Implementation of Advanced Encryption Standard using FPGA

Prof. Shilpa.A.Ingole[1,] Prof.V.T.Gaikwad[2]

Sipna's college of Engg. And Technology ,Amravati.
Corresponding Addresses
shilpa.ingole@gmail.com, vtgaikwad@rediffmail.com

Abstract*: Advanced Encryption Standard (AES) algorithm was developed by Vincent Rijmen and Joan Daeman and named as Rijndael cipher algorithm. AES consists of 128 block length of bits and supports 128,192 and 256 key length bits and consists of 10, 12 or 14 iteration rounds, respectively. Each round mixes the data with a round key, which is generated from the encryption key. For security and fast transmission of data over an insecure path, cryptography methods have been used so far. For this reason, we are trying to implement secure, fast and efficient cryptographic algorithms in hardware. In this project we are implementing advanced encryption standard (AES) algorithm using a Field Programmable Gate Array (FPGA).*

The objective is to implement an efficient realization of AES using very high speed integrated circuit hardware description language (VHDL). A fast and area efficient composite field implementation of the byte substitution phase is designed using an optimum number of stages for FPGA implementation. This seminar proposes an efficient solution to combine Rijndael encryption in one FPGA design, with a strong focus on low area constraints.

To ensure that our implementation gives a better result in terms of area and speed, we compare the two encryption codes, original and modified. This comparison is done by considering two criteria: chip speed and area utilization. The design will be implemented by using VHDL frontend and backend tools.

Keywords*: Cryptography, AES, DES, FPGA, compact encryption/decryption implementation, embedded Systems.*

1. Introduction

1.1 The AES algorithm

The Advanced Encryption Standard (AES, Rijndael) algorithm is a symmetric block cipher that processes data block of 128, 192 and 256 bits using, respectively,keys of the same length[3]. In this paper,only the 128 bit encryption version (AES-128) is considered. The 128-bit data block and key are considered as a byte array, respectively called State and RoundKey, with four rows and four columns.Let a 128-bit data block in the ith round be defined as:

data_block ^{I} $=d^i15|d^i14|d^i13|d^i12|d^i11|d^i10|d^i9|d^i8|d^i7|d^i6|d^i5|d^i4|d^i3|d^i2|d^i1|d^i\,0$

where di 15 represents the most significant byte of the data block of the round i. The corresponding State is:

$$State^i = \begin{bmatrix} d^i_{15} & d^i_{11} & d^i_7 & d^i_3 \\ d^i_{14} & d^i_{10} & d^i_6 & d^i_2 \\ d^i_{13} & d^i_9 & d^i_5 & d^i_1 \\ d^i_{12} & d^i_8 & d^i_4 & d^i_0 \end{bmatrix}$$

AES-128 consists of ten rounds. One AES encryption round includes four transformations: SubByte,ShiftRow, MixColumn and AddRoundKey. The first and last rounds differ from the other ones Indeed there is an additional AddRoundKey transformation at the beginning of the first round and noMixColumn transformation is performed in the last round. This is done to facilitate the decryption process. SubByte (SB) is a non-linear byte substitution. It operates with every byte of the State separately. The substitution box (S-box) is invertible and consists of two transformations:

1. Multiplicative inverse in GF(2^8). The zero element is mapped to itself.
2. An affine transform over GF(2^8).

The SubByte transformation applied to the State can be represented as follows:

$$SB(State^i) = \begin{bmatrix} SB(d^i_{15}) & SB(d^i_{11}) & SB(d^i_7) & SB(d^i_3) \\ SB(d^i_{14}) & SB(d^i_{10}) & SB(d^i_6) & SB(d^i_2) \\ SB(d^i_{13}) & SB(d^i_9) & SB(d^i_5) & SB(d^i_1) \\ SB(d^i_{12}) & SB(d^i_8) & SB(d^i_4) & SB(d^i_0) \end{bmatrix}$$

The inverse transformation is defined InvSubByte (ISB). ShiftRow (SR) performs a cyclical left shift on the last three rows of the State. The second row is shifted of one byte, the third row is shifted of two bytes and the fourth row is shifted of three bytes. Thus, the ShiftRow transformation proceeds as follows:

$$SR(SB(State^i)) = \begin{bmatrix} SB(d^i_{15}) & SB(d^i_{11}) & SB(d^i_7) & SB(d^i_3) \\ SB(d^i_{10}) & SB(d^i_6) & SB(d^i_2) & SB(d^i_{14}) \\ SB(d^i_5) & SB(d^i_1) & SB(d^i_{13}) & SB(d^i_9) \\ SB(d^i_0) & SB(d^i_{12}) & SB(d^i_8) & SB(d^i_4) \end{bmatrix}$$

The inverse ShiftRow operation (InvShiftRow (ISR)) is trivial.MixColumn (MC) operates separately on every column of the State. A column is considered as a polynomial over GF(2^8) and multiplied modulo x4+1 with the xored polynomial c(x): c(x) ='03'x3 +'01'x2 +'01'x +'02'

As an illustration, the multiplication by 0020 corresponds to a multiplication by two, modulo the irreductible polynomial m(x) = x8 + x4 + x3 + x + 1.

This can be represented as a matrix multiplication

$$R^i = MC(SR(SB(State^i))) = \begin{bmatrix} '02' & '03' & '01' & '01' \\ '01' & '02' & '03' & '01' \\ '01' & '01' & '02' & '03' \\ '03' & '01' & '01' & '02' \end{bmatrix} \otimes \begin{bmatrix} SB(d^i_{15}) & SB(d^i_{11}) & SB(d^i_7) & SB(d^i_3) \\ SB(d^i_{10}) & SB(d^i_6) & SB(d^i_2) & SB(d^i_{14}) \\ SB(d^i_5) & SB(d^i_1) & SB(d^i_{13}) & SB(d^i_9) \\ SB(d^i_0) & SB(d^i_{12}) & SB(d^i_8) & SB(d^i_4) \end{bmatrix}$$

To achieve the inverse operation (InvMixColumn (IMC)), every column is transformed by multiplying it with a specific multiplication polynomial d(x), de-fined by
c(x) d(x) =`01`
d(x) =`0B`x3 +` 0D`x2 +'09`x +` 0E`
AddRoundKey (AK) performs an addition (bitwise XOR) of the State with the RoundKey:

$$AK(R^i) = \begin{bmatrix} R^i_{15} & R^i_{11} & R^i_7 & R^i_3 \\ R^i_{14} & R^i_{10} & R^i_6 & R^i_2 \\ R^i_{13} & R^i_9 & R^i_5 & R^i_1 \\ R^i_{12} & R^i_8 & R^i_4 & R^i_0 \end{bmatrix} \oplus \begin{bmatrix} rk^i_{15} & rk^i_{11} & rk^i_7 & rk^i_3 \\ rk^i_{14} & rk^i_{10} & rk^i_6 & rk^i_2 \\ rk^i_{13} & rk^i_9 & rk^i_5 & rk^i_1 \\ rk^i_{12} & rk^i_8 & rk^i_4 & rk^i_0 \end{bmatrix}$$

The inverse operation (InvAddRoundKey
(IAK)) is trivial.
RoundKeys are calculated with the key schedule
for every AddRoundKey transformation. In AES-128, the original cipher key is the first RoundKey` (rk0) used in the additional AddRoundKey at the beginning of the first round. RoundKey, where 0 < i _ 10, is calculated from the previous
RoundKey 1. Let p(j) (0 _ j _ 3) be the column
j of the RoundKey 1 and let w(j) be the column
j of the RoundKey. Then the new RoundKey is
calculated as follows:
w(0) = p(0) _ (Rot(Sub(p(3))) _ rcon;
w(1) = p(1) _ w(0)
w(2) = p(2) _ w(1)
w(3) = p(3) _ w(2)
Rot is a function that takes a four byte input
[a0; a1; a2; a3] and rotates them as [a1; a2; a3; a0].The function Sub applies the substitution box (S-box) to four bytes. The round constant rconi contains values [(`02`)i1;`00`;`00`;`00`].

1.2 AES Decryption Algorithm

The Cipher transformation can be inverted to produce the Inverse Cipher which is the Decryption. The key Expansion remains the same for Encryption and Decryption if the data is decrypted by the Inverse Cipher. The AES encrypted data can be decrypted by an equivalent Inverse Cipher in which the transformation used is in the same sequence of the cipher whereas the sequence of transformation in the Key expansion changes. The Decryption generates the Plain text from the Cipher text. It also operates on 128 bits of Cipher text which is the output of the Cipher and uses the Key which is generated at the last iteration of the Cipher. The transformations used are
1. Inverse shift rows
2. Inverse Sub bytes Transformation
3. Inverse Mix Column transformation
4. Add round key Transformation

2. Related work:

In embedded applications, it is required to minimize the area rather than to maximize the throughput. Pramstaller et al presented a compact implementation of AES encryption and decryption with all key lengths using a novel State representation, which solves the problem of accessing both rows and columns of the State. Therefore, FPGAs are considered one of the integral part of the cryptographic hardware implementation in order to achieve further acceleration in throughput and efficient utilization of memory and other hardware resources.

Uinversity of Lethbridge [1] proposed a three stage sub pipelined architecture which is the first entry. The forte for this architecture is that it includes both encryption and decryption. The architecture is divided into the data module, the key generator module and the input/output module. The important block which implements the AES algorithm is the data module.

The S – box implemented is based on the combinational circuit proposed by J. Wolkerstorfer et al [2]. The column mixing is based on their own architecture put forth in The Key generator module also has 3 stages of pipelining and generates the same key for three consecutive cycles. The same architecture can also be extended to 192 and 256 bit key AES. The encoding rate is 1.57 Gbps and the latency is 30 cycles for 3 blocks of 128 bit key.

3. Proposed work and objectives:

The goal of this project is to compare c code with the FPGA output.

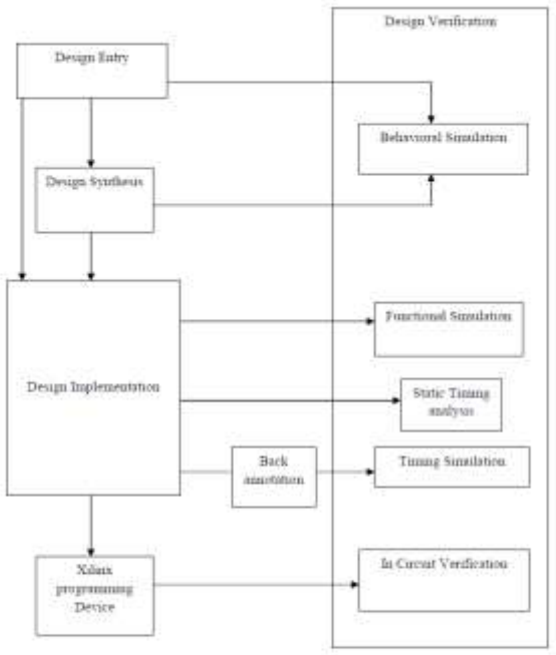

Figure 1 Block diagram

3.1 Design Entry The design entry for this project is basically the HDL codes for the AES

3.2 Design Synthesis and Implementation

The design is synthesized and implemented using Xilinx ISE. Once the design is placed and routed, the Post placed and routed design is simulated and the Output is cross checked with outputs after the simulation and the output from the C code. The same process is repeated for all the architectures of both the encryption and the Decryption module. The implementation is constrained for the maximum speed.

3.3 In Circuit verification

After the implementation of the Design a bit stream is generated. This bit stream programs the FPGA and makes the FPGA work as the AES. The circuit is synthesized and the bit stream is generated. This bit stream is used to program the FPGA. The Output is viewed verified using the C- code.

4. Performance Analysis:

The data input to the encryption module is (0000_0000_0000_0000_0000_0000_0000_0000) h and the key is also (0000_0000_0000_0000_0000_0000_0000_0000)h. The above data is encrypted to (66E9_4BD4_EF8A_2C3B_884C_FA59_CA34_2B2B)h. The encrypted data is also verified by the AES algorithm implemented in C. Figure 2: Online verification of basic AES (Encryption). Similarly the same data is sent as the input to the decryption module and the basic AES (decryption) is also verified. Figure 3 shows the online verification of Basic AES (decryption).

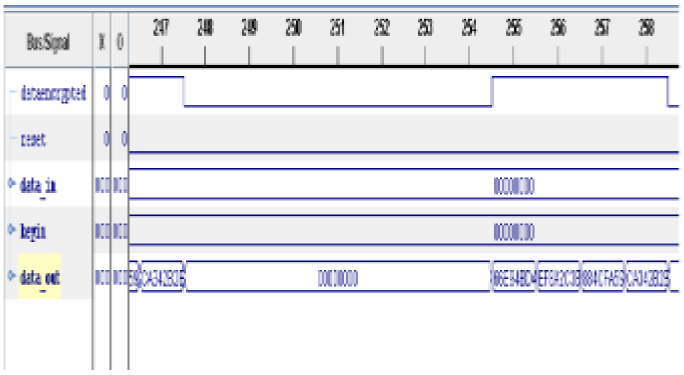

Figure 2: Online verification of basic AES (Encryption).

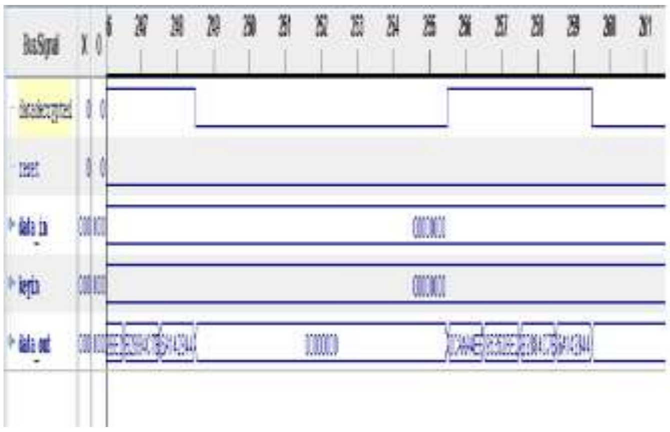

Figure 3: The online verification of Basic AES (decryption).

5.Conclusion

For the AES design, a software model of the AES algorithm was initially developed in C which would read a binary text file and then output the encoded bit stream into another binary text file. The same piece of code also decoded the encrypted data to plain text. This C code serves as a check for validating the behavioral output generated from the Simulation of the design. Once the results from the C code matches with the results in the behavioral simulation the design is further synthesized and checked for behavioral simulation again.. The Design is then implemented in a FPGA. The Xilinx development kit was used for verifying the designs. The design is then placed and routed on a FPGA.

6.References:

[1] "A High Performance Sub-Pipelined Architecture for AES" Hua Li and Jianzhou Li Department of Mathematics and Computer Science, University of Lethbridge, Canada T1K 3M4

[2] J. Wolkerstorfer, E. Oswald and M. Lamberger, "An ASIC Implementation of the AES SBoxes," CT-SA 2002, LNCS2271, pp. 67-78, 2002.

[3] National Institute of Standards and Technology: Specification for the Advanced Encryption Standard (AES). http://csrc.nist.gov/publications/fips/fips197/ fips-197.pdf, (2001).

[4] The Mathworks: Galois Field Computations. http://www.mathworks.com/ access/helpdesk/help/toolbox/comm/tutor3.shtml, Communications Toolbox,(2001)

Author Biographies

First Author :Prof.S.A.Ingole received her Bachelor of Information technology in 2008 from GCOE0 Amravati,Currently pursuing her M.E in Information technology and working as a lecturer in Sipna's C.O.E.T, Amravati.

Second Author: Prof.V.T.Gaikwad received his Bachelor of Electronics & telecommunication in 1994 from SSGM College of engg, shegaon. Masters degree in Electronics from GCOE ,Amravati in 2001 ,Currently working as a Asst.Prof in sipna's COET Amravati.