Signal Processing and Renewable Energy
December 2017, (pp.1-7)
ISSN: 2008-9864

Very High Throughput Implementation of Advanced Encryption Standard (AES) Algorithm on FPGA

Mahdi Rahmanpour[1], Amir Amirabadi Zavare[2*]

[1,2] Electrical Engineering Department, South Tehran Branch, Islamic Azad University, Tehran, Iran.

Abstract
An Advanced Encryption Standard (AES) algorithm is one of the most popular and most commonly used encryption algorithms. This algorithm can be implemented on microcontroller chips and FPGAs with various specifications. Also, the goals of implementing this algorithm are varied according to the application and requirements. In this paper, a project has been given that output very high data transfer rate equal 192 Gbps on the FPGA of the Virtex-7 (XC7VX330T-3FFG1157) from Xilinx. The extracted results of the implementation of the algorithm in the ISE 14.7 software show the maximum achievable clock frequency 500 MHz, with the parallel implementation of than three AES algorithms cores on a chip, higher speeds are also available.

Keywords: ; Encryption; AES algorithm; High-Speed AES; Rijndael.

1. INTRODUCTION

Encryption Algorithm is the most important part of security in confidential communications. One of the most important and well-known Encryption algorithms is the Advanced Encryption Standard (AES) that was approved by the National Institute of Standards and Technology of the United States in 2000 as the standardized encryption algorithm, and after its effective implementation of the software and hardware of this algorithm has been considered.

The Advanced Encryption Standard algorithm is a symmetric block cipher that can process data blocks of 128 bits through the use of cipher keys with lengths of 128 bits. The usage of 128 bit cipher key is to achieve the high security, because 128 bit cipher key is difficult to be broken. AES algorithm can resist any kinds of password attacks with a strong practicability in information security and reliability. AES provides better security and has less implementation complexity; it has emerged as one of the strongest and most efficient algorithms in existence today. The AES algorithm can be efficiently implemented by software. Software implementations cost the smallest resources, but they offer a limited physical security and the slowest process. Besides, growing requirements for high speed, high volume secure communications combined with physical security, hardware implementation of cryptography takes place [9].

In the field of correcting and optimizing the speed of implementation of this algorithm, various articles have been presented so far, and each has been presented with the changes in the above-mentioned conversions and their relationship method, to increase the efficiency of the algorithm. Correcting the last floor architecture,

*****Corresponding Author's Email: A_Amirabadi@azad.ac.ir

parallelizing part of the structure of the algorithm, using the pipeline method, using the search table in the S-Box, combining simultaneous, parallel processing, using pipeline method, and Samples Speed algorithm optimization methods are used. In this paper, by studying the above methods, we introduce combined and innovative methods to increase the speed of implementation of the AES algorithm.

2. METHODS FOR SPEEDING UP

In this paper, we tried to use the experiences in related articles and use parallelization methods, pipelines, optimal VHDL code, proper synthesis software configuration, proper data and key pin arrangement, family selection of FPGA chips Virtex Xilinx, with its new technology and minimalistic latency, and creativity in the implementation and integration of methods, is rapidly reaching for the implementation of the AES algorithm. Although all optimization solutions are provided, a certain improvement in the execution speed of the algorithm is created, but it should be noted that the sum of all of these cases can be significant in optimizing results.

2.1. Choosing the Right FPGA

The FPGA chip with the XC7VX330T-3FFG1157 number from Xilinx Company is selected as the processor of this article. Its 28-nanometer chip technology is one of the new products of the company. The most important reasons for choosing this chip are the up-to-date technology of making it and the ability to achieve higher speed in implementing the algorithm. The maximum clock frequency is 800 MHz. The speeds available on the standard LVCMOS18 port for inputs and outputs of this FPGA range are below 300 MHz.

2.2. Correct Clock Connection to FPGA

The main clock pulse connection to the FPGA is an important point for optimization. The connection of the clock to its specific stand can improve the speed of implementation of an implemented algorithm several times. High-speed and low-latency hardware routes move the clock to the entire FPGA internal block with minimal error. In fact, the use of existing FPGA-based hardware paths with a minimum delay is crucial for transmitting and distributing high-speed clock pulses in the chip and for all synchronous functions.

2.3. Proper Setup of Synthesis Software

The VHDL synthesizer software makes it possible to apply important settings to implement an algorithm. Optimization of the implementation should be adjusted according to the requirements of the synthesizing software. In this paper, with the goal of achieving high speed in the implementation of the algorithm, the following settings have been applied to the software. In the Design Goal and Strategies window, you have to select the Timing Performance toolbar in the Design Goal toolbar and consider the Performance without IOB Packing.

By performing the above settings in the VHDL synthesis tool, the optimization priority will be assigned to the implementation of the algorithm on the hardware. Therefore, initial considerations such as adjoining blocks of processing and their non-dispersion at the level of the chip's gates, fixing connections from the shortest possible paths to reduce the delay time, matching the hardware corresponding to the code of the description and some other important points will be considered by default on the software.

2.4. Set the Speed of the Clock

The applicable speed limits are added to the project by the constraints rules and specify the implementer's request for the synthesis software. The frequency limit set in the UCF file, along with the proper layout of the bases, will create the working frequency domain for achieving a more optimal speed.

In this paper, due to the design, the maximum speed of 500 MHz for the clock range is considered.

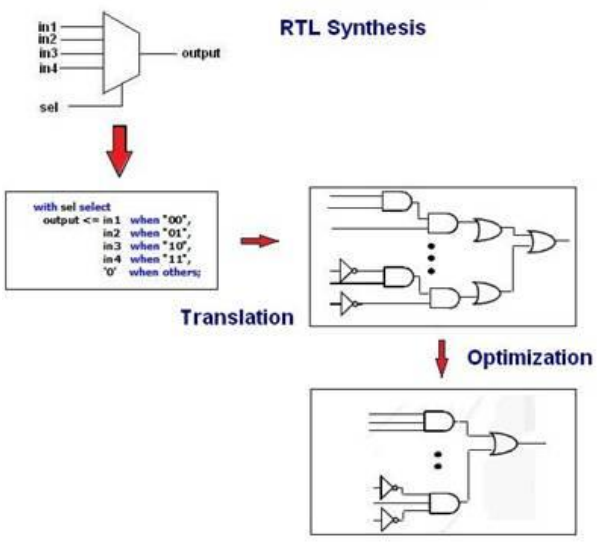

Fig. 1. Software optimization.

2.5. Optimal Implementation of the Algorithm

Another important issue that is effective in optimizing the speed of the algorithm is how to design the hardware implemented with the VHDL code. A significant part of the speed of the algorithm is related to the correct design, the lack of use of complex mathematical functions, the inability to use the successive classes of logical gates, Failure to use unnecessary registers, the non-integration of parts that can be merged with each other, and so on.

Obviously, a strong implementation requires good mastery of the algorithm and the way its functions are implemented by the hardware description language (VHDL). In many cases, functions that are easily implemented with rational circuits are replaced by complex mathematical functions, which not only slow down the execution of the algorithm but also occupy the space on the chip and its power consumption Increases. The goal in this article is achieving the maximum clock speed of the clock, which the algorithm can work with and produce the highest output correct data rates. Given the importance of this issue, a more complete description of how the optimal implementation of the AES code algorithm is presented below.

The input and output ports of the algorithm on the FPGA include 128 bit data inputs with 128-bit key encryption and the 128-bit encrypted output, along with clock and reset signals, connected to the FPGA. The asynchronous reset signal at the start determines all the signals used in implementing the algorithm with a zero value. In the input data and encryption key, there is a register that will prepare the data for the Initial Round portion with each clock. According to the blocks of the AES algorithm, initially, the XOR should be executed between the encryption key and the input data in the Initial Round. In the next step, the calculated S-Box values, which have been mapped to FPGA memory on ROM, are extracted. The method of putting the calculated values of the 256 possible modes into memory would appear to result in a higher rate than the calculated mode by multiplying the matrix. The Shift Row, Mix columns, and Add round Key operations should continue. To increase the speed of the Shift Row, simultaneously extract the S-Box values from the ROM and set it to a newly defined signal. With this synchronous operation, the algorithm will be partially executed. Then the extracted data from the S-Box table, which is applied to the required shift, is included in the Mixcolumns section. In this section, the matrix multiplication is performed on the data and the output is in the defined array and its placement in the variables of this function is performed according to the algorithm. At the end, XOR of the extended key with the Mixcolumns transferred to a register. This loop is repeated 9 times, and eventually in the final step without mix columns operation, the data is placed in the output register. Implementation of key expansion steps can also be done at first.

The important points to be mentioned in the implementation of speed optimization are the successive 9-repeat loop repetition of the algorithm, in the form of burst blocks and their registration, which has a significant effect on the speed of the algorithm. With this method, when the data is being processed, the next data is processed after the successive clocks behind the first data.

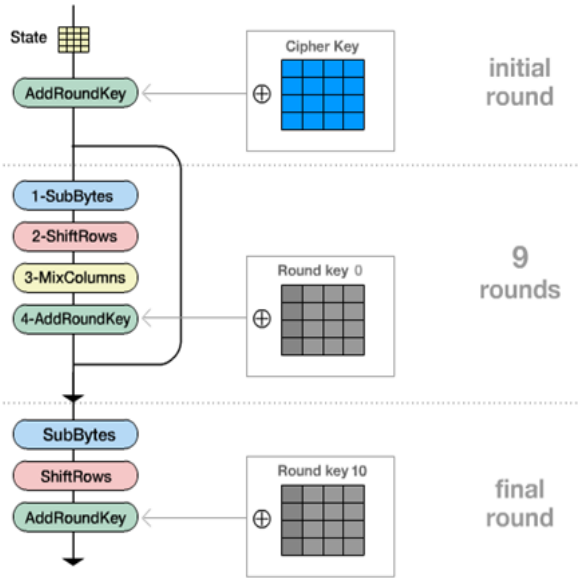

Fig. 2. Stages of AES.

Another significant issue is the fact that it is not necessary to use a separate key in executing the encryption algorithm for any independent data. With this in mind, one can only calculate the expanded key once and avoid for the next input data, from the point of view of the key changes.

2.6. Implementation as Combinational Logic Circuits

Parts of the AES encryption algorithm, such as XOR operations or shifts, as well as the replacement of values in tables and variables is be able to implement as an integrated and independent circuits of the clock. This independence from the input clock results in a quick access to the output of the function with any changes in its input. The delay in this part of the implementation is noticeable, the same as Propagation Delay. This delay is allowed to be extended since it does not restrict the original clock system and the factor which limits the clock is in another part of the circuit. Otherwise for causing a large release delay, the pipeline operation must be performed on the signal path.

2.7. Use Parallelism

In this paper, due to the high capability of FPGA chips in performing parallel processing, and in order to increase the speed of implementation of the AES encryption algorithm, parallel implementation was considered. Unlike sequential processors that execute this algorithm as a byte, all of the 128-bit inputs are processed concurrent in the FPGA chip. The lack of limitation in the number of registry bits defined in the FPGA provides the ability to perform operations such as XOR the key input data as a concurrent hybrid circuit.

In this implementation, for increase total throughput, three AES core running parallel on one FPGA. With this method, with same frequency clock, throughput increased triple.

2.8. Use Pipeline Method

The pipeline method is one of the most effective methods for optimizing the speed of the algorithm on the FPGA. In this paper, to achieve maximum speed, 52 register were used in the input-to-output data path. Obviously, these 52 registry steps will cause the initial delay, but the speed of the algorithm will increase significantly. These registers are executed once after each step of the algorithm, and a total of 52 registry steps are used.

2.9. Separating the Key Expansion Clock from Main Clock

Considering that in a cryptographic module, the key does not change at high speed, and the change in the message sender requires a change of key in the receiver and requires an independent mechanism, the clock frequency necessary to create the extended key, is separated from the original clock and has been shifted to the key expansion circuit with half its frequency. By this method, the speed bottleneck on running of the implemented algorithm is not in the key expansion circuit and the possibility of higher optimization can be achieved by making corrections in the input data path to the output.

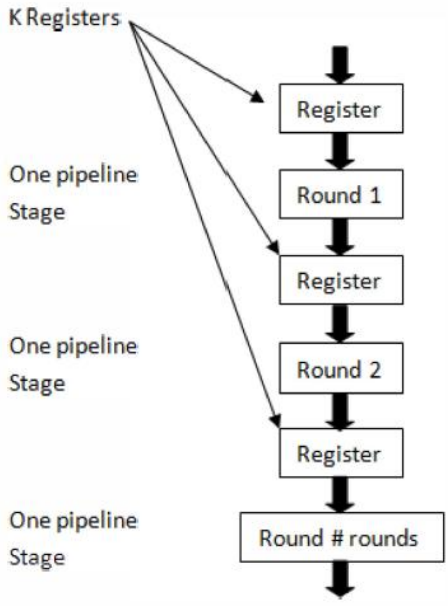

Fig. 3. Pipeline method [8].

3. RESULTS

The maximum execution speed of AES algorithm on the XC7VX330T-3FFG1157 chip is 500 MHz as a result, the throughput of this algorithm is 192 Gbps. Also, the initial delay for output is 52 clocks (equivalent to about 104 nanoseconds) at the beginning of the pipeline, and all outputs generated per clock cycle appear at this output delay. When placing the output data on the port, the selected FPGA speed limit should be considered for communication with the output port. It should be noted that the data and the input key must be changed at the negative edge of the clock so that the FPGA can registered it at the positive edge of the registry entries.

The output waveform of the VHDL code implemented in the ISIM software is similar to the following:

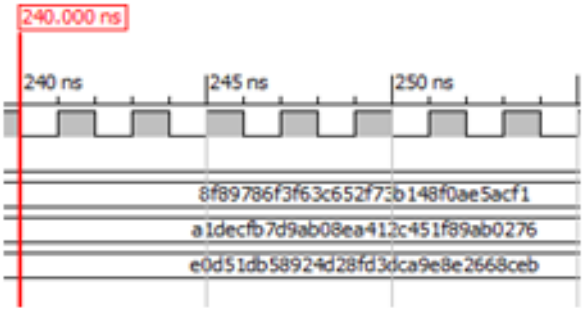

Fig. 4. Simulation result.

In this simulation, the input data of a 128-bit is hexadecimal number:

A1DECFB7D9AB08EA412C451F89AB0276

and the key is 128-bit:

8F89786F3F63C652F73B148F0AE5ACF1

and the result of outputting 128-bit is:

E0D51DB58924D28FD3DCA9E8E2668CEB

which is achieved by clock with 2 nanoseconds period (equivalent to 500 MHz).

FPGA consumables are used in accordance with the table 2.

Table 1. ***Maximum frequency of AES algorithm on chip XC7VX330T-3FFG1157.***

```
Data Sheet report:
------------------
All values displayed in nanoseconds (ns)

Clock to Setup on destination clock clk
---------------+---------+---------+---------+---------+
               | Src:Rise| Src:Fall| Src:Rise| Src:Fall|
Source Clock   |Dest:Rise|Dest:Rise|Dest:Fall|Dest:Fall|
---------------+---------+---------+---------+---------+
clk            |   1.999|         |         |         |
---------------+---------+---------+---------+---------+


Timing summary:
---------------

Timing errors: 0  Score: 0  (Setup/Max: 0, Hold: 0)

Constraints cover 130294 paths, 0 nets, and 137022
connections
Design statistics:
   Minimum period:   1.999ns{1}
  (Maximum frequency: 500.250MHz)
```

Table 2. ***Resources used on FPGA.***

Device Utilization Summary			
Slice Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice Registers	21,828	408,000	5%
Number of Slice LUTs	23,081	204,000	11%
Number of occupied Slices	8,476	51,000	16%

4. COMPARISON

The studies on a large number of articles show that the speed of implementation of the optimization carried out in this paper is relatively high compared to other papers, one of the highest achievable speeds on FPGA. Also, in some arti-

cles that have cluster frequencies in this range [1, 2], the encoded data is not encountered at the output per clock, and therefore the final output data rate is much less than the results of this paper. In order to better evaluate, this algorithm compared with other result.

In the paper [3], the implementation of AES algorithm was carried out on the FPGA of the Virtex-6 family with the number XC6VSX315T and the clock frequency was 254 MHz.

In the paper [4], the implementation of AES algorithm was performed on the FPGA of the Virtex-6 family and data throughput is 1.3 Gbps.

In the paper [5], the implementation of AES algorithm was performed on the FPGA of the Virtex-4 family with the number XC4VFX140-11FF1517 and the data output rate was 1.3 Gbps.

In articles [6] [7], AES algorithm was implemented on the FPGA of the Spartan-3 with numbers 3S400TQ144-5 and XC3S1500L-4FG676, and the clock frequency was 116.4 MHz and 188.8 MHz.

5. CONCLUSION

As presented in this paper, using the proposed methods, it is possible to increase the speed of AES standard encryption algorithm with proper implementation and applications of synthesizing tools for FPGA. FPGA chips with new technology, modern synthesis tools, and precise implementation methods with the Very High-Speed Integrated Circuit Hardware Description Language (VHDL) make it possible in the future to run faster. It is recommended that people interested in this topic re-implement the process of implementation of this algorithm with the Vivalo synthesis tool provided by Xilinx Company, and in addition to applying creativity in the implementation and use of tools capabilities new, with the introduction of the ultra-scale and ultra-scale + chips, will capture new experiences in speeding up the implementation of the Encryption standard algorithm.

Definitely, the ideas used to speed up optimization are worthwhile and significant differences in the results of work with non-optimal implementations will be observed. In addition to the above, it is suggested that, with the full parallelization of a multiple encryption algorithm on a chip, and the management of the input data and the key for it, it will be possible to get higher speed encryption chips for specific applications.

ACKNOWLEDGMENT

The first author would like to thank and appreciate Dr. Amir Amirabadi Zavare. What has been presented is a small reflection of his efforts and his teachings during the graduate years.

REFERENCES

[1] Kirat Pal Singh, Shiwani Dod, "An Efficient Hardware design and Implementation of Advanced Encryption Standard (AES) Algorithm", Special Issue on International Journal of Recent Advances in Engineering & Technology (IJRAET) V-4 I-For National Conference on Recent Innovations in Science, Technology & Management (NCRISTM) ISSN (Online): 2347-2812, Gurgaon Institute of Technology and Management, February 2016.

[2] Mradul Upadhyay, Utsav Malviya, "High Speed Advanced Encryption Standards Using Piplining", International Journal of Science and Research (IJSR), 2012.

[3] Vijaya Kumar. B., T. Thammi Reddy, "FPGA Implementation of High Speed AES Algorithm for Improving The System Computing Speed", International Journal of Computer Trends and Technology (IJCTT),V4(9):2981-2985, September Issue 2013.

[4] Mradul Upadhyay, Utsav Malviya, "High Speed Advanced Encryption Standards Using Piplining", International Journal of Science and Research (IJSR), 2012.

[5] Ms.Anuradha Balasubramaniam, "High Speed AES Cipher Engine", International Journal of Engineering and Applied Sciences (IJEAS) ISSN: 2394-3661, Volume-2, Issue-7, July 2015.

[6] Sujatha Hiremath, M.S.Suma, "Advanced Encryption Standard Implemented on

FPGA", Second International Conference on Computer and Electrical Engineering, 2009.

[7] S. M. Umar Talha, Mir Asif, Hammad Hussain, Ali Asghar, Hadi Ameen, "Efficient Advance Encryption Standard (AES) Implementation on FPGA Using Xilinx System Generator", IEEE, 2016.

[8] A. P. Anusha Naidu, B. Prof (Mrs.) Poorvik. Joshi, "FPGA Implementation of Fully Pipelined Advanced Encryption Standard" IEEE ICCSP conference, 2015.

[9] Sonali A. Varhade, N. N. Kasat, "Implementation of AES Algorithm Using FPGA & Its Performance Analysis" International Journal of Science and Research (IJSR), Volume 4 Issue 5, May 2015.