Fully Pipelined High Speed SB and MC of AES Based on FPGA

S.Sankar Ganesh [#1], J.Jean Jenifer Nesam[2]
[1]Assistant.Professor,VIT University
Tamil Nadu,India.
[1]s.sankarganesh@vit.ac.in
[2]jeanjenifer@rediffmail.com

Abstract:

A new implementation scheme of high speed mixcolumn based on sharing the use of sbox is introduced in this paper. The single MC (mixcolumn) shares the single SB(sbox-subbyte) based on the time slot. For each time slot SB and MC performed parallelly. Earlier they use 16 individual sbox for each input. In our paper, we introduce sharing concept of sbox which can eliminate the use of 16 individual sbox and reduce the delay and cost. Normal AES uses shiftrows followed by sbox needs 128 bit for their operations that consumes large time. By eliminating shiftrows, we can increase the speed of the AES operation. LUT based sbox consumes more than 75% of power. In our paper we design the Composite field sbox which reduces the power consumption of AES architecture. Sbox is the main source of information leakage since the values are fixed one. In our paper the values of sbox are masked by using particular fixed value thus increase the system security.

Keywords: AES, high speed MC, time slot, composite field sbox, vhdl, shiftrow elimination.

I. INTRODUCTION

In this modern world, communication between each and everyone is very important. Internet, satellite made the communication much easier than past decades. Even communication between each get easier, the security transmission of the message is very important today. Transmitting and receiving the secured data becomes tougher in nowadays. Encryption algorithms are used to protect the data from hackers. There are so many algorithms present like,Triple DES,AES,,,,etc.. Among them AES is very strong encryption standard that will give more secure encrypted data [4]. Even though AES is very strong algorithm, the hardware implementation sometimes leaks the information. The hackers attack the data in different ways to trace the key or the plaintext. In AES sbox is the main thing that leaks the message information or key information. In our paper sbox values are masked with some fixed value that useful to increase the data security and reduce the side channel attacks. In our paper we introduce sbox time sharing method that can eliminate the use of 16 individual sboxes. For each time slot the sbox gets the input and performs the subbyte operations on that input within that time slot. By sending Add Round Key output in proper way as a input of sbox gives output which is equal to shitrows output. Thus we can eliminate shiftrows and increases speed and system clock frequency and also throughput because the sbox output always 8 bit and shifting the rows need total of 128 bit. Then sbox output is directly given to mixcolumn. MC and SB both are performed in parallel.

II. ADVANCED ENCRYPTION STANDARD

A. Brief Explanation of Rijndael Algorithm

The Rijndael as Advanced Encryption Standard (AES) was published by NIST(National Institute of Standards and Technology) in 2001[3].The AES is strong security standard that become effective on May 26, 2002 by NIST to replace DES. The AES uses 128 bit input and the key length is 128 bit, 192 bit or 256 bits. AES can be implemented easily on software and also the hardware. Rijndael algorithm consists of encryption and decryption and key schedule algorithm. The main operations among three parts of Rijndael algorithm have four main operations. They use a) Byte substitution (sub bytes) b) The shiftrows c) Mixcolumns d) Round key adding (Add round key).

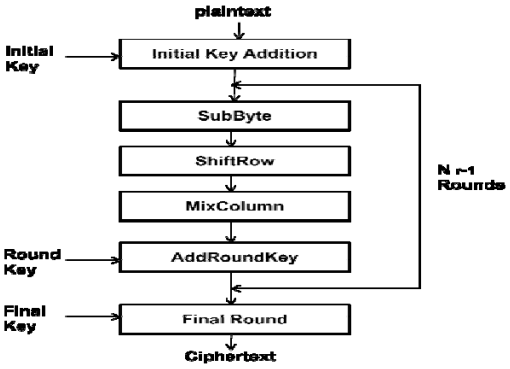


Fig.1. AES Encryption Structure

AES-128 encryption consists of 10 rounds of transmission of the input plaintext for the cipher text. For AES-128 bit the corresponding key length is 128 bits. In this paper only AES-128 encryption scheme with 128 bit key is considered.

III.COMPOSITE FIELD SBOX

The hardware implementation of LUT based sbox uses ROM or RAM to store the sbox values. This usage of memory consumes more than 75% power of overall AES architecture. In our paper the composite field sbox design is used which replace the need of memory. The composite field sbox and inverse sbox includes two main operations

1. Subbyte→multiplicative inverse in GF(2^8)→ Affine Transformation.
2. Invsubbyte→inverse affine transformation
   →Multiplicative inverse in GF(2^8).

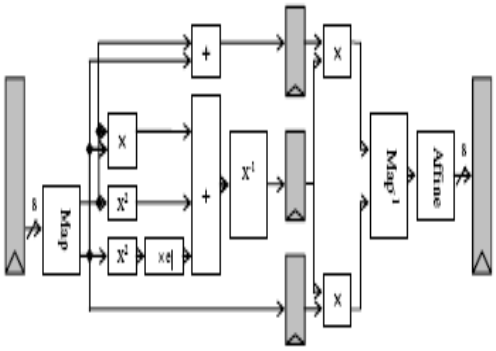


Fig.2. Two stages pipelined composite field sbox

Hence both the operations needs multiplicative inverse module so we can share the multiplicative inverse module and calculation of both can be separated by enable. Composite field sbox consumes less area and power compare to ROM based sbox design and also it reduces the construction charges. The main drawback of composite field sbox is logic delay. This can be eliminated during hardware implementation by inserting registers between multiplicative inverse and affine operation.

IV. SPEED and SECURITY IMPROVED SBOX

The sbox input and output always 8 bit and performing mixcolumn operation needs at least 32 bit at a time. 8 bit sbox output and use of registers slow down the mixcolumn operation. Instead of using separate sbox for each input, single sbox is shared by each input by a time slot. The inputs are separated by a time slot and for a particular time period it will take one value as an input. The sbox uses fixed ROM cases the hackers easily track the information by adding simple resisers parallel with that ROM. By reading the power consumed by each input to sbox we can trace out the key information this is known as power analysis side channel attacks. There are so many side channel attacks present, they can trace the information at any stage of AES but sbox is the main thing because it uses fixed known values. In our paper we use another values instead of original values. The masked values obtained by this way, the sbox values are xored with some particular value to reduce the side channel attacks [1].

Instead of using original value another one value is transmitted from sbox increases the system security. Another thing to increase the speed of AES algorithm, we eliminate the shiftrows stage. The output of Add Round Key is rearranged equalling to shiftrows output then given to sbox as an input. Then the sbox will give the output which is equal to shiftrow output. The performance of normal add round key and our proposed add round key output is explained in below figure3 and figure4.

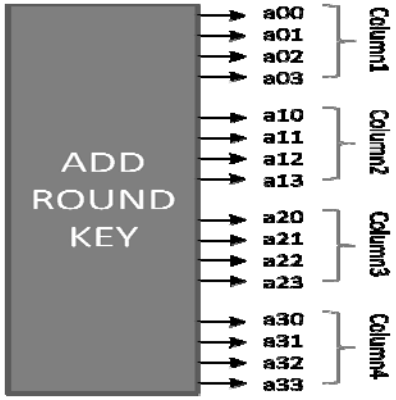


Fig.3. Original Add Round Key output

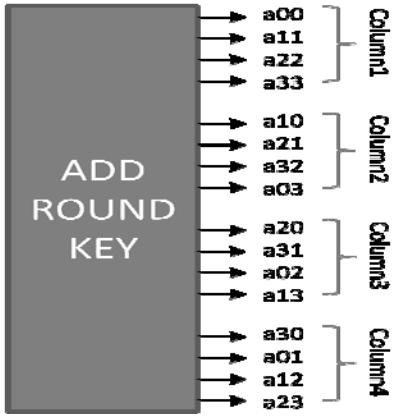


Fig.4. Rearranged Add Round Key output

V. PIPELINING MIXCOLUMN OPERATION

Normal mixcolumn operation performed on each column and each module gives one column output. Thus normal implementation needs 3 times replication of MC module. Replication of modules increases the speed but cost and area of the system architecture also get increased. Single MC based on time sharing concept is introduced in this paper. The time sharing MC will get the input for each time slot and what are the operations to be performed on that input also done on the same time slot. The MC has some complicated calculation compare to other stages in AES. Once the MC gets output from sbox it will starts its calculations and these calculations are performed parallel with the sbox operations.

The basic idea in this part of AES is all incoming bit to the MC have the GF over all fixed coefficients. In normal MC performs in column wise means it read the input for maximum of 4 times but in our paper the input is taken once and performs four of it operations within the same time period. The naming the signal plays important role in this concept. Naming and rearranging the signals are done in software itself we need not worry about the extra hardware thus does not increase the system area and cost. The same time we reduce the use of MC module from 4 to 1 will reduce the cost of constructions and area. The normal one round stages in AES is given in the below diagram.

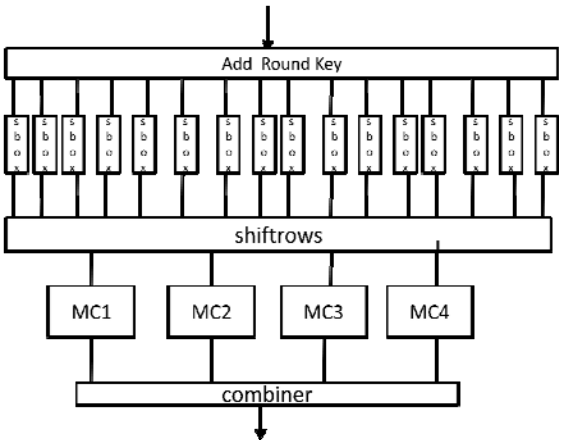


Fig.5. normal operation of subbyte and mixcolumn

In our proposed model each round key output share the same sbox and input to sbox based on time slot. The 16 x 1 mux is used to separate the 16 eight bit output and selection pins of mux is controlled by internal clock signal [7]. Then the output of sbox is directly given to mixcolumn stage. The operation of mixcolumn can be explained as below.

The MixColumns transformation operates on the State column-by-column, treating each Column as a four-term polynomial [9]. The columns are considered as polynomials over GF(2^8) and multiplied modulo $x4 + 1$ with a fixed polynomial $a(x)$, given by

$a(x) = \{03\}x3 + \{01\}x2 + \{01\}x + \{02\}$

This can be written as a matrix multiplication. Let

$s¢(x) = a(x)Äs(x)$:

$$\begin{bmatrix} s'_{o,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 02\ 03\ 01\ 01 \\ 01\ 02\ 03\ 01 \\ 01\ 01\ 02\ 03 \\ 03\ 01\ 01\ 02 \end{bmatrix} \begin{bmatrix} s_{o,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix}$$

for $0 £ c < \boldsymbol{Nb}$.

As a result of this multiplication, the four bytes in a column are replaced by the following:

S'0,c = ({ 0 2} • S0,c)⊕({ 0 3} • S1,c) ⊕ S2,c ⊕ S3,c

S'1,c = S0,c ⊕ ({ 0 2} • S1,c)⊕ ({ 03} • S2,c) ⊕S3,c

S'2,c = S0,c ⊕ S1,c ⊕ ({ 02} • S2,c)⊕ ({ 03} • S3,c)

S'3,c = ({ 0 3} • S0,c) ⊕ S1,c⊕ S2,c ⊕ ({0 2 • S3,c).

Sbox gets the input at a interval of 5 ns. Each 5 ns it gets the input and produces the 8 bit output then the mixcolumn performed on the output parallelly. For example "11010100" is the first output of sbox comes out at 5 ns and the mixcolumn performed on "11010100". "10111111" comes out next at 10 ns and MC performed parallelly. Since the inputs are pipelined with time slot[2] the MC also produce pipelined output[6], hence the speed of the AES architecture increased. By combining the signal properly we will get the required information.

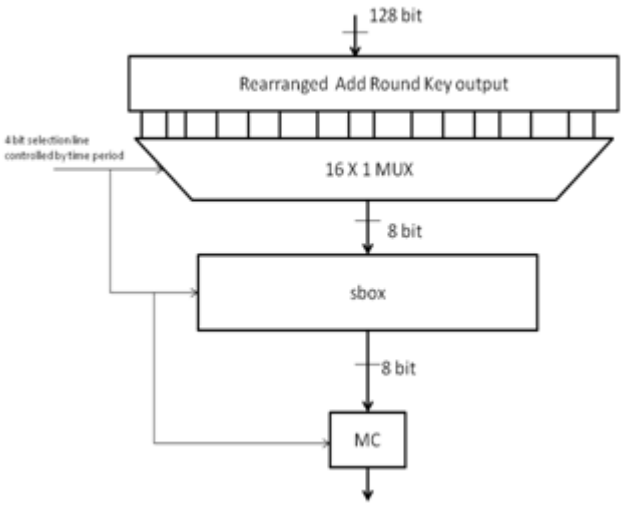


Fig.6. Sharing subbyte and high speed mixcolumn

Our proposed architecture eliminates shiftrows and input to MC is based on time slot. Thus each column shares the same mixcolumn module. These timing operations are controlled by clock period.

VI. ELIMINATION OF SHIFTROWS

We are all known that sbox can produce 8 bit output at a time. But shiftrows need total of 128 bit since each row is shifted according to their row number.

1. row0→ no change
2. row1→ one left shift
3. row2→ two left shifts
4. row3→three left shifts

Thus the stage between subbyte and mixcloumn consume large time period. Shifting the row is nothing but a left shift based on their row number. There no arithmetic operations, we can eliminate this stage [5] by rearrange the Add Round Key output which produces the same output as shiftrows. Add Round Key have 128 bit output always thus we rearrange the bit in this stage itself we can eliminate the use of internal registers between sbox and shiftrows. This can be explained by following example

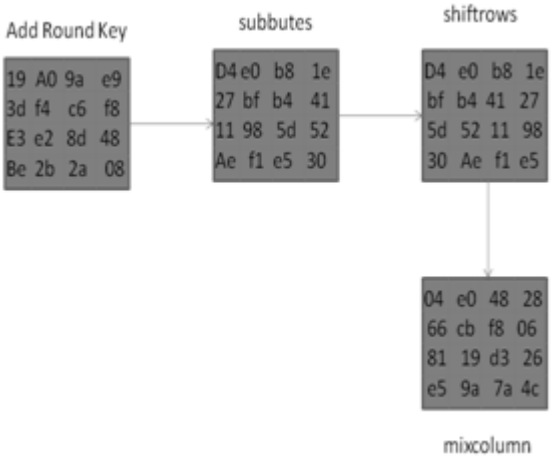


Fig.7. Normal operation of AES algorithm

Above the example we will know that the shift rows start its shifting operation only after getting 128 bit. This is the time consuming process same time there are no arithmetic operation. So we just shift the position of shiftrows before the subbyte operations where are 128 bit available for any time. The following diagram explains how is the output of Add Round Key rearranged and how can we eliminate the shiftrows. The output of both gives the same results.

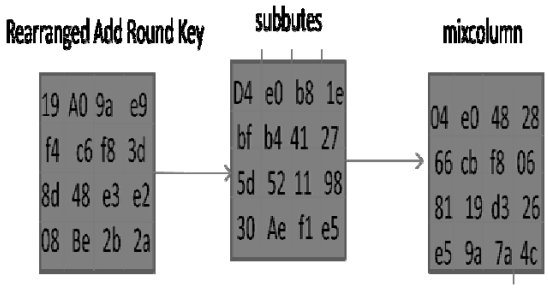


Fig.8. Improved AES algorithm

VII. FUNCTIONAL SIMULATION and SYNTHESIS RESULTS

A. Simulation result using Modelsim Altera 6.6c:

The coding is written by using VHDL language and then the code is simulated by using Modelsim Altera 6.6c version. The obtain waveform is given below

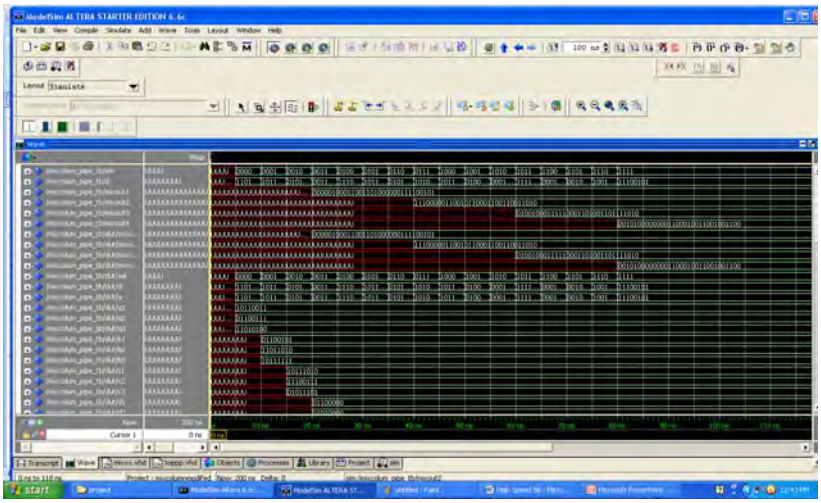


Fig.9. Simulation results of Improved SB and MC stage

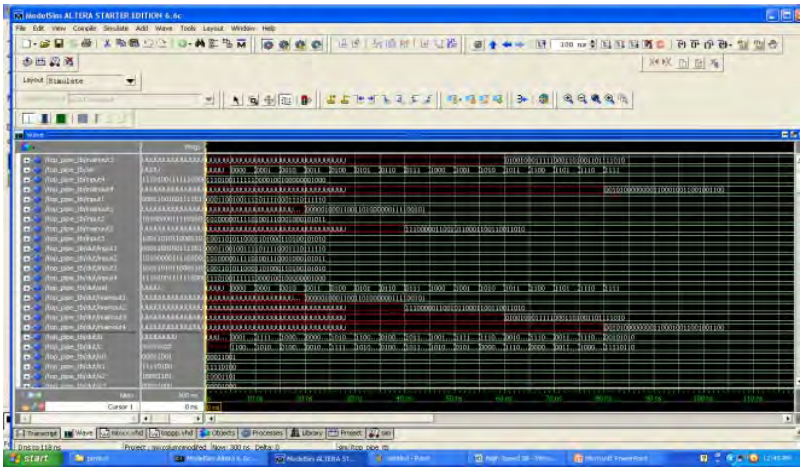
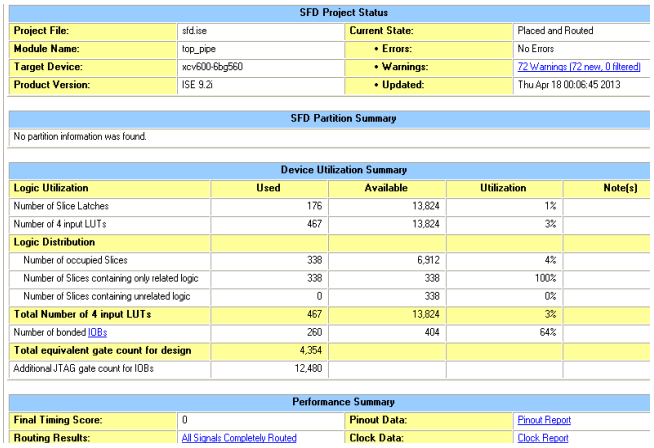


Fig.10. Pipelined MC output

The simulation result shows that the VHDL coding simulates properly and the test vector fed for the simulation gave the correct output.

B. Synthesis result of high speed SB and MC:

The simulated output then synthesized using ISE 9.2i. The Target Device is Virtex XCV600 BG 560– 6 Speed Grade:-6[8]. The synthesis results shows that all inputs are fitted correctly and all mapping functions and routing functions are done successfully.



SFD Project Status

Project File:	sfd.ise	**Current State:**	Placed and Routed
Module Name:	top_pipe	**• Errors:**	No Errors
Target Device:	xcv600-6bg560	**• Warnings:**	72 Warnings (72 new, 0 filtered)
Product Version:	ISE 9.2i	**• Updated:**	Thu Apr 18 00:06:45 2013

SFD Partition Summary

No partition information was found.

Device Utilization Summary

Logic Utilization	Used	Available	Utilization	Note(s)
Number of Slice Latches	176	13,824	1%	
Number of 4 input LUTs	467	13,824	3%	
Logic Distribution				
Number of occupied Slices	338	6,912	4%	
Number of Slices containing only related logic	338	338	100%	
Number of Slices containing unrelated logic	0	338	0%	
Total Number of 4 input LUTs	467	13,824	3%	
Number of bonded IOBs	260	404	64%	
Total equivalent gate count for design	4,354			
Additional JTAG gate count for IOBs	12,480			

Performance Summary

Final Timing Score:	0	**Pinout Data:**	Pinout Report
Routing Results:	All Signals Completely Routed	**Clock Data:**	Clock Report

Fig.11. synthesis results of time sharing SB and MC

The synthesis & mapping results of AES design are summarized in Table I.

Target Device	Virtex XCV600 BG 560–6
Optimization Goal	speed
Number of slices	270/6912 (3%)
Number of 4 input LUTs	467/3824 (3%)
Number of bounded IOBs	260/404 (64%)
Total memory usage	151304 kilobytes

Table.I. synthesis summary of time sharing SB and MC

V. CONCLUSION

This paper presented a fully pipelined implementation of the AES S-box and mixcolumn based on time sharing concept. Composite field S-Box becomes very compact and dissipates low power. We used a simple XOR function with fixed value improve the security of the S-Box of Wolkerstorfer et al. Time sharing concept reduces the resource requirement and elimination of shiftrow reduces the delay. The presented S-Box and MC was combine simulated using modelsim 6.6c altera technology. The simulation and synthesis results show that our design is the best choice for applications requiring small silicon area, low power consumption and high security.

REFERENCES

[1] Abdel alim kamal and Amr M.Youssel *"An Area-Optimized implementation for AES with Hybird countermeasures against power analysis"* IEEE 9781-4244-3786-3/09.

[2] Ahmed Rady, Ehab EL Shehely,A.M EL Hennawy *"Design and implementation of area optimized AES algorithm on reconfigurable FPGA"*IEEE volume 3 2007.

[3] Announing the Advanced Encryption standards (AES),Federal Information processing standards publication,2001.

[4] J.Deamen and Vincent Rijimen, *"A Specification for the AES algorithm Rijidael"*.

[5] Krishnamurthy GN,V.Ramaswamy *"Study of Removal of shiftrows and mixcolumn stage of AES and AES-KDS on their Encryption and hence security"* World Academy of Science and Technology 50,2011.

[6] M M.Wong and M.L.D *Wong "A High Throughput Low Power Composite field Arithmetic and Algebraic Normal Form Reprentation* "IEEE vol 8,2010.

[7] M.R.M Rizk,M.Morsy *"Optimized Area and Optimized speed Hardware Implementation of AES on FPGA"* IEEE vol 1,2007.

[8] Pravin B. Ghewari et al *"Efficient Hardware Design and Implementation of AES Crypo System"*IJESTvol2(3),2010.

[9] W.Stallings ,Cryptography and Network Security, Prentice Hall, 3rd ed,2003.