High Performance Hardware Implementation of AES Algorithm Using Minimal Resources

T. Anusha 1*, N. Satyanarayana 2*
1. II. M.Tech (VLSI), Dept of ECE, AM Reddy Memorial College of Engineering & Technology, Petlurivaripalem.

Abstract: -

Increasing need of data protection in computer networks led to the development of several cryptographic algorithms hence sending data securely over a transmission link is critically important in many applications. Hardware implementation of cryptographic algorithms are physically secure than software implementations since outside attackers cannot modify them. In order to achieve higher performance in today’s heavily loaded communication networks, hardware implementation is a wise choice in terms of better speed and reliability. In order to achieve higher speed and lesser area, Sub Byte operation, Inverse Sub Byte operation, Mix Column operation and Inverse Mix Column operations are designed as Look Up Tables (LUTs) and Read Only Memories (ROMs). This approach gives a throughput of 3.74Gbps utilizing only 1% of total slices.

I. Introduction

The main Objective of this project is to code a Data Encryption System using Advanced Encryption Standard (AES) Algorithm in Hardware Description Language and to test it according to a predetermined standard stimulus so that it meets requirements. This standard specifies the Rijndael algorithm and, a symmetric block cipher that can process data blocks of 128 bits, using cipher keys with lengths of 128, 192, and 256 bits. Rijndael was designed to handle additional block sizes and key lengths, however they are not adopted in this standard. Throughout the remainder of this standard, the algorithm specified herein will be referred to as “the AES algorithm.” The algorithm may be used with the three different key lengths indicated above, and therefore these different “flavors” may be referred to as “AES-128”, “AES-192”, and “AES-256”. Security attacks against network are increasing significantly with time. Our communication media should also be secure and confidential. For this purpose, these three suggestions arrive in every one’s mind:

(i) one can transmit the message secretly, so that it can be saved from hackers, (ii) the sender ensures that the message arrives to the desired destination, and (iii) the receiver ensures that the received message is in its original form and coming from the right sender. For this, one can use two techniques: (i) one can use invisible ink for writing the message or can send the message through the confidential person, and (ii) one can use a scientific approach called “Cryptography”. Cryptography is the technique used to avoid unauthorized access of data. For example, data can be encrypted using a
cryptographic algorithm in conjunction with the key management. It will be transmitted in an encrypted state, and later decrypted by the intended party. If a third party intercepts the encrypted data, it will be difficult to decipher. The security of modern cryptosystems is not based on the secrecy of the algorithm, but on the secrecy of a relatively small amount of information, called a secret key. The fundamental and classical task of cryptography is to provide confidentiality by encryption methods. Cryptography is used in applications present in technologically advanced societies; examples include the security of ATM cards, computer passwords, and electronic commerce, which all depend on cryptography. Cryptanalysis is the study used to describe the methods of code-breaking or cracking the code without using the security information, usually used by hackers.

II. PROPOSED ADVANCED ENCRYPTION STANDARD

The AES algorithm is a round-based, symmetric block cipher. It processes data blocks of fixed size (128 bits) using cipher keys of length 128, 196 or 256 bits. Depending on the key used, it is usually abbreviated as AES-128, AES-196 or AES-256 respectively. In this project only AES-128 is considered, as it is the most popular variant of the algorithm. The functional blocks of the algorithm are Key expansion and encryption. In this project we are concentrating on the key generation algorithm. The initial 128-bit cipher key has to be expanded to new eleven round keys of same length. In order to produce a new round key, two transformations have to be performed, RotWord and SubWord. The first one simply cyclically shifts the bytes of the first 32-bit word of the previous key by one position to the left. SubWord on the other hand performs the SubBytes transformation to each byte of the rotated word. Simple bit wise xors are then needed in order to produce the final round key. The SubWord (SubBytes) transformation is implemented with a ROM (LUT).

Fig 1 Architecture of Advanced Encryption Standard

2.1 Advanced Encryption Standard Algorithm

The architecture of AES is shown in figure1 AES is an iterated block cipher with a fixed block size of 128 and a variable key length. The different transformations operate on the intermediate results, called state. The state is a rectangular array of bytes and since the block size is 128 bits, which is 16 bytes, the rectangular array is of dimensions 4x4. (In the Rijndael version with variable block size, the row size is fixed to four and the number of columns varies. The number of columns is the block size divided by 32 and denoted Nb). The cipher key is similarly pictured as a rectangular array with four rows. The number of columns of the cipher key, denoted Nk, is equal to the key length divided by 32.
2.2 Encryption

Encryption is the conversion of data into a form, called a cipher text, that cannot be easily understood by unauthorized people. Decryption is the process of converting encrypted data back into its original form, so it can be understood.

The use of encryption/decryption is as old as the art of communication. In wartime, a cipher, often incorrectly called a code, can be employed to keep the enemy from obtaining the contents of transmissions. (Technically, a code is a means of representing a signal without the intent of keeping it secret; examples are Morse code and ASCII.) Simple ciphers include the substitution of letters for numbers, the rotation of letters in the alphabet, and the "scrambling" of voice signals by inverting the sideband frequencies. More complex ciphers work according to sophisticated computer algorithms that rearranges the data bits in digital signals. In order to easily recover the contents of an encrypted signal, the correct decryption key is required. The key is an algorithm that undoes the work of the encryption algorithm. Alternatively, a computer can be used in an attempt to break the cipher. The more complex the encryption algorithm, the more difficult it becomes to eavesdrop on the communications without access to the key.

Encryption/decryption is especially important in wireless communications. This is because wireless circuits are easier to tap than their hard-wired counterparts. Nevertheless, encryption/decryption is a good idea when carrying out any kind of sensitive transaction, such as a credit-card purchase online, or the discussion of a company secret between different departments in the organization. The stronger the cipher -- that is, the harder it is for unauthorized people to break it -- the better, in general. However, as the strength of encryption/decryption increases, so does the cost.

This means everyone who uses a cipher would be required to provide the government with a copy of the key. Decryption keys would be stored in a supposedly secure place, used only by authorities, and used only if backed up by a court order. Opponents of this scheme argue that criminals could hack into the key-escrow database and illegally obtain, steal, or alter the keys. Supporters claim that while this is a possibility, implementing the key escrow scheme would be better than doing nothing to prevent criminals from freely using encryption/decryption.

2.3. Key Generation

Key generation is the process of generating keys for cryptography. A key is used to encrypt and decrypt whatever data is being encrypted/decrypted. Modern cryptographic systems include symmetric-key algorithms (such as DES and AES) and public-key algorithms (such as RSA). Symmetric-key algorithms use a single shared key; keeping data secret requires keeping this key secret. Public-key algorithms use a public key and a private key. The public key is made available to anyone (often by means of a digital certificate). A sender encrypts data with the public key; only the holder of the private key can decrypt this data.

The simplest method to read encrypted data is a brute force attack—simply attempting every number, up to the maximum length of the key. Therefore, it is important to use a sufficiently long key length; longer keys take exponentially longer to attack, rendering a brute force attack impractical.
Currently, key lengths of 128 bits (for symmetric key algorithms) and 1024 bits (for public-key algorithms) are common.

III. RESULTS

3.1 Simulation results:

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>0ns</th>
</tr>
</thead>
<tbody>
<tr>
<td>aes_enc</td>
<td>76bde35a1</td>
<td>00000000</td>
</tr>
<tr>
<td></td>
<td>01234567</td>
<td></td>
</tr>
<tr>
<td></td>
<td>00000000</td>
<td></td>
</tr>
</tbody>
</table>

Simulation waveform for encryption

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>0ns</th>
</tr>
</thead>
<tbody>
<tr>
<td>aes_dec</td>
<td>01234567</td>
<td>00000000</td>
</tr>
<tr>
<td></td>
<td>76bde35a1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>00000000</td>
<td></td>
</tr>
</tbody>
</table>

Simulation waveform for Decryption

3.2. Timing report

Timing report for Encryption

Timing Summary:

- Speed Grade: -4
- Minimum period: 3.154ns (Maximum Frequency: 317.058MHz)
- Minimum input arrival time before clock: 3.314ns
- Maximum output required time after clock: 79.515ns
- Maximum combinational path delay: No path found

Timing report for Decryption

Minimum period: 54.978ns (Maximum Frequency: 18.189MHz)
- Minimum input arrival time before clock: 4.613ns
- Maximum output required time after clock: 148.173ns
- Maximum combinational path delay: No path found

3.3 Area report

Area report for Encryption

Selected Device: 3s500efg320-4

- Number of Slices: 13758
  - out of 4656 295% (*)
- Number of Slice Flip Flops: 128
  - out of 9312 1%
- Number of 4 input LUTs: 26999
  - out of 9312 289% (*)
- Number of IOs: 385
- Number of bonded IOBs: 385
  - out of 232 165% (*)
- Number of BRAMs: 10
- Number of GCLKs: 1 out of 24 4%

WARNING: Xst:1336 - (*) More than 100% of Device resources are used

Area report for Encryption

Selected Device: 3s500efg320-4

- Number of Slices: 16243
  - out of 4656 348% (*)
- Number of Slice Flip Flops: 144
  - out of 9312 1%
- Number of 4 input LUTs: 31226
  - out of 9312 335% (*)
- Number of IOs: 385
- Number of bonded IOBs: 385
  - out of 232 165% (*)
- Number of BRAMs: 10
  - out of 20 50%
Number of GCLKs: 1 out of 24 4%
WARNING: Xst:1336 - (*) More than 100% of Device resources are used

CONCLUSION

AES-128 algorithm for encryption and decryption is implemented Xilinx. With the designing of all the operations as LUTs and ROMs, the proposed architecture achieves a throughput of 3.74 Gbps and thereby utilizing only 1% of slices in the targeted FPGA. Since the speed is higher than the already reported systems, hence the proposed design serves as the best high speed encryption algorithm and is thus suitable for various applications. Moreover with less area utilization, the proposed design can be embedded with other larger designs as well.

REFERENCES

[2] Yang Jun Ding Jun Li Na Guo Yixiong School of Information Science and Engineering, Yunnan University Kunming, China - "FPGA based design and implementation of reduced AES algorithm" (IEEE 2010).