A Security Based Cryptographic AES using Fully Combinational Pipelined Substitution Box

1Vasanth Kumar Y.B, 2R. Sundaraguru
1,2Dept.of ECE, Sir.MVIT, Bengaluru, Karnataka, India

Abstract—A superior substitution box (S-Box) FPGA usage utilizing Galois Field GF (28) is displayed in this paper. An ideal number of pipeline registers in view of Spartan-3E FPGA is tended to in this paper. The plan is completely synthesizable utilizing Verilog and can undoubtedly be changed over to ASIC execution. The AES is an type of symmetric block encryption technique which is based on a plaintext of 128 bits and an key of size 128,192,256 bits. For high speed operation non look up table based substitution box based on Galois field is used in this paper. The AES is made up of four blocks sub-byte substitution, shift row, mix column and add round key. The only non linear block in AES is sub-byte substitution using pre computed look up tables. The complexity of hardware for AES is dominated by single s-box. The maximum clock frequency which can be applied to the design is 701.262MHz, with an output delay that is equal to 2 clock cycles.

Keywords—Substitution Box, AES, Galois Field, FPGA.

I. INTRODUCTION

Cryptography is the science of information and communication security. Cryptography is the science of secret codes, enabling the confidentiality of communication through an unconfident channel. It protects against unauthorized parties by preventing unauthorized alteration of use. It uses a cryptographic system to transform a plaintext into a cipher text, using most of the time a key [1]. Byte substitution and Inverse Byte Substitution are the most difficult steps in the encryption and decryption processes. In these steps each byte of the state array will be replaced with its equivalent byte in the S-box or the Inverse S-box. As AES algorithm use elements within the GF (28), each element in the state array represents a byte with a value that varies between 00H-FFH. The S-box has a fixed size of 256 bytes represented as (16 * 16) bytes matrix [2]. In this paper propose an optimized and pipelined architecture for calculating multiplicative inverse in a Galois Field GF(28) followed by an affine transformation in the binary extension field [4], which is based on combinational logic as existing in [7]. Such examples of work that implements the S-Box using this process are discussed in [3] and other publications. This implementation of S-Box has the advantage having small area occupancy; in addition to the possibility of being pipelined by inserting register stages for increased performance in clock frequency. So, this paper presents the implementation of pipeline S-Box based on composite field approach on Spartan-3E FPGA and it addresses an best possible number of pipeline stages expected to improve the performance in term of throughput per area.
II. THE S-BOX CONSTRUCTION.

1. THE S-BOX

The content of the S computed based on Galois Field GF(2^8) by undergoing two Transformations first taking a multiplicative inverse in the Galois Field GF(2^8) with irreducible polynomial.

\( m(x) = x^8 + x^4 + x^3 + x + 1 \)

Then applying a standard affine transformation over Galois Field GF(2^8). The polynomial representation in GF.

\( b(x) = b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 \)

and the multiplication in GF(2^8) is straightforward operation and can be done using the polynomial representation above by simply multiplying the two bytes representative polynomials together modulo with the irreducible polynomial m(x).

### Table 1: S-Box look up table

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0b</td>
<td>0c</td>
</tr>
<tr>
<td>0c</td>
<td>0d</td>
</tr>
<tr>
<td>0d</td>
<td>0e</td>
</tr>
<tr>
<td>0e</td>
<td>0f</td>
</tr>
<tr>
<td>0f</td>
<td>0a</td>
</tr>
<tr>
<td>0a</td>
<td>0b</td>
</tr>
</tbody>
</table>

A. Affine Transformation.

The affine transformation f is defined in a matrix form which can also be relating as a polynomial multiplication, followed by affine transformation.

B. Multiplicative Inversion in GF(2^8).

The Composite field mechanism which can be used for calculating multiplicative inverse in AES S-Box is an efficient method. The plan behind using composite field to calculate multiplicative inverse in GF(2^8) is then by decomposing the illustration of field elements. All Galois Field representations of the same order, for example, GF(2^8), GF((2^4)^2) and GF(((2^2)^2)^2), are isomorphic, otherwise they are structurally the same, only different in their field element representation.

C. Mapping and Inverse Isomorphic mapping

Both isomorphic mapping (δ mapping (δ^−1) can be represented as 8 × 8 matrix. Let q be the element in GF(2^8), then the isomorphic mappings and its inverse can be written as δq and δ^−1q consecutively, which are a case of matrix multiplication as shown below in (6) and (7).
Shift Rows() Transformation

In the Shift Rows() transformation, the bytes in the last three rows of the State are cyclically shifted over different numbers of bytes (offsets). The first row, $r = 0$, is not shifted. Specifically, the Shift Rows() transformation proceeds as follows:

$$ s = s' \text{cr}, (c + \text{shift}(r, Nb)) \text{ mod } Nb \text{ for } 0 < r < 4 \text{ and } 0 \leq c < Nb $$

Mix Columns() Transformation

The MixColumns() transformation operates on the State column-by-column, treating each column as a four-term polynomial. The columns are considered as polynomials over GF($2^8$) and multiplied modulo $x^4 + 1$ with a fixed polynomial $a(x)$, given by

$$ a(x) = \{03\}x^3 + \{01\}x^2 + \{01\}x + \{02\} $$

Add Round Key Transformation

In the Add RoundKey() transformation, a Round Key is added to the State by a easy bitwise XOR operation.

III. RESULTS

The figure shows the simulation Results of S-Box implementation with the inverse multiplication in gallois field and standard affine transformation.

**Simulation Results**

The Figure shows the simulation results using ModelSim for AES-128 bit by considering the following inputs.

Plain Text – 00112233445566778899aaabbbccddeeff

Key Input – 000102030405060708090a0b0c0d0e0f

The following Encrypted output obtained is as shown.

Cipher Key – 13111d7fe3944a17f307a78b4d2b30c5

Cipher Text – 69c4e0d86a7b0430d8c0b7807b4c55
IV. CONCLUSION AND FUTURE WORK

The designed core supports both encryption and decryption standards. Its functionality has been verified using simulation, by taking various inputs and is synthesized by using Xilinx 13.2. The design is targeted on FPGA (spartan-3E). The design uses 2439 slices, 2252 flip flops, 4647 4-input look up tables and operates at 2.84 Gbps (Throughput). Progress of physical design of AES-128 bit is done using cadence SoC encounter. Performance estimation of the physical design with respect to area, power, and time has been done. The core consumes 10.21 mW of power for the core area of 332128.742 μm2.

This Implementation of 128 bit AES using Rijndael algorithm, and the same can be extended to encrypt 192 and 256 bits of plain text data with proper key length, which makes even tougher to decrypt the original data form an unauthorized receivers.

REFERENCES


