Implementation of T-box/T^{-1}-Box Based AES Design on Latest Xilinx FPGA

DUR-E-SHAHWAR KUNDI*, AND ARSHAD AZIZ*

RECEIVED ON 12.01.2015 ACCEPTED ON 28.05.2015

ABSTRACT

This work presents an efficient implementation of the AES (Advance Encryption Standard) based on T-box/T^{-1}-box design for both the encryption and decryption on FPGA (Field Programmable Gate Array). The proposed architecture not only make efficient use of full capacity of dedicated 32 Kb BRAM (Block RAM) of latest Xilinx FPGAs (Virtex-5, Virtex-6 and 7 Series) but also saves considerable amount of BRAM and logical resources by using multiple accesses from single BRAM in one cycle of system clock as compared to conventional LUT (Look-Up-Table) techniques. The proposed T-box/T^{-1}-box based AES design for both the encryption and decryption fits into just 4 BRAMs on FPGA and results in good efficiency TPS (Throughput per Slice) with less power consumption.

Key Words: Advance Encryption Standard, BRAM, Field Programmable Gate Array, T-Box/T^{-1}-Box

1. INTRODUCTION

The Rijndael algorithm [1] was announced as the AES Algorithm in November, 2001 by NIST (National Institute of Standard and Technology) [2], for the protection of sensitive information. It is a symmetric block cipher that takes 128-bit block of data as input that is arranged in a 4x4 matrix of 16 bytes called a State matrix. The algorithm consists of round function that are executed for specific number of iterations followed by a final round as shown in Fig. 1. Each normal round is composed of a sequence of four transformations: SubByte, ShiftRow, MixColumn and AddRoundKey while the last round composed of only three transformations; SubByte, ShiftRow and AddRoundKey. The number of rounds to be executed depends upon the key size as shown in Table 1.

Reconfigurable platforms such as FPGAs are ideal for implementation of cryptographic algorithms. Modern Xilinx FPGAs offers unique features such as DCM (Digital Clock Manager), on-chip BRAM memories, DSP slices and multipliers etc that can be efficiently utilized to enhance the performance of cryptographic systems [3].

The dedicated memory elements (BRAM) in latest Xilinx FPGAs (Virtex-5, Virtex-6 and 7 series) have storage capacity of 32 Kb that can be used to implement large LUT and is almost double than the BRAM available in other FPGA families (Spartan-3, Virtex-4, Virtex-II). Using the dedicated FPGA resources to improve the performance of LUT based FPGA implementations is not new but if previous proposed designs are implemented on latest Xilinx FPGAs, they result in an inefficient utilization of only 25-50% of 32 Kb memory. So there is need to propose a new design that utilizes the BRAM more efficiently.

* Department of Electrical Engineering, National University of Sciences & Technology, Islamabad.

In this paper we present the efficient implementation of a LUT based AES design for both the encryption and decryption with fast accessing. The rest of this paper is organized as follows; section 2 presents a survey of LUT based AES designs while in section 3, we explain our design implementation. In section 4 implementation results and comparisons are given and section 5 contains the conclusions.

2. LOOK-UP-TABLE BASED DESIGN

The implementation of both MixColumn and SubByte transformation are a critical part in the design of an efficient AES encryption core. The SubByte alone consumes 75% of the area resources [4], when implemented using FPGAs

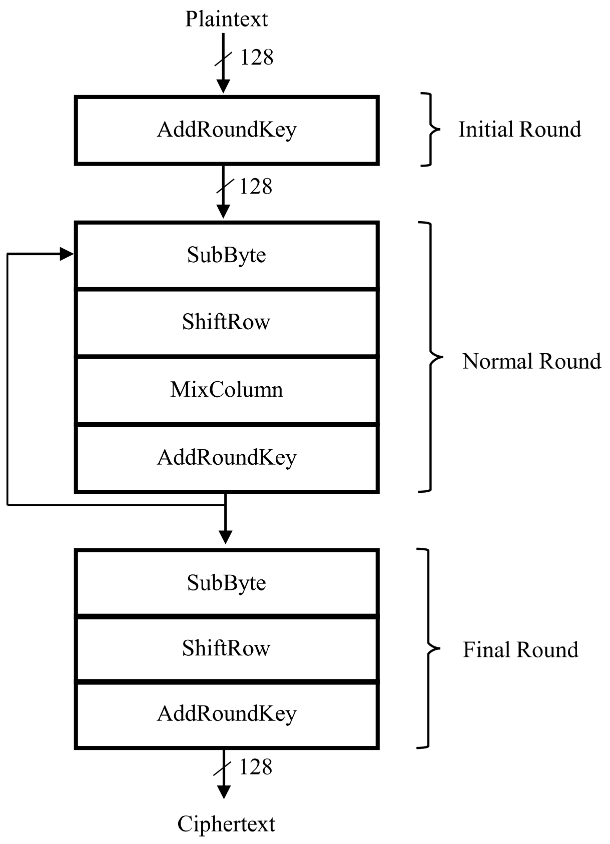

FIG. 1. AES ENCRYPTION ARCHITECTURE

TABLE 1. NUMBER OF ROUNDS AS A FUNCTION OF KEY SIZE

Key Length	128-bits	192-bits	256-bits
Round Number	10	12	14

logical resources. So for the efficient implementation of AES encryption and decryption cores on FPGA, LUT based techniques are used [5], in which all the computational complexities and critical delays of SubByte and MixColumn are replaced by a simple pre-computed tables called the S-box [6-8] and the T-box [9-11]. An S-box is a LUT for SubByte step only with a total size of 2 Kb (256 x 8-bit) while a T-box is the combination of both SubByte and MixColumn transformations into a single LUT of 8 Kb (256 x 32-bit).

To achieve maximal performance it is essential to access the S-box or T-box table 16 times concurrently. Therefore, 16 instances of such single table have to be implemented, which occupies considerable device resources. Similarly for AES decryption, the S^{-1}-box or T^{-1}-box also consumes further area resources due to asymmetric S-box and S^{-1}-box tables and also different encryption and decryption multiplication coefficients [2]. In [11], the author has presented 24-bit T-box entries instead of using 32-bit values for each byte and this approach consumes 3107 slices for AES encryption only but same idea will not be applied to T^{-1}-box as it contains four different multiplication constants (09, 0b, 0d, 0e) and will consumes more than the above reported slices. Also the asymmetric hardware architecture of 24-bit T-box and 32-bit T^{-1}-box will results in unshared XORed hardware [12].

However BRAMs in the latest Virtex series [13] of Xilinx FPGAs are ideal for implementation of the T-box as they can stores up to 36 Kb (32 Kb for data and 4 Kb for parity) and can be configured as either two independent 18 Kb RAMs or one 36 Kb RAM. So it will be advantageous to use the latest families of FPGA as its BRAM have double memory storage ability as compared to other FPGAs family devices. However, the main problem with BRAMs is that they are synchronous in nature and each memory location can be accessed a single time in one clock cycle.

The conventional implementation of AES encryption can only access one memory location per port and accommodates only one T-box per BRAM using a single port configuration or at most two T-boxes using dual-port configuration there by utilizing 25-50% space of 32 Kb BRAM and resulting in an inefficient design. Similarly for the AES decryption also. In recent T-box based iterative AES architectures [14-16], the authors used 8 BRAMs in dual-port mode for encryption by saving one T-box table per port and another 8 BRAM for decryption by saving one T^{-1}-box table per port, thus results in inefficient BRAM utilization. The LUT based iterative designs used a maximum of 16 BRAMs but if the same design is employed in an unrolled architecture it will result in 10 times more area usage [17-19]. A well-known T-box based unrolled AES architecture presented by McLoone and McCanny [17] used 224 BRAMs (storing T-box tables) for encryption only while an S-box based unrolled AES architecture presented in [18] used 400 BRAMs for both encryption and decryption. Similarly in [19], Ali, et. al. consumed 50% of the Block memory resources and 3560 ALMs to implement the encryption module, which make impossible to implement a decryption module on the same device.

3. DESIGN IMPLEMENTATION

Our proposed AES T-box/T^{-1}-box design includes the integration of all these transforms (MixColumn and SubByte, Inverse MixColumn and Inverse SubByte) into one single-unit for both the encryption and decryption. This not only utilizes BRAM to its full capacity but also reduces the BRAM resources by 50%. Further BRAM requirement for the design is reduced down to 50% by increasing the frequency of BRAM clocks to enable multiple memory accesses in one clock cycle.

The architecture of our proposed efficient T-box/T^{-1}-box based AES design is shown in Fig. 2. It consists of three main sections; Selection Circuitry, LUT, Registers Bank and a DCM. The selection circuitry consists of five 2x1 multiplexers (*M0-M4*). The *M0* multiplexer is used to select initially 64- bit plaintext and then feedback data for the remaining rounds with its control signal (*Ctrl*) while the remaining four 2x1 multiplexers (*M1-M4*) are used to make selection between two bytes (16-bit) on the basis of its controlling signals (C_0).

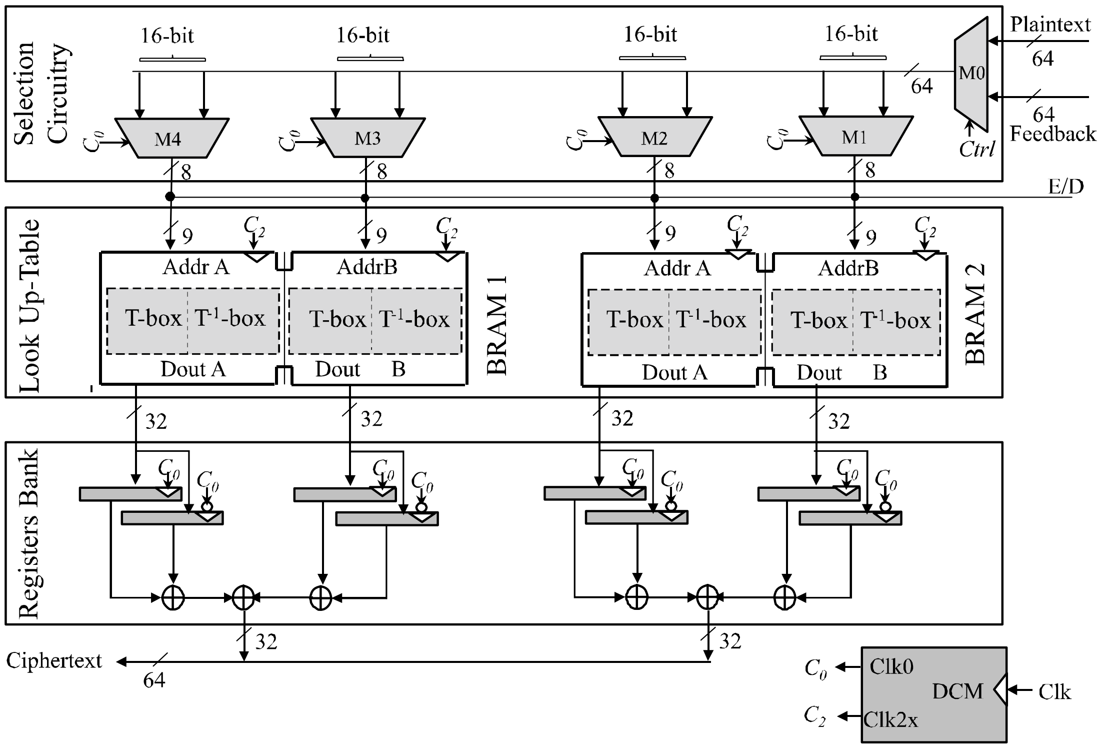

FIG. 2. 64-BIT AES T-BOX/T^{-1}-BOX DESIGN

The selected bytes are then applied to the LUT section that consists of two BRAMs. Each BRAM is configured as ROM and each port of it, is initialized with a copy of 16 Kb joint T-box/T^{-1}-box look-up values using coefficient (COE) file. Using the ROM configuration, we only read look-up data from BRAM by applying corresponding input bytes at its address ports; *Addr A* and *Addr B*. The 32-bit outputs corresponding to each applied address are obtained from output ports; *Dout A* and *Dout B*. Each dual port BRAM is used as two separate single-port BRAMs and each single-port BRAM accommodates one T-box and one T^{-1}-box table. Thus a single dual-port BRAM accommodates two T-box and two T^{-1}-box tables used equally during the encryption and decryption. The selection between the T-box and T^{-1}-box is done through the input signal (*E/D*). In order to accommodate 16 Kb (512x32-bit) look-up data per BRAM port, 9-bit addressing is used for both *Addr A* and *Addr B*. The *E/D* signal in combination with the input byte value forms a total of 9-bit for addressing.

To produce multiple outputs from a single BRAM in one cycle of system clock (*Clk*), we have used DCM [20]. A DCM in Xilinx FPGA provides a variety of advance features such as being able to multiply or divide an input clock and to produce multiple phase-shifted clocks. It is used here to produce two clock variants such as *Clk0* and *Clk2x*. The frequency of *Clk0* (also as *C0*) is same as *Clk* which is used as select inputs for multiplexers and registers while *Clk2x* (also as *C2*) whose frequency is twice the frequency of *Clk* is applied at BRAM clock ports that enables fast access of BRAM. The frequency of the BRAM clock has a direct relation with the number of BRAMs used for 16 look-ups. By using the simple *Clk*, 8 BRAMs will be used for 16 look-ups while in case of using double frequency clock (*Clk2x*) for BRAMs clocking, the number of BRAMs will be reduced by 50% i.e. 4 BRAMs.

The 32-bit T-box/T^{-1}-box output from each BRAM port are then stored in register banks that consists of eight 32-bit registers; 4 positive edge triggered and 4 negative edge triggered. The appropriate positive or negative edge 32-bit register is selected with the help of the controlling signal *C0*. Each four 32-bit T-box/T^{-1}-box registered outputs are then XORed according to MixColumn transformation [2] to produce an output of a single 32-bit (1 column) of AES encryption or decryption and further with pre-computed round key. This section is designed in such a way as to maintain the symmetry of operation during both encryption and decryption. Finally 64-bit ciphertext is obtained in one clock cycle while in meantime next applied 64-bit feedback is also in pipelined and will be obtained at the next clock cycle there by maintaining the latency of one clock cycle. For full 128-bit AES, the 64-bit instant is replicated twice to utilize total of four BRAMs.

4. IMPLEMENTATION RESULTS AND COMPARISONS

The design has been implemented and simulated using Xilinx ISE Foundation 14.7 and Xilinx ISim 14.7 respectively. Our design is operating at a clock speed of 350 MHz on Virtex-7 FPGA and it takes 20 clock cycles to process two 128-bit data block of AES in pipelined. The full AES design for both encryption and decryption fits into four BRAMs and 193 Slices with the help of multiple accessing circuitry and symmetric register bank architecture. And also the full 32 Kb capacity of single BRAM is efficiently utilizes up to 100%.

A comparison of our results with reported LUT based AES designs is given in Table 2. The comparison table clearly shows that our efficient T-box/T^{-1}-box based AES for both encryption and decryption utilizes 50% less BRAMs resources without requiring any additional rearrangement circuitry to reuse the register bank during both encryption and decryption. McLoone and McCanny [17] used 8 BRAMs each for storing 2 Kb tables of multiplication constants 01, 02 and 03 during encryption and 8 BRAMs each for storing 2 Kb tables of decryption multiplication constants 09, 0b, 0d, 0e. So they utilized a total of 56 (7x8) BRAMs for 16 parallel substitutions in a single round. Implementations [14-16] used 8 BRAMs for storing a full 8 Kb T-box containing all possible multiplication constant during encryption and 8 separate BRAMs for storing 8 Kb decryption LUT (T^{-1}-box). If the same designs are implemented using the 32 Kb BRAM, [17] will inefficiently consume 25% of BRAM storage while [14-16] will consume only 50%.

Further Rouvroy, et. al. [21] combined the encryption and decryption LUT and presented 32-bit design in which they used 2 BRAMs in dual port mode, so for 128-bit it is estimated to require 8 BRAMs. Their implementation results in 50% less BRAM resources as compared to other implementations.

However, they used extra hardware as there is no symmetric relation in their LUT entries for encryption and decryption which in turn makes use of selection circuitry to arrange the LUT entries in order to use the same XOR circuitry and also used an extra multiplexer to select the final encrypted or decrypted output that will impose extra delays.

In order to further compare the effectiveness of our proposed design, we provided TPS parameter which is used to compare two FPGA implementations and also the estimated power. For the calculation of TPS, a 32 Kb BRAM is taken equivalent to 128 slices while 16 Kb BRAM to 64 slices. It is cleared from Table 2, our proposed AES design has highest TPS as compared to the previous reported implementations and also results in less power consumption.

5. CONCLUSION

In this work we have presented an efficient T-box/T^{-1}-box based AES implementation for both the encryption and decryption with a multiple access and symmetric hardware architecture that not only results in efficient utilization of 32 Kb BRAMs of latest Xilinx FPGAs but also reduces the overall memory requirements for both encryptor and decryptor as compared to previous conventional LUT based approaches. Our proposed design results in good efficiency (TPS) and also consumes less power which can be used in any compact design. Our future work includes the performance enhancement of the proposed design to maximize throughput.

ACKNOWLEDGMENT

Authors express thanks to anonymous reviewers for their valuable comments in improving this work.

TABLE 2. COMPARISON RESULTS OF LUT IMPLEMENTATION

Authors	Device/ BRAM Size	Data Path	E/D	Area (# BRAMs, Slices)	Frequency (MHz)	Throughput (Mbps)	TPS (Mbps/Area)	Power (mW)
[21]	Virtex-2/16Kb	32	E/D	2, 146	123	358	1.306	-
[17]	Virtex-E/4Kb	128	E & D	(24+32),...	93.3	1194	0.666	-
[14]	Virtex-4/16Kb	128	E & D	(8+8), 1920	250	2900	0.985	-
[15]	Virtex-5/32Kb	128	E	(8,624) x 2	550	6704	2.033	503
[16]	Spartan-2/16Kb	128	E/D	16, 301	321.67	4117	3.107	-
Our Work	Virtex-7/32Kb	128	E/D	4, 193	350	4480	6.354	274
	Virtex-6/32Kb	128	E/D	4, 210	300	3840	5.318	283
	Virtex-5/32Kb	128	E/D	4, 332	251.42	3218	3.813	314

REFERENCES

[1] Daemen, J., and Rijmen, V., "AES Proposal: The Rijndael Block Cipher", pp. 1-45, 1999.

[2] FIPS-197, "Advanced Encryption Standard", National Institute of Standards and Technology, 2001.

[3] Xilinx, "7 Series FPGAs Overview v1.15", Technical Report, 2014.

[4] Aziz, A., and Ikram, N., "An Efficient FPGA Based Sequential Implementation of Advanced Encryption Standard", ITI 3rd International Conference on Information and Communications Technology, pp. 875-882, Cairo, Egypt, 2005.

[5] Fischer, V., and Drutarovsky, M., "Two Methods of Rijndael Implementation in reconfigurable hardware", Proceedings of 3rd International Workshop on Cryptographic Hardware and Embedded Systems, pp. 77-92, Springer-Verlag, 2001.

[6] Standaert, F.X., Rouvroy, G., Quisquater, J.J., and Legat, J.D., "Efficient Implementation of Rijndael Encryption in Reconfigurable Hardware: Improvements and Design Tradeoffs", Cryptographic Hardware and Embedded Systems Volume 2779, Lecture Notes in Computer Science, pp. 334-350, Springer Berlin Heidelberg, 2003.

[7] Kundi, D.S., Zaka, S., and Aziz, A., "A Compact AES Encryption Core on Xilinx FPGA", International Conference on Computer, Control and Communication, pp. 1-4, Karachi, Pakistan, 2009.

[8] Zambreno, J., Nguyen, D., and Choudhary, A., "Exploring Area/Delay Tradeoffs in an AES FPGA Implementation", Proceedings of 14th Annual International Conference on Field-Programmable Logic and Applications, Lecture Notes in Computer Science, Volume 3203, pp. 575-585, Springer, 2004.

[9] McLoone, M., and McCanny, J., "High Performance Single-Chip FPGA Rijndael Algorithm implementations", Cryptographic Hardware and Embedded Systems, Lecture Notes in Computer Science, pp. 65-76, Springer Verlag, 2001.

[10] Wang, J.F., Chang, S.W., and Lin, P.C., "A Novel Round Function Architecture for AES Encryption/Decryption Utilizing Look-Up Table", IEEE 37th Annual International Carnahan Conference on Security Technology, pp. 132-136, Taipei, Taiwan, ROC, 2003.

[11] Shastry, P., Somani, N., Gadre, A., Vispute, B., and Sutaone, M., "Rolled Architecture Based Implementation of AES Using T-Box", IEEE 55th International Midwest Symposium on Circuits and Systems, pp. 626-630, Boise, Idaho, 2012.

[12] Morioka, S., and Satoh, A., "A 10 Gbps Full-AES Crypto Design with a Twisted-BDD S-Box Architecture", Proceedings of IEEE International Conference on Computer Design, pp. 98-103, Freiburg, Germany, 2002.

[13] Xilinx, "7 Series FPGAs Memory Resources User Guide v1.10", Technical Report, 2014. http://www.xilinx.com/support/documentation/user_guides/ug473_7Series_Memory_Resources.pdf.

[14] Bulens, P., Standaert, F.X., Quisquater, J.J., Pellegrin, P., and Rouvroy, G., "Implementation of the AES-128 on Virtex-5 FPGAs", Progress in Cryptology AFRICACRYPT, Lecture Notes in Computer Science, Volume 5023, pp. 16-26, Springer Berlin Heidelberg, 2008.

[15] Drimer, S., Guneysu, T., and Paar, C., "DSPs, BRAMs, and a Pinch of Logic: Extended Recipes for AES on FPGAs", ACM Transactions on Reconfigurable Technology System, Volume 3, No. 1, pp. 3-27, 2010.

[16] Aziz, A., and Ikram, N., "A Look-Up-Table Implementation of AES", International Conference on High Performance Computing, Networking and Communication Systems, pp. 187-191, Orlando, Florida, USA, 2007.

[17] McLoone, M., and McCanny, J.V., "Rijndael FPGA Implementation Utilizing Look-Up Tables", IEEE Workshop on Signal Processing Systems, pp. 349-360, Antwerp, Belgium, 2001.

[18] Yoo, S.M., Kotturi, D., Pan, D.W., and Blizzard, J., "An AES Crypto Chip using a Highspeed Parallel Pipelined Architecture", Journal of Microprocessors and Microsystems, Volume 29, pp. 317-326, 2005.

[19] Ali, L., Aris, I., Hossain, F.S., and Roy, N., "Design of an Ultra-High Speed AES Processor for Next Generation IT Security", Journal of Computers & Electrical Engineering, Volume 37, pp. 1160-1170, 2011.

[20] Xilinx, "7 Series FPGAs Clocking Resources User Guide v1.9", Technical Report, 2014. http://www.xilinx.com/support/documentation/user_guides/ug472_7Series_Clocking.pdf.

[21] Rouvroy, G., Standaert, F.X., Quisquater, J.J., and Legat, J., "Compact and Efficient Encryption/Decryption Module for FPGA Implementation of the AES Rijndael Very Well Suited for Small Embedded Applications", International Conference on Information Technology: Coding and Computing, Volume 2, pp. 583-587, Las Vegas, Nevada, 2004.