International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 2 Issue 7, July - 2013

Efficient Implementation of Hummingbird Cryptographic Algorithm on a Reconfigurable Platform

Kutuboddin Jinabade
Dept. of Electronics & Communication
KLE DR.MSSCET Belgaum, Karnataka

Krupa Rasane
Dept. of Electronics & Communication
KLE DR.MSSCET Belgaum, Karnataka

Abstract

For the resource constrained devices such as Radio Frequency Identification (RFID) tags, smart cards and wireless sensor networks, novel ultra lightweight Hummingbird cryptographic algorithm is presented. In this paper an attempt has been made to improve the architectural area as compared to previous hardware implementation on low cost Field Programmable Gate Array (FPGA) devices. Hummingbird provides design security with small block size and it is resistant to the most common attacks. Our results are compared with FPGA implementations of the various cryptographic algorithms. The results show that the work presented in this report gives optimization with respect to area utilization and hence cost effective in comparison to the referred papers.

This paper is the reimplementation of the concept described in[1] with optimized programming in VHDL.

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

1. 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 refer[1].

The design and implementation is described in Section 2, Section 3 gives the round-based architecture of 16-bit block cipher, The results and discussion is presented in Section 4. Finally, the paper is concluded in Section 5.

2. 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.

3. 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:

i) 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.

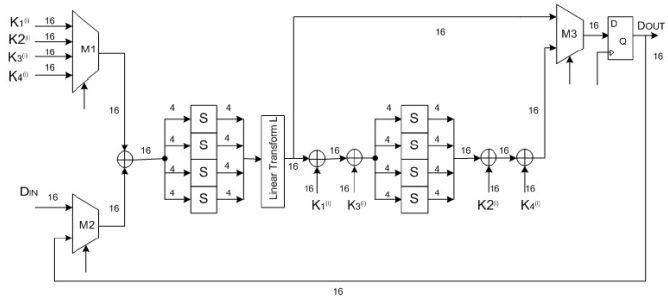

Fig 1. *Round-based Architecture of 16-bit Block Cipher*

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 1 summarizes our experimental results.

Table 1: Area Requirement for the Round-based Architecture of 16-bit Block Cipher on the Spartan-3 XC3S200 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 enabled (i.e., CE = '1') and four rotors RSi (i = 1, 2, 3, 4) are 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 internal state. The 64-bit sub keys ki (i = 1, 2, 3, 4) are read from an external register based on the value of a round counter 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 is taken care by the multiplexer M1 where the appropriate operands to form the correct input of the 16-bit block cipher are selected by M2 and M3. The system initialization completes and the READY signal becomes high level. The first 16-bit plaintext block is then read from an external register for encryption. , the corresponding cipher text is obtained from the encryption core and the valid output(Vo) signal changes from '0' to '1'.

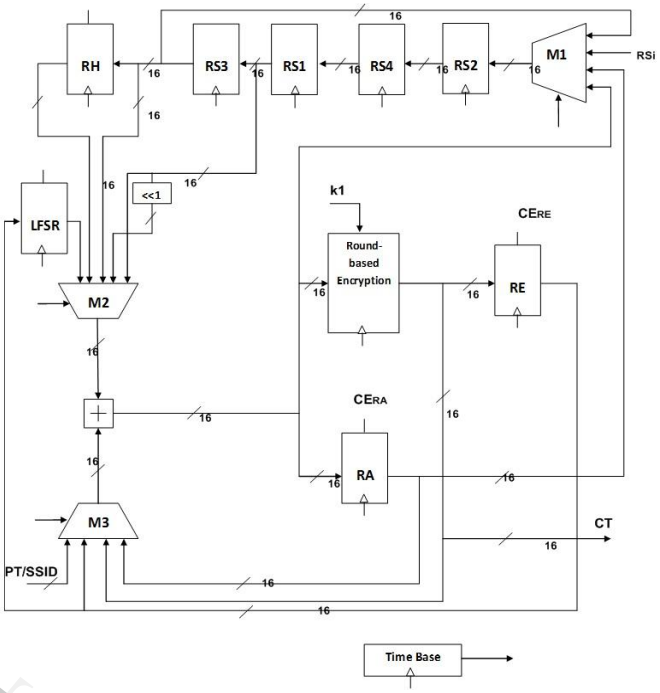

Fig 2: The Datapath of Area Optimized Hummingbird Encryption Core Using the Round-based Architecture of 16-bit Block Cipher

4. 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 Spartan-3 platform with speed grade -5.

Table 2: Implementation results of Hummingbird algorithm in our work

Methodology	Mode	S-box and it's Implementation	#LUTs	#FFs	Total Occupied Slices	Time Period (ns)	% decrease in delay of opt. cores w.r.t traditional core	%decrease in delay of speed w.r.t area
Traditional	ENC	$S_i(x)$, (i=1, 2, 3, 4) with LUT	4114	352	2116	27.055	-	-
	DEC		4930	352	2546	31.730	-	-
Speed Optimized	ENC	$S_3(x)$ with LUT	458	192	250	2.359	91.28%	69.07%
	ENC/DEC		992	320	540	2.359	-	-
Area Optimized	ENC	$S_1(x)$ with LUT	396	168	211	7.629	71.80%	-

For the Hummingbird encryption-only core, Table 2 shows that the time period of the speed optimized implementation is 69.07 % faster than that of the area optimized implementation at the cost of additional 39 slices on the similar target platform. The speed optimized core is 91.28 % faster than traditional core

and the area optimized core is 71.80% faster than the traditional core.

The performance comparison of our Speed optimized Hummingbird implementation with the existing FPGA implementations of [1], [8], [9] and [2] are summarized in Table 3.

Table 3: Performance comparison of FPGA Implementation of Cryptographic Algorithms

Cipher	Key Size	Block Size	FPGA Device	Total Occupied Slices	Max freq [MHz]	% Decrease in Slices	No.of times Increase in Freq.
OUR WORK (speed opt.)	**256**	**16**	**Spartan -3XC3S200-5**	**250**	**423.935**	-	-
Hummingbird[1]	256	16	Spartan -3XC3S200-5	273	40.1	8.42	10.57
PRESENT[8]	80	64	Spartan-3e XC3S500-5	271	-	7.5	-
XTEA[9]	128	64	Spartan-3 XC3S50-5	254	62.2	1.57	6.77
XTEA[9]	128	64	Virtex-5 XC5VLX85-3	9647	332.2	97.40	1.27
AES[2]	128	128	Spartan-3 XC3S2000-5	17425	196.1	98.56	2.16
AES[2]	128	128	Spartan 3	1800	150	86.11	2.82

The frequency increase in our work is 10.57 times more than the work[1], 6.77 times than [9] on Spartan-3 XC3S50-5, 1.27 times than [9] on Virtex-5 XC5VLX85-3, 2.16 times more than the work[2] on Spartan-3 XC3S2000-5 and 2.82 times more than the work [2] on Spartan 3.

The performance comparison of our Area optimized Hummingbird implementation with the existing FPGA implementations of [1], [8], [9] and [2] are summarized in Table 4.

Table 4: Performance Comparison of FPGA Simulation results of Cryptographic Algorithms

Cipher	Key Size	Block Size	FPGA Device	Total Occupied Slices	Max freq [MHz]	% Decrease in Slices	No. of times change in Freq
OUR WORK (area opt.)	**256**	**16**	**Spartan-3XC3S200-5**	**211**	**131.078**	-	-
Hummingbird[1]	256	16	Spartan -3XC3S200-5	273	40.1	22.71	3.26↑
PRESENT[8]	80	64	Spartan-3e XC3S500-5	271		22.14	-
XTEA[9]	128	64	Spartan-3 XC3S50-5	254	62.2	16.92	2.09↑
XTEA[9]	128	64	Virtex-5 XC5VLX85-3	9647	332.2	97.81	2.53↓
AES[2]	128	128	Spartan-3 XC3S2000-5	17425	196.1	98.78	1.49↓
AES[2]	128	128	Spartan 3	1800	150	88.27	1.14↓

From the analysis and comparison of results, it can be concluded that the work presented here has better area optimization than the results previously obtained and also has a better speed with tremendous increase in frequency. Thus the project presents a performance enhanced Hummingbird cryptographic algorithm with both speed and area optimization.

Discussion : From the above results it can be concluded that the speed and area optimized encryption cores have better performance in terms of time period and number cryptographic algorithm with both speed and area optimization.

5. Conclusion

The Hummingbird encryption algorithm has been implemented with area optimization. The hardware implementation has been done on FPGA. The synthesis and simulation results have been compared and improvement has been observed. Compared to other lightweight FPGA implementations of block ciphers PRESENT, XTEA and AES, Hummingbird can be implemented with smaller area requirement. Consequently, Hummingbird can be considered as an ideal cryptographic primitive for resource-constrained environments. As the future research, we intend to conduct Encryption/Decryption core of Hummingbird cipher as well as propose low power ASIC implementations for low-cost RFID tags.

ACKNOWLEDGMENT

We are grateful to KLE DR M S. Sheshagiri College of engineering and Department of Electronics and communication for providing the necessary facilities for carrying out the project. We are grateful to all those who are directly or indirectly part of this paper.

REFERENCES

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

[2] P. Bulens, F.-X. Standaert, J.-J. Quisquater, and P. Pellegrin, "Implementation of the AES-128 on Virtex-5 FPGAs", Progress in Cryptology - AFRICACRYPT 2008, LNCS 5023, pp. 16-26, 2008

[3] P. Chodowiec and K. Gaj, "Very Compact FPGA Implementation of the AES Algorithm", The 5th International Workshop on Cryptographic Hardware and Embedded Systems - CHES 2003, LNCS 2779, pp. 319-333, 2003.

[4] 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, 2010

[5] T. Good and M. Benaissa, "AES on FPGA from the Fastest to the Smallest", The 7th International Workshop on Cryptographic Hardware and Embedded Systems - CHES 2005, LNCS 3659, pp. 427-440, 2005.

[6] X. Guo, Z. Chen, and P. Schaumont, "Energy and Performance Evaluation of an FPGA-Based SoC Platform with AES and PRESENT Coprocessors", Embedded Computer Systems: Architectures, Modeling, and Simulation - SAMOS'2008, LNCS 5114, pp. 106-115, 2008.

[7] 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.

[8] 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.

[9] Daniel Engels, Xinxin Fan, Guang Gong, Honggang Hu, and Eric M. Smith,"Ultra-Lightweight Cryptography for Low-Cost RFID Tags: Hummingbird Algorithm and Protocol",Department of Electrical and Computer Engineering University of Waterloo Waterloo, Ontario, N2L 3G1, CANADA.

[10] Xinxin Fan-" Efficient Cryptographic Algorithms and Protocols for Mobile Ad Hoc Networks"-A Ph.d A thesispresented to the University of Waterloo, Canada.