DOI: 10.35598/mcfpga.2019.014

A VHDL Implemetation of the Advanced Encryption Standard

Hanna Loban
Department of Computer Radio Engineering and Technical Information Security Systems
Kharkiv National University of Radio Electronics
Kharkiv, Ukraine
hanna.loban@nure.ua

***Abstract*—A new original approach to realization of AES algorithm on FPGA is proposed. Problems of VHDL modeling of AES ciphering and deciphering are considered.**

Keywords—AES, triple DES, encryption, decryption, FPGA, VHDL

I. Introduction

Modern cryptography is the cornerstone of computer and communications security. Its foundation is based on various concepts of mathematics such as number theory, computational-complexity theory, and probability theory.

It came out on top among several competitors and was officially announced in 2001 by the new AES encryption standard. The algorithm is based on several substitutions, permutations and linear transformations, each of which is performed on data blocks of 16 bytes, hence it takes its term “blockcipher”. These operations are repeated several times, called "rounds." During each round, a unique round key is calculated from the encryption key and is included in the calculation. Based on the AES block structure, modifying a single bit, either in the key or in the plaintext block, results in a completely different block of ciphertext - a clear advantage over traditional stream cipher methods. Finally, the difference between AES-128, AES-192, and AES-256 is the key length: 128, 192, or 256 bits — all radical improvements over the 56-bit DES key. And cracking a 128-bit AES key using a modern supercomputer will take longer than the estimated age of the universe. [1]

II. AES Standard

The more popular and widely adopted symmetric encryption algorithm likely to be encountered nowadays is the Advanced Encryption Standard (AES). It is found at least six time faster than triple DES. [1]

A replacement for DES was needed as its key size was too small. With increasing computing power, it was considered vulnerable against exhaustive key search attack. Triple DES was designed to overcome this drawback but it was found slow.[1,2]

The features of AES are as follows:

- symmetric key and symmetric block cipher;
- 128-bit data, 128/192/256-bit keys;
- stronger and faster than Triple-DES;
- provide full specification and design details;
- software implementable in C and Java.

The Advanced Encryption Standard can be programmed in software or built with pure hardware. However Field Programmable Gate Arrays (FPGAs) offer a quicker, more customizable solution. This research investigates the AES algorithm with regard to FPGA and the Very High Speed Integrated Circuit Hardware Description language (VHDL). Altera Max+plus II software is used for simulation and optimization of the synthesizable VHDL code. All the transformations of both Encryptions and Decryption are simulated using an iterative design approach in order to minimize the hardware consumption. Altera ACEX1K Family devices are utilized for hardware evaluation.

III. The algorithm

A. Encryption

The AES algorithm represents a data block in the form of a two-dimensional byte array of 4x4. All operations are performed on individual bytes of the array, as well as on independent columns and rows. In each round of the algorithm, the following transformations are performed:

1. SubBytes operation, which is a tabular replacement of each byte of the data array

2. The ShiftRows operation, which performs a cyclic left shift of all rows of the data array, with the exception of zero. Shift i-th row of the array (for i = 1,2,3) is performed on i byte.

3. MixColumns operation. Performs multiplication of each data array column by a fixed polynomial a(x):

$$a(x) = 3x^3 + x^2 + x + 2.$$

Multiplication is performed by modulo $x^4 + 1$

4. The AddRoundKey operation imposes a key material on the data array. Namely, on i -th column of the data array (i = 0 ... 3), a bit-by-bit logical exclusive or XOR operation,

superimposes a specific extended key word W_{4r+i}, where r is the number of the current round of the algorithm, starting with 1. Before the first round of the algorithm, a preliminary key material is superimposed using the AddRoundKey operation, which overlays the clear four words of the extended key $W_0 \ldots W_3$ on the plaintext.

The last round is different from the previous ones in that it does not perform the operation MixColumns.

B. Decryption

Decryption is performed using reverse operations in reverse order. Accordingly, before the first round of decryption, an AddRoundKey operation (which is the reverse of itself) is performed, performing overlay on the ciphertext of the last four words of the extended key, i.e. $W_{4r} \ldots W_{4r+3}$. Then r decryption rounds, each of which performs the following transformations:

1. The InvShiftRows operation performs a cyclic right shift of the last three rows of the data array by the same number of bytes that the ShiftRows operation was shifted during encryption.

2. The InvSubBytes operation performs a table-by-byte reverse tabular replacement.

3. The AddRoundKey operation, as well as when encrypting, imposes four words of the extended key $W_{4r} \ldots W_{4r+3}$ on the processed data. However, the numbering of rounds r when decrypting is performed in the opposite direction - from $r-1$ to 0.

4. The InvMixColumns operation multiplies each column of the data array in the same way as the direct MixColumns operation, however, the multiplication is performed by the polynomial $a^{-1}(x)$, defined as follows:

$$a^{-1}(x) = Bx^3 + Dx^2 + 9x + E.$$

Similar to encryption, the last decryption round does not include the InvMixColumns operation.

C. Key extension

AES uses encryption keys of three fixed sizes: 128, 192, and 256 bits. Depending on the key size, a specific variant of the AES algorithm may be referred to as AES-128, AES-192 and AES-256, respectively.

The task of the key expansion procedure is to form the necessary number of words of the extended key for their use in the AddRoundKey operation. As mentioned above, the “word” here means a 4-byte fragment of the extended key, one of which is used in the primary imposition of the key material and one by one in each round of the algorithm. Thus, in the process of key expansion, $4 * (r + 1)$ is formed of words.

Key expansion is performed in two stages, the first of which is the initialization of the words of the extended key (denoted as W_i): the first N_k (N_k is the size of the original encryption key K in words, that is, 4, 6 or 8) words W_i (t. e. $i = 0 \ldots N_{k-1}$) are formed by their successive filling with key bytes.

The following figure 1 represents complete hardware implementation of the both encryption and decryption with key generation modules.

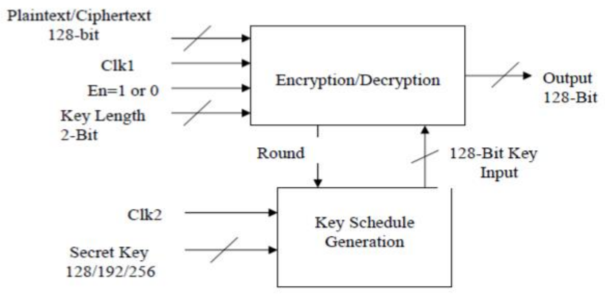

Fig. 1. Block Diagram of AES Hardware Implementation.

Key Schedule Generation block can generate the required keys for the process with secret key and Clk2 as inputs; these generated keys are stored in internal ROM and read by Encryption/Decryption block for each round to obtain a distinct 128-bit key with Round counter, where Encryption/Decryption module takes 128-bit plaintext or ciphertext as input with respective to the Clk1 (If En=1 or 0 process is encryption or decryption respectively). In order to distinguish the number of rounds, a 2-bit Key Length input is given to this module where 00, 01, 10 represents 10(128-bit key), 12(192- bit key), 14(256-bit key) rounds respectively, generates the final output of 128-bit cipher or plaintext.

IV. CONCLUSION

Optimized and Synthesizable VHDL code is developed for the implementation of both encryption and decryption process. Each program is tested with some of the sample vectors provided by NIST and output results are perfect with minimal delay. Therefore, AES can indeed be implemented with reasonable efficiency on an FPGA, with the encryption and decryption taking an average of 320 and 340 ns respectively (for every 128 bits). The time varies from chip to chip and the calculated delay time can only be regarded as approximate. Adding data pipelines and some parallel combinational logic in the key scheduler and round calculator can further optimize this design.

REFERENCES

[1] Tilborg, Henk C. A. van. “Fundamentals of Cryptology: A Professional Reference and Interactive Tutorial”, New York Kluwer Academic Publishers, 2002.

[2] Peter J. Ashenden, “The Designer's Guide to VHDL”, 2nd Edition, San Francisco, CA, Morgan Kaufmann, 2002.

[3] Joan Daemen and Vincent Rijmen, The Design of Rijndael, AES - The Advanced Encryption Standard, Springer-Verlag 2002 (238 pp.).