Journal of The Institute of Electronics and Information Engineers Vol.53, NO.10, October 2016 http://dx.doi.org/10.5573/ieie.2016.53.10.103
ISSN 2287-5026(Print) / ISSN 2288-159X(Online)

논문 2016-53-10-13

새로운 초경량 블록 암호의 하드웨어 설계 및 구현

(The Hardware Design and Implementation of a New Ultra Lightweight Block Cipher)

Gookyi Dennis A. N.*, 박 승 용*, 류 광 기**

(Gookyi Dennis A. N., Seungyong Park, and Kwangki Ryoo©)

요 약

미래의 것으로 여겨지던 pervasive 컴퓨팅이 현재 널리 이용되고 있다. Pervasive 컴퓨팅의 단점으로 여겨지는 데이터의 유출문제는 데이터의 확실한 보호가 이루어진다면 크게 부각되지 않겠지만 해커들의 홈 네트워크를 통한 정보 수집 등과 같은 문제들이 발생하고 있다. Pervasive 디바이스는 일반적으로 소비 전력, 공간 및 비용 측면에서 제약을 가지고 있고 완벽한 암호화 환경의 구현은 현실적으로 불가하다. 따라서 연구의 초점은 가능한 적은 메모리를 필요로 하는 암호화 경량화에 집중하고 있다. 본 논문은 새로운 경량 블록 암호의 설계 및 구현에 초점을 두고 치환-순열(S-P) 네트워크와 파이스텔 구조의 장단점을 연구하여, 두 가지 네트워크의 이용시 가장 적합한 방향을 제시한다. 알고리즘은 S-박스 및 P-박스와 함께 파이스텔 구조를 사용한다. 본 논문에서는 백도어 아이디어가 알고리즘에 사용되는 것을 방지하기 위해 S-박스를 사용하였다. P-박스와 달리 S-박스는 키 디펜던트 원 스테이지 오메가 네트워크를 사용하여 보안 단계를 향상하였다. 본 논문에서 제안하는 하드웨어는 Verilog HDL로 설계되었으며 Virtex6 XC4VLX80 FPGA iNEXT-V6 테스트 보드를 사용하여 검증하였다. Simple core design은 337 MHz의 최대 클록 주파수에서 196 슬라이스를 합성한다.

Abstract

With the growing trend of pervasive computing, (the idea that technology is moving beyond personal computers to everyday devices) there is a growing demand for lightweight ciphers to safeguard data in a network that is always available. For all block cipher applications, the AES is the preferred choice. However, devices used in pervasive computing have extremely constraint environment and as such the AES will not be suitable. In this paper we design and implement a new lightweight compact block cipher that takes advantage of both S-P network and the Feistel structure. The cipher uses the S-box of PRESENT algorithm and a key dependent one stage omega permutation network is used as the cipher's P-box. The cipher is implemented on iNEXT-V6 board equipped with virtex-6 FPGA. The design synthesized to 196 slices at 337 MHz maximum clock frequency.

Keywords : Lightweight ciphers, AES, S-box, P-box, PRESENT algorithm

Ⅰ. Introduction

Today, there is rise in the mass-deployment of pervasive computing. In pervasive computing, devices with computing power are attached to house hold items. These devices record, store and update sensitive information about the medium they are attached to. It is therefore sad to note that though the information available in pervasive devices are sensitive, not much work have been done to protect such information from adversaries.

Due to cost-constraints in the mass-deployment of pervasive computing, it is inherent to use devices that are constraint in terms of computing capabilities, memory capacitance and power supply. One technology that is widely used in pervasive computing is the

* 학생회원, ** 정회원, 한밭대학교 정보통신공학과
(Department of Information and Communication Engineering, Hanbat National University)
© Corresponding Author (E-mail : kkryoo@gmail.com)
※ 본 연구는 미래창조과학부 및 정보통신기술진흥센터의 해외ICT전문인력활용촉진사업(IITP-2015-R0134-16-1019)과 해외인재스카우팅사업(IITP-2016-R2418-16-0007)의 연구결과로 수행되었음.
Received ; June 28, 2016 Revised ; October 5, 2016
Accepted ; October 5, 2016

RFID technology. RFID tags have a user memory capacity of about 512 bits and also the response time should be less than a 100us according to the ISO/IEC 18000 standard[1]. With these constraints, standard cryptographic algorithms like AES[2] cannot be implemented on such devices. Therefore the research focus has shifted to lightweight cryptography. According to the ISO/IEC standard on lightweight cryptography, the number of slices of LUTs should be 100-300[3].

An entirely new design that has been accepted as an ISO/IEC standard is the PRESENT[4] algorithm. The PRESENT has one of the smallest S-boxes available. The downside to the PRESENT algorithm is that it requires 31 rounds to encrypt a block of data which translates to 310 us per block at 100 KHz. This falls short of the response time for RFID tags. Another ISO/IEC standard algorithm is CLEFIA [5] developed by SONY. This uses two S-boxes and two P-boxes and therefore requires much more memory than the PRESENT algorithm. Table 1 summaries some encryption algorithms in existence.

The aim of this paper is to describe the hardware design and implementation of a new lightweight block cipher that meets the requirements of both ISO/IEC 18000 standard of less than 100us response time for RFID tags and ISO/IEC standard of 100-300 LUTs for lightweight ciphers. Our cipher uses the Feistel structure together with an S-box and P-box. To prevent the idea of a backdoor into the algorithm, the PRESENT S-box. Our P-box uses a key dependent one stage omega permutation network. This entirely new permutation box was designed to not only meet the strict avalanche criterion[6] but to also make cryptanalysis of the cipher a lot more difficult.

The cipher takes a 64 bit user input data and 128 bit user input key. The 64 bit data passes through 8 rounds of encryption with each round using a new round key generated by the key generation algorithm.

표 1. 암호화 알고리즘
Table1. Encryption Algorithms

Algorithm	S-box memory requirement (bits)	Cycles per block	Time for 1 encryption@100 KHz (us)
AES	2048	12	120
PRESENT	64	31	310
TEA	-	64	640
CLEFIA	4096	18	180
DESL	256	16	160
HIGHT	-	32	320

Ⅱ. New Lightweight Block Cipher

1. Encryption/Decryption Algorithm Description

The new lightweight block cipher uses the Feistel network structure and consists of only 8 rounds. The block length is 64 bits and it uses a 128 bits key length. Each round consists of two stages. The first stage passes data through four operations: AddRoundKey, S-Box, P-Box and a second AddRoundKey. The second stage consists of the same set of operations with only a change in the keys used. The beauty about the Feistel structure is that the encryption routine and the decryption routine of the algorithm are virtually the same. Fig. 1 shows a top level algorithm description of the encryption routine and Fig. 2 shows a top level algorithm description routine of the new lightweight block cipher.

2. AddRoundKey Layer

The new lightweight block cipher uses a 128 bit key size which is enough to discourage a brute force attack. The 128 bit key K for each round is divide into four keys K0, K1, K2 and K3. Each of the four keys consists of 32 bits. K is allocated to the four key locations as follow: K0=K[31:0], K1=K[63:32], K2=K[95:64] and K3=K[127:96]. The AddRoundKey operation which is a simple XOR is applied four times in each round of the algorithm.

For the encryption routine, Stage 1 in each round uses K0 and K1 while stage 2 in each round uses K2 and K3 and for the decryption routine, Stage 1 in each round uses K2 and K3 while stage 2 in each round uses K0 and K1 Given the 32 bit key K=K31...K0 and the current state S=S31...S0, the AddRoundKey operation is shown in Fig. 3.

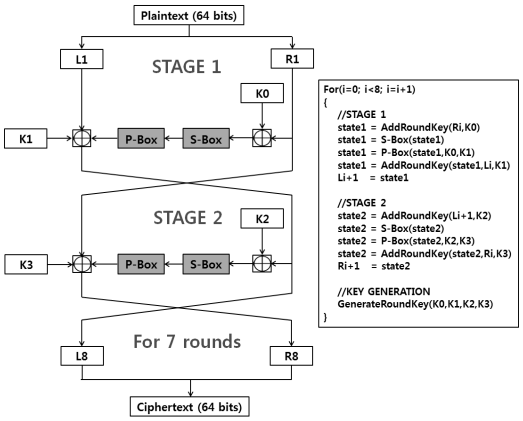

그림 1. 기존 암호화 과정
Fig. 1. Encryption Routine Description.

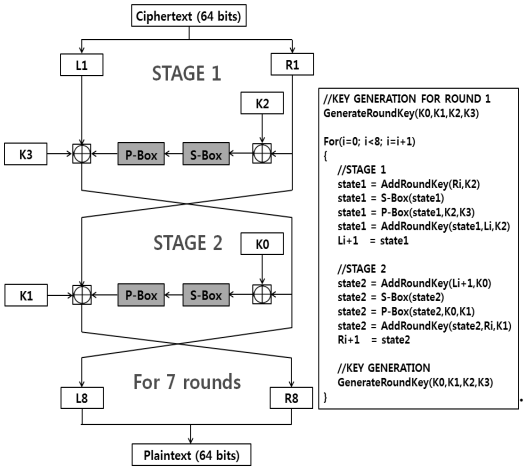

그림 2. 기존 복호화 과정
Fig. 2. Decryption Routine Description.

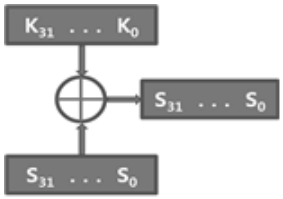

그림 3. AddRoundKey 연산 방법
Fig. 3. AddRoundKey Operation.

3. S-Box Layer

In symmetric key algorithms, the substitution-box (S-Box) is the basic component which performs substitution of bits. The S-Box is used to mask the relationship between the key and the ciphertext[7] (Shannon's property of confusion). Generally, the S-Box takes some number of input bits and maps it to some number of output bits with output bits value representing the transformed value of the input. The eight S-Boxes used in Data Encryption Standard algorithm were studied intensely because experts were sure that there was a backdoor to the algorithm which later proved to be false. To clear all doubts about a backdoor into the our cipher, the decision was made to employ the S-Box of the PRESENT algorithm. The PRESENT S-Box is resistant to both linear and differential cryptanalysis.

Table 2 shows the PRESENT S-Box used by the new lightweight block cipher. The S-Box is a 4 bit to 4 bit S-Box S: $F_2^4 \rightarrow F_2^4$.

For the S-Box layer, the current state $S_{31}...S_0$ is considered as 4 bit words $W7...W0$ where $Wi=S_{4xi+3}||S_{4xi+2}||S_{4xi+1}||S_{4xi}$ for $0 \leq i \leq 7$. The output nibble $S[Wi]$ provides the updated state. The encryption algorithm and the decryption algorithm use the same S-Box.

표 2. Substitution 박스 (S-박스)
Table2. Substitution Box. (S-Box)

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

4. P-Box Layer

In cryptography, a permutation box (P-Box) is used to permute bits across S-Box inputs. If the P-Box is implemented carefully, the outputs of the S-Boxes are distributed to as many S-Box inputs as possible. Though the S-Box layer is the most critical part of S-P network block cipher, the P-Box is essential in meeting the strict avalanche criterion of the cipher. The strict avalanche criterion (SAC) is satisfied whenever a single input bit is inverted, each output bit changes with a probability of 50%.

Many block ciphers employ the use of fixed P-Boxes. With the advent of differential and linear cryptanalysis, a fixed P-Box is no longer secure. A completely new kind of P-Box is proposed in this paper. The P-Box proposed is the Key Dependent One Stage Omega Permutation Network.

As shown in Fig. 4, the P-Box layer consist of 32 2-to-1 multiplexers. A 16 bit variable KEY_BITS serve as the select signals to the multiplexers. Two

multiplexers are controlled by one select signal (from KEY_BITS).

For the encryption routine the lower half of the 128 bit round key (*key[63:0]*) is used to calculate the *KEY_BITS* as follows: *KEY_BITS[15:0] = key[63:48] ^ key[47:32] ^ key[31:16] ^ key[15:0]*, where the operator (^) is the exclusive-or operator.

The algorithm for generating the outputs of the P-Box is shown in Fig. 4. In the algorithm, in_data[31:0] indicates the outputs of the S-Box while pbox[31:0] indicates the output of the P-Box. Here again, the encryption algorithm and the decryption algorithm use the same P-Box.

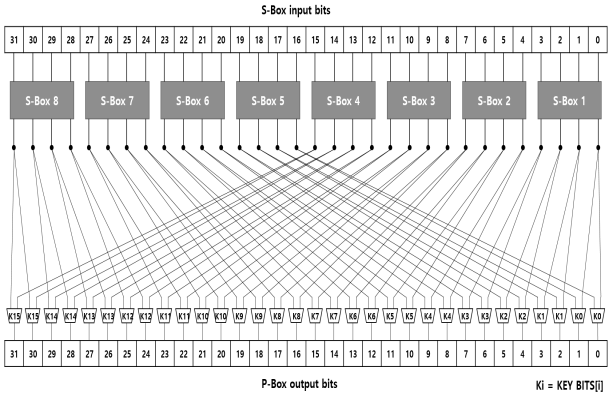

그림 4. Permutation 박스 (P-박스)
Fig. 4. Permutation Box. (P-Box)

4. Key Schedule Algorithm

In implementing an algorithm for pervasive computing, performance is rated above security. In the design of the new lightweight block cipher some of the considerations that were made before designing the key schedule algorithm include: The key schedule algorithm should be very simple, the key schedule algorithm should be fast, the key schedule algorithm should use minimal hardware resources.

For the encryption key schedule, the 128 bit user supplied key is stored in a register. The key schedule algorithm for the encryption routine as shown in Fig. 5 while the key schedule algorithm for the decryption routine is shown in Fig. 6.

Since the algorithm involves only a circular shift operations, it requires very few hardware resources to implement. Also, a completely unique key is generated for each round of encryption and decryption. This discourages the use of related key attacks to cryptanalyze the key schedule algorithm.

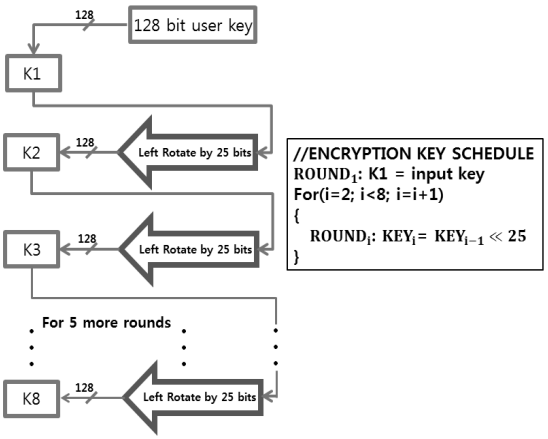

그림 5. 암호화 키 스케줄 알고리즘
Fig. 5. Encryption Key Schedule Algorithm.

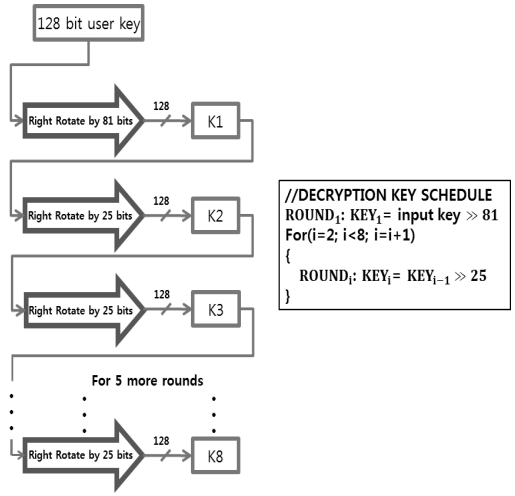

그림 6. 복호화 키 스케줄 알고리즘
Fig. 6. Decryption Key Schedule Algorithm.

Ⅲ. Experiment

To evaluate the performance of the proposed design, we synthesized the design using Xilinx ISE 14.3 on an iNEXT-V6 board equipped with virtex 6 FPGA. Mentor Graphics Modelsim SE-64 10.1c was used for the purpose of simulation. Fig. 7 shows the simulation results of an encryption/decryption core of the propose cipher. From the figure, it can be seen

that it takes only 8 clock cycles for each encryption and decryption routines. The synthesis results compared to other algorithms is shown in Table 3.

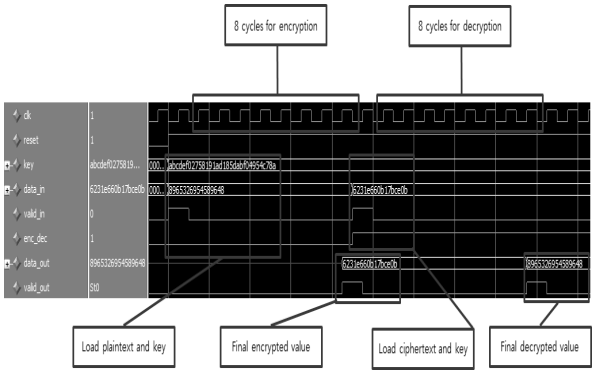

그림 7. 암호화/복호화 시뮬레이션
Fig. 7. Encryption/Decryption Core Simulation.

표 3. 제안하는 알고리즘 하드웨어 결과 비교
Table3. Decryption Key Schedule Algorithm.

Design	Max delay ns	Clock cycles per block	Block size bits	Key size bits	Area slices	Throughput Mbps	Mbps per area Mbps/ slice
PRESENT[8]	3.94	32	64	128	202	508	2.5
AES[9]	20.00	46	128	128	222	139	0.62
CLEFIA[10]	5.4	36	128	128	270	658	2.40
Camellia[9]	7.95	875	128	128	318	18.41	0.06
Tiny XTEA[9]	15.97	112	64	128	254	35.78	0.14
Proposed	2.97	8	64	128	196	2693	13.76

1. Image Encryption/Decryption Application

In this experiment the proposed design is used to perform image encryption and decryption in other to evaluate the real time processing performance of the design. The experiment is carried out on the iNEXT-V6 test board. The test image is an image of the actress Angelina Jolie in 480x272 JPEG format. In the experiment, the image is first converted into a coe file and stored in on-chip block ROM. Data from the ROM is sent to the encryption module to be encrypted. The encrypted data is stored in a RAM and the data is sent to the decryption module to be decrypted. The decrypted data is also stored in a RAM. The original image, encrypted image and decrypted image take turns to be displayed on the TFT LCD of the iNEXT-V6 test board. The image displayed on the TFT LCD is shown in Fig. 8.

To investigate the time taken for both encryption and decryption of the image, the clock period of the simulation was set at 10ns and the time for the entire encryption and decryption was taken. The result of this experiment is shown in Table 4.

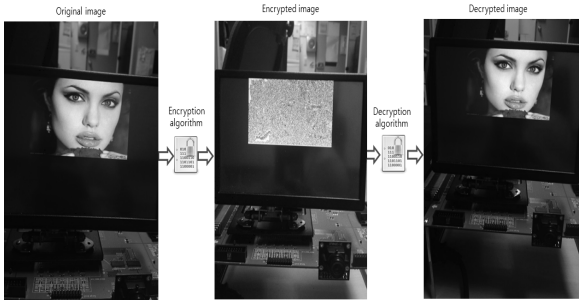

그림 8. 제안하는 암호화/복호화 이미지 검증
Fig. 8. Encryption/Decryption Images.

표 4. 암호화/복호화 어플리케이션 비교
Table4. Encryption/Decryption Application Comparison.

Algorithm	Image size (Pixels)	Clock period (ns)	Cycle/block	Required time for ENC/DEC(s)
AES[9]	480x272	10	46	0.120
PRESENT[8]	480x272	10	32	0.080
TEA[9]	480x272	10	64	0.160
Proposed	480x272	10	8	0.021

2. Investigating the Avalanche Effect

In the following, we give ten test vectors for the new lightweight block cipher. Here, the cipher key “ABCDEF02758191AD185DABF04954C78A” remain constant but the plaintext change by one bit. This is to test the avalanche nature of the new cipher. The hamming distance (bit change in two ciphertexts in which their plaintext differ by one) is calculated by taking the exclusive-or of the two ciphertexts. All data are expressed in hexadecimal notation.

표 5. 평문 변화에 따른 암호화문 비교
Table5. Investigating the Avalanche Effect.

Plaintext	Ciphertext	Hamming Distance
0000 0000 0000 0000	D0EBBFB002FC211E	28
0000 0000 0000 0001	A8013D5725FC8496	
0000 0000 0000 0002	1B9A72BACD398D34	29
0000 0000 0000 0003	D25667B09B4DB802	
0000 0000 0000 0004	936F3EDA60197859	35
0000 0000 0000 0005	8959DC899621A7CF	
0000 0000 0000 0006	EB5B8A40466BAEC6	30
0000 0000 0000 0007	A88459CA673E9FA2	
0000 0000 0000 0008	8906477C8E40B4C7	24
0000 0000 0000 0009	45AF2659DDC09CF5	

Ⅳ. Conclusion

In this paper, we design and Implement a new

lightweight block cipher. The FPGA implementation leads to compact results requiring 196 slices while providing a maximum frequency of 337 MHz. Though the cipher was designed to use very few hardware resources (targeting constraint devices like the RFID tag), it is equally preferable for high-speed and high-throughput application. The design of the new lightweight cipher is so simple yet incorporates many techniques used in cryptography. It is therefore encouraged to be used as tutorials for students studying basic cryptography. Cryptanalysis of the cipher is left for future work.

REFERENCES

[1] M. Feldhofer, S. Dominikus and J. Wolkerstorfer, "Strong Authentication for RFID Systems Using AES Algorithm," Cryptographic Hardware and Embedded Systems-CHESS, LNCS 3156, Springer-Verlag, pp. 357-370, 2004.

[2] National Institute of Standards and Technology (NIST), Advanced Encryption Standard (AES), Federal Information Processing Standards Publication 197, [Available Online]: http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf, 2001.

[3] B. Gauray, R. Nishchal and P. Narayan, "Implementation of a New Lightweight Encryption Design for Embedded Security," IEEE Transaction on Information Forensics and Security, 2015.

[4] A. Bogdanov et al., "PRESENT-An Ultra-Lightweight Block Cipher," Cryptographic Hardware and Embedded Systems-CHESS, Springer-Verlag, pp. 450-466, 2007.

[5] SONY Corporation, "The 128 bit Block Cipher CLEFIA: Algorithm Specification," SONY Corporation, Tokyo, Japan, 2007.

[6] A. F. Webster and S. E. Tavares, "On the Design of S-boxes," Advances in Cryptography -Cypto '85 (Lecture Notes in Computer Science), Vol. 219, pp. 523-534, 1985.

[7] C. E. Shannon, "A mathematical theory of communication," ACM SIGMOBILE Mobile Computing and Communications Review, Vol. 5, No. 1, pp. 3-55, 2001.

[8] M. Sbeiti, M. Silbermann, A. Poschmann and C. Paar, "Design Space Exploration of PRESENT Implementation for FPGAs," Southern Conference on Programmable Logic - SPL, 2009.

[9] Y. Panasayya and K. Jen-Peter, "Lightweight Cryptography for FPGAs," International Conference on Reconfigurable Computing and FPGAs, 2009.

[10] P. Paulo and C. Ricardo, "Compact CLEFIA Implementation on FPGAs," 21st International Conference on Field Programmable Logic Applications, 2011.

저 자 소 개

Gookyi Dennis A. N.(Member)

2013 BSc Degree in Computer Engineering, Kwame Nkrumah Univ. of Science and Technology, Ghana.

2014~Studying for M. S. Degree in Information and Comm. Engineering, Hanbat National Univ., South Korea

<Research Interests : SoC Design and Verification, Lightweight Cryptography>

Seung-yong Park(Member)

2010 BS Degree in Information and Comm. Engineering, Hanbat National Univ.

2012 M. S. Degree in Information and Comm. Engineering, Hanbat National. Univ.

2012~Studying for PhD Degree in Information and Comm. Engineering, Hanbat National Univ.

<Research Interests: SoC Design and Verification for Image Processing, Multimedia Codec Design>

Kwangki Ryoo(Member)

1986, 1988, 2000 B. S., M. S. and Ph. D. in Electronic Engineering, Hanyang University.

1991~1994 Assistant Professor, Korea Military Academy

2000~2002 Senior Researcher, ETRI IC Design Team

2010~2011 Visiting Professor, Univ. of Texas at Dallas

2003~Currently Professor, Hanbat National Univ.

<Research Interests: Engineering education, SoC Design and Verification for Image Processing, Multimedia Codec Design>