**ISBN** 978-981-11-3671-9
**Proceedings of 2017 the 7th International Workshop on Computer Science and Engineering (WCSE 2017)**
*Beijing*, *25-27 June*, *2017*, *pp. 977-982*
doi: 10.18178/wcse.2017.06.169

# An Enhanced Smart Card Memory Ciphering System Implementation on FPGA

Wira Firdaus Yaakob [1]and Jahariah Sampe [2+]

[1, 2] Institute of Microengineering and Nanoelectronics, Universiti Kebangsaan Malaysia (UKM), Universiti Kebangsaan Malaysia (UKM), 43600 Bangi, Selangor, Malaysia

**Abstract.** This paper describes about the design and implementation of an enhanced memory ciphering system of a smart card prototype in Cyclone V System-on-a-Chip (SoC) 5CSEMA5F31C6N Field Programmable Gate Array (FPGA) device. The comparison between the design implementation in this device and in Xilinx's Zynq-7000 XC7020-1-CLG484 FPGA device is explained in this paper in terms of resource utilization and time requirements. The memory ciphering system in the smart card is capable to complete in 40ns that is a single cycle CPU of the smart card. The implementation in the Cyclone V SoC has the least utilized logics and the highest maximum frequency (Fmax) that are 8,313 slices and 195 MHz respectively. The smart card memory ciphering system provides significant enhancement in terms of security and performance especially for smart card secured applications like national identification (ID), financial transactions and health insurance.

**Keywords:** Smart Card; AES; ARM Cortex A9; XC7Z020; XC4VLX60; 5CSEMA5F31C6N

## 1. Introduction

Smart card is used widely in security-sensitive applications including payment systems [1]. Due to the confidential electronic data that resides in the smart card, many aggressive attacks have taken place till these days in order to attain it. Thus, its security is a major concern and a lot of studies has been done by worldwide researchers in order to enhance the security measures of the smart card system without degrading its performance [2]. One of the studies is about the enhancement of the smart card memory ciphering system [3].

The smart card design in this paper is developed with an enhanced memory ciphering system that contains three major units; Advanced Encryption Standard (AES) cipher block, Random Number Generator (RNG) Key Generation and Scrambler/Descrambler. This system is developed in the Memory Management Processing Unit (MMPU) of the smart card. By having the AES cipher unit, the plaintext from the central processing unit (CPU) is encrypted or decrypted using a random key that is generated by the RNG Key Generation unit.

The encrypted data also called as ciphertext is scrambled with the data from the Scrambler/Descrambler unit before being written into the memory during the write mode. Meanwhile during the read mode, the secured data from the data memory is descrambled with the data from the Scrambler/Descrambler unit into the ciphertext. For memory types that allow for data reading only, e.g. ROM typically storing executable code, the process will be one way only i.e. descrambling and decryption. User Personal Identification Number (PIN) is utilized in the scrambling and descrambling processes. This prototype system is implemented in the Zynq-7000 XC7020-1-CLG484 and Cyclone V System-on-a-Chip (SoC) 5CSEMA5F31C6N FPGA devices.

+ Corresponding author..
*E-mail address*: jahariah@ukm.edu.my.

The system is managed to complete the process within a single cycle CPU that is about 40ns with less hardware resources and high throughput of 9085 (Mbits/s). The descriptions of the memory ciphering system with the FPGA implementation results are explained in the following sections. The overall smart card diagram is illustrated in Figure 1.

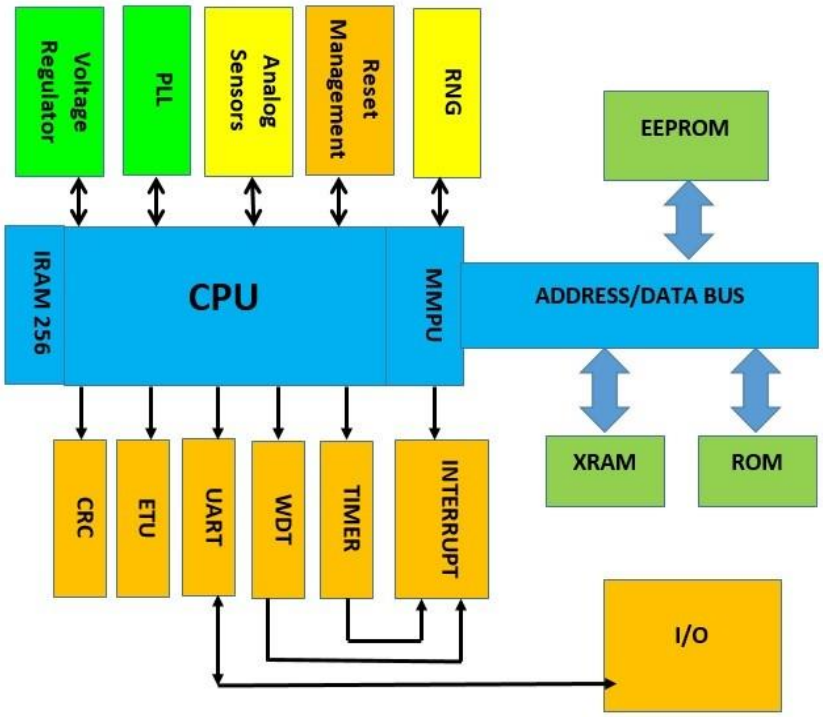


Fig. 1: Smart Card block diagram.

## 2. Memory Ciphering System Functions

The complete processing cycle of the system as shown in Figure 2 has been reduced from more than 20 clock cycles to a single cycle CPU only that is approximately 40ns or 25MHz. In the previous design, the 20 clock cycles are due to the AES-128 cipher, scrambling and descrambling processes [4].

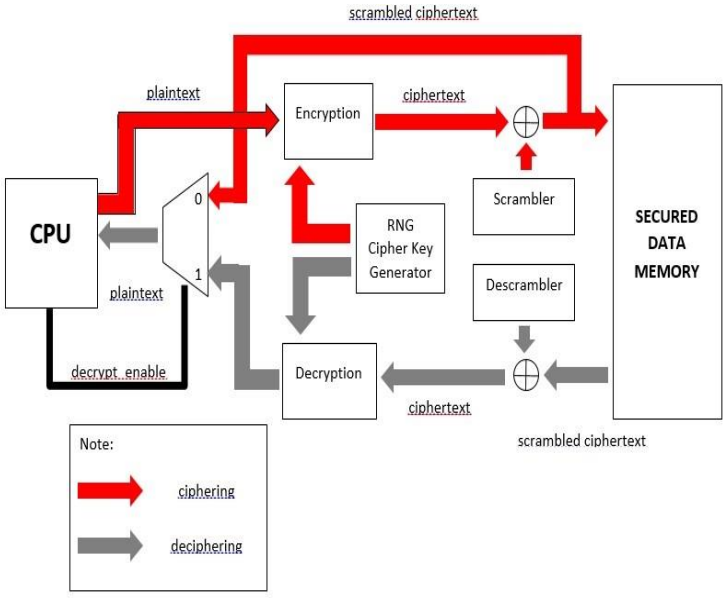


Fig. 2: Memory Ciphering System in the Smart Card

## 2.1. AES-128 Cipher unit

The AES-128 cipher is based on the AES Rijndael algorithm and is compliant to Federal Information Processing Standard Publications (FIPS PUB) 197. The cipher is a symmetric block cipher that can process plaintext blocks of 128 bits. Rijndael is a key-iterated block cipher, meaning that the initial block of plaintext and cipher key traverses through multiple rounds of transformation before generating the output [5]. A state is the intermediate result at the end of each round. There are 9 rounds of transformation for this 128-bit key AES. Each round consists of multiple operations like byte substitution, shift row, mix column and key addition. Byte substitution is the operation that runs independently on each byte of the state using a substitution table that is called as S-box. Shift row's operation runs by shifting bytes of the last three rows of the state cyclically while the first row is not shifted. Mix column's operation is the transformation process that runs on the state column-by-column. Each column is treated as a four-term polynomial. The final operation for each round that is key addition, operates by adding the round key to the state by a simple bitwise XOR. Other than key size 128 bits that is used in this study, Rijndael supports also for 192 and 256 bits key to encrypt plaintext blocks that are 192-bit and 256-bit respectively [6].

AES Rijndael cipher's algorithm as shown in Figure 3 can perform very well in both hardware and software with a very low memory requirements that helps to avoid high cost design development due to the large memory requirement and slow processing performance. The key or plaintext block size decides the number of rounds. The inverse cipher is called as decryption [7]. During decryption process, the encryption key schedules' form is the same as encryption but the sequence of the transformations is done in reverse order. AES Rijndael is also adopted in the FIPS standard that is documented by National Institute of Standards and Technology (NIST). The cipher has proven reliable due to its high computational complexity [8].

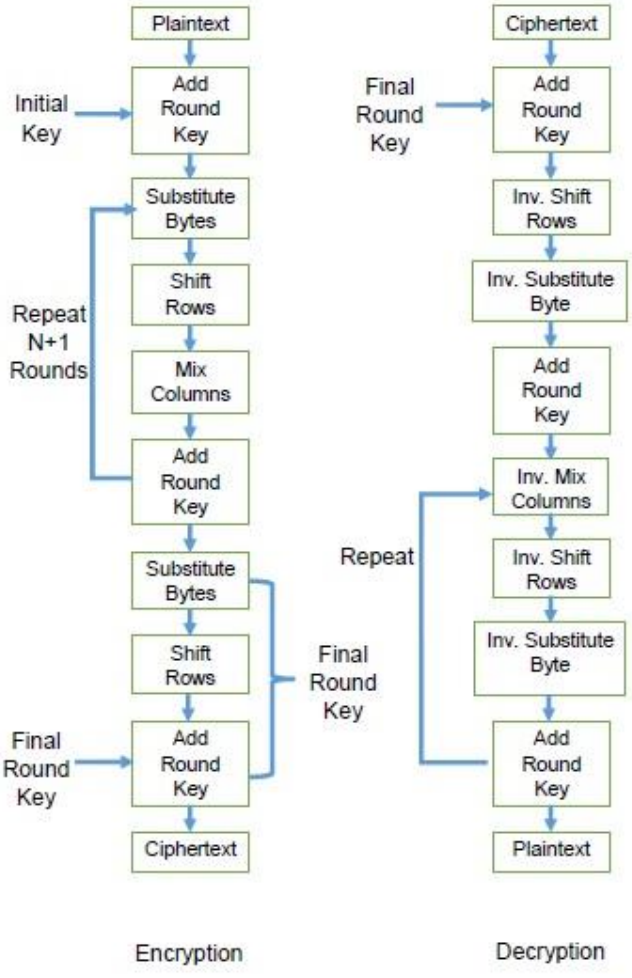


Fig. 3: AES-128 Rijndael Flow

## 2.2. RNG Key Generator unit

The unit utilizes 128 bits of RNG data as the seed for cipher key generation. In the FPGA implementation, the RNG is prototyped as Pseudo RNG (PRNG) using a Linear Feedback Shift Register (LFSR) with a fixed seed. However, in the ASIC implementation later, the PRNG unit is replaced with a mixed-signal design of a True RNG (TRNG) IP for the seed generation. TRNG IP is a physical source for randomness that ensures the maximum guessing effort depending on the key length [9]. For security reasons, the generated key that is used for complementary encryption and decryption operations is randomly generated for every transactions [10].

## 2.3. Scrambler/Descrambler unit

The unit generates 128 bits scramble data that is mixed with the ciphertext in order to make it more difficult for the attacker to recognize the data from outside. The process of scrambling and descrambling requires user PIN input in order to provide more secured access [11]. Without the correct PIN keys, the secured data from the memory cannot be descrambled into ciphertext and results the data cannot be accessed.

# 3. Design Simulation

As shown in Figure 4, the complete process of the proposed ciphering system takes only a single cycle CPU that is approximately 40 ns or 25MHz. The complete process cycle starts from the plaintext encryption and ends with the decryption of the ciphertext. The signal kld_cipher is used as a flag to enable the encryption process and the signal 'done' is used to notify the encryption process is completed.

Figure 5 shows the decryption result of ciphertext. As can be seen from the figure, the decrypted ciphertext is similar with the original plaintext that is in hexadecimal value "01". The decrypted data and the plaintext value is highlighted with the circle. The signal 'done2' is a flag to notify the decryption process has been completed. The 'encrypted_mem' signal is the encrypted data or ciphertext that goes into the Scrambler/Descrambler unit.

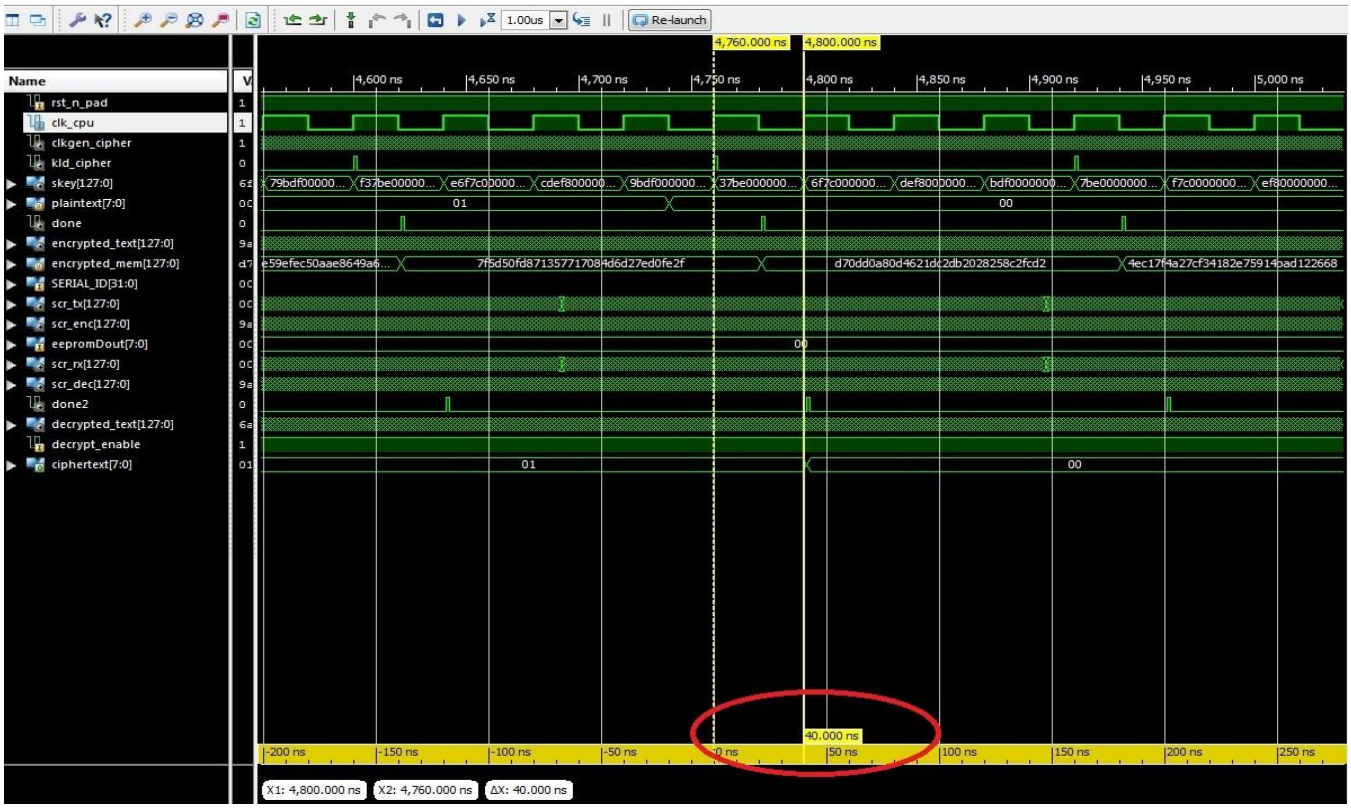


Fig. 4: Completion of a single ciphering cycle is one clock period of the CPU, 40 ns.

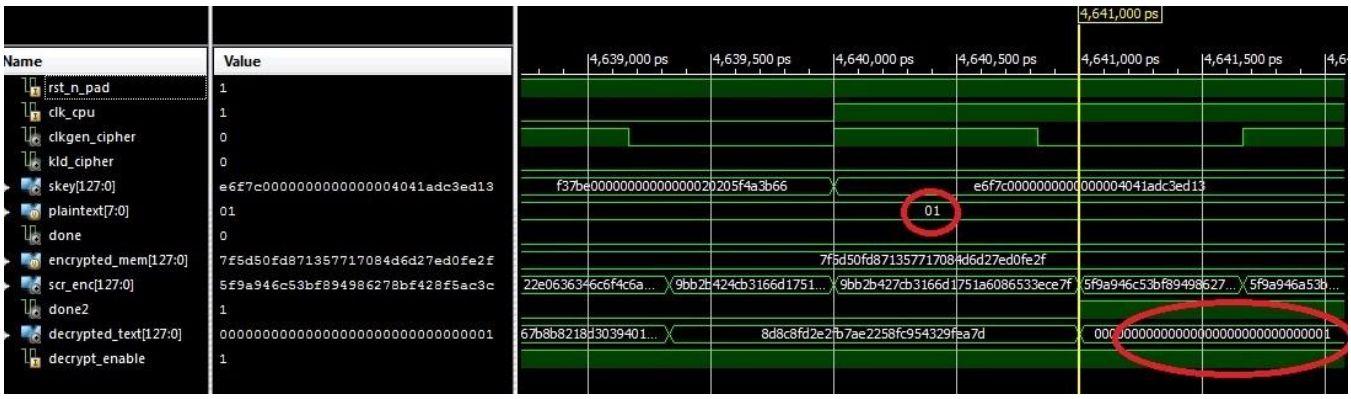


Fig. 5: Secured data is successfully decrypted to its plaintext value i.e. Hex 01.

## 4. Results & Discussion

The comparison results of the ARM-based FPGA implementations are shown in Table 1. The adaptive logic module (ALM) utilization percentage in the Altera FPGA is much less than look-up table (LUT) utilization in the Xilinx FPGA. The difference is about 10.37 %. This means the logic slices is reduced twice in the Altera FPGA fabric. This is due to in ALM architecture, less area is utilized for more combinational logic, thus, gaining a higher logic density than a standard four-input LUT [12]. The equation 1 is the conversion formula from ALM into LUT slice, normalized relatives to logic capacity for these types of FPGA devices. The equation means the logic capacity of one Altera Cyclone V SoC ALM is equivalent to that of 2.55 Xilinx Zynq SoC slices [13].

$$\text{Number of LUT slice} = 2.55 \text{ X ALM} \qquad (1)$$

Table I. Table Compariosn Results Of The Smart Card Design In Xilinx Zynq SOC And Altera Cyclone V SOC

| Resource | Xilinx Zynq SoC (xc7z020-1clg484) | Altera Cyclone V SoC (5CSEMA6F31C6) |
|---|---|---|
| Slice LUT/ALM | 20.54 % | 10.17 % |
| Slice Register (FF) | 2.33 % | 1.97 % |
| BRAM/M10K | 37.5 % | 30.22 % |
| DSP | 1.36 % | 2.3 % |
| Max. Freq (Fmax) | 104 MHz | 195 MHz |
| Combinational Path Delay | 10.81 ns | 13.74 ns |

From Table 1 as well, the percentage of utilized registers mainly flip-flops (FF) in the Altera FPGA is 0.36 % less than the Xilinx FPGA. The Altera FPGA is also less than the Xilinx FPGA in terms of memory block utilization that is 7.28 %. This means that the memory prototyping for the smart card design is more optimized in the Altera FPGA than the Xilinx FPGA. The Altera FPGA utilizes configurable M10K block for memory prototyping while the Xilinx FPGA utilizes configurable block random access memory (BRAM). The DSP utilization result is quite different from the three previous resources' results. The table shows that the utilization in the Altera FPGA is higher compared to the Xilinx FPGA that is 0.94% difference.

The Fmax of ARM-based Altera FPGA is 91MHz higher than the Fmax of the ARM-based Xilinx FPGA. The Fmax is a clock frequency that is achieved without internal setup (tSU) and hold (tH) time violations. On the other hand, the Xilinx FPGA's worst path delay is 2.93 ns less than the Altera FPGA. The path delay in the Altera FPGA implementation is higher than the Xilinx FPGA due to the longer routing path.

## 5. Conclusion

From the FPGA implementation results in Table 1, it shows the smart card with the memory ciphering system on the ARM-based Altera platform has achieved the smallest area due to the least number of the total hardware used. The platform has also attained the highest maximum frequency that is the best time

requirement. This can be a good reference as well for other smart card or SoC design implementations on the SoC-based type FPGA platform in achieving design optimization with high performance.

## 6. Acknowledgements

This research is funded by Ministry of Education Malaysia under grant FRGS/2/2014/TK03/UKM/02/1 and GUP-2015-021.

## 7. References

[1] Yaakob, Wira Firdaus Hj, Hafizul Hasni Manab and Siti Noorashikin Md Adzmi. Smart card chip design implementation on ARM processor-based FPGA. *3rd Global Conference on Consumer Electronics (GCCE), 2014 IEEE*. 2014, pp. 294-297.

[2] Guillou, L. C., M. Ugon, and J. J. Quisquater. Cryptographic authentication protocols for smart cards. *Computer Networks,* 2011, pp. 437-451.

[3] Jyrwa, Banraplang, and Roy Paily. An Area-Throughput Efficient FPGA Implementation of Block Cipher AES Algorithm. *International Conference on Advances in Computing, Control and Telecommunication Technologies,* 2009, pp. 328-332.

[4] Druyer, R., L. Torres, P. Benoit, P. V. Bonzom, et al. A Survey on Security Features in Modern FPGAs. *10th International Symposium on Reconfigurable and Communication-centric Systems-on-Chip (ReCoSoC),* 2015.

[5] Peng, Zhang, and Jia Jian Fang. Comparing and Implementation of Public Key Cryptography Algorithms on Smart Card. *Computer Application and System Modeling (ICCASM),* 2010, pp. 508-510.

[6] J Sampe and M Othman. Fast detection anti-collision algorithm for RFID system implemented on chip. *Journal of Applied Sciences,* 2008, 8 (7): 1315-1319.

[7] Borst, Johan, Bart Preneel, and Vincent Rijmen. Cryptography on smart cards. *Computer Networks,* 2001, 36(4): 423-435.

[8] J Sampe. Reliable and higher throughput anti-collision technique for RFID UHF tag. *World Applied Sciences Journal,* 2012, 20 (3): 445-449.

[9] J Sampe and M Othman. Fast detection anti-collision algorithm for RFID system implemented on-chip. *Journal of Applied Sciences,* 2008, 8 (7): 1315-1319.

[10] Messerges, Thomas S., Ezzat A. Dabbish, and Robert H.Sloan. Examining Smart-card Security Under the Threat of Power Analysis Attacks. *IEEE Transactions on Computers,* 2002, 51(5): 541-552.

[11] Xilinx. Vivado Design Suite User Guide. *VivadoUG1*, 2013, 901:1-120.

[12] Altera. Introduction to the Quartus II Software. *Quartus II UG,* 2010. Pp. 1-136.

[13] Altera. Stratix II Vs. Virtex-4 Density Comparison. *Quartus II UG*, 2005, pp. 1-19.