회로 크기 축소를 기반으로 하는 저 전력 암호 설계

Low Power Cryptographic Design based on Circuit Size Reduction

유영갑, 김승열, 김용대, 박진섭
충북대학교 정보통신공학과

Younggap You(ygyou@cbnu.ac.kr), Seung-Youl Kim(kimsy@hbt.chungbuk.ac.kr),
Yong-Dae Kim(ydkim@hbt.chungbuk.ac.kr), Jinsub Park(jspark@hbt.chungbuk.ac.kr)

요약

본 논문은 기존의 블록 암호 프로세서를 128-bit 구조에서 32-bit구조로 소형화시킨 저 전력 구조를 제안하였다. 본 논문의 목적은 암호 이론 연구가 아닌 실용화 연구로서 실용화 결과를 보이는 것이다. 제안된 구조는 하드웨어 크기를 줄이기 위해 데이터 패스와 확산 함수가 수정되었다. 저전력 암호회로의 예로서 ARIA 알고리즘을 고쳐서 4개의 S-box가 사용되었다. 제안된 32-bit ARIA는 13,893 게이트로 구성되어있으며 기존 128-bit 구조보다 68.25% 더 작다. 설계된 회로는 매그너칩스의 0.35um CMOS 공정을 기반으로 표준 셀 라이브러리를 이용하여 합성되었다. 트랜지스터 레벨에서 전력 시뮬레이션 결과 이 회로의 전력 소모는 71MHz에서 기존의 128-bit ARIA구조의 9.7%인 61.46mW으로 나타났다. 이 저 전력 블록 암호 회로는 전원이 없는 무선 센서 네트워크 또는 RFID 정보보호에 핵심요소가 될 것이다.

■ 중심어 : | ARIA | 저전력 | 블록암호 |

Abstract

This paper presented a low power design of a 32bit block cypher processor reduced from the original 128bit architecture. The primary purpose of this research is to evaluate physical implementation results rather than theoretical aspects. The data path and diffusion function of the processor were reduced to accommodate the smaller hardware size. As a running example demonstrating the design approach, we employed a modified ARIA algorithm having four S-boxes. The proposed 32bit ARIA processor comprises 13,893 gates which is 68.25% smaller than the original 128bit structure. The design was synthesized and verified based on the standard cell library of the MagnaChip's 0.35um CMOS process. A transistor level power simulation shows that the power consumption of the proposed processor reduced to 61.4mW, which is 9.7% of the original 128bit design. The low power design of the block cypher processor would be essential for improving security of battery-less wireless sensor networks or RFID.

■ keyword : | ARIA | Low Power | Block Cypher |

I. Introduction

Wireless sensor networks carry security features mainly relied on software. Hardware handling cryptic operations will reduce the processing time and power

* This work was supported by the research grant of the Chungbuk National University in 2005.

접수번호 : #061211-001
접수일자 : 2006년 12월 11일
심사완료일 : 2007년 02월 06일
교신저자 : 유영갑, e-mail : ygyou@cbnu.ac.kr

yielding more efficient and wider applications. Traditional crypto-designs have been focused on high performance block cipher circuitry with some pipe-line structure. The designs are not proper for wireless sensor networks and RFID. A low-power and small-size design is better than high-performance structure, due to power consumption, transaction time and the number of gates in wireless sensor networks and RFID.

The block cipher algorithms such as ARIA and AES are 128bit block ciphers with the SPN (Substitution- Permutation Network) structure[1][2]. ARIA is known to be relatively stronger than AES against differential crypt-analysis and linear crypt-analysis. According to security and performance analysis, ARIA and AES are two times faster than competing alternatives such as SEED and Camellia[1].

In this paper, we propose an architecture to implement block cipher efficiently. We place a strong focus on a very tight area and power consumption constraints of portable applications. The design proposed in this paper is based on a reduced hardware size consuming less power than the original design.

The ARIA[5] based on a 1-round loop is implemented with Xilinx VirtexE-1600 FPGA. Its throughput is 496 Mbps using 1,491 logic slices and 16 Block RAM's. We have similar results from AES implementation. From now on we are focus on the reduced ARIA implementation. We use four S-boxes and 32-bit diffusion function, and modify its data-path to eliminate the diffusion function generating a round key used for ciphering data.

This paper is organized as follows. In section II, the ARIA algorithm is briefly described. In section III, the proposed ARIA implementation is presented. Section IV explains simulation results of the proposed ARIA design. Finally, concluding remarks are in Section V.

II. Original ARIA Algorithm

The ARIA algorithm comprises a round function and a key scheduler for encryption and decryption. The round function is based on the involution SPN structure. Input and output data is 128bits. The key size is one of 128, 192 or 256 bits[3].

The ARIA design can process data blocks of 128 bits, using cipher keys with lengths of 128, 192 and 256 bits as in AES systems. The number of rounds depends on the key size. The round function comprises three parts of AddRoundKey, Substitution, and Diffusion. [Fig. 1] shows the encryption and decryption of the ARIA algorithm. The AddRoundKey part performs exclusive-OR operations on 128-bit inputs and 128-bit keys. The Substitution part includes S-box and Inverse S-box.

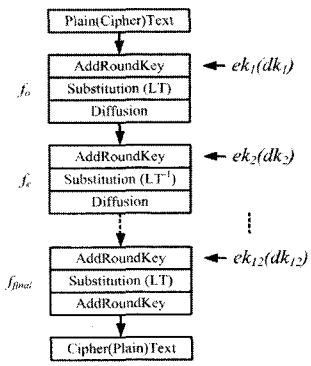


Fig. 1. Data path of ARIA

$$\begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \\ y_8 \\ y_9 \\ y_{10} \\ y_{11} \\ y_{12} \\ y_{13} \\ y_{14} \\ y_{15} \end{pmatrix} = \begin{pmatrix} 0&0&0&1&1&0&1&0&1&1&0&0&0&1&1&0 \\ 0&0&1&0&0&1&0&1&1&1&0&0&1&0&0&1 \\ 0&1&0&0&1&0&1&0&0&0&1&1&1&0&0&1 \\ 1&0&0&0&0&1&0&1&0&0&1&1&0&1&1&0 \\ 1&0&1&0&0&1&0&0&1&0&0&1&0&0&1&1 \\ 0&1&0&1&1&0&0&0&0&1&1&0&0&0&1&1 \\ 1&0&1&0&0&0&0&1&0&1&1&0&1&1&0&0 \\ 0&1&0&1&0&0&1&0&1&0&0&1&1&1&0&0 \\ 0&0&1&1&1&0&0&1&0&0&1&0&0&1&0&1 \\ 0&0&1&1&0&1&1&0&0&0&0&1&1&0&1&0 \\ 1&1&0&0&0&1&1&0&1&0&0&0&0&1&0&1 \\ 1&1&0&0&1&0&0&1&0&1&0&0&1&0&1&0 \\ 0&1&1&0&0&0&1&1&0&1&0&1&1&0&0&0 \\ 1&0&0&1&0&0&1&1&1&0&1&0&0&1&0&0 \\ 1&0&0&1&1&1&0&0&0&1&0&1&0&0&1&0 \\ 0&1&1&0&1&1&0&0&1&0&1&0&0&0&0&1 \end{pmatrix} \bullet \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \\ x_{10} \\ x_{11} \\ x_{12} \\ x_{13} \\ x_{14} \\ x_{15} \end{pmatrix}$$

Fig. 2. 16x16 binary matrix

The diffusion part comprises a simple 16×16 involution binary matrix as shown in [Fig. 2]. The key initialization employs a Feistel structure using the round function as its f function. The initialization determines the values of W_0, W_1, W_2, and W_3. The round key is from the combination of the four values. The same algorithms is used both in encryption and decryption further simplifying circuit structure, whereas AES uses different procedures. The decryption round keys are different from the encryption round keys and are derived from the encryption round keys. The ordering of round keys are reversed followed by the output of the diffusion function A to all round keys except for the first and the last. The decryption round keys are the same as following when the round number is n[3].

$$dk_1=ek_{n+1},\ dk_2=A(ek_n),\ \cdots,\ dk_n=A(ek_2),\ dk_{n+1}=ek_1 \quad (1)$$

III. Modification of the Algorithm

The hardware size reduction addresses the low power design goal of the block cipher processor. A smaller hardware is a powerful approach reducing power consumption per unit time.

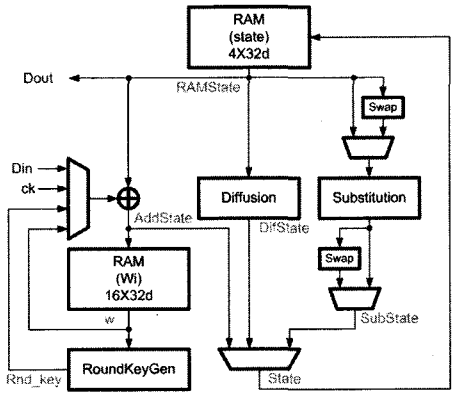
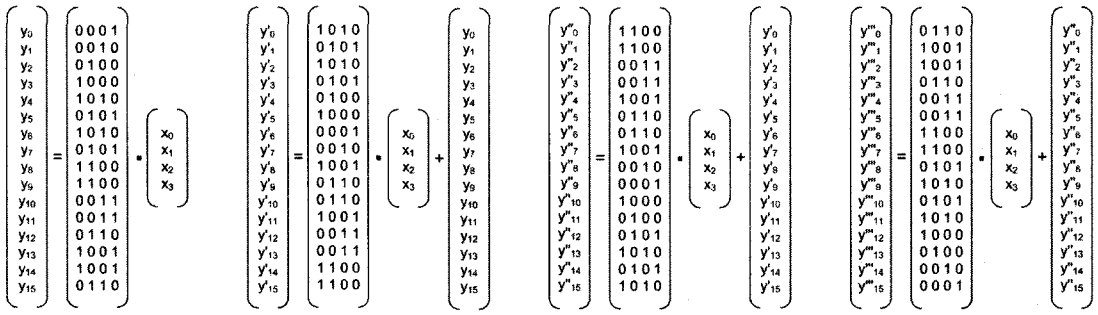


Fig. 3. Block diagram of ARIA

ARIA hardware implementations can be tailored for a smaller die-size. The algorithm can meet different system bus widths. Because ARIA uses minimum two S-Boxes, the modified version of implementation is compatible to 16-bit, 32-bit and 128-bit platforms. This paper presents a 32-bit structure of ARIA. [Fig. 3] shows the reduced 32bit ARIA structure.

The reduced ARIA design is based on the standard cell library of the MagnaChip's 0.35um CMOS process. The synthesis tool is the design compiler from Synopsys Inc.



(a)on the first word (b)on the second word (c) on the third word (d)on the fourth word

Fig. 4. 4x16 binary matrix for 32-bit operation of diffusion

1. Modification of Round Function and Memory

A Round function comprises AddRoundKey, Substitution, and Diffusion. The AddRoundKey is an exclusive-OR that adds a round key to the state in each iteration, where the round keys also are generated each round.

The substitution, LT, includes four S-Boxes, S1, $S1^{-1}$, S2, and $S2^{-1}$. An S-Box is implemented as a look-up table. The substitution of the 128-bit structure requires 16 S-Boxes. The number of S-boxes of the proposed 32bit design is four. The LT and LT^{-1} of an S-Box can be shared by re-ordering the array sequences.

Diffusion is designed for 32-bit operations. The diffusion of the proposed ARIA differs from other block cipher which is the original 16×16 binary matrix shown in [Fig. 2]. The diffusion proposed in this paper is a 4x16 binary matrix that divides the original 16×16 matrix by 4. [Fig. 4] shows the diffusion process using the 4×16 matrix. It needs a 128-bit register for intermediate values.

The ARIA needs four 128 bits memories to save the values of W0, W1, W2, and W3 and a 128-bit memory for an intermediate state of the round function. The ARIA in this paper uses 16×32-bit registers and 4×32-bit registers as memory. The memory block for registers comprises 5,709 gates.

2. Modification of Data Path

The ARIA algorithm is an involution SPN structure using the same circuit for encryption and decryption. The difference between the encryption and the decryption is the use of diffusion function in the round key generation for decryption.

The data path modification proposed in this paper eliminates the diffusion function from the round key generation for decryption. The diffusion block

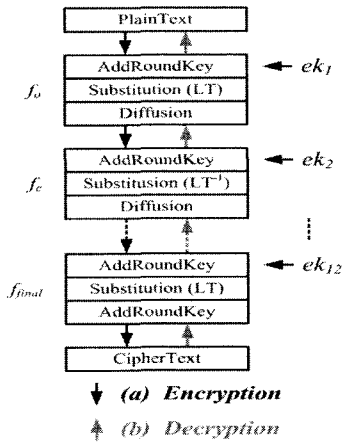


Fig. 5. Structure of ARIA algorithm

including 128bit registers takes 2,262 gates.

[Fig. 5] shows the modified ARIA data path: Figure 5a and 5b are for encryption and decryption, respectively. Inputs of functional blocks are switched using a multiplexer reflecting the encryption or decryption process.

The input of AddRoundKey is W, output of RAM for initialization, the output of diffusion function for encryption, the output of the substitution function for decryption, and data from outside sources. Substitution attaches a swap function at the input and output to share LT and LT^{-1}. The swap function swaps the upper 16 bits with the lower 16 bits of input at every even round. We suggest a data path that can remove diffusion generating a decryption round key. The decryption is executed using an encrypting round key.

3. Modification of Key Schedulers

The Key Scheduler produces a round key using both encoding and decoding. The combination of the four Wi values makes the round key.

In case of the proposed 32-bit ARIA, the round key is generated by using a barrel shifter and XOR gates.

Data from the RAM are rotated by a predetermined number of positions. The rotated data goes through the excusive-OR gate with the previous output value stored in the register. To produce a 32-bit round key, three items have to be fetched from the RAM. For example, to make $ek_{1,0}$, the upper 32 bits of ek_1, needs $W_{0,0}$, $W_{1,0}$, $W_{1,3}$ where $W_{i,j}$ is the jth 32-bit part of W_i. The $ek_{1,0}$ is the sum of $W_{0,0}$, and upper 32-bit of W_1 right-shifted as 19 bit positions denoted {$W_{1,3}$[18:0], $W_{1,0}$[31,19]}. The rotation count is fixed by the barrel shifter. [Fig. 6] is the block diagram of RndKeyGen that generate round keys using the barrel shifter for a rotated word. A 32-bits round key generation takes 4 clock cycles. It takes 3 clock cycles to access the RAM getting three values, and 1 clock cycle to initialize the register.

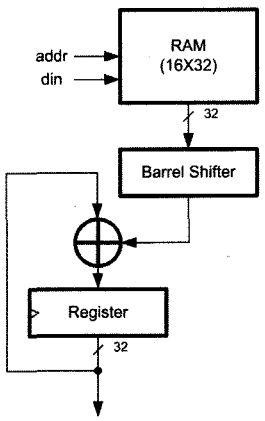
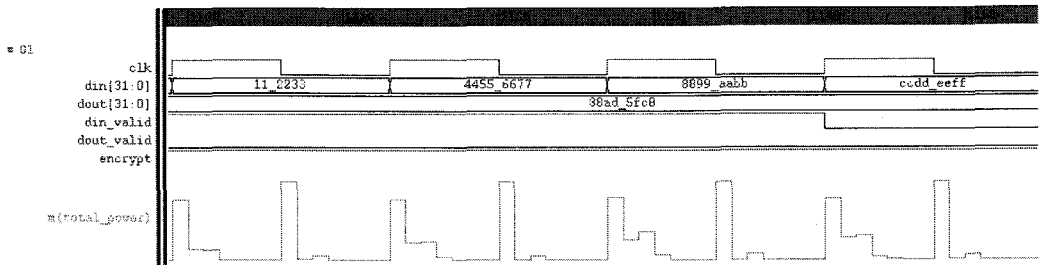


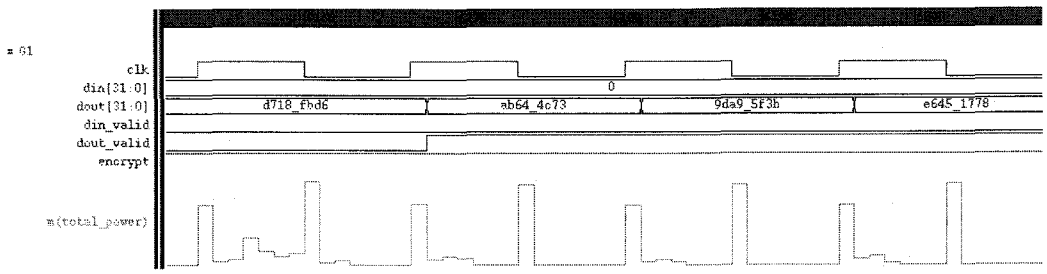
Fig. 6. Block diagram of RndKeyGen

Ⅳ. Performance Analysis

We now turn to the performance issue of the proposed 32bit ARIA design comparing to the original 128bit processor. The circuit synthesis is based on the standard cell library of the MagnaChip's 0.35um CMOS process.



(a) data input



(b) data output

Fig. 7. Simulation results

The circuit synthesis and power simulation employ the design compiler and NanoSim of Synopsys Inc., respectively.

The simulation results shows the promising performance. [Fig. 7] illustrates the simulation results of the proposed 32-bits ARIA processor. The test vectors for the evaluation are from the standard inputs defined in the official ARIA specification[3]. [Fig. 7a][Fig. 7b] show the input and output results compared to the specification.

- Encryption test vector : 128 bit encryption key
- Key : 000102030405060708090a0b0c0d0e0f
- Input : 00112233445566778899aabbccddeeff
- Output : d718fbd6ab644c739da95f3be6451778

[Table 1] shows a comparison of the proposed 32-bit version and a 128-bit version re-synthesized with the CMOS library. The required hardware complexity of 32-bit version is estimated to be 13,893 equivalent gates. The proposed 32-bit ARIA is 68.25 % smaller than the original 128-bit system.

The substitution block that is the biggest one in 128-bit version is saved remarkably by using four S-Boxes. The biggest block in the 32-bit version is the memory block. The ARIA needs for four 128-bit values to generate round keys and a 128-bit value for the state. The diffusion block gets bigger because it has registers to save intermediate values.

Table 1. The size and power characteristic of ARIA

Block	Equivalent Gates		Power (mW)@71MHz	
	128-bit	32-bit	128-bit	32-bit
AddRoundKey	2,217	390	52.99	1.07
Substitution	25,538	3,493	497.78	1.96
Diffusion	1,888	2,543	15.64	21.28
Initial (Key MEM)	5,489	4,372	39.88	12.56
RoundKeyGen	6,212	594	40.53	6.26
State MEM	-	1,337	-	9.23
Control circuit	2,416	1,282	32.33	9.1
Total	43,760	13,893	679.15	61.46

Table 2. Comparison of ARIA and AES [11]

Algorithm	Bus Width	Area (Gates)	Throughput (Mbps)	Frequency (MHz)	Process (um)
ARIA	32-bit	13,893	22	71	0.35
AES	32-bit	15,493	241	64	0.6

Table 3. The mission time and energy characteristics

	Mission time	Energy@ 71 MHz
128-bit	12 cycles	114 nJ
32-bit	419 cycles	362 nJ

The power consumption of the 32-bit ARIA is about 9.7% of the 128-bit version. The values of power simulation are an average power for encrypting a plain-text.

[Table 2] shows a comparison between AES (Advanced Encryption Standard) and ARIA implementations. The two algorithms are similar in structure. The proposed 32-bit ARIA circuit has a slightly smaller size as the AES case[11].

It is difficult to find power consumption data for published designs since most designs did not go through silicon implementation but relied on the synthesis of circuitry. Comparison of power consumption is difficult.

[Table 3] shows the comparison of delay and energy consumption when simulated at 71 MHz and 2.5 V. [Table 3] shows that the 32bit design consumes more power than the 128bit design. It is due to the longer mission time for the 32bit design. The smaller average power is an important feature for RFID applications rather than total energy consumed for each transaction[13] since unlimited energy is available for RFID tags from air interface. However, the energy in a given unit time is limited for a RFID tag due to its on-chip energy collection mechanism. It is necessary to limit the average power consumption.

The maximum clock frequency of 71 MHz allows a data throughput rate of 25 Mbps after key initialization. The 32-bit ARIA module needs 63 clock cycles for initialization and 356 clock cycles for encryption. After generating a 32-bit word of a round key that takes 3 cycles, it need a cycle for a 32-bit word of AddRoundKey and initialization of a register of the RndKeyGen is executed simultaneously. Therefore, AddRoundKey spends 16 clock cycles. The substitution needs 4 clock cycles. It takes 4 clock cycles for diffusion, and the 4 clock cycles to output 32-bit pieces. The one round function of ARIA designed in this paper needs 28 clock cycles

V. Conclusion

This paper presented a low power block cipher design for battery-less wireless sensor networks or RFID tags demanding lower average power. A 32-bit architecture of ARIA reduced from the original 128-bit block cipher was developed for verification purposes. The ARIA module in this paper needs 63 clock cycles for initialization and 356 clock cycles for encryption. The hardware size is estimated to be 13,893 gate equivalents. The 32-bit ARIA proposed in this paper is 68.25 % smaller than the original 128-bit ARIA. The power consumption is 61.46mW, 9.7% of the original 128-bit system.

This work aims at the evaluation of physical implementation rather than crypto-theoretic aspects. The secrecy of the algorithm doe not changes as the circuit size reduces. However, we expect stronger immunity on some attacks such as differential power analysis since smaller current variation demands higher sensitivity of current sensing mechanism and thereby makes it difficult to analyze externally. Future research will address this immunity issues as the circuit size becomes smaller.

References

[1] http://www.nsri.re.kr/ARIA/

[2] D. Kwon, J. Kim, S. Park, S. Sung, Y. Sohn, J. Song, Y. Yeom, E. Yoon, S. Lee, J. Lee, S. Chee, D. Han, and J. Hong, "New block cipher: ARIA," in Proc. ICISC2003, pp.432-445, Nov. 2003.

[3] http://www.nsri.re.kr/ARIA/doc/ARIA-specification.pdf/

[4] W. Nebel and J. Mermet, Low Power Design in Deep Submicron Electronics, Kluwer Academic Publishers, 1997.

[5] J. Park, Y. Yun, Y. D. Kim, S. Yang, T. Chang, and Y. You, "Design and implementation of ARIA cryptic algorithm," Journal of IEEK, Vol.42-SD, No.4, pp.22-36, Apr. 2004.

[6] M. Feldhofer, S. Dominikus, and J. Wolkerstorfer, "Strong authentication for RFID systems using the AES algorithm," in Proc. CHES2004, pp.357-370, Aug. 2004 .

[7] F. X. Standaert, G. Rouvroy, J. J. Quisquater, and J. D. Legat, "A methodology to implement block ciphers in reconfigurable hardware and its application to fast and compact AES Rijndael," in Proc. FPGA'03, pp.216-224, Feb. 2003.

[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," *in Proc. ITCC2004*, pp.583-587, Apr. 2004 .

[9] G. P. Saggese, A. Mazzeo, N. Mazocca, and A. G. M. Strollo, "An FPGA based performance analysis of the unrolling, tiling and pipelining of the AES algorithm," in Proc. FPL2003, 2003.

[10] T. F. Lin, C. P. Su, C. T. Huang, and C. W. Wu, "A high-throughput low-cost AES cipher chip," in Proc. 3rd IEEE Asia-Pacific Conf. ASIC, pp.85-88, Aug. 2002.

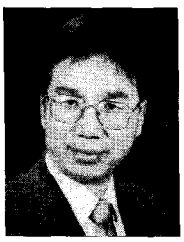
[11] S. Mangard, M.Aigner, and S. Dominikus, "A Highly Regular and Scalable AES Hardware Architecture," IEEE Trans. Comp., Vol.52, No.4, pp.483-491, Apr. 2003.

[12] I. Verbauwhede, P. Schaumont, and H. Kuo, "Design and Performance testing of a2.29Gb/s Rijndael Processor," IEEE Journal of Solid-State Circuits, Vol.32, No.3 pp.569-572, Mar. 2003.

[13] M. Feldhofer, J. Wolkerstorfer, and V. Fijmen, "AES implementation on a grain of sand," IEE Proceedings on Information Security, Vol.152, pp.13-20, Oct. 2005.

저 자 소 개

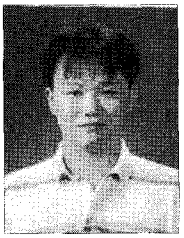
유 영 갑(Younggap You) **정회원**



- 1975년 8월 : 서강대학교 전자공학과 (공학사)
- 1975년 ~ 1979년 : 국방과학연구소 연구원
- 1981년 8월 : Univ.of Michigan, Ann Arbor 전기전산학과 (공학석사)
- 1986년 4월 : Univ.of Michigan, Ann Arbor 전기전산학과 (공학박사)
- 1986년 ~ 1988년 : 금성반도체(주) 책임 연구원
- 1993년 ~ 1994년 : 아리조나 대학교 객원교수
- 1998년 ~ 2000년 : 오레곤 주립대학교 교환교수
- 1988년 ~ 현재 : 충북대학교 정보통신공학과 교수

<관심분야> : VLSI 설계 및 Test, 고속 인쇄회로 설계, Cryptography

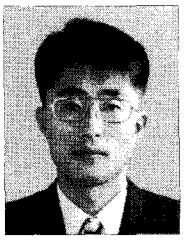
김 승 열(Seung-Youl Kim) **정회원**



- 2002년 2월 : 충북대학교 정보통신공학과 (공학사)
- 2004년 8월 : 충북대학교 정보통신공학과 (공학석사)
- 2005년 3월 ~ 현재 : 충북대학교 정보통신공학과 박사과정

<관심분야> : 디지털 회로설계, Cryptography

김 용 대(Yong-Dae Kim) **정회원**



- 1990년 2월 : 충북대학교 정보통신공학과 (공학사)
- 1993년 2월 : 충북대학교 컴퓨터공학과 (공학석사)
- 1989년 ~ 1998년 : 신흥기술연구소 팀장
- 2000년 ~ 현재 : 충북대학교 정보통신공학과 박사과정

<관심분야> : Computer arithmetic, ASIC 설계, 암호시스템

박 진 섭(Jinsub Park) **정회원**

- 2004년 8월 : 충북대학교 전기전자공학부(공학사)
- 2006년 8월 : 충북대학교 정보통신공학과 (공학석사)

<관심분야> : 디지털 회로설계, 암호시스템