Improving the Efficiency of AES Encryption Algorithm by Using the co-designed Strategy

Ricardo Peixoto Robaina
UNIPAMPA – Federal University of Pampa
Bagé-RS, Brazil
ricardorobaina11@gmail.com

Bruno Silveira Neves
UNIPAMPA – Federal University of Pampa
Bagé-RS, Brazil
brunoneves@unipampa.edu.br

Fábio Livi Ramos
UNIPAMPA – Federal University of Pampa
Bagé-RS, Brazil
fabioramos@unipampa.edu.br

ABSTRACT

Currently, the need for mechanisms for information security is indisputable. In addition, it is noted that with high-speed communication, wireless technology carries out a predominant role in data transmission. In the wireless environment, one of the most commonly used security algorithms in the MAC (Medium Access Control) layer is Advanced Encryption Standard (AES). Therefore, increasing the efficiency of this algorithm means increasing at the same time the efficiency of much of the communication. The present study proposes a solution based on the co-design technique for the AES cipher algorithm. An analysis and implementation of an advanced hardware architecture was performed to process part of the algorithm, the rest was processed in software. A cost-benefit analysis was made from the implemented solution. Tests and validation of the created component were obtained through testbenchs also developed. As expected, the version of the algorithm in co-design achieved a higher performance compared to the software version at the cost of a small increase in the processor area.

KEYWORDS

Algorithm; Encryption; co-design.

1 INTRODUCTION

Throughout the history of information security, several cryptographic algorithms have been used to protect data in different categories. At the present, the AES (Advanced Encryption Standard) algorithm is the most widely used in the industry, being also the widely cited in the academic environment [7-11]. In the current context, the AES receives great attention not only for providing a high level of security for the data, but also for its high efficiency during the process of encryption/decryption of the information, and for its low memory consumption, which enables its large use in the mobile devices.

The cryptographic process of the data with a symmetric key performed by AES is subdivided into 10, 12 or 14 steps or rounds, depending on the size of the key used by the algorithm. As shown in Figure 1, four basic operations are performed on each round: AddRoundKey, SubBytes, ShiftRows, and MixColumns, except in the last round, when the MixColumns operation is not performed.

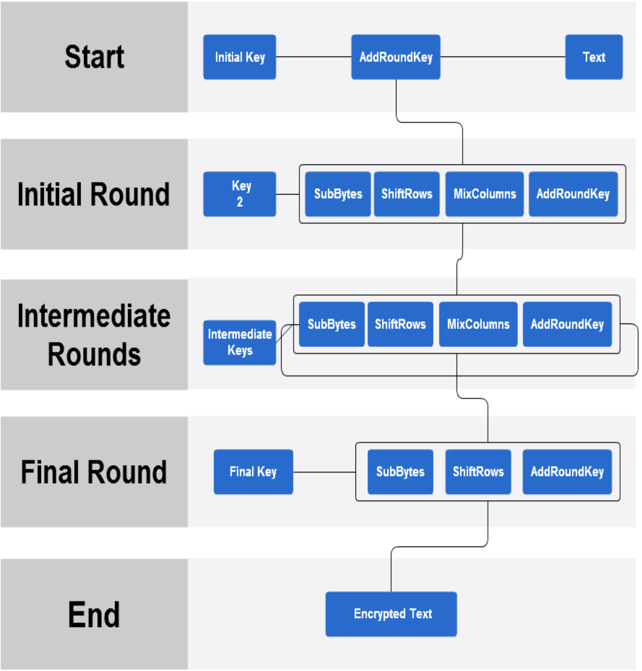

Figure 1: AES Algorithm Operation Flow.

2 INTERNAL DESCRIPTION OF MACRO OPERATIONS PERFORMED BY AES

The complexity of the AES algorithm is given by the combination of the results from each of its macro operations. The description of each of them is as follows:

• AddRoundKey, in this step are combined the columns of the block to be encrypted with the key of the round, generated in the expansion routine.

• SubBytes, there is the transformation that replaces the bytes of the state array by bytes of the S-Box, an auxiliary array. The calculation of the index of the matrix to be replaced is done by dividing the current byte into two parts. The 4 most significant bits of the byte indicate the line and the 4 least significant bits indicate the column to be replaced from the S-box.

• ShiftRows, such a step acts on the state rows by shifting the bytes in each row of a given number of positions. In AES, the first line is unchanged. Each byte of the second line is shifted to the left of a position. Similarly, the third and fourth rows are offset from two and three positions respectively.

• MixColumns, this step operates on each column of the state array, multiplying them by a fixed array.

3 METHODOLOGY

The methodology used to develop this study consists of the following steps:

1) Selection of the tools for implementation, validation and evaluation of the co-design based solution for the target application.

2) Execution of application partitioning in software and hardware components.

3) Implementation of the hardware component.

4) Development, execution and analysis of tests for the hardware component developed in step 3.

5) Execution of the co-synthesis followed by the integration of the components.

6) Validation of the proposed new solution and also the verification and evaluation of its charge-area-performance ratio.

7) Comparison of the results obtained at the end of the work with the data available in the literature for related works.

4 DEVELOPMENT

Based on the steps provided in the methodology for the development of this work, the following productions were obtained for this study up to the present moment:

Step 1 - Atera's Qsys [5] environment choice as a co-design tool for implementing the hardware and software components of the target application. It was determined that both the validation and the evaluation of the final application achieved should be performed using a prototype built from a Nios II Embedded kit [6], which is based on a field-programmable device (FPGA).

Step 2 - To establish the partitioning, a study was carried out in a software version of the algorithm. In this study, the impact of each step on the processing time of the algorithm was measured. The most costly stage, in terms of software execution time, is SubBytes, accounting for 74% of the total execution time of the algorithm. As can be seen in figure 2.

Therefore, it established that a hybrid arrangement in which the SubBytes operation is implemented in hardware, because such a step has a greater impact on the AES execution time. The rest of the application runs in software on a general purpose processor.

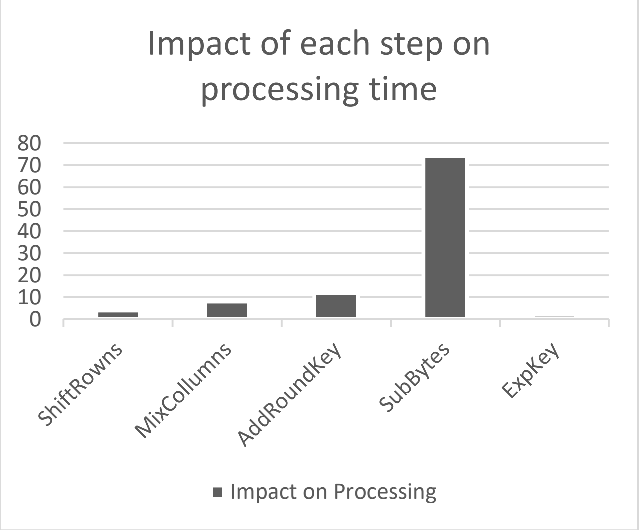

Figure 2: Impact of each step on processing time.

Step 3 - The hardware component of the SubBytes step has been developed in a fully combinational, in such a way that the full SubBytes operation is performed in only one clock cycle.

Step 4 - To validate the correct functioning of the component created, a specific TestBench was developed for such application. For this purpose, it was implemented a comparison of the outputs produced by the hardware SubBytes component with the outputs produced by the software version of this component, using several different input stimuli.

Step 5 - In the co-synthesis stage, the selected toolset was used to generate the software, the hardware and the interface logic for the hybrid AES system.

Step 6 - The validation of the solution was made by comparing the results produced by the encryption of data blocks using the co-design solution with the results produced by the encryption of the same blocks using an previously validated software version.

5 RESULTS AND DISCUSSIONS

The area occupied in projects synthesized in FPGA's is given by the number of logic elements used in it. For the intended analysis of area, it was considered a ciclone III EP3C25F423C6 FPGA from Altera. Thus, initial estimates of the implementation produced by this work point to an increase in the FPGA area of 5.48% compared to a conventional implementation in which the entire algorithm is executed in software on a general purpose processor, as can be noticed in figure 3.

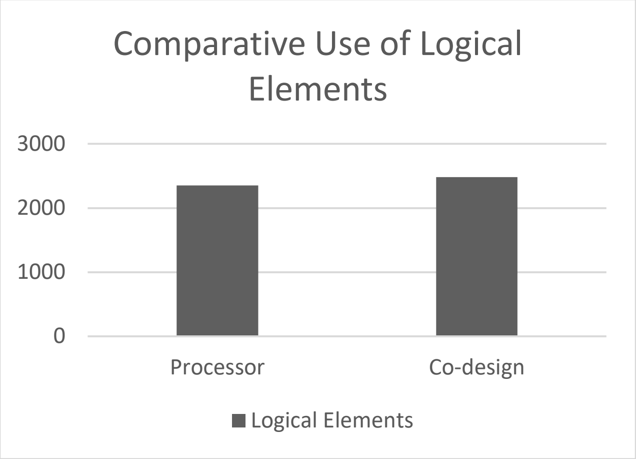

Figure 3: Comparative use of logical elements

Although a combinational solution tends to lead to an expressive area increase, the small additional cost obtained in area is justified by the use of the internal memory of the FPGA to implement the Sbox matrix out of the logical elements space of the device. On the other hand, the estimated execution time is about 64% less than that of the conventional solution, executed entirely in software, as is shown in figure 4.

The expressive increase in efficiency is given by the fact that the hardware component is fully combinational. For this reason, the entire operation runs in only one clock cycle, compared to the same operation in software that takes approximately 117 cycles to be completely executed.

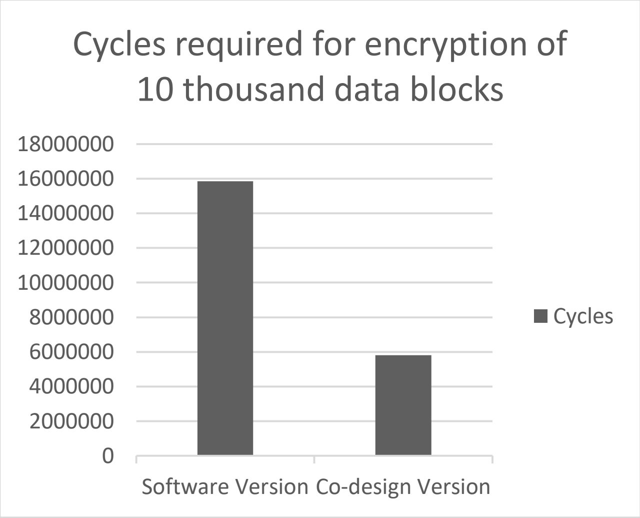

Figure 4: Comparison of number of cycles required to perform the operation

When analyzing the abovementioned data together, the area and the execution time, a favorable charge-benefit ratio is observed. With the use of the proposed project, despite a small area increase, there is a significant improvement in performance compared to the fully software version. Figure 5 shows this relation.

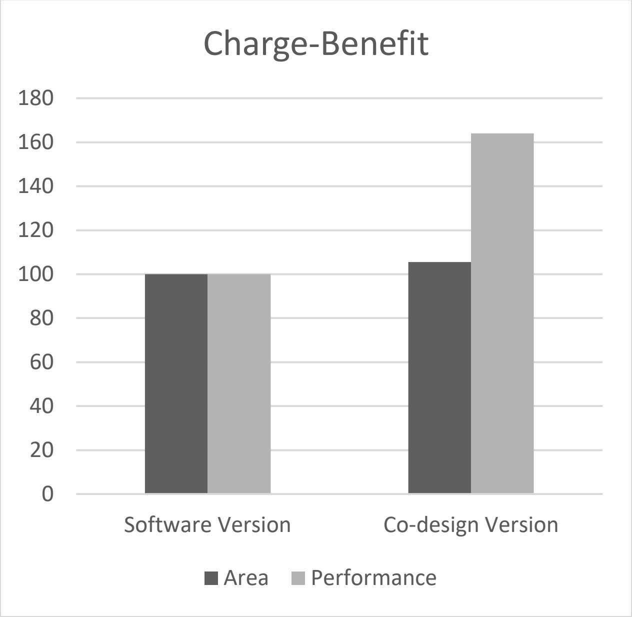

Figure 5: Comparative of Charge-Benefit

6 CONCLUSION

Through the study, it was observed that, by using the co-design technique efficiently, an expressive performance improvement in the AES encryption algorithm is obtained, at the cost of a small increase in the processor area. The cost (in area) of a co-design solution tends to be lower compared to a fully-developed hardware design.

It is believed to be important to improve the set of results produced in this study, through the use of energy consumption reduction techniques balanced with other key factors: performance, area and price. In this way, better inputs are created for a more effective final comparative evaluation with results in the literature.

REFERENCES

[1] El Maraghy, Mazen; Salma Hesham; Mohamed A. Abd El Ghany. "Real-time efficient FPGA Implementation of AES Algorithm." SOC Conference (SOCC), 2013 IEEE 26th International. IEEE, 2013.

[2] Gomes, Leandro As; Bruno S. Neves; Leonardo B. Pinho. "Empirical Analysis of Multicore CPU and GPU-Based Parallel Solutions to Sustain Throughput Needed by Scalable Proxy Servers for Protected Videos." Computer Systems (WSCAD-SSC), 2012 13th Symposium on. IEEE, 2012.

[3] Tonde, Ashwini R.; Akshay P. Dhande. "Implementation of Advanced Encryption Standard (AES) Algorithm Based on FPGA." International Journal of Current Engineering and Technology 4.2, 2014.

[4] Seshadrinathan, M., and Dempski, K. L., “Implementation of Advanced Encryption Standard for Encryption and Decryption of Images and Text on a GPU”, IEEE CVPRW´08, Anchorage, AK, USA, 2008.

[5] Altera, “Quartus Prime Features” Available: https://www.altera.com/products/design-software/fpga-design/quartus-prime/features/qts-qsys.html

[6] Altera, “Products Kit CVC3 embedded” Available: https://www.altera.com/products/boards_and_kits/dev-kits/altera/kit-cyc3-embedded.html

[7] Drimer, Saar; Tim Güneysu; Christof Paar. “DSPs, BRAMs, and a Pinch of Logic: Extended Recipes for AES on FPGAs. Journal ACM Transactions on Reconfigurable Technology and Systems, USA, 2010.

[8] Hoang, Anh-Tuan; Takeshi Fujino. “Intra-Masking Dual-Rail Memory on LUT Implementation for SCA-Resistant AES on FPGA”. Journal ACM Transactions on Reconfigurable Technology and Systems, USA, 2014.

[9] Yingxi Lu; O’Neil, Maire; McCanny, John. “Evaluation of Random Delay Insertion against DPA on FPGAs”. Journal ACM Transactions on Reconfigurable Technology and Systems, USA, 2010.

[10] Peter Hellekalek; Wegenkittl, Stefan. “Empirical evidence concerning AES”. Journal ACM Transactions on Modeling and Computer Simulation, USA, 2003.

[11] Ted Huffmire et al. “Security Primitives for Reconfigurable Hardware-Based Systems”. Journal ACM Transactions on Reconfigurable Technology and Systems, USA, 2010.