# A HARDWARE IMPLEMENTATION IN FPGA OF THE RIJNDAEL ALGORITHM

*Cristian Chiţu*[*,**], *David Chien*[**], *Charles Chien*[**], *Ingrid Verbauwhede*[*,**], *Frank Chang*[*,**]

[*]University of California Los Angeles, Department of Electrical Engineering
420 Westwood Plaza, Los Angeles, CA 90095, USA
[**]G-Plus, Inc., 3420 Ocean Park Blvd., Suite 3070
Santa Monica, CA 90405, USA
e-mail: cochitu@ee.ucla.edu

## ABSTRACT

Implementation in FPGA of the new Advanced Encryption Standard, Rijndael, was developed and experimentally tested using the Insight Development Kit board, based on Xilinx Virtex II XC2V1000-4 device. The experimental clock frequency was equal to 75 MHz and translates to the throughputs of 739 Mbit/s for Rijndael with block size and key size of 128 bits, respectively. This circuit has capability to handle encryption/decryption and fitted in one FPGA taking approximately 84 % of the area. Our work supplements and extends other research efforts [1] [2].

## 1. INTRODUCTION

In 1997, The National Institute of Standards and Technology (NIST) initiated an effort towards developing a new encryption standard, called Advanced Encryption Standard (AES). The development of the new standard was organized in the form of a contest coordinated by NIST. In October 2000, Rijndael [3] was announced as the winner of the contest and a future Advanced Encryption Standard. Rijndael proved to be one of fastest and most efficient algorithms. It is also easily implemented on a wide range of platforms and is extendable to other key and block lengths.

This paper evaluates the Rijndael cipher implementation from the viewpoint of its hardware mapping into high performance Xilinx FPGA. An FPGA implementation can be easily upgrated to incorporate any protocol changes without the need for expensive and time consuming physical design, fabrication, and testing required in case of ASICs. Our paper is organized as follows. A brief overview of Rijndael cipher algorithm and its basic building blocks is given in Section 2. Section 3 outlines the design of the pipelined Rijndael implementation. Performance results and the test setup are given in Section 4. Finally, in Section 5, possible future work is described and concluding remarks are made.

## 2. IMPLEMENTATION OF THE RIJNDAEL ALGORITHM

Rijndael is a symmetric key block cypher with a variable key size and a variable input/output block size. Our implementation supports only one key size of 128 bits and is limited to the block size of 128 bits, which is the only block size required by Advanced Encryption Standard. Implementing other block sizes, specified in the original, non standardized description of Rijndael is not justified from the economical point of view, as it would substantially increase circuit area and cost without any substantial gain in the cipher security.

The Rijndael algorithm is a substitution linear transformation cypher based on S-boxes and operations in the Galois Fields. Below we describe the way of implementing all component operations of Rijndael.

Implementation of the encryption round of Rijndael requires realization of four component operations: *Substitution*, $ShiftRow$, $MixColumn$, and $KeyAddition$. Implementation of the decryption round of Rijndael requires four inverse operations: $InvSubstitution$, $InvShiftRow$, $InvMixColumn$, and $KeyAddition$.

*Substitution* is composed of sixteen identical S-boxes working in parallel. $InvSubstitution$ is composed of the same number of inverse S-boxes. Each of these S-boxes can be implemented independently using a 256x8 bit look-up table.

$ShiftRow$ and $InvShiftRow$ change the order of bytes within a 16 byte (128 bit) word. Both transformations involve only changing the order of signals, and therefore they can be implemented using routing only, and do not require any logic resources, such as Configurable Logic Blocks (CLBs) or dedicated RAM.

The $MixColumn$ transformation as well as $InvMixColumn$ can be expressed as a matrix multiplication in the Galois Field GF($2^8$). The $InvMixColumn$ transformation has a longer critical path compared to the

0-7803-7523-8/02/$17.00 ©2002 IEEE

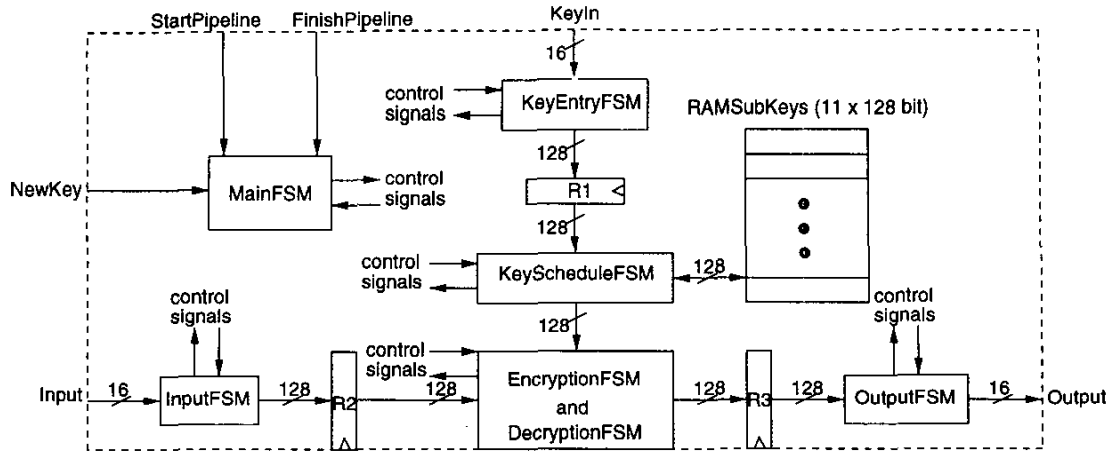


Figure 1: Architecture of the circuit.

*MixColumn* transformation, and therefore the entire decryption is more time consuming than encryption.

*KeyAddition* is a bitwise XOR of two 128 bit words.

## 3. ARCHITECTURE OF THE ENCRYPTION/DECRYPTION

The organization of the hardware implementation of the circuit is shown in Figure 1. The organization includes the following units:

- *EncryptionFSM* and *DecryptionFSM* (Encryption and Decryption Finite State Machines), used to encipher and decipher input blocks of data.
- *KeyScheduleFSM* (Key Scheduling Finite State Machine), used to compute a set of internal cipher keys based on a single external key.
- *RAMSubKeys* (RAM memory) of internal keys, used to store internal keys computed by the *KeyScheduleFSM*, or load the initial key to the FPGA through the Key Entry Interface.
- *InputFSM* (Input Interface Finite State Machine), used to load blocks of input data and to store input blocks awaiting encryption/decryption.
- *KeyEntryFSM* (Key Entry Interface Finite State Machine), used to load the external key.
- *OutputFSM* (Output Interface Finite State Machine), used to temporarily store output from the encryption/decryption unit.
- *MainFSM* (Main Control Finite State Machine), used to generate control signals for all other units.

The registers *R*1 through *R*3 allows the circuit to process data in parallel with an encryption or decryption.

Both the input and output channels are 16 bits wide. Therefore, in order to read in the whole cipher or key, a handshaking protocol is used. The top level controller *MainFSM* provides control signals necessary for the handshaking protocol.

| Module | Round | *sel*1 | *sel*2 |
|---|---|---|---|
| *EncryptionFSM* | Initial | 0 | 0 |
| | Iteration | 1 | 0 |
| | Final | 1 | 1 |
| *DecryptionFSM* | Initial | 0 | 0 |
| | Iteration | 1 | 1 |
| | Final | 1 | 0 |

Table 1: Selection signals for encryption and decryption.

As mentioned in the abstract, we selected 128 bit size for the key. *KeyEntryFSM* loads the key and *KeyScheduleFSM* produces subkeys of 128 bits. The subkey pieces are then passed and stored into the RAM memory *RAMSubKeys*. The memory has eleven locations: ten locations are for the subkeys and one for the key. It is important to store the initial key since is used in the decryption process.

The implementation of the encryption and decryption Finite State Machines is shown in Figure 2 and Figure 3, respectively. The values of the selection signals *sel*1 and *sel*2 for the multiplexers are presented in Table 1. The critical path is located in the decryption module, and includes *KeyAddition* (an XOR operation), *InvMixColumn*, multiplexer, *InvShiftRow*, *InvSubstitution*, and multiplexer. There are one initial round, nine iterations and one final round.

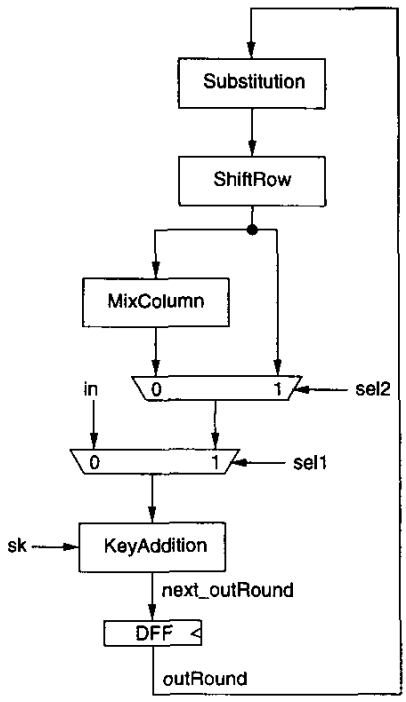


Figure 2: Block diagram of the $EncryptionFSM$.

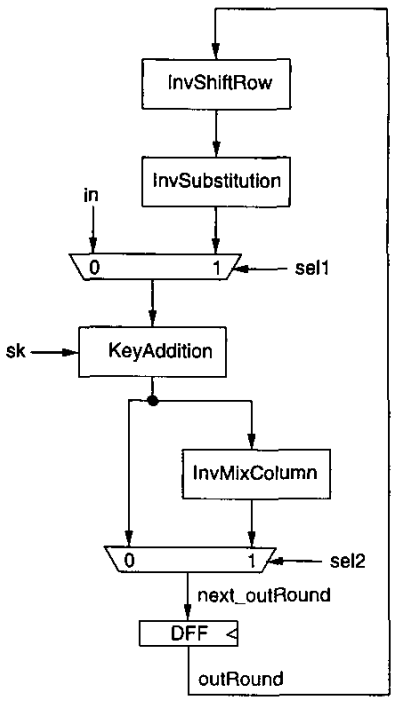


Figure 3: Block diagram of the $DecryptionFSM$.

## 4. TEST SETUP

The Rijndael cipher was first described in VERILOG, and its description verified using the VERILOG-XL simulator from Cadence Design Systems. Test vectors from the reference software implementations [4] were used for debugging and verification of VERILOG codes. The revised VERILOG code became an input to Xilinx ISE Series 4.1i software [5] performing the logic synthesis, mapping, placing, and routing. These tools generated reports describing the area and speed of implementation, a netlist used for timing simulations, and a bitstream to be used to program the FPGA device Virtex II XC2V1000-4.

Figure 4 shows a block diagram of the measurement setup. In order to program the FPGA, a SUN workstation was connected to the Insight Virtex II Development Kit board [6]. The board was connected to the Logic Analysis System Agilent 16702B [7] which provided and displayed signals during measurements.

The results of the FPGA implementation are summarized in Table 2. The throughput [8] of the circuit is given by:

$$Throughput = 128\ bits/13\ clocks \cdot 75\ MHz,$$

where 13 clocks is the number of cycles used by the modules $EncryptionFSM$ or $DecryptionFSM$ for an encryption or decryption.

## 5. CONCLUSIONS

In this paper we have evaluated the Rijndael cipher from the point of view of its implementation in FPGA.

The new architecture presented allows the implementation of the Rijndael cipher with high speed encryption and decryption. The experimental procedure demonstrated that the total encryption and decryption throughput of 739 Mbit/s can be achieved using a single FPGA device. Only up to 84 % of resources of this single FPGA are required by all cryptographic modules.

Future development will include integration of the modes of operation CBC, CFB, and OCB which are considered secure for transmission of large volumes of data.

## 6. ACKNOWLEDGMENTS

This work was supported by G-Plus, Inc., which is a rapidly growing fabless semiconductor provider offering optimized RF solutions to enable broadband wireless networking and fixed wireless access system applications.

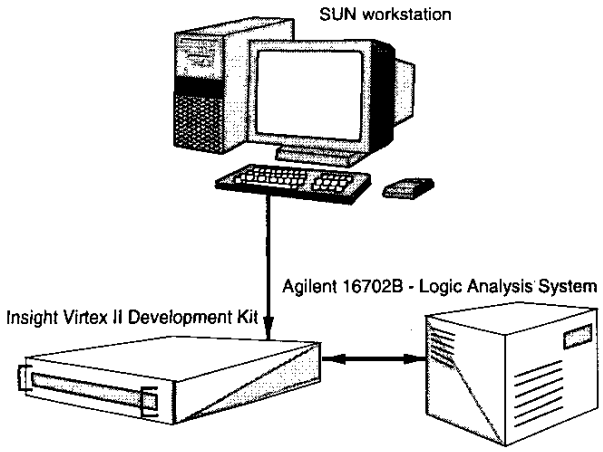


Figure 4: Experimental setup.

## 7. REFERENCES

[1] H. Kuo, I. Verbauwhede, "Architectural Optimization for a 1.82 Gb/s VLSI Implementation of the AES Rijndael algorithm", Cryptographic Hardware and Embedded Systems (CHES 2001), LNCS 2162, pp. 51-64, Springer-Verlag.

[2] P. Schaumont, H. Kuo, I. Verbauwhede, "Unlocking the Design Secrets of a 2.29 Gb/s Rijndael Core", Design Automation Conference 2002.

[3] National Institute of Standards and Technology, "Advanced Encryption Standard (AES)", http://csrc.nist.gov/publications/fips, FIPS 197, November 2001.

[4] Vincent Rijmen, "The block cipher Rijndael", http://www.esat.kuleuven.ac.be/~rijmen/rijndael.

[5] Xilinx, "ISE 4 User Guide", Xilinx, Inc., 2001.

[6] Insight Electronics, "Virtex II Development Kit", http://www.insight-electronics.com/virtexII, Memec Inc., San Diego, CA 92121.

[7] Agilent Technologies, "Training Kit for the Agilent Technologies 16700 - Series Logic Analysis System, Making Basic Measurements", 16700-97020, Agilent Technologies, August 2001.

[8] Kris Gaj and Pawel Chodowiec, "Fast implementation and fair comparison of the final candidates for Advanced Encryption Standard using Field Programmable Gate Arrays", *Proceedings RSA Security Conference*, San Francisco, CA, April 8-12, 2001.

| Target FPGA device | Virtex II XC2V1000-4 |
|---|---|
| Maximum clock frequency | 75 MHz |
| Encryption/decryption throughput | 739 Mbit/s |
| Area | |
| CLB slices | 4,325 |
| Block ROMs | 37 |
| Block RAMs | 1 |
| Percentage occupied in device | 84% |

Table 2: Results of FPGA implementation.