Design of Hummingbird Algorithm for Advanced Crypto Systems

[1]M.Rabbani, [2]R.Ramprakash
[1]M.tech student, [2]Assistant Professor
Dept. of ECE, Anurag Group of institutions, Jntuh, Hyderabad, India

***Abstract* - Hummingbird is a new ultra-light cryptographic algorithm targeted for resource constrained devices like Radio Frequency Identification (RFID) tags, smart cards and wireless sensor nodes. In this project we implement an encryption and decryption core on the low cost Xilinx FPGA series Spartan-3.This project present a technique to reduce number of clock cycles to encrypt and decrypt the message.This reduction in number of clock cycles allows faster encryption and decryption of the message. The result based on a rough theoretical analysis and logic synthesis showed its efficiency in encrypting and decrypting the message.**

***Index Terms* — RFID, Cryptography, Hummingbird, Encryption, Decryption.**

I. INTRODUCTION

RFID tags, smart cards, and wireless sensor nodes and various such smart devices, with their wide range of applications, have gained significant importance in home automation and health care. Many applications involve complicated processing of sensitive personal information and biological data. Moreover it is necessary to maintain confidentiality as much as its necessary to process it. Thus there is an ever increasing demand for integrating cryptographic functions into embedded applications and improvising them. But the important issue that is to be considered is that the smart devices have extremely constrained resources with respect to memory, power supply etc.,thus it is impractical to use classical cryptographic algorithms. Moreover, the tight cost constraints also bring forward impending requirements for designing new cryptographic primitives that can perform strong authentication and encryption, and provide other security functionality for ultra-low-power applications in the era of pervasive computing. This emerging research area is usually referred to as lightweight cryptography. Hummingbird is the recently proposed lightweight cryptographic algorithm. For detail description of the Hummingbird cryptographic algorithm refers[1].The design and implementation is described in Section 2, Section 3 gives the round-based architecture of 16-bit block cipher,

II. DESIGN AND IMPLEMENTATION

In this section an encryption only core and an encryption/decryption core have been implemented on the low-cost Xilinx FPGA series Spartan-3.

2.1 Speed Optimized Hummingbird Encryption **Core**

The top-level description of an speed optimized Hummingbird encryption core is as in[1]. In our work the traditional modulus operator is replaced by XOR operation.

The working of the speed optimized Hummingbird encryption core is: The initialization process begins when the chip enable signal CE changes from „0" to „1" and within four clock cycles the four rotors RSi (i = 1, 2, 3, 4) are first initialized by four 16-bit random NONCE through the interface RSi(15:0). From the fifth clock cycle, the core starts encryption RS1 XOR RS3 for four times and each iteration requires four clock cycles to finish encryptions by four 16-bit block ciphers as well as the internal state updating. The correct computation results to update four rotors depending on the value of a round counter are chosen by the multiplexer M5 and other multiplexers select appropriate inputs to feed the 16-bit block cipher. The full update of the rotor RS2 involves successive encryptions of two plaintext blocks to save chip area for the encryption-only core. More accurately, the rotor RS2 is updated by V12t and RS4t+1 when encrypting two successive plaintext blocks, respectively. After 20 clock cycles the initialization process completes and the READY signal changes from „0" to „1 then the first 16- bit plaintext block is read from an external register for encryption. The required cipher text is obtained from the encryption core after another four cycles and the valid output signal VO becomes high level. Consequently, after an initialization process of 20 clock cycles, the proposed speed optimized Hummingbird encryption core can encrypt one 16-bit plaintext block per 4 clock cycles.

2.2 Speed Optimized Hummingbird **Encryption/Decryption Core**

The top-level description of an speed optimized Hummingbird encryption/decryption core is as in[1].In our work the traditional modulus operator is replaced by XOR operation. The four operation modes supported by Hummingbird encryption/decryption core are: i) Encryption only; ii) Decryption only; iii) Encryption followed by decryption; and iv) Decryption followed by encryption. The same initialization procedure is shared by both encryption and decryption routines that first takes 4 clock cycles to load four random nonce into rotors through multiplexers M5 and M11 followed by 16 clock cycles for four iterations. The architecture of the encryption/decryption core is similar to that of the encryption only core apart from the following several aspects. Firstly, while encrypting two successive plaintext blocks in the encryption-only core the rotor

RS2 completes the update, But in the encryption/decryption core, every rotor is fully updated each time a plaintext block is encrypted or decrypted to support the four operation modes. Hence, in order to fully update the rotor RS2 after each encryption/ decryption, two multiplexers M10 and M11 are introduced. Secondly, we include an XOR, which facilitates the execution of the corresponding arithmetic as per the operation modes of the core.Thirdly, the correct values to the encryption and decryption routines of the 16-bit block cipher is fed using two multiplexers M7 and M8, respectively. Finally, the appropriate inputs are selected by all the other multiplexers based on the value of a round counter as well as the operation modes. This indicates that the working of the encryption/decryption core matches that of encryption- only core. Accordingly one 16-bit plaintext or cipher text block can be encrypted or decrypted by the speed optimized Hummingbird encryption/decryption core, after an initialization process of 20 clock cycles.

III. ROUND-BASED ARCHITECTURE OF 16- BIT BLOCK CIPHER

The round-based architecture is as in [10], is Used to further reduce the chip area and power consumption. This architecture repeatedly uses only one round function block as shown in Figure 1 and also consists of four regular rounds which shares the common hardware resources of one substitution and permutation layer and the final round is composed of another substitution layer and four XORs. Therefore, there are totally 5 XORs, 8 S-boxes, and one permutation layer for the datapath. Furthermore, three 16-bit multiplexers are introduced for different purposes.

1. A 4-to-1 multiplexer M1 is utilized to switch among the required round keys; ii) A 2-to-1 multiplexer M2 is employed to choose between the input and the computation result of each round; and iii) A 2-to-1 multiplexer M3 is used to export either the computation result of each round or the final ciphertext that is then stored into a 16-bit register. The whole encryption can be performed in four clock cycles for the round-based architecture.

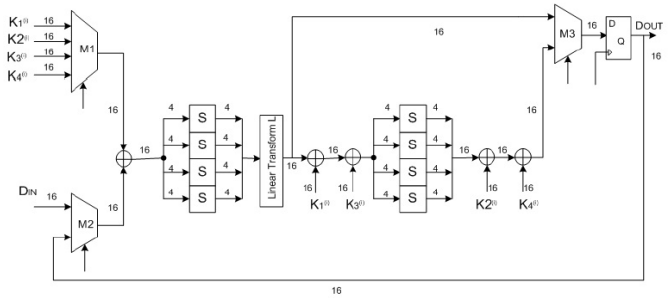

Fig 1. Round-based Architecture of 16-bit Block Cipher

The round-based architecture of the 16-bit block cipher is implemented on the Spartan-3 XC3S200 FPGA. The area requirement of the round-based architecture is tested by four S-boxes and two implementation options, respectively. Table 1summarizes our experimental results.

Table 1: Area Requirement for the Round-based Architecture of 16-bit Block Cipher on the Spartan-3XC3S200 FPGA in our work.

S-box	Implementation strategy	#LUT's	#FF's	Total Occupied slices
$S_1(x)$	LUT	187	20	98
	BFR	187	20	100
$S_2(x)$	LUT	187	20	99
	BFR	187	20	99
$S_3(x)$	LUT	186	20	99
	BFR	187	20	99
$S_4(x)$	LUT	186	20	99
	BFR	187	24	100

3.1 Area Optimized Hummingbird Encryption Core

The depiction in Figure 2 shows the top-level description of an area optimized Hummingbird encryption core[10]. Working of the area optimized Hummingbird encryption core is as follows: The initialization process starts when the chip is (i.e., CE = „1") and four rotors RSi (i = 1, 2, 3, enabled) 4are first initialized by four 16-bit random nonce through the interface RSi(15:0). The message RS1 RS3 is encrypted when the core executes four iterations. To complete encryptions, four 16-bit block ciphers are executed with the 64-bit key and also updating the internalstate.

IV. RESULTS AND DISCUSSION

A summary of our implementation results is presented in Table 2, where the area requirements (in slices) and the time period are given. All experimental results were extracted using ISE Design Suite from Xilinx on a XC3S200-5ft256 Spartanplatform with speed grade -5 -3 The 64-bit sub keys ki (i = 1, 2, 3, 4) are read from an external register based on the value of a round counterr through the interfaces KEY1 (15:0) to KEY4 (15:0) and under the control of the signal KEYSEL (1:0). In addition, the corresponding inputs are also chosen from multiplexers M1, M2 and M3 and temporary registers RH, RA and RE under the control of the round counter. While the updating of four rotors.

Comparison between AES Method and Hummingbird Method:

As compared to AES Method, hummingbird algorithm takes less Number of clock cycles to encrypt and decrypt the bits. So hummingbird algorithm is fast in encrypting and decrypting the bits.

Table 2

Name of the cryptographic algorithm	Key size	No of bits applied	Total no of clock cycles Required to encrypt and decrypt are
AES	128	128	1616
16 BIT HUMMINGBIRD	64	16	4
256 BIT HUMMINGBIRD	1024	256	16

V. CONCLUSION

This paper presents the implementations, simulation results of the ultra-lightweight cryptographic algorithm Hummingbird. The proposed Hummingbird encryption and decryption cores can encrypt and decrypt a 256-bit message block with 14 clock cycles. As compared to other lightweigh fpga implementation of blockcipher AES, Hummingbird can encrypt and decrypt in less number of clock cycles.

REFERENCES

[1] X. Fan, G. Gong, K. Lauffenburger, and Hicks "FPGA Implementations of the Hummingbird Cryptographic Algorithm", IEEE International Symposium on Hardware-Oriented Security and Trust (HOST), 2010.

[2] D. Engels, X. Fan, G. Gong, H. Hu, and E. M. Smith, "Hummingbird: Ultra-Lightweight Cryptography for Resource-Constrained Devices", to appear in the proceedings of The 14th International Conference on Financial Cryptography and Data Security - FC 2010.

[3] X. Guo, Z. Chen, and P. Schaumont, "Energy and Performa Evaluation Platform with AES and PRESENT Coprocessors Embedded Computer 115, 2008.

[4] F. Mace, F.-X. Standaert, and J.-J. Quisquater, "FPGA Implemenation(s) of a Scalable Encryption Algorithm", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 16, no. 2, pp. 212-216, 2008.

[5] G. Rouvroy, F.-X. Standaert, J.-J. Quisquater, and J.-D.Legat, "Compact and Efficient Encryption/Decryption Module For FPGA Implementation of the AES Rijndael Very Well Suited for Small Embedded Applications", International Conference on Information Technology: Coding and Computing - ITCC 2004, pp. 583-587, 2004. F.-X. Standaert, G. Piret, G. Rouvroy, and J.-J. Quisquater, "FPGA Implementations of the ICEBERG Block Cipher", Integration, the VLSI Journal, vol. 40, iss. 1, pp. 20-27, 2007.