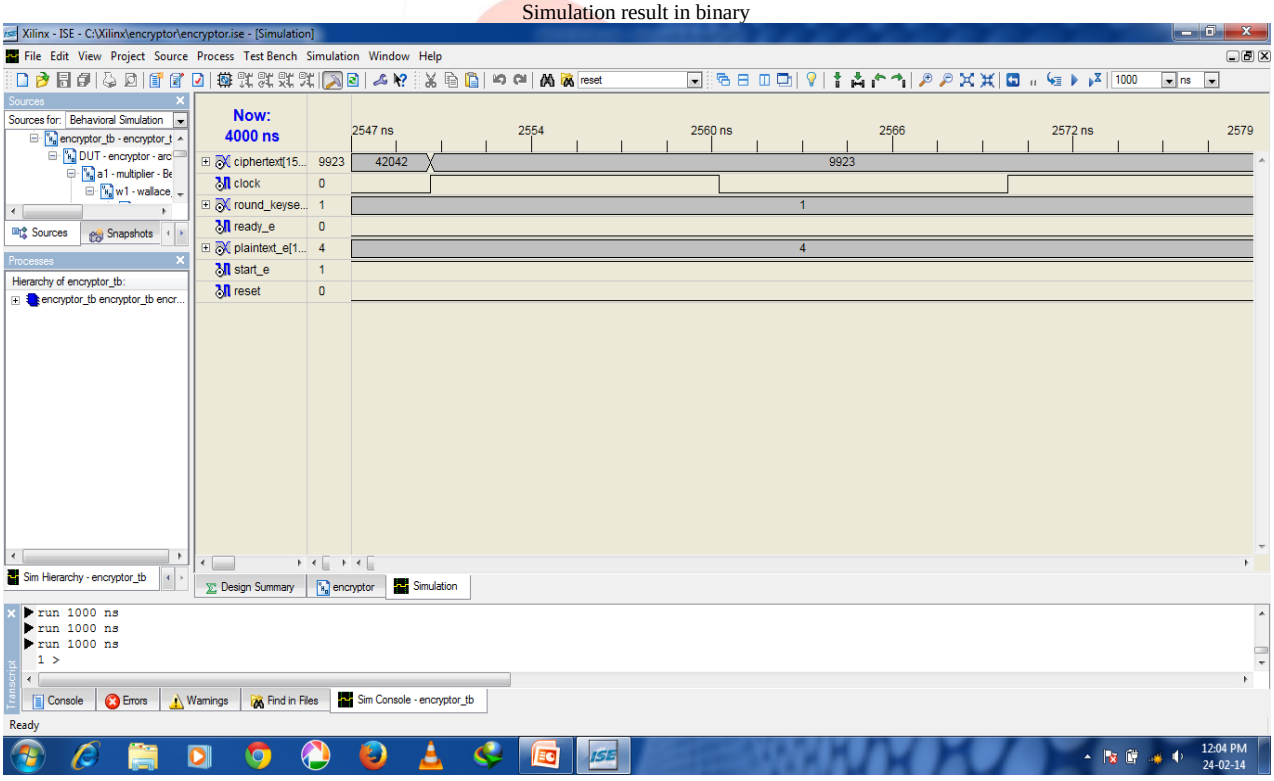


Fig 4 Simulation result in decimal

Device Utilization Summary (estimated values)			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slices	1479	13312	11%
Number of Slice Flip Flops	214	26624	0%
Number of 4 input LUTs	2454	26624	9%
Number of bonded IOBs	52	221	23%
Number of GCLKs	1	8	12%

Fig 5 Device utilization summary

Fig 6 RTL Schematic

V. CONCLUSION

A VHDL-Xilinx behavioral model of Encryption of AES algorithm is presented in this paper. Numbers of slices used are very less and design with minimum utilization is presented. This design offers minimum period of 13.345ns (Maximum Frequency-74.934MHz). Multiplication and addition is the major bottlenecked as far as the speed of Encryption in RC6 concern. However up to some extent this short coming was handled using the pipelining in our design.

REFERENCES

[1] William Stallings, "Cryptography and Network Security – Principles and Practices", 4th Edition, Pearson Education Asia – 2006.
[2] Behrouz Forouzan, "Cryptography and network security"
[3] Fouad Ramia, Hunar Qadir, "RC6 Implementation including key scheduling using FPGA", ECE 646 Project, December 2006
[4] LIU Niansheng, GUO Donghui, HUANG Jiaxiang, "Implementation scheme with Java is feasible for the PDA", 1-4244-1035-5/07/2007 IEEE.
[5] Yuan Kun, Zhang Han, Li Zhaohui, "Improved AES algorithm based on chaos", DOI 10.1109/MINES.2009.219, 2009IEEE
[6] Leonard Gibson Moses S, Thilagar M, "VHDL implementation of high performance RC6 algorithm using ancient
[7] Indian vedic mathematics", 978-1-4244-8679-3/11,2011 IEEE
[8] Obaida Mohammad Awad Al-Hazaimeh, "a new approach for complex encrypting and decrypting data",
[9] International Journal of Computer Networks & Communications (IJCNC) Vol.5, No.2, March 2013
[10] J. Bhasker Third Edition, A VHDL Primer.