Communications on Applied Electronics (CAE) – ISSN : 2394-4714
Foundation of Computer Science FCS, New York, USA
International Conference on Communication Technology (ICCT 2015) – www.caeaccess.org

Combined Architecture for AES Encryption and Decryption using FPGA

Poonam Kadam
Assistant Professor, EXTC,
DJ Sanghvi College of Engineering
Vile Parle (W), Mumbai, India

Nilima D. Parmar
PG student, EXTC
DJ Sanghvi College of Engineering
Vile Parle (W) Mumbai, India

ABSTRACT
This paper presents a combined architecture of Advanced Encryption Standard-128 encryption and decryption for high speed application. A select line named enc/dec is used to select either of the two operations. If enc/dec is 0, then encryption will take place and if it's 1 then decryption. Pipelining and sub-pipelining is used to enhance the speed of operation. Use of 9 stage sub-pipelining per round unit gives a throughput of 18.773 Gbps on Xilinx Virtex XCV3200E-8-BG560 device whereas it gives a throughput of 24.930 Gbps on SPARTAN 3 XC3S4000-5fg676 device.

Keywords
AES; Rijndael; pipelining; encryption; decryption

1. INTRODUCTION
In 2001 the National Institute of Standards and Technology declared Rijndael cipher also known as Advanced Encryption Standard (AES), as Federal Information Processing Standard (FIPS-197). It is a symmetric-key cipher which encodes or decodes 128 bits of data with the help of cryptographic key of length 128, 192 or 256 bits.

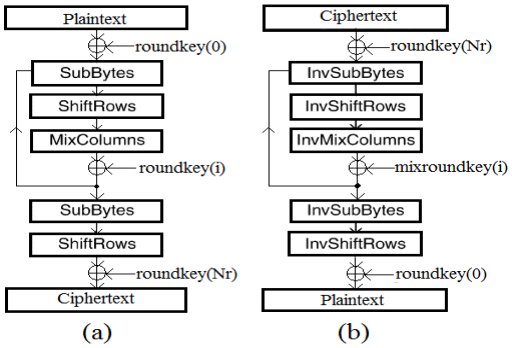

Fig.1: (a) Encryption; (b) Decryption.

Ever since the announcement of Rijndael in 2001, continuous attempts have been made to improve its performance by implementing on various platforms from software to hardware. The works reported in [2]-[6] used look-up tables (LUTs) for the implementation of SubByte and InvSubByte. However this approach is not suitable and practical for high speed and low area requirement since they introduce an unbreakable delay within the architecture and also makes pipelining difficult. Hence further works were done to look for an alternative, and as a solution, [7] and [8] introduced combinational logic only implementation of SubByte and InvSubByte using composite field arithmetic. This was further explored in [9]. Compared to software implementations, hardware implementations of the AES algorithm provide more physical security as well as higher speed [9]. Also many works have been published studying the effects of introducing pipelining and sub-pipelining to enhance the speed of AES as done in [9] and [12].In the paper [22], the AES-128 encryptor was designed for loop unrolled architecture. Further, 8 sub-stages of pipelining were implemented on Xilinx Virtex XCV3200E-8-BG560 device. This increased the maximum operating clock frequency by around 13% as compared to that reported in [14]. This was achieved due to reduction in the delay along the critical path. In this paper, a combined architecture for AES-128 which can do encryption or decryption with the same hardware by using a select line named as enc/dec. If enc/dec=0 then encryption will occur and if it's 1 then decryption. A total of 9 sub-pipelining stages are used to achieve operating frequency of 146.67 MHz on Xilinx Virtex XCV3200E-8-BG560 device compared to just 66 MHz as proposed in [14] and 25.3 MHz as in [13].

2. LOGICAL DESIGN
The Fig.2 shows the logical design and flow of the algorithm for combined architecture. As discussed above enc/dec helps select the type of operation. The Key Expansion unit takes 128 bits key as input and generates 128 bits of key for every round (roundkey). During encryption, for round 0 the roundkey (0) is the original input key. This key is then used further to generate roundkeys for future rounds. In decryption process, the key used for round 0 is the roundkey (10) as used in encryption. For $1 \le i \le 9$, InvMixColumn needs to be applied to produce the mixroundkeys. However for final round the original input key by the user is used.

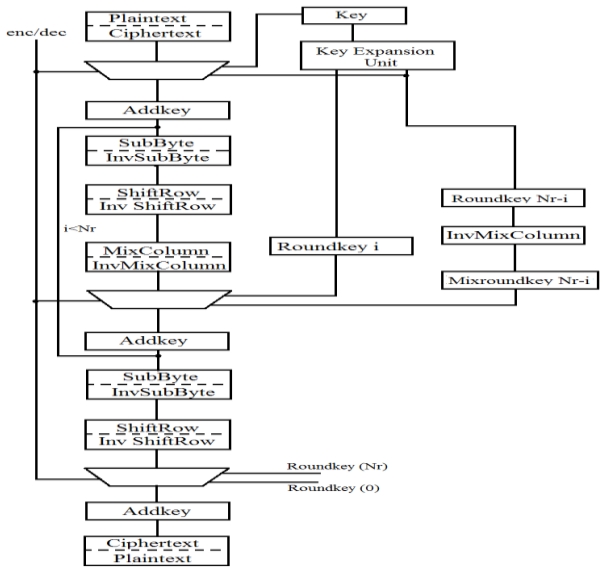

Fig.2: Logical flow for combined encryption and decryption

3. AES ARCHITECTURE

3.1 SubByte / InvSubByte Transformation

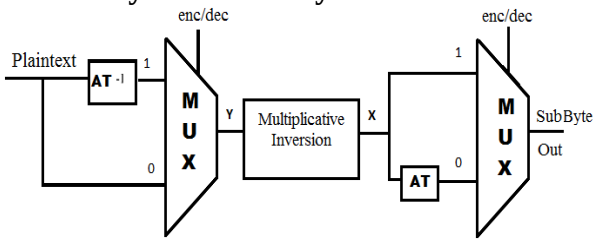

Fig.3: Combined SubByte and InvSubByte [9].

The Multiplicative Inversion block is common to both SubByte and InvSubByte hence it is shared with both the transformations resulting in reduced usage of slices. To speed up the operation 6 stages of sub-pipelining is implemented in this component.

3.2 ShiftRow/InvShiftRow

ShiftRow shifts the rows cyclically left. The first row is left untouched while the second, third and fourth rows are shifted towards left by one, two and three bytes respectively. The same process happens in InvShiftRow except it shifts towards right.

3.3 MixColumn/InvMixColumn

The equations for MixColumn are as discussed in [22]. Equation 1 represents the equations for each element of InvMixColumn.

$$s'_{0,c} = \{02\}_{16}(s_{0,c} + s_{1,c}) + (s_{2,c} + s_{3,c}) + s_{1,c} + \{02\}_{16}(\{04\}_{16}(s_{0,c} + s_{2,c}) + \{04\}_{16}(s_{1,c} + s_{3,c})) + \{04\}_{16}(s_{0,c} + s_{2,c})$$

$$s'_{2,c} = \{02\}_{16}(s_{2,c} + s_{3,c}) + (s_{0,c} + s_{1,c}) + s_{3,c} + \{02\}_{16}(\{04\}_{16}(s_{0,c} + s_{2,c}) + \{04\}_{16}(s_{1,c} + s_{3,c})) + \{04\}_{16}(s_{0,c} + s_{2,c})$$

$$s'_{1,c} = \{02\}_{16}(s_{1,c} + s_{2,c}) + (s_{3,c} + s_{0,c}) + s_{2,c} + \{02\}_{16}(\{04\}_{16}(s_{0,c} + s_{2,c}) + \{04\}_{16}(s_{1,c} + s_{3,c})) + \{04\}_{16}(s_{1,c} + s_{3,c})$$

$$s'_{3,c} = \{02\}_{16}(s_{3,c} + s_{0,c}) + (s_{1,c} + s_{2,c}) + s_{0,c} + \{02\}_{16}(\{04\}_{16}(s_{0,c} + s_{2,c}) + \{04\}_{16}(s_{1,c} + s_{3,c})) + \{04\}_{16}(s_{1,c} + s_{3,c})$$

$0 \le c \le 4$ (1) [9]

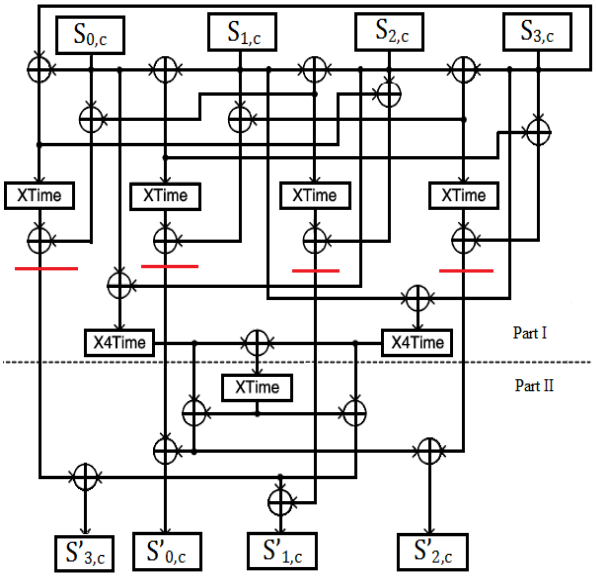

Fig.4: Combined architecture for MixColumn and InvMixColumn [9]

The Fig. 4 above shows the combined architecture for the Mix and InvMixColumn transformation. The Part I is used for MixColumn where the red line indicates the corresponding MixColumn output. Similarly Part II uses Part I to produce corresponding output for InvMixColumn. The dotted line indicate a pipelined register to cut down the complexity and increase the speed of operation.

3.4 Key Expansion

This unit is responsible to produce the roundkeys for each round which are added with the output after each round during encryption. Similarly mixroundkeys are generated during decryption as already explained in section II.This design uses a Key Expansion module which can generate the roundkeys on fly i.e its dynamic in nature. The 128 bits input key is divided into four group of 32 bits key as shown in Fig. 5 SubWord: It's a SubByte transformation on 32 bits word. In order to reduce the slices usage the same SubByte transformation is used as in the main algorithm. This also takes care of pipelining and no additional pipelining is required in the key expansion unit.RotWord: It rotates the 32 bits key left by 8 bits.RCon: FIPS-197 defines a round constant for every round. It has a unique value for every round [1]. It can be either implemented as LUT or on fly.

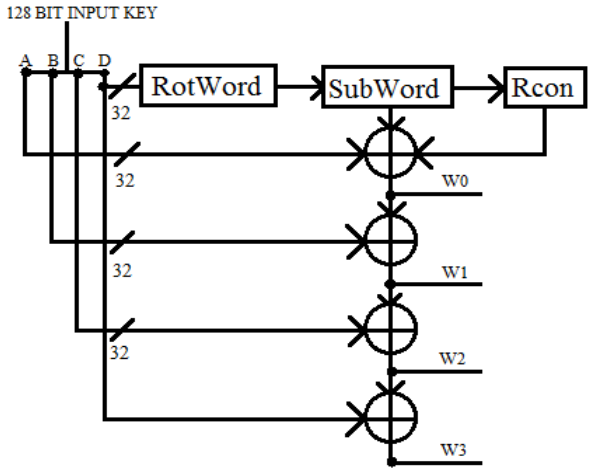

Fig.5: Key expansion

The Fig. 6 shows the hardware implementation of the combined encryption and decryption for one round unit along with the placement of the sub-pipelining registers. The registers are placed approximately at equal time delay to distribute the complexity equally throughout the architecture.

4. IMPLEMENTATION AND RESULTS

The combined architecture for AES-128 is implemented by creating 10 copies of the hardware for one round unit and one key expansion unit. 10 copies implies repeating the hardware for one round unit 10 times. This complete design is implemented on two FPGA platforms viz, VirtexE and SPARTAN 3. The tool used for synthesis and post implementation and timing result is Xilinx ISE 9.2i. The sub-pipelined structure can continuously accept input and produce output at every clock cycle. However for the initial 128 bits of input, it takes m x Nr +1 clock cycles to produce a valid output, where m is the number of sub-pipelining stages. In this design m=9.There is an initial additional delay in decryption process compared to encryption process. For this design in terms of clock cycles it takes 68 cycles more than that in encryption. This is due to the requirement of the last roundkey for the first round. Hence all the keys for all the rounds need to be generated at the beginning itself. On the FPGA VirtexE platform (XCV3200E-8-BG560 device), the proposed architecture operates at a maximum frequency of 146.67 MHz as per the synthesis report and gives a throughput of 18.773 Gbps. When implemented on SPARTAN 3 platform (XC3S4000-5fg676 device) an operating frequency of 194.77 MHz is obtained with a throughput of 24.930 Gbps. These results, as shown in table 1, obtained are better than the previous proposed design in [13] [14] for combined architecture in terms of maximum clock frequency, throughput and throughput/slice.

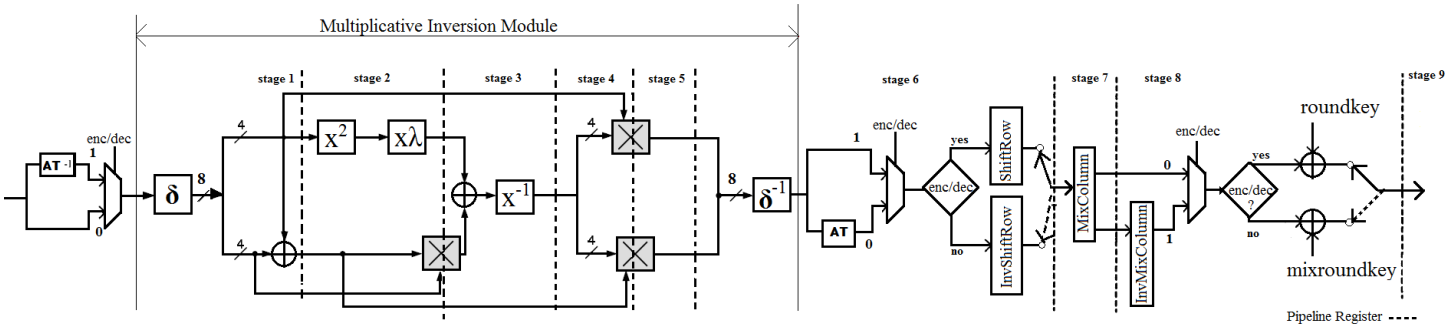

Fig.6: Hardware implementation of sub-pipelined Encryptor/Decryptor for one round unit.

Table 1. Comparison of the FPGA implementation of AES algorithm

Design	Slices	Frequency (MHz)	Throughput (Gbps)	Mbps/slice
Mccanny[13] XCV3200e-8	7576 + 102 BRAM	25.3	3.239	0.1569
Deng[14] CMOS	NA	66	844.8	NA
This work XCV3200e-8	16208	146.67	18.773	1.15825
This work XCS4000-5	17003	194.77	24.930	1.4662

5. CONCLUSION

A combined architecture which can perform either encryption or decryption with the help of a select line has been successfully presented in this paper. Pipelining and sub-pipelining has been explored to obtain high speed of operation. Increasing the number of pipelining and sub-pipelining stages need not necessarily increase the operating clock frequency. The placement of pipelining register and careful construction of the logic within the available resources also contributes to an efficient architecture. Further, resource sharing has been done to keep the total area consumption under control. Future work will address to bringing down the slice usage even further without compromising the speed.

6. REFERENCES

[1] FIPS 197, "Advanced Encryption Standard (AES)", November26,2001 http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf

[2] K. Gaj and P. Chodowiec, "Comparison of the hardware performance of the AES candidates using reconfigurable hardware". Presented at Proc.3rd AES Conf. (AES3).

[3] J. Elbirt, W. Yip, B. Chetwynd, and C. Paar, "An FPGA implementation and performance evaluation of the AES block cipher candidate algorithm finalist", presented at Proc. 3rd AES Conf. (AES3).

[4] H. Kuo and I. Verbauwhede, "Architectural optimization for a 1.82 Gbits/sec VLSI implementation of the AES Rijndael algorithm," in Proc. CHES 2001, Paris, France, May 2001, pp. 51–64.

[5] M. McLoone and J. V. McCanny, "Rijndael FPGA implementation utilizing look-up tables," in IEEEWorkshop on Signal Processing Systems, Sept. 2001, pp. 349–360.

[6] V. Fischer and M. Drutarovsky, "Two methods of Rijndael implementation in reconfigurable hardware," in Proc. CHES 2001, Paris, France, May 2001, pp. 77–92.

[7] Akashi Satoh, Sumio Morioka, Kohji Takano and Seiji Munetoh, "A Compact Rijndael Hardware Architecture with S-Box Optimization.", Springer-Verlag Berlin Heidelberg, 2001

[8] A. Rudra, P. K. Dubey, C. S. Jutla, V. Kumar, J. R. Rao, and P. Rohatgi,"Efficient implementation of Rijndael encryption with composite field arithmetic," in Proc. CHES 2001, Paris, France, May 2001, pp. 171–184.

[9] Xinmiao Zhang and Keshab K. Parhi, "High-Speed VLSI Architectures for the AES Algorithm," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol.12, No. 9, Septemper 2004.

[10] K. U. Jarvinen, M. T. Tommiska, and J. O. Skytta, "A fully pipelined memoryless 17.8 Gbps AES-128 encryptor," in Proc. Int. Symp. Field-Programmable Gate Arrays (FPGA 2003), Monterey, CA, Feb. 2003, pp. 207–215.

[11] G. P. Saggese, A. Mazzeo, N. Mazocca, and A. G. M. Strollo, "An FPGA based performance analysis of the unrolling, tiling and pipelining of the AES algorithm," in Proc. FPL 2003, Portugal, Sept. 2003.

[12] F. Standaert, G. Rouvroy, J. Quisquater, and J. Legat, "Efficient implementation of Rijndael encryption in reconfigurable hardware: Improvements & design tradeoffs," in Proc. CHES 2003, Cologne, Germany,Sept. 2003.

[13] John McCanny, Maire McLoone, "Apparatus for selectably encrypting or decrypting data", US Patent 20030053623 A1, Mar 20, 2003

[14] Liang Deng', Hongyi Chen,"A new VLSI implementation of the AES algorithm", IEEE 2002 International Conference on Communications, Circuits and Systems and West Sino Expositions, Volume 2,2002

[15] Naga M. Kosaraju, Murali Varanasi and Saraju P. Mohanty "A High-Performance VLSI Architecture for Advanced Encryption Standard (AES) Algorithm," IEEE Proceedings of the 19th International Conference on VLSI Design (VLSID'06).

[16] Amruta Page, P. V. Sriniwas Shastry, "AES-128 Key Expansion with LUT and OTF S-Box," International Journal of Computer Technology and Electronics Engineering (IJCTEE) Volume 4, Issue 3, June 2014, An ISO 9001: 2008 Certified Journal

[17] Edwin NC Mui, "Practical Implementation of Rijndael S-Box Using Combinational Logic."

[18] Marian Cretu1and Cristian-Gabriel Apostol "A Modified Version of Rijndael Algorithm Implemented to Analyze the Cyphertexts Correlation for Switched S-Boxes"IEEE conference on Communication (COMM), Bucharest,2012

[19] K. U. Jarvinen, M. T. Tommiska, and J. O. Skytta, "A fully pipelined memoryless 17.8 Gbps AES-128 encryptor," in Proc. Int. Symp. Field-Programmable Gate Arrays (FPGA 2003), Monterey, CA, Feb. 2003, pp. 207–215.

[20] Ion Sima, Adrian- viorel Diaconu and Marian Cretu. "Analysis of Modified ShiftRows and MixColumns Transformations in Rijndael Algorithm" IEEE conference on Electronics, Computers and Artificial Intelligence (ECAI), 2013.

[21] Poonam Kadam, Nilima Parmar, "Pipelined Implementation of Dynamic Rijndael S-Box" in International Journal of Computer Applications, Vol. 111 (19578-1384), 2015

[22] Nilima D. Parmar and Poonam Kadam, "High Speed Architecture Implementation of AES using FPGA", Proceedings of ICCT-2015, 25th and 26th September 2015, Mumbai, India, (IJCA – ICCT-2015, No. 7, pp. 27 – 30, September 2015, http://www.ijcaonline.org/proceedings/icct2015/number7/22684-1590, ISBN : 973-93-80888-65-6