2012 10th International Conference on Frontiers of Information Technology

An Efficient High Throughput FPGA Implementation of AES for Multi-Gigabit Protocols

Ulfat Hussain, Habibullah Jamal
Department of Electrical Engineering
University of Engineering and Technology
Taxila, Pakistan
ulfat092003@yahoo.com, habibullah.jamal@uettaxila.edu.pk

***Abstract*—Due to the requirement of high throughput architecture for encrypted channels, an efficient implementation of hardware is needed. This can be achieved by using smart utilization of high end reconfigurable platforms. To achieve convincingly high throughput, an efficient non-pipelined style implementation of Advanced Encryption Standard (AES) with key size of 128-bit, for multi-gigabit protocols on Field Programmable Gate Array (FPGA) is presented.**

Keywords: FPGA, AES, CBC, ECB, Multi-gigabit, HDL

I. INTRODUCTION

High speed data communication networks are no more a dream. Networks with data transmissions speeds of tens of gigabits are already in use. The urge of mankind to achieve high speeds is increasing with each passing day. Data communication speeds in gigabits is pushing for new methods, designs and implementation techniques in the field of data security. Securing data in high speed communications impose strict requirements on cryptographic designs. The requirements include minimum latency, resources merging, flexible design, and scalability in cryptographic parameters and of course reliable security.

Various algorithms and cryptographic designs have been proposed by the researchers to cope the growing demands of high speed communications. However AES [1], due to its distinct design, has the capability of securing data at high speeds without compromising security. Designing efficient AES architectures for reconfigurable platforms is a promising choice of achieving high-speed data security. Design flexibility offered by FPGAs and data security provided by AES results in practical implementations of high speed data security.

AES can be implemented in hardware by using pipelined or non-pipelined approach. A pipelined architecture helps to achieve high throughput, but major drawback is that it is difficult to handle it in feedback modes and its throughput significantly reduces in feedback modes [9]. On the other hand, non-pipelined structural design gives relatively less speed then pipelined approach but can easily be implemented in both feedback and non-feedback modes.

II. RELATED DESIGNS

There are various hardware implementations of AES on FPGA. We have mainly focused on high throughput non-pipelined AES implementations.

In [2], AES-512 architecture is proposed to increase the security and throughput. The throughput of AES-512 and AES-128 on Virtex-7 platform is 1.163 Gbps and 0.495 Gbps respectively. In [3], AES encryption/decryption is implemented in iterative style and took 51cycles to Encrypt/Decrypt data. The throughput of 0.352 Gbps is achieved for both encryption/decryption on Xilinx Virtex XCV600 platform. In [4], an iterative approach of AES is mentioned and throughput of 1.4 Gbps is achieved on XC2VP30 platform. In [5], Content-addressable memory (CAM) is used for high speed sub-bytes block, new hardware sharing structural design is proposed to implement mix-column and efficient low cost Add Round Key is employed for real time key generation. The given sequential design throughput is 0.876 Gbps. In [6], a tradeoff between area and speed optimized AES is given. The throughput of area optimized (non-pipelined) encryption and decryption is 1.664 and 1.598 Gbps respectively on a Xilinx Virtex 4 platform. In [7], implemented AES is in non-pipelined style on a Xilinx Virtex 4 platform and achieves throughput of 0.582 Gbps. In [8], an iterative implementation of AES is done on a Xilinx Virtex-II FPGA platform in which off-line key expansion is used and the throughput of 0.969 Gbps is achieved. In [9], AES is implemented in non-pipelined style having the throughput of 0.832 Gbps on Xilinx XC2v8000-5 platform. In [10], AES is implemented in pipelined and non-pipelined style. In non-pipelined method the throughput achieved is 0.258 Gbps on a Xilinx Virtex-E XCV812 platform.

III. AES ALGORITHM

AES [1] is a symmetric block cipher having 128-bit data block and key lengths of 128, 192 and 256-bits. AES consists of rounds that execute in sequential manner. The number of rounds (Nr) depends upon the key length e.g., for 128 bit key length, the Nr is 10.

For both encryption and decryption, AES uses four different byte-oriented transformations: 1) sub-bytes/inverse-sub-bytes (SB/iSB), 2) shift-rows/inverse-shift-rows (SR/iSR), 3) mix-columns/inverse-mix-columns (MC/iMC),

978-0-7695-4927-9/12 $26.00 © 2012 IEEE
DOI 10.1109/FIT.2012.45

and 4) add round key (ARK). The basic architecture diagram of AES encryption and equivalent decryption is given in Figure 1.

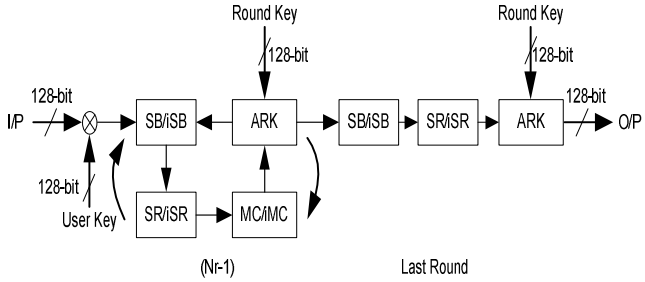

Figure 1. Basic AES Algorithm Flow

The ARK is performed initially on the input data, and then this data is passed to rounds. For Nr-1 rounds, all four transformations (SB/iSB, SR/iSR, MC/iMC, and ARK) are used. For last round MC/iMC is not used.

IV. Proposed High-Throughput AES Architecture

In our implementation we have used 128-bit key length for encryption and decryption. Equivalent inverse cipher has been used for the symmetry of both encryption and decryption processes. We have used the technique mentioned in [1] in key expansion module so that the keys for decryption can be accustomed according to the equivalent inverse cipher.

The SB/iSB unit consists of S-Box/iS-Box. There are different methods to implement the S-Box/iS-Box. To make our design faster, we have used look-up table based approach for S-Box. For the same reason, we used look-up tables for the MC/iMC tables (02, 03, 09, 11, 13 and 14).

A. Key Expansion

In the proposed design we have used off line key expansion method. In this approach, we have expanded the user key into 10 round keys and stored them on a Block RAM. These keys are then used in encryption and decryption processes. For the equivalent inverse cipher, we have used the key expansion modification given in [1].

B. Throughput Improvement Using Proposed Coalesced Transformation Module

AES 128-bit encryption and decryption consists of 10 rounds. Generally, optimized designs take 1 clock cycle for each transformation process, SB/iSB, SR/iSR, MC/iMC, and ARK i.e. first 9 rounds take 36 clock cycles, the last round takes 3 clock cycles, and the initial ARK takes 1 clock cycle. This results in a total of 40 clock cycles to accomplish the encryption and decryption process.

We have proposed a coalesced transformation module (CTM) for the AES that efficiently merges the four transformations into a single unit. The SB/iSB look-up tables (S-Box/iS-Box) and MC/iMC look-up tables (02, 03, 09, 11, 13) are fused together to form new coalesced Tables (CT). The CTs are pre-generated look-up tables according to the pseudo code shown in Figure 2.

```
While (i < 256)
{
        Z       ←   S-Box[i];
        CT_xx[i] ←  Table_xx [Z];
        i       ←   i+1;
}
For Encryption xx = 02 and 03 and for decryption xx = 09, 11, 13, 14.
For Decryption S-Box is replaced with iS-Box.
```

Figure 2. Pseudo code for Coalesced Tables

The CTM is shown in Figure 3. The 128-bit round data is entered into the CTM, the data is divided into 8-bit chunks, and these 8-bit data are then processed through CT_xx and S-Box/iS-Box look-up tables. After that MC/iMC, SR/iSR and ARK is performed on the data and 128-bit round output is generated. The CTM takes 1 clock cycle to perform the transformation that is equivalent to the four AES transformations.

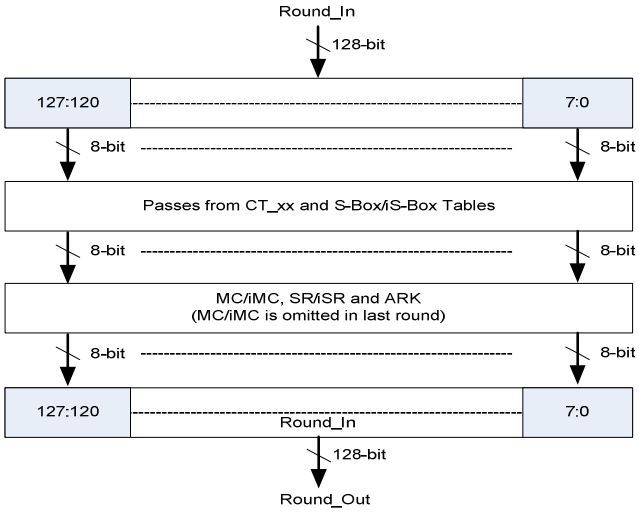

Figure 3. Block Diagram of Coalesced Transformation Module

Our encryption and decryption design takes 11 clock cycles to generate output. The flow of the given approach is given in Figure 4.

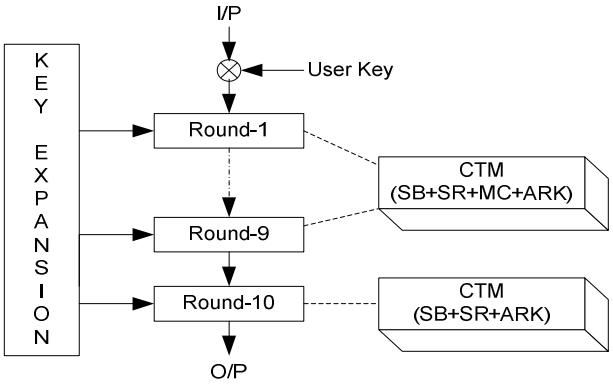

Figure 4. Block Diagram of Proposed AES Architecture

C. *Modes of Operation*

To enhance the confidentiality of AES, five modes of operation for symmetric key block cipher algorithm were suggested in [11]. From those given modes we have implemented Electronic Codebook (ECB), and Cipher Block Chaining (CBC). Block diagram of ECB and CBC mode is shown in Figure 5.

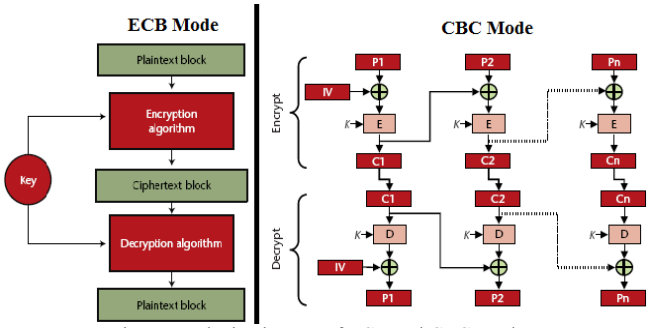

Figure 5. Block Diagram of ECB and CBC Mode

In ECB mode, the plain data is directly applied to the encryption module and it is independent on any other factor. The resultant output is the encrypted data. This inverse process is done in the decryption process. The main advantage of ECB mode is that multiple encryption and decryption modules can be computed in parallel. The major drawback is that, under a given key, any given plain data block always gets encrypted to the same encrypted data block.

In CBC mode, the plain data block is XORed with the previous encrypted data block. An Initialization Vector (IV) is required to be XORed with the first plain data block. The IV need not be secret, but it must be unpredictable; the generation of such IVs is discussed in [11]. This inverse process is done in the decryption process. The major advantage of CBC mode is its ability to hide statistical properties of the plain data blocks. The major drawback is that the multiple encryption blocks cannot be computed in parallel but however multiple decryption modules can be performed in parallel.

V. IMPLEMENTATION & RESULTS

Verilog HDL is used for the AES encryption and decryption realization. The design has been simulated on the ModelSim 6.5 and implemented using Xilinx ISE Design Suite 13.2.

The Xilinx Virtex-7 (speed grade -3) FPGA device has been used to implement our design. The V7 series is fourth-generation Advanced Silicon Modular Block (ASMBL™) column-based architecture that reduces system development and deployment time with simplified design portability.

The simulation results are shown in Figures 6, and 7. Both encryption and decryption takes 11 clock cycles to compute the respective output.

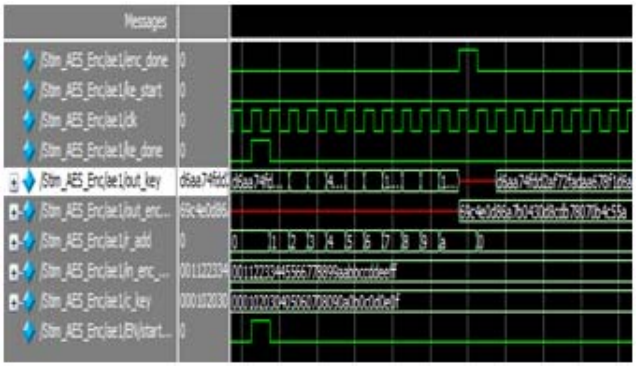

Figure 6. AES-128 Encryption and Key Expansion

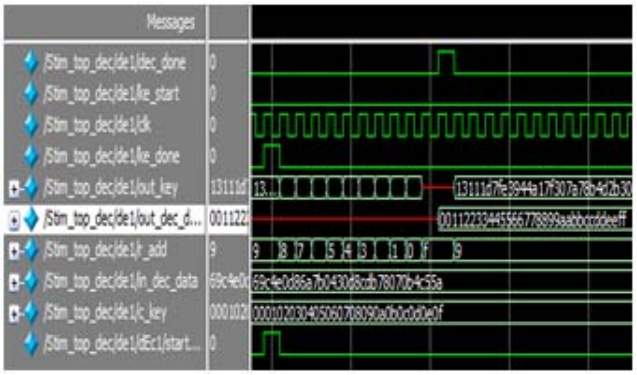

Figure 7. AES-128 Decryption and Key Expansion

Each look-up table used in the CTM utilizes 32 slice LUTs on Virtex-7. The implementation results of encryption and decryption of our design in ECB and CBC mode on Virtex-7 platform are given in Table I and II respectively. The results include design frequency, throughput and number of slice registers.

TABLE I. RESULT OF ENCRYPTION MODULE ON VIRTEX-7

Mode	Freq (MHz)	No. of Slices Registers	Throughput (Gbps)
ECB	456	596	5.30
CBC	450	731	5.23

TABLE II. RESULTS OF DECRYPTION MODULE ON VIRTEX-7

Mode	Freq (MHz)	No. of Slices Register	Throughput (Gbps)
ECB	418	891	4.86
CBC	416	986	4.84

Table III and IV shows the comparison of encryption and decryption respectively of our design with the previous designs. The comparison results include design frequency and number of slices.

TABLE III. COMPARISON OF NON-PIPELINE ENCRYPTION ON FPGA

Design	Platform	No. of Slices	Frequency (MHz)
[2]	Virtex-7	3243	378.4
This	Virtex-7	2444	456
[3]	XCV600-6	1853	140.39
This	XCV600-6	5096	98.6
[4]	XC2VP30	6211	142.5
This	XC2VP30	4591	196.375
[5]	XC2V3000-6	7617	73.5
This	XC2V3000-6	4582	153.167
[6]	XC4VLX60	1468	130
This	XC4VLX60	4582	193.291
[7]	XC4VFX12-10		50
This	XC4VFX12-10	4582	144.937
[8]	XC2V1000 -5	1122	159
This	XC2V1000 -5	4582	135.613
[9]	XC2v8000-5	8378	65
This	XC2v8000-5	4582	135.613
[10]	XCV812	2744	22.41
This	XCV812	5096	56.970

TABLE IV. COMPARISON OF NON-PIPELINE DECRYPTION ON FPGA

Design	Platform	No. of Slices	Frequency (MHz)
[3]	XCV600-6	1853	140.39
This	XCV600-6	6196	98.6
[4]	XC2VP30	6211	142.5
This	XC2VP30	5554	162.985
[5]	XC2V3000-6	7617	73.5
This	XC2V3000-6	5554	128.179
[6]	XC4VLX60	2752	124.844
This	XC4VLX60	5554	157.162
[9]	XC2V8000-5	8378	65
This	XC2V8000-5	5554	113.011

Figure 8 and 9 shows the comparison of non-pipelined encryption and decryption of our design with the previous designs in the form of charts respectively. The results include the throughput of the designs.

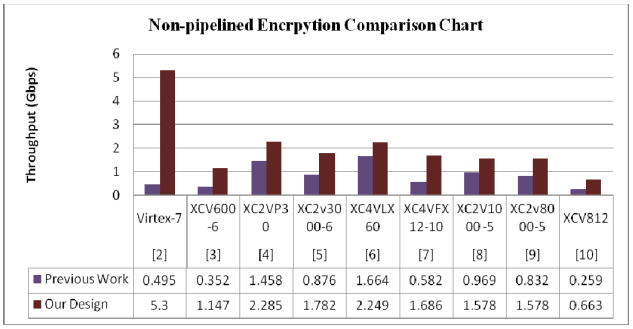

Figure 8. Comparison Chart of Non-pipelined Encryption

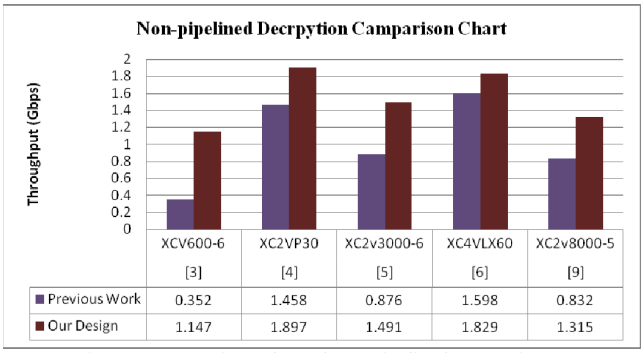

Figure 9. Comparison Chart of Non-pipelined Decryption

VI. CONCULSIONS

In this paper, we have presented an efficient non-pipelined implementation of AES-128 to achieve high throughput so that it can be used in gigabit protocols. We have implemented our design of AES-128 encryption and decryption on a Xilinx Virtex-7 FPGA and achieved throughput of 5.30/ 4.86 Gbps in ECB mode and 5.23/4.84 Gbps in CBC mode.

REFERENCES

[1] FIPS 197, Federal Information Processing Standards Publication (FIPS) 197, Specification for the Advanced Encryption Standard (AES), NIST's AES, November 26, 2001. http://www.nist.gov/aes.

[2] Moh'd A., Jararweh Y., Tawalbeh L.: "AES-512: 512-bit Advanced Encryption Standard algorithm design and evaluation," *7th International Conference on Information Assurance and Security (IAS),* pp. 292 – 297, 2011.

[3] Borkar A.M., Kshirsagar R.V., Vyawahare M.V.: "FPGA implementation of AES algorithm," *3rd International Conference on Electronics Computer Technology (ICECT) 2011,* pp. 401 – 405, 2011.

[4] Jyrwa B., Paily R.: "An Area-Throughput Efficient FPGA Implementation of the Block Cipher AES Algorithm," *International Conference on Advances in Computing, Control, & Telecommunication Technologies, ACT 09,* pp. 328 – 332, 2009.

[5] Chih-Peng Fan, Jun-Kui Hwang: "Implementations of high throughput sequential and fully pipelined AES processors on FPGA," *International Symposium on Intelligent Signal Processing and Communication Systems, ISPACS 07,* pp. 353 – 356, 2007.

[6] Rizk M.R.M., Morsy M.: "Optimized Area and Optimized Speed Hardware Implementations of AES on FPGA," *2nd International Design and Test Workshop, IDT 07,* pp. 207 – 217, 2007.

[7] Wiebe J.H.: "AES-128 Implementation on a Virtex-4 FPGA," *IEEE International Symposium on Signal Processing and Information Technology,* pp. 68 – 73, 2007.

[8] Brokalakis A., Kakarountas A.P., Goutis C.E.: "A high-throughput area efficient FPGA implementation of AES-128 Encryption," *IEEE Workshop on Signal Processing Systems Design and Implementation,* pp. 116 – 121, 2005.

[9] Sever R., Ismailglu A.N., Tekmen Y.C., Askar M., Okcan B.: "A high speed FPGA implementation of the Rijndael algorithm," *Euromicro Symposium on Digital System Design, DSD04,* pp. 358 – 362, 2004.

[10] Saqib N.A: "AES algorithm implementation - an efficient approach for sequential and pipeline architectures," *Proceedings of the Fourth Mexican International Conference on Computer Science,* pp. 126-130 , 2003.

[11] NIST, National Institute of Standards and Technology Special Publication 800-38A, Recommendation for Block Cipher Modes of Operation: Methods and Techniques, December 01, 2001.