A Novel Area-Throughput Optimized Architecture for the AES Algorithm

Hadi Samiee
Department of Electrical Engineering
Amirkabir University of Technology
Tehran, Iran
samiee.hadi@gmail.com

Reza Ebrahimi Atani
Department of Computer Engineering
The University of Guilan
Rasht, Iran
rebrahimi@guilan.ac.ir

Hamidreza Amindavar
Department of Electrical Engineering
Amirkabir University of Technology
Tehran, Iran
hamidami@aut.ac.ir

***Abstract*—In this paper, an efficient method for high speed hardware implementation of AES algorithm is presented. So far, many implementations of AES have been proposed, for various goals that effect the $SubByte$ transformation in various ways. These methods of implementation are based on combinational logic and are done in polynomial bases. In the proposed architecture, it is done by using composite field arithmetic in normal bases. In addition, efficient key expansion architecture suitable for 6 subpipelined round units is also presented. By using the proposed architecture in non-feedback mode, a fully subpipelined encryptor with 6 substages in each round unit can achieve a throughput of 43.71 Gbps on a Xilinx XC2VP20-7fg676 device, which is the fastest compared to previous FPGA implementations known to date. The efficiency in terms of equivalent throughput/slice is improved by a factor of 143%. Also in this design, key variation during the encryption operation is supported and it doesn't have any effect on continued throughput of the encryptor.**

I. Introduction

Cryptography plays an important role in the security of data transmission. The development of computing technology imposes stronger requirements on the cryptography schemes. The Data Encryption Standard (DES) has been the U.S. government standard since 1977. However, now, it can be cracked quickly and inexpensively. In 2000, the Advanced Encryption Standard (AES) [1] replaced the DES to meet the ever-increasing requirements for security [2].

The AES algorithm has broad applications, such as smart cards and cell phones, WWW servers, fiber network, satellite communication, and digital video recorders. Considering the variety of presented application, different requirement and limitation of each of them, implementation of different structure are noticed by researchers. Regarding this, numerous architectures have been proposed for the hardware implementations of the AES algorithm [3] – [17]. Among these architectures, the design in [12] can achieve the highest speed while it is more efficient than the prior designs. The key idea in [12] is to employ composite field arithmetic in the computation of the multiplicative inversion in the $SubBytes$/$InvSubBytes$ transformation of the AES algorithm. As a result, deep subpipelining is enabled, and hardware complexity is reduced.

Different construction schemes for composite fields are proposed for the AES algorithm in [9], [7], and [10]. In the design in [9], $GF(2^8)$ is decomposed into $GF((2^4)^2)$, and composite field arithmetic is applied to all the transformations in the AES algorithm. The optimum construction scheme for $GF((2^4)^2)$ is selected based on minimizing the total gate count in the implementation of all transformations. However, it is more efficient to apply composite field arithmetic only in the computation of the multiplicative inversion in the $SubBytes$ and $InvSubBytes$ transformations [7]. In this case, the construction scheme selected in [9] is no longer optimum. The schemes proposed in [7] and [10] apply composite field arithmetic only to the multiplicative inversion. In [7], $GF(2^8)$ is decomposed into $GF((2^4)^2)$, while in [10], $GF(2^8)$ is decomposed into $GF(((2^2)^2)^2)$. Nevertheless, each of them proposed only one possible way to construct the composite field. There exist other construction schemes with smaller gate counts and shorter critical paths.

In applications that the rate of receive and transmit data are very high, using of high speed hardware encryptors are usual. In these encryptors, three techniques of loop-unrolling, pipelining and subpipelining are used for increase of output bitrates. From these techniques, the third one has the maximum influence on the increment of encryptor throughput and causes the more optimization in throughput/area ratio. In this paper, all three presented methods with resource sharing technique are used for area and throughput optimization of encryptor hardware. In prior designs, the implementation of $SubByte$ transformation is done in polynomial bases but in this design normal bases form is used for this purpose.

The rest of the paper is organized as follows: the section II briefly describes the advance encryption standard. In section III, the Implementation of full pipeline AES is explained. The novel implementation of key expansion unit is presented in section IV. The final implementation results and comparison with prior works are provided in Section V. Finally, a conclusion is drawn in Section VI.

II. Rijndael: The Advance Encryption Standard

The Rijndael algorithm is composed of N_r times repetition of four transformations $SubByte$, $ShiftRow$, $MixColumn$, $AddRoundKey$ in each round of algorithm except the last one. Fig. 1 shows the Rijndael encryption process for a 128-bit secret key. The value of N_r for key length of 128-bit is equal to ten. Each round and the initial stage requires a 128-

978-1-61284-389-6/11/$26.00 ©2011 IEEE

bit round key, and thus 11 sets of round keys are generated from the secret key. In the AES algorithm, the plain text data string is divided into blocks of 128 bits. Each block is divided into 16 bytes (4×4 matrix of bytes), and each byte is considered as an element of $GF(2^8)$. Although different irreducible polynomials can be used to construct $GF(2^8)$, the one specified for the AES algorithm is shown in (1).

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

The primitive functions SubBytes, ShiftRows and MixColumns are based on byte-oriented arithmetic, and AddRoundKey is a simple 128-bitwise XOR operation.

$SubBytes$ is a nonlinear transformation that uses 16 byte substitution tables (S-Boxes). An S-Box is the multiplicative inverse of a Galois field $GF(2^8)$ followed by an affine transformation. In the decryption process, the affine transformation is executed prior to the inversion. $m(x)$ is the irreducible polynomial used by a Rijndael S-Box.

$ShiftRows$ is a cyclic shift operation of the last three rows by different offsets. $MixColumns$ treats the 4-byte data in each column as coefficients of a 4-term polynomial, and multiplies the data modulo $x^4 + 1$ with the fixed polynomial given by

$$c(x) \quad = \quad \{03\}x^3 + \{01\}x^2 + \{01\}x + \{02\} \qquad (2)$$

In the decryption process, $InvMixColumns$ multiplies each column with the polynomial

$$c^{-1}(x) \quad = \quad \{0B\}x^3 + \{0D\}x^2 + \{09\}x + \{0E\} \quad (3)$$

and $InvShiftRows$ shifts the last three rows in the opposite direction from $ShiftRows$.

The key expander in Fig. 1 generates 11 sets of 128-bit round keys from one 128-bit secret key by using a 4-byte S-Box.

III. Implementation of full pipeline AES

The key step in the implementation of this algorithm is $SubByte$ transformation, the only nonlinear step in each round of encryption algorithm. Consideration on high delay and required gates of this transformation, different implementation methods are presented. The most of traditionally hardware implementation of this transformation are done with using of Look Up Tables (LUT) [3] – [5]. Using of LUTs in high speed applications has two main problems. The first one is the high volume of required gates for implementation of this transformation. The second one is inherent and unbreakable existent delay in this LUT structures. This delay is longer than the total delay of the rest of the transformations in each round unit and prohibits each round unit from being divided into more than two substages to achieve any further speedup. Non-LUT-based approaches, which employ combinational logic only, can be used to avoid the unbreakable delay of LUTs. However, these approaches involve inversion in Galois Field $GF(2^8)$, which may have high hardware complexities. Therefore, composite field arithmetic is presented, in which the field operations are implemented in lower order fields such as $GF(2^4)$ or $GF((2^2)^2)$ and by lower cost subfield operations. In [6]-[12] the $SubByte$ transformation is implemented with using composite field arithmetic and in polynomial bases.

In [13], this implementation is done by using Composite field arithmetic in normal bases. The main advantages of the new bases than to older one is that it doesn't require any hardware for implementation of square operation. This operation is used in $\nu * \gamma^2$ block of $GF(2^8)$ inversion operation. The subpipelined architecture can achieve maximum speed up if each round unit is divided into more substages with equal delay. By increment the number of these substages, the critical path and clock pulse width of system can be decreased and as a result the throughput is increased. Fig. 2 shows a full pipeline AES encryptor with 6 subpipeline stages and key length of 128-bit. This architecture uses the composite field arithmetic in normal bases. In [11] and [12] because of the long critical path in the key expansion unit, the implementation of the architecture is done in 7 substages. In this design the implementation is done in a way to decrease the substages to 6. We have shown the idea in section IV. By application of 6 substages subpipelined AES architecture we can reduce the intermediate registers to 1408 compared to the ones presented so far. 1280 of the registers are for the encryptor and 128 registers for the key expansion unit.

IV. Implementation of key expansion

The round keys that are used in each encryption round unit, can be either generated beforehand and stored in memory or generated on the fly. In applications that key variation is required during encryption operation, the second approach is suitable.

If we want the round keys generation to be done simultaneously with encryption operation, it must be considered that the key expansion architecture is divided the same as the number of existent substages in encryption round unit. Fig. 3 shows the key expansion architecture suitable for 6 substages subpipelined AES algorithm with the key length of 128-bit.

Assuming the same subpipelined $SubByte$ transformation is used in the key expansion unit, the remaining path of this unit has one 2-to-1 Mux and 1 Xor gate. The 2-to-1 Mux operation is executed simultaneously with first substage of encryption round unit and the Xor operation is executed simultaneously with 6th substage of encryption round unit after M_IA operation. The proposed key expansion architecture is capable of generating all required $N_r + 1$ roundkeys after $6 \times N_r$ clock cycle. The roundkey i is available at output of R_{5J}-column ($(1 \leq J \leq 4)$) registers at clock cycle $6 \times i$ ($0 \leq i \leq N_r$) and at the moment, with activation C_i signal, it is loaded into $RndKey(i)$ registers. These roundkeys are held there for using in the entire encryption process. Also, the Xor operation of R_{const} with each row of input state matrix are executed simultaneously with second substage of key expansion unit.

V. Implementation results and comparison

In this section, a fully subpipelined AES encryptor with 6 substages and key length of 128-bit with the key expansion

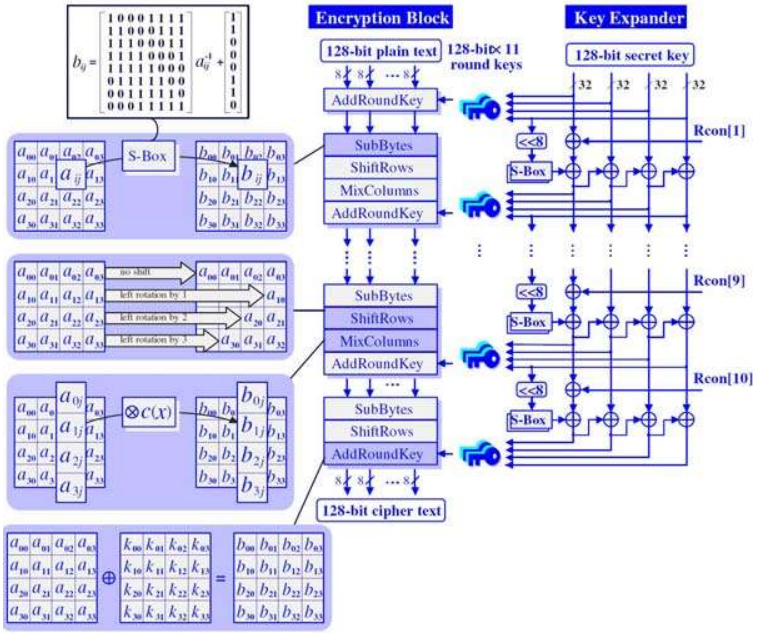

Fig. 1. Encryption process of Rijndael algorithm [7]

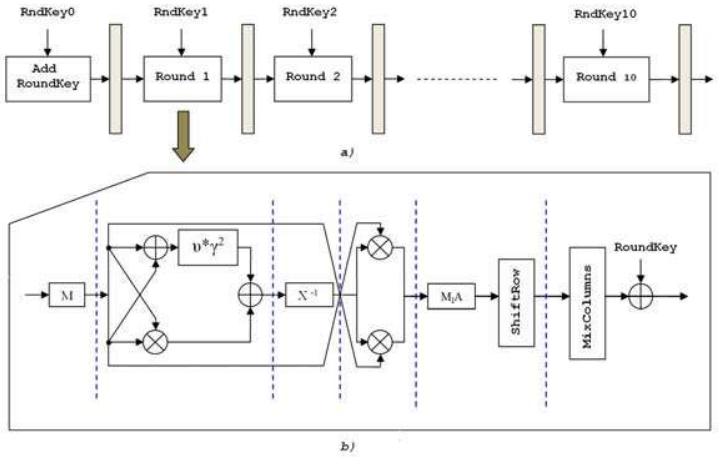

Fig. 2. The AES algorithm a) Fully pipelined architecture b) The subpipelined composite field architecture of round unit

unit are implemented on FPGA devices. Xilinx ISE 9.2i is used to synthesize the design and provide post-placement timing result. In the proposed architecture, after an initial delay of $6 \times N_r + 1$ clock cycles, the corresponding ciphertext blocks appear at the output every clock cycle. The fully subpipelined architecture is implemented on Xilinx XCV1000e-8bg560 and XC2VP20-7fg676 devices. These implementation can achieve a throughput of 21.76 and 43.71 *Gbps* , respectively. As can be observed from table I, our architecture can achieve higher speed than all prior FPGA implementations known to the authers and is 143% more efficient than the best design [12] in term of equivalent throughput/slice.

VI. Conclusion

In this paper, an efficient subpipelined architecture of AES algorithm with its key expansion unit is presented. The key expansion architecture is suitable for 6 substages subpipelined AES architecture. Using a Virtex-II Pro XC2VP20-7 target device with only 7865 slices the clock rate of 341.53 MHz is acheived. Against prior implementations, this architecture uses composite field arithmetic in normal bases representation to reduce the required hardware.

References

[1] National Institute of Standards and Technology (NIST), Information Technology Laboratory (ITL), *Advanced Encryption Standard (AES)*, Federal

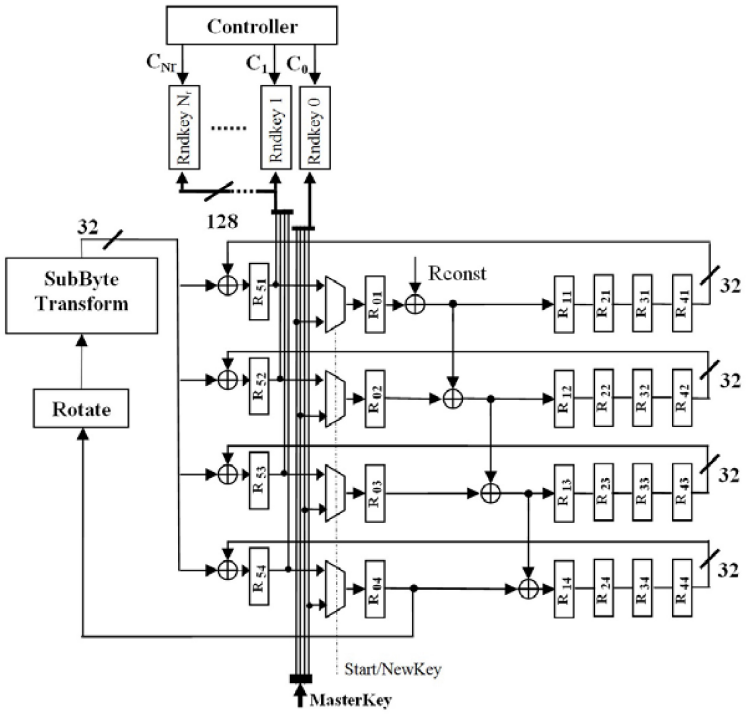

Fig. 3. The key expansion architecture for 6 substages subpipelined AES algorithm with 128-bit key

TABLE I
COMPARISON WITH OTHER FPGA IMPLEMENTATION

Design	Target FPGA Device	Frequency MHz	Slice	Mbps/ Slice
Saggesse et al [14]	Virtex-E XCV2000e-8	158	18610	1.091
Standaert et al [15]	Virtex-E XCV3200e-8	145	15112	1.228
Zambreno et al [16]	Virtex-II XC2V4000	184.1	16938	1.391
Jarvinen et al [17]	Virtex-E XCV1000e-8	129.2	11719	1.408
Zhang ($r = 7$) [11]	Virtex-E XCV1000e-8	168.4	11022	1.956
Hodjat et al(r=7) [12]	Virtex-II Pro XC2VP20-7	169.1	9446	2.29
This work ($r = 6$)	Virtex-E XCV1000e-8	170.01	8099	2.687
This work ($r = 6$)	Virtex-II Pro XC2VP20-7	341.53	7865	5.558

Information Processing Standards (FIPS) Publication 197, November 2001

[2] X. Zhang and K. K. Parhi, *On the Optimum Constructions of Composite Field for the AES Algorithm*, IEEE Transactions on Circuits and Systems-II: Express Briefs, VOL. 53, NO. 10, OCTOBER 2006.

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

[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. Rijmen, *Efficient Implementation of the Rijndael S-box*, 2000. Available online at www.iaik.tugraz.at/RESEARCH/krypto /AES/old / rijmen/rijndael/sbox.pdf.

[7] A. Satoh, S. Morioka, K. Takano, S. Munetoh, *A Compact Rijndael Hardware Architecture with S-Box Optimization*, Proceedings of ASIACRYPT 2001, LNCS Vol.2248, pp. 239 - 254, Springer-Verlag, December 2001.

[8] N. Mentens, L. Batina, B. Preneel, and I. Verbauwhede, *A Systematic Evaluation of Compact Hardware Implementations for the Rijndael S-Box*, In Alfred Menezes, editor, CT-RSA, volume 3376 of LNCS, pages 323-333. Springer, 2005.

[9] 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, Paris, France, May 2001, pp. 171-184.

[10] J. Wolkerstorfer, E. Oswald, and M. Lamberger, *An ASIC implementation of the AES S-boxes*, in Proc. RSA Conf., San Jose, CA, Feb.2002, pp. 67-78.

[11] X. Zhang, K. K. Parhi, *High-speed VLSI architectures for the AES algorithm*, IEEE Trans. VLSI Systems, Vol. 12, Iss. 9, pp. 957 - 967, Sept. 2004.

[12] A. Hodjat, I. Verbauwhede, *A 21.54 Gbits/s Fully Pipelined AES Processor on FPGA*, 12th Annual IEEE Sypmosium on Field-Programmable Custom Computing Machines (FCCM'04), pp. 308-309, April 2004.

[13] D. Canright, *A Very Compact S-Box for AES* In Josyula R. Rao and Berk Sunar, editors, CHES, volume 3659 of LNCS, pages 441-455. Springer, 2005.

[14] 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*, Proc. FPL 2003, Portugal, Sept. 2003.

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

[16] J. Zambreno, D. Nguyen, A. Choudhary, , *Exploring Area/Delay Tradeoffs in an AES FPGA Implementation*, Proc. FPL 2004, 2004.

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