VLSI Implementation of Lightweight Cryptography Algorithm

Hinpreet kaur and Sakthivel R

School of Electronics Engineering, VIT University, Vellore, Tamil Nadu, India.

Abstract
Lightweight algorithms for cryptography are popular for resource stringent devices. Now days, radio frequency identification techniques are gaining popularity because of their small size and low cost applications. In this paper, Hummingbird algorithm is used to provide security in such devices like RFID tags, smart cards, etc. It is a Hybrid algorithm which provides security against most of the common attacks encountered such as linear attacks and differential attack. The hybrid combination of block and stream cipher makes this algorithm more secure using lesser number of clock cycles. The algorithm is implemented on both FPGA and ASIC platform. The main aim is to reduce the number devices utilized and using lesser area on the chip. The algorithm is implemented in Xilinx Vertex-5 family. This design consumes the total standard cell area of 0.255 mm^2. The design is placed and route using Cadence SoC Encounter at TSMC90nm technology.
Keywords Hummingbird algorithm; RFID tag; Lightweight cryptography; ASIC; FPGA.

1 Introduction

The motivation behind this work is the increase in demand of resource stringent devices such as smart cards and devices using the RFID (Radio Frequency Identification) technology. RFID tags are now being used in libraries to keep the records of the book, in hospitals for medical equipment detection, etc. The small size and low cost of the tags makes it popular.

RFID devices consist of mainly three elements, a tag, a reader and a database system. There exist a communication between the tag and the reader. This communication is affected by outside environment if a third person or attacker tries to leak the information. So it becomes essential to incorporate an algorithm in order to make the communication secure. This work proposes an area efficient architecture on both FPGA and ASIC platform.

2 Background

Hummingbird is proved to be very secure algorithm because it is a hybrid algorithm of stream cipher and block cipher. Despite of presence of many cryptographic algorithms such as DES (Data Encryption Standards) and AES (Advanced Encryption standard)[**?**], we need algorithm which is lightweight on both hardware

and software side, which is not fulfilled by the mentioned algorithms. Since RFID devices are small in size, the cryptographic unit thus used should be hardware friendly and provides the same level of security as in other non-resource constrained devices.

Several work has been carried out till related to this algorithm. Many other lightweight algorithms such as PRESENT, KLEIN, HIGHT, LED, ICEBERG are implemented in FPGA platform but Hummingbird is proved to provide the highest degree of security and is resistant to many attacks such as birthday attacks, algebraic attacks, structural attack and cube attack. The work related to Hummingbird are tabulated in table1.The first Hummingbird algorithm was implemented in 4-bit microcontroller with low power consumption[**?**]. Many implementations are being carried out on FPGA platform are shown in the table1.

Table 1 Different Implementations of Hummingbird algorithm

Paper	Year of Publication	Implementation platform	Key Feature
Ref[**?**]	2009	Microcontroller	4-bit microcontroller , low power consumption
Ref[**?**]	2010	FPGA	Larger throughput ; smaller area
Ref[**?**]	2011	FPGA	Co-processor approach
Ref[**?**]	2011	FPGA	Throughput oriented and area oriented
Ref[**?**]	2011	FPGA	Gen2 protocols for MAC
Ref[**?**]	2014	FPGA	Low power and high speed

3 Algorithm Steps

The Hummingbird algorithm consists of a 256-bit secret key shared between the tag and the reader. There are four internal registers of 16-bit and a 16-bit Galois field LFSR. The registers are first initialized in the initialization process with some random initial vectors which are then being updated by the LFSR in the encryption process. The input to encryption is the plaintext and the output is the 16-bit cipher text. Fig.1 shows the working of the algorithm. The 16-bit plain text is given as the input along with the 256-bit key, which is segmented into four 64-bit subkeys. There are three blocks which perform initialization, encryption and decryption. In the initialization process, the registers are updated for encryption. After each set of plaintext and cipher text, the internal status registers are updated by the 16-bit LFSR. The initialization block and the encryption block consist of the block substitution box or S-Box and the linear permutation. The predefined s-box and the inverse s-box is shown in Table 2.

The message is encrypted by the tag using the encryption process and the resultant cipher text is sent to the reader. The reader decrypts the message using

the same key (as the key is being shared between tag and the reader).

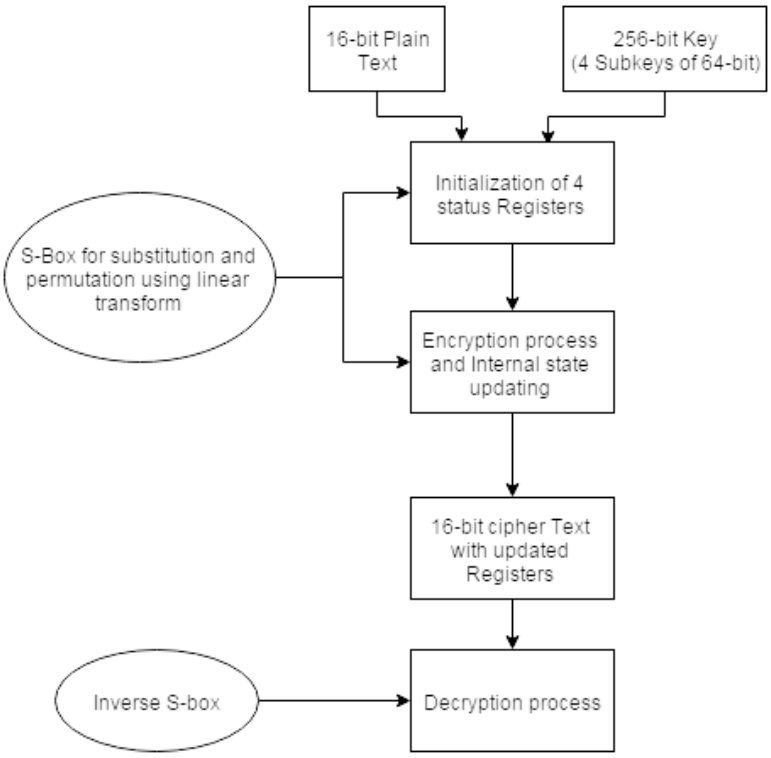

Fig. 1 Algorithm flow of Hummingbird cryptography

Table 2 S-box and the inverse S-box used in the algorithm

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
S(x)	2	E	F	5	C	1	9	A	B	4	6	8	0	7	3	D
S(x)	C	5	0	E	9	3	A	D	B	6	7	8	4	F	1	2

4 Proposed Architecture

The proposed architecture is shown in Fig.2. This architecture works on the encryption only and decryption only processor. The initialization block consists of four status registers and a ready pin. A 5-bit counter is used to count the number of clock cycles. When the *data_rdy* pin goes high, the counter starts counting and the status registers are initialized with some random values. The registers are then updated by undergoing encryption round of block cipher in next 16 clock cycles. For the algorithm refer[**?**].

The Encryption block uses the initialized status registers to encrypt the plain

text into cipher text. Encryption undergoes the modulo 2^{16} addition of register RS1 and plain text and undergoes through the block encryption using the secret key. This is repeated four times and the resulting cipher text is given to the decryption module making the *enc_complete* signal high. The decryption is just reverse of encryption. The input is cipher text and output is plain text. Modulo 2^{16} subtraction is used in decryption. Thus when both *dec_complete* and *enc_complete* signals are high the output is given in one clock cycle. The registers are updated for the next process.

Table 3 Device utilization summary sheet

Logic Utilization	Used		Available	Utilization	
	Ref.[?]	This work		Ref.[?]	This work
Number of slice register	4242	74	12480	33%	1%
Number of slice LUTs	3504	2309	12480	28%	18%
Number of Bonded IOBs	278	117	172	161%	68%
Number of fully used LUT-FF pair	1907	70	Ref.[?]-5839 This work-2313	32%	3%
Number of BUFG/ BUFCTRLS	8	1	32	25%	3%

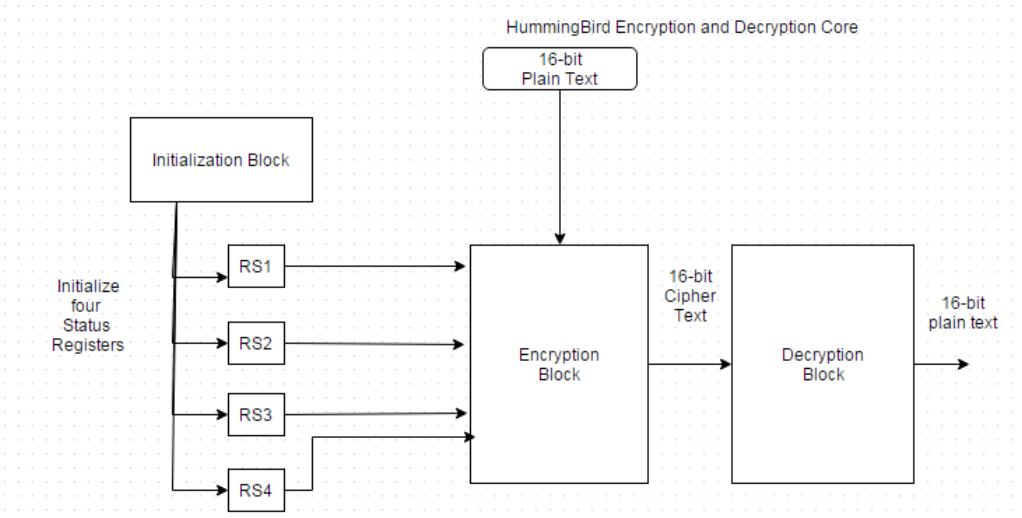

Fig. 2 Proposed architecture of Hummingbird encryption and decryption

5 Simulation Results

The encryption block along with the initialization module of the Hummingbird algorithm are designed and simulated using the S-box using the LUT based ap-

proach. LUT based s-box uses less area and power. The simulation results are shown in Fig.3 The 16-bit plain text is encrypted to 16-bit cipher text. When the reset is high, there will be no initialization process. After the reset signal changes to high, the initialization starts and the 16-bit plain text is converted to its cipher text using a 256-bit secret key. The first cipher text is obtained after 8-clock cycles and then at the subsequent clock cycles we will get the other cipher texts. The simulation carried out using Modelsim 6.5b. In the simulation results shown in Fig.3, a plaintext is encrypted using a 64-bit key and the respective cipher text and the decrypted data is obtained after 20-clock cycles. The result of the console window is given below:

Hummingbird INPUT data==0011000000111001

Hummingbird

key==11100110111001110010000011011011101110000011011010110001110110 01

Hummingbird encrypt data==0101100100001110

Hummingbird decrypt data==0011000000111001

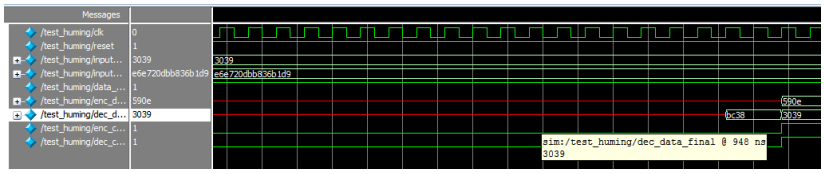

Fig. 3 Simulation result of encryption process

The entire algorithm is designed and implemented in Xilinx 14.7 ISE suite with Vetex-5 XC5VLX20T in package FF-323 and speed grade -2. The results obtained were compared with the mentioned results in [**?**]. The device utilization sheet is shown in Table.3. The total number of slices occupied in our design is 917, which is very less as mentioned in [**?**].

The design works at a frequency of 14MHz and the power consumption is 322.80 mW at 2.5V. The device utilization summary sheet mentioned in Table.3 shows that our design utilizes less space and is hardware friendly. The lightweight algorithm such as Hummingbird can be used in resource stringent devices and are now a days used in password identification, library management system, hospitals embedded in the RFID tag.

The ASIC implementation of the Hummingbird encryption and decryption core is done using Cadence SoC Encounter using TSMC 90nm technology. The final chip layout is shown in Fig.4.

The design is successfully placed and route using Cadence SoC Encounter tool using TSMC 90nm technology with no setup and hold violations. The Cadence RC compiler results in the area, power, timing and the gate counts used in the design. The die area, power, area in gate counts and maximum frequency of

operation are tabulated in Table 4.

The design has worst case skew of 19psec and the best case skew of 16.1psec with no timing violations. After the place and route the design was verified for the geometry and there were no violations. This proposed design is thus area efficient at both FPGA and ASIC platform.

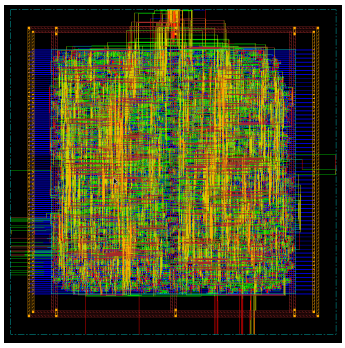

Fig. 4 Final chip layout

Table 4 Specifications

Area	.2556mm2
Power	2.234mW
Gate counts	11363
Max. Frequency	2.129 GHz

6 Conclusion and Future Scope

The Hummingbird encryption and decryption module is implemented in both FPGA and ASIC platform. The design is coded using Verilog HDL and implemented in Xillinx 14.7 ISE suite in Vertex-5 family. The results show reduced device utilization in slices. The ASIC implementation is done using Cadence SoC Encounter using TSMC90nm technology library. The design is successfully placed and route with no timing violations and the area power of the final chip is noted. Thus this design is suitable for resource stringent devices and hardware friendly as compared to AES and other cryptographic algorithms.

The algorithm used here S-Boxes, which can be designed using Boolean expression instead of LUT based approach. So the area and power can be reduced by using BDD reduction unit. The variable reordering reduces the Boolean functions according to the variables being selected. Learning CAD tools to perform BDD reduction module is further scope of study.

References

[1] P.Chodowiec and K.Gaj. (2003), "Very compact FPGA implementation of AES algorithm", *in Cryptographic Hardware and Embedded Systems CCHES 2003*, C. Walter, C. Koy and C. Paar(Ed.), Springer Berlin Heidelberg, Vol. 2779, pp. 319-333.

[2] F. Xinxin, G. Guang, K. Lauffenburger and T. Hicks. (2010), "FPGA implementations of the Hummingbird cryptographic algorithm", *in Hardware-*

Oriented Security and Trust (HOST), 2010 IEEE International Symposium on, pp. 48-51.

[3] T. San and N. At.(2011). "Compact hardware architecture for Hummingbird cryptographic algorithm", *in International Conference on Field Programmable Logic and Applications (FPL)*, pp. 376-381.

[4] M. Biao, R. C. C. Cheung and H. Yan. (2011). "FPGA-based high throughput and area-efficient architectures of the Hummingbird cryptography", *in IECON 2011 - 37th Annual Conference on IEEE industrial Electronics Society*, pp. 3998-4002.

[5] X. Mengqin, S. Xiang, W. Junyu and J. Crop.(2011), "Design of a UHF RFID tag baseband with the Hummingbird cryptographic engine", *In 2011 IEEE 9th International Conference on ASIC (ASICON)* , pp. 800-803.

[6] X. Mengqin, S. Xiang, W. Junyu and J. Crop.(2011), "Design of a UHF RFID tag baseband with the Hummingbird cryptographic engine", *In 2011 IEEE 9th International Conference on ASIC (ASICON)*, pp. 800-803.

[7] Nikita Arora and Yogita Gigras. (2014), "FPGA implementation of low power and high speed Hummingbird cryptographic algorithm", *International Journal of Computer Applications* , Vol. 92, No. 16, pp. 0975-8887.

[8] Xinxin Fan. (2010), *Efficient cryptographic algorithms and protocols for mobile ad hoc networks*, Canada: Ontario.

Corresponding author
Hinpreet kaur can be contacted at: hinpreetkaur@gmail.com