KIPS Trans. Comp. and Comm. Sys.
Vol.8, No.11 pp.271~276 pISSN: 2287-5891



https://doi.org/10.3745/KTCCS.2019.8.11.271

# Design of Optimized ARIA Crypto-Processor Using Composite Field S-Box

Kang Min Sup†

## ABSTRACT

Conventional ARIA algorithm which is used LUT based-S-Box is fast the processing speed. However, the algorithm is hard to applied to small portable devices. This paper proposes the hardware design of optimized ARIA crypto-processor based on the modified composite field S-Box in order to decrease its hardware area. The Key scheduling in ARIA algorithm, both diffusion and substitution layers are repeatedly used in each round function. In this approach, an advanced key scheduling method is also presented of which two functions are merged into only one function for reducing hardware overhead in scheduling process. The designed ARIA crypto-processor is described in Verilog-HDL, and then a logic synthesis is also performed by using Xilinx ISE 14.7 tool with target the Xilnx FPGA XC3S1500 device. In order to verify the function of the crypto-processor, both logic and timing simulation are also performed by using simulator called ModelSim 10.4a.

Keywords : ARIA Crypto-Processor, Composite Field S-Box, Key Scheduling, Verilog HDL

# 합성체 S-Box 기반 최적의 ARIA 암호프로세서 설계

강 민 섭†

## 요 약

LUT 기반의 S-Box를 사용하는 기존의 ARIA 알고리듬은 처리속도는 빠르지만 회로의 크기가 매우 커지게 되어 저면적이 요구되는 소형의 휴대용 기기에는 적용하기 어렵다. 본 논문에서는 하드웨어 면적의 감소를 위해 개선된 합성체 S-Box를 기반으로 한 최적의 ARIA 암호프로세서 설계를 제안한다. ARIA 알고리듬에서의 키 스케쥴링 과정에서 확산 및 치환 계층에서 반복적으로 사용한다. 여기에서는 또한, 키 스케쥴링 과정에서의 사용 면적을 최소화하는 방안으로 치환과 확산 계층에서 하드웨어 자원의 공유 방법을 제안한다. 설계된 ARIA 암호프로세서는 Verilog-HDL을 이용하여 회로를 기술하였고, Xilinx XC3S1500을 타겟으로 하여 논리 합성을 수행하였다. 설계된 시스템의 기능 검증을 위해 Mentor사의 Modelsim 10.4a 툴을 이용하여 논리 및 타이밍 시뮬레이션을 수행하였다.

키워드 : ARIA 암호프로세서, 합성체 S-Box, 키 스케쥴링, Verilog HDL

## 1. 서 론

암호화 기술은 데이터의 기밀성, 무결성, 디지털 서명 및 개인 정보보안 등 을 위한 필수 기술로서 인터넷 기반의 유선망이나 무선 통신망, 정보 유통, 그리고 전자상거래 등에 널리 사용되고 있다[1, 2]. 이러한 암호화 기술은 하드웨어나 소프트웨어로 구현이 가능하다. 마이크로프로세서 기반의 소프트웨어 구현은 구현 비용이 비교적 저렴하지만 데이터 처리속도 면에서 한계가 있다. 하드웨어 구현 기술은 소프트웨어 구현에 비해 비용이 많이 들지만 대용량이나 실시간의 데이터를 처리가 용이하므로 고속 하드웨어 설계나 스마트카드, NFC와 같은 휴대용 장치의 설계에 적합하다[3, 4].

국내에서 개발된 ARIA 암호 알고리듬은 대칭키 방식이며, SEED와 함께 많이 사용되고 있다[5, 6]. 향후 스마트 카드, 사물 인터넷(IOT) 그리고 모바일 기기 등의 초경량 환경에서 사용하기 위해서는 저면적, 고성능의 암호 시스템 개발이 필요하다[7, 8].

본 논문에서는 개선된 합성체 S-Box를 기반으로 하여 면적 및 처리속도 면에서 최적의 ARIA 암호프로서 설계를 제안한다. 또한, 키 스케쥴링 과정에서 확산계층과 치환 계층에서 하드웨어 자원을 공유하는 설계방법을 제안한다. 설계된 ARIA 암호시스템은 Verilog HDL을 이용하여 구조적 모델링

※ 이 논문은 연구년 기간 중 연구되었음.
† 종신회원 : 안양대학교 컴퓨터공학과 교수
Manuscript Received : June 18, 2019
First Revision : July 10, 2019
Accepted : July 17, 2019
* Corresponding Author : Kang Min Sup(mskang@anyang.ac.kr)

을 수행하였고, Xilinx ISE 14.7 툴을 이용하여 Xilinx XC3S1500를 타겟으로하여 논리 합성을 수행하였다.

## 2. ARIA 암호 알고리듬

ARIA(Academy Research Institute Agency)는 국내에서 개발한 블록 암호 알고리듬으로서 국가 표준 암호 알고리듬으로 사용하고 있다[1]. ARIA알고리듬은 암호화 및 복호화 과정이 동일한 구조를 갖는 Involution SPN(Substitution-Permutation-Networks)로 되어 있다. 이 알고리즘의 데이터 블록은 128비트이며, 암호화 키는 128/196/256 비트를 지원한다. 이 알고리즘의 라운드 횟수는 키의 크기에 따라 12, 14, 또는 16회 반복하는 구조를 가진다[1].

ARIA 알고리듬은 라운드 키 덧셈연산, 치환 계층, 그리고 확산 계층의 3 모듈로 구성되며, 마지막 라운드에서는 확산계층 대신에 라운드 키 덧셈 연산을 수행하는 구조로 되어있다[1].

이 알고리듬의 각 라운드에서는 3개의 라운드 함수, 즉 홀수, 짝수, 그리고 최종 라운드 함수를 사용하게 된다. 그리고 복호화 과정의 라운드 키는 확산함수를 통과하여 계산된 암호화 라운드 키를 사용한다[1].

라운드 키 덧셈은 128비트의 키와 128비트의 암호화될 입력 데이터가 비트별로 XOR(⊕) 연산을 수행한다. 또한, 치환 계층(Substitution layer)은 2개의 유형(type 1, type 2)으로 되어 있는데 이 유형들은 모두 4개의 S-Box ($S_1, S_2, S_1^{-1}, S_2^{-1}$)로 구성되어 있다. S-Box는 32비트 단위로 사용되며, S1과 S1-1은 서로 역 관계를 가진다[1].

ARIA 암호 알고리듬은 키 길이에 따라서 라운드 수가 변화하며, 라운드 함수는 홀수(odd) 라운드 함수(Fo), 짝수(even) 라운드 함수(Fe), 최종(final) 라운드 함수(Ff)로 구성이 되어 있다. 이 세 함수에서 사용하는 $LT$와 $LT^{-1}$은 32비트로 구성된 치환 함수로서 8비트씩 나누어 2 종류의 S-Box와 그들에 대한 역변환을 수행한다[1].

이때 치환계층은 항상 교대로 사용되며, 확산 계층을 포함한 전체가 involution 구조가 되도록 구성한다. 미국의 표준 암호화체계인 AES는 유한체 $GF(2^8)$ 상의 함수 $x^{-1}$에 아핀 변환 방법을 사용하고 있으나 ARIA에서는 $x^{-1}$과 $x^{247}$에 대해서 아핀 변환을 취한 후 S-Box를 생성하는 방법을 취하고 있다[1].

확산 계층(Diff Layer)은 16×16 Involution 이진(binary) 행렬(matrix)을 사용한다. 확산함수는 16 바이트의 입력에 대해 바이트 단위로 곱셈 연산을 수행하고, 결과로서 16바이트의 출력을 생성한다.

ARIA 알고리듬의 키 초기화 과정은 키 스케쥴링과 라운드 키 생성 과정으로 나눌 수 있다. 키 스케쥴링 과정에서는 주어진 암호화키로부터 128비트 값으로 구성된 4개의 라운드 키 즉, W1, W2, W3, W4 를 생성하게 된다. 라운드 키 생성 과정에서는 4 개의 라운드 키를 기반으로 하여 암호화 라운드 키 $ek_i$와 $dk_i$를 생성하여 사용한다[1-2].

본 논문에서는 사용 면적의 최소화를 위해 키 스케쥴링 과정에서 치환계층과 확산계층에서 하드웨어 자원을 공유하는 방법을 제안한다.

## 3. 합성체 S-Box 기반 ARIA 암호 프로세서 설계

### 3.1 개선된 합성체 기반의 S-Box 구조

ARIA 암호프로세서에는 2개의 S-Box($S_1, S_2$)와 그의 역의 관계인 S-Box($S_1^{-1}, S_2^{-1}$)가 필요하다. 이러한 S-Box는 ARIA 알고리듬에서 라운드 함수를 사용하는 곳에서 모두 필요하며 이 알고리듬 내에서 가장 큰 면적을 차지한다. 일반적으로 LUT(LookUp Table) 기반의 S-Box는 2차원의 메모리 형태로 구현되므로 처리속도는 매우 빠르지만, 하드웨어 설계 면적이 증가되는 단점이 있다[5]. 본 논문에서는 면적 및 처리속도 면에서 최적의 암호프로세서를 구현하기 위해 기존의 LUT 기반의 S-Box 의 단점을 보완한 개선된 합성체 기반의 S-Box를 제안한다.

S-Box를 구성하기 위한 Rijndael 알고리듬의 모든 연산은 $GF(2^8)$의 다항식으로 표현하여 사용한다.

계수가 $GF(2^4)$의 1차 다항식인 경우, 기약 다항식을 $x^2$+Ax+B와 같은 형태를 가지며, 임의의 다항식 bx+c 형태의 곱셈에 대한 역원 $(bx+c)^{-1}$ 는 Equation (1)과 같은 결과식을 얻을 수 있다[4]. 단, A=1, B=1100(λ)로 대치시킨다.

$$b(b^2\lambda+(b+c)c)^{-1}x+(c+b)(b^2\lambda+(b+c)c)^{-1} \quad (1)$$

Fig. 1은 Equation (1)을 기반으로 하여 구성된 합성체 S-Box의 일반적인 구조를 나타낸다[4].

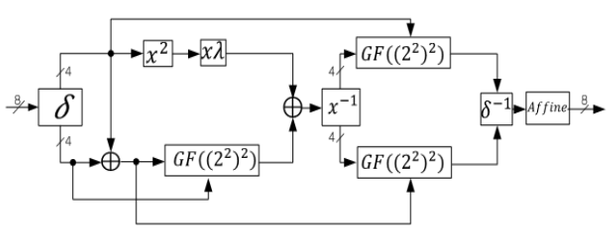


Fig. 1. Structure of Conventional Composite Field S-Box [4]

이 방법은 $GF(2^8)$의 구조를 $GF(((2^2)^2)^2)$의 형태로 변환 후 곱의 역원을 계산한다. 다음 단계에서는 체 역변환 행렬 $(X^{-1})$을 수행하고 다시 $GF(2^8)$로 변환한 후에 최종적으로 Affine 연산을 수행하게 된다.

$GF(((2^2)^2)^2)$상의 곱셈의 역원은 $GF((2^2)^2$상에서 3개의 곱셈 연산기와 $GF(2^4)$상의 역원 회로($x^{-1}$), 제곱($x^2$) 및 XOR 연산을 필요로 한다. Fig. 1에서 δ 모듈은 체 변환 행렬을, $\delta^{-1}$ 모듈은 체 역변환 행렬을 나타낸다[4]. 그리고 $x^2$ 모듈은

$GF(2^4)$ 상에서 제곱 연산을 수행하게 되고, λ 모듈은 2진수 1100으로 선택하여 사용한다.

Fig. 1의 $GF((2^2)^2$의 곱셈모듈은 3개의 $GF(2^2)$의 곱셈모듈, 파이(ϕ) 모듈, 그리고 4개의 XOR 연산모듈로 구성된다[4].

본 논문에서는 $GF((2^2)^2$ 곱셈기의 크기를 줄이기 위해서 유한체로 입력된 S-Box 입력을 합성체로 변환하여 4 비트의 곱셈 연산을 수행하는 효율적인 S-Box 구조를 제안한다. 즉, 기존의 방법[4]과는 달리 단지 곱셈 및 XOR 연산을 비트 단위로 수행하여 4 비트의 곱셈결과($OP\_GF$(i), i = 0, 1, 2, 3))를 구한다(Equation (3) 참조). Fig. 2는 $OP\_GF((2^2)^2)$ 형태의 합성체를 사용한 S-Box의 전체 블록도를 나타낸다.

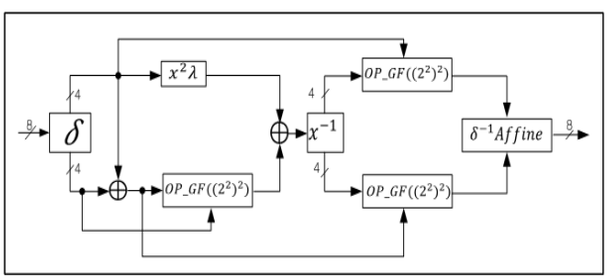


Fig. 2. Structure of ARIA S-Box Based on Modified Composite Field

$GF((2^2)^2)$상의 4비트 곱셈기를 설계할 경우, w{$w_3$, $w_2$, $w_1$, $w_0$}, q{$q_3$, $q_2$, $q_1$, $q_0$}, k{$k_3$, $k_2$, $k_1$, $k_0$}일 때 출력은 k=wq가 된다. 여기에서 $x^2$+x+ϕ를 사용하면 k는 Equation (2)에 나타난 결과식을 얻을 수 있다[4].

$$k=(wH q_H+w_L q H+w_H q_L)x+w_H q_H \phi+w_L q_L \quad (2)$$

기존의 $GF((2^2)^2$ 곱셈모듈에서[4]에는 Equation (2)를 이용하여 곱셈기를 구현하였다. 그러나 본 논문에서는 Equation (3)을 이용하여 합성체 기반의 $OP\_GF((2^2)^2)$ 곱셈모듈을 구현한다. Equation (3)은 최하위 출력 비트 $OP_GF(0)$을 계산하기 위한 계산식을 나타낸다.

$$((w(3)\&q(3)\&\oplus w(2)))\oplus(q(2)\&w(3))\oplus(q(1)\&w(1))\oplus(q(0)\ \&w(0)) \quad (3)$$

Equation (3)에서 연산자 &는 bitwise AND 연산을 나타낸다. 여기서 알 수 있듯이 기존의 곱셈기(Equation (2) 참조)에서 사용되는 3개의 $GF(2^2)$ 곱셈 연산기와 파이(ϕ) 연산기가 제거되므로 면적 및 처리속도 면에서 성능이 개선된다. 또한, $x^2$와 $x\lambda$를 한 개의 모듈로 통합하고($x^2\lambda$모듈)다시 $\delta^{-1}$과 $Affine$을 통합 ($\delta^{-1}Affine$모듈)하여 구성하여 S-Box의 연산 속도를 빠르게 한다.

### 3.2 최적의 ARIA 암호프로세서 설계

본 논문에서는 128-bit기준으로 하여 ARIA 암/복호프로세서를 구현하였으며, Fig. 3은 개선된 합성체 S-Box를 이용한 ARIA 암호프로세서의 구조를 나타낸다.

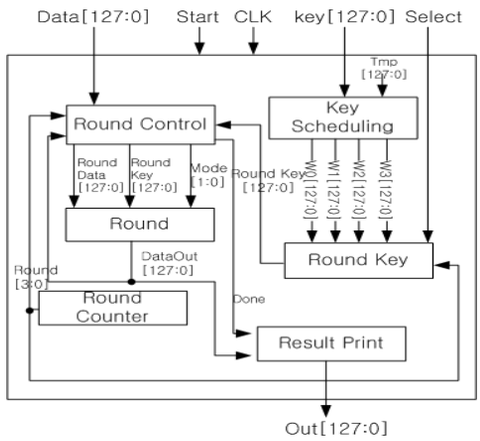


Fig. 3. Structure of the Proposed ARIA Crypto-processor

본 논문에서 설계한 ARIA 암호프로세서는 Key Scheduling, Round Counter, Round Key, Round Control, Round, 그리고 Result Print 모듈 등으로 구성되어 있다.

이 암호프로세서의 동작과정은 다음과 같다. 먼저 Key scheduling 모듈에서 128 비트 킷값을 2개의 버퍼($K_L$과 $K_R$)에 저장한 후, 4개의 128비트 $W_0$, $W_1$, $W_2$, $W_3$을 스케쥴링한다. $W_0$, $W_1$, $W_2$, $W_3$는 Round Counter에서 발생한 Round 값과 Round Key 모듈에 입력으로 쓰인다.

Round Key 모듈은 select 입력을 통해 암호화 Round Key 또는 복호화 Round Key를 생성한다. Round Control 모듈에서는 Round Key와 Round 값을 입력을 받아 Round Key 덧셈을 수행하고 RoundData, RoundKey, 그리고 Mode를 출력한다.

Round 모듈에서 2.1절에서 설명한 3개의 라운드 함수($F_o$, $F_e$, $F_f$) 가 포함되어 S-Box 연산이 수행되는 핵심 모듈로서 Mode 값에 따라 3개의 모듈을 단일 모듈로 성하여 사용된다. Result Print 모듈은 Round Control 모듈에서 생성된 Done의 값에 따라 완료 여부를 확인하고 Out을 출력한다.

본 논문에서는 키 스케쥴링 과정에서 발생하는 사용 면적의 증가를 줄이기 위하여 각각의 $F$함수에서 공통으로 사용되는 치환계층과 확산계층의 하드웨어 자원을 공유하도록 하는 개선된 Key Scheduling 구조를 제안한다.

ARIA 알고리듬에서 키 스케쥴링 과정은 키 스케쥴링과 RoundKey 생성과정으로 구분된다. 먼저, 키 스케쥴링 모듈에서는 $W_0$, $W_1$, $W_2$, $W_3$를 생성하게 되고, 여기에서 생성된 값들이 RoundKey 모듈로 입력으로 사용하게 된다. RoundKey 모듈에서는 입력인 평문과 각 라운드 키를 사용하여 덧셈 연산(XOR)을 수행하게 된다.

앞에서 설명했듯이 키 스케쥴링 과정은 feistal구조를 가지며 2회의 홀수 라운드, 1회의 짝수 라운드가 수행된다. 이 과정에서 치환계층과 확산계층이 반복되기 때문에 사용 면적이 증가하게 된다. 이러한 문제점의 개선을 위하여 여기에서는 스케쥴링 과정의 라운드 함수에서 사용되는 이 두 계층의 하드웨어 자원을 공유하도록 설계한다.

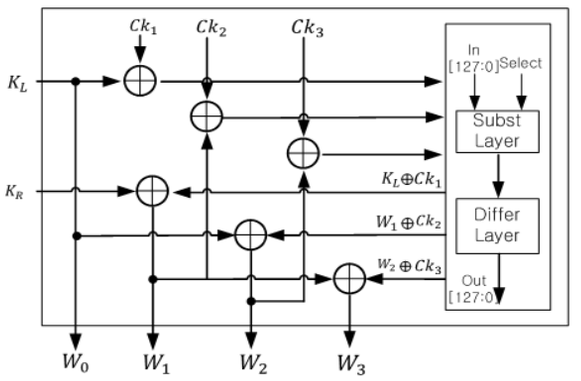


Fig. 4. Proposed Key Scheduling Sharing Method

Fig. 4는 제안하는 키 스케쥴링을 위한 두 계층의 하느웨어 자원의 공유 방법을 나타낸다. 키 스케쥴링을 위해서는 치환계층(Sub: Subst Layer) 모듈과 확산계층(Dif: Differ Layer) 모듈이 필요하다. Fig. 4에서 Subst Layer의 128 비트의 입력은 In[127:0]이며, Select(Sel)은 제어선이다. 두 계층의 출력값으로 나온 $K_L \oplus Ck_1$, $W_1 \oplus Ck_2$, $W_2 \oplus Ck_3$은 각각의 XOR 연산에 대한 입력 값을 의미한다. Sub는 XOR 연산의 입력 값(In)과 두 개의 치환계층을 선택하기 위한 Sel이 입력 값이 된다. 이와 같이 키 스케쥴링 과정에서는 4개의 128비트 값 ($W_0$, $W_1$, $W_2$, $W_3$)을 생성하게 된다.

## 4. 암호시스템 구현 및 성능평가

### 4.1 암호시스템 구현

본 논문에서 제안한 개선된 S-Box를 사용한 ARIA 암호 프로세서는 Verilog-HDL을 사용하여 회로를 기술하였고, Xilnx FPGA XC3S1500 상에서 구현되었다. 구현된 ARIA 암호프로세서의 동작검증을 위해 Mentor's Modelsim 10.4a를 사용하여 타이밍 시뮬레이션을 수행하였다.

Fig. 5는 구현된 ARIA 암호 프로세서의 S-Box(S1)에 대한 시뮬레이션 결과를 나타낸다.

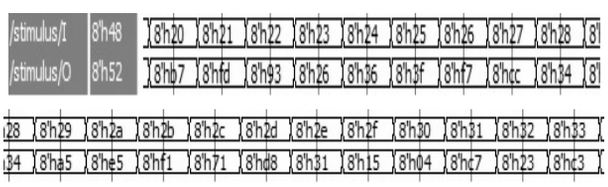


Fig. 5. Simulation Result of S-Box(S1) of Implemented ARIA

Fig. 5에서 "/sttimulus/I"는 입력 데이터를 나타내며, "/sttimulus/O"는 개선된 S-Box 에서의 치환된 결과를 나타낸다. 예를 들어서 입력 데이터 20(Hexa decimal)에 대한 S-Box 출력은 b7((Hexa decimal)이 된다.

Fig. 6은 설계된 S-Box를 사용하여 ARIA 시스템의 복호화를 위한 시뮬레이션 결과를 나타낸다.


Fig. 6. Simulation Result of ARIA Decryption using the Designed S-Box

Fig. 6에서 'data'는 평문 입력값과 key 값을 사용하여 얻어진 암호화된 결과를 나타내며, 'data_out'은 복호화된 결과를 나타낸다. 복호화된 시뮬레이션 결과를 통하여 입력된 평문 데이터와 암호화 출력이 일치되어 설계된 암호 시스템이 정상적으로 동작하고 있음을 나타내고 있다.

Table 1은 입력으로 ARIA 테스트 벡터[10]를 사용한 Fig. 6의 시뮬레이션 결과를 요약한 것이다.

Table 1. Summary of Simulation Result of Fig 6

| | |
|---|---|
| Input | 11111111aaaaaaaa11111111bbbbbbbb |
| Key | 00112233445566778899aabbccddeeff |
| Data | 6ecd08e22c30abdb215cf74e2075e6e |
| Data-out | 11111111aaaaaaaa11111111bbbbbbbb |

Table 1에서 data는 128비트의 평문(input) 입력과 키(key) 값을 사용하여 암호를 수행한 결과 값을, data_out은 복호화된 결과를 나타낸다.

### 4.2 성능 평가

Table 2는 기존의 키 스케쥴링 방법[11]과 본 논문에서 제안한 두 함수의 하드웨어 자원의 공유방법에 대한 합성결과를 나타낸다.

Table 2. Synthesis Results of Three Key Scheduling Methods

| Items / Methods | # of Slices | Delay (ns) |
|---|---|---|
| LUT[11] | 3,852 | 31.03 |
| CF_SBox+ KeySch | 2,858 | 77.36 |
| CF_SBox+ SH_KeySch | 2,150 | 16.7 |

Methods의 LUT[11]은 기존의 키 스케쥴링 과정에서 S-Box을 LUT로 구현한 방법이고, "CF_SBox+KeySch"은 기존의 키 스케쥴링 과정에서 개선된 S-Box를 기반으로 한 비교 결과이다. 그리고 "CF_SBox+SH_KeySch"는 본 논문에서 제안하는 개선된 S-Box와 두 계층의 하드웨어의 공유방법을 동시에 적용한 결과를 나타낸다. Table 2의 합성 결과로부터

"CF_SBox+SH_KeySch" 방법은 LUT[11] 기반과 비교하여 delay는 약 47% 정도가 개선되었으며, Slices도 약 44% 정도로 사용면적이 감소되었음을 알 수 있다.

Table 3은 Xilinx FPGA 소자(XC2v500) 상에서 S-Box의 구현 방식을 비교한 결과를 나타낸다.

Table 3. Comparison Results of Three S-Box Implementations

| Items / Methods | Slices (Out of 13312) | Maximum Comb. Path Delay (ns) | Process |
|---|---|---|---|
| Saurabh[4] | 45 | 19.255 | Xilinx XC2v500 |
| Edwin[5] | 45 | 21.117 | Xilinx XC2v500 |
| CF_SBox | 37 | 17.098 | Xilinx XC2v500 |

Table 3에서 "CF_SBox"는 본 논문에서 제안하는 개선된 합성체를 기반으로 하는 S-Box를 사용한 것이다. 하드웨어 사용 면적(Slices)은 기존의 방법들[4,5]에 비해 제안한 방법이 약 12% 정도 감소되었다. 또한, 최대 지연 시간(delay)은 속도가 비교적 빠른 Saurabh[4]의 방법과 비교했을 때 약 11% 정도 개선되었다.

Table 4는 기존의 ARIA 암호시스템과 제안한 시스템에 대해 성능을 비교한 결과이다.

Table 4. Performance Comparison of Crypto-processors

| Items / Methods | # of Slices | BRAMs | Max. Freq. (MHz) | Process |
|---|---|---|---|---|
| Kang[2] | 9,217 | 0 | 61.9 | XCV1600E |
| Park[7] | 1,491 | 16 | 46.5 | XCV1600E |
| Ha[11] | 6,437 | 128 | 192.9 | XC2VP30-7 |
| CF_SBox | 6,095 | 0 | 53.3 | XC3S1500 |
| CF_SBox+ S_KeySch | 5,328 | 0 | 55. | XC3S1500 |

본 논문에서 제안한 방법인 "CF_SBox+S_KeySch"와 기존의 방법[2]과의 동작속도 비교에서 최대 주파수(Maximum Frequency)는 약 10% 정도 감소하였지만, Slice는 약 35% 정도 개선되었다. 다른 기존의 방법[7]과의 동작속도 비교에서는 약 16% 정도가 개선되었다.

하드웨어 사용 면적 비교(Slices)에서 제안한 방법은 BRAMs을 사용하지 않았기 때문에 기존의 방법[7]과의 정량적인 비교는 어렵지만, 객관적으로 제안한 방법이 어느 정도의 하드웨어 오버헤드를 가진다고 판단된다. Ha[11] 등이 제안한 방법은 시스템 코어 부분을 pipeline으로 처리하기 때문에 동작속도는 매우 빠르지만 사용면적은 제안한 방법에 비해 매우 증가됨을 알 수 있다.

## 5. 결 론

본 논문에서 설계한 ARIA 암호프로세서는 저면적이 요구되는 소형의 휴대용 기기에 적용이 가능하도록 XC3S1500을 소자를 이용하여 FPGA 상에서 구현되었다. 보다 최적화된 암호 시스템의 구현을 위해 본 논문에서는 2가지 방안, 즉 개선된 합성체 기반의 S-Box 구조와 치환계층과 확산계층에서의 하드웨어 자원의 공유방법을 제안하였다.

논리 합성 및 타이밍 시뮬레이션 결과를 통하여 설계된 ARIA 암호프로세서가 정확히 동작함을 확인하였다.

구현된 128비트 ARIA 암호프로세서의 최대 동작 주파수는 55.4 Mhz이며, 약 645.5Mbps의 처리율을 보였다.

향후의 연구방향으로서 점점 소형화가 요구되는 스마트 카드, 사물 인터넷(IOT), 그리고 모바일 시스템 환경에서 사용이 가능한 저전력, 저면적, 고성능 암호 시스템에 대한 연구가 계속되어야 할 것이다.

## References


[1] ARIA Algorithm Specification, May 2004 [Internet], http://www.nsri.re. kr/ARIA/doc/ARIA-specification.pdf.

[2] M. S. Kang, "Design of a High-speed FPGA-Based ARIA Cipher Processor," *Journal of Security Engineering*, Vol.11, No.3, pp.195-206, 2014.

[3] R. J. Robles and T. H. Kim, "Applying Asymmetric Key Encryption to Secure Internet based SCADA," *International Journal of Internet, Broadcasting and Communication*, Vol.4, No.2, pp.17-21, 2012.

[4] S. Kumar, V. K. Sharma, and K. K. Mahapatra, "Low latency VLSI architecture of S-box for AES encryption," *International Conf. on Circuits, Power and Computing Technologies (ICCPCT)*, Mar. 2013.

[5] Edwin NC Mui, "Practical Implementation of Rijndael S-Box Using Combinational Logic," Custom R&D Engineer Texco Enterprise Pvt. Ltd, 2007.

[6] Y. G. You, S. Y. Kim, Y. D. Kim, and J. S. Park, "Low Power Cryptographic Design based on Circuit Size Reduction," *Journal of the Korea Contents Association*, Vol.11, No.2, pp.92-99, 2007.

[7] J. S. Park, Y. S. Yun, Y. D. Kim, S. W. Yang, T. J. Chang, and Y. G. You, "Design and Implementation of ARIA Cryptic Algorithm," *The Institute of Electronics Engineers of Korea - Semiconductor and Devices*, Vol.42, No.4, pp.29-36, 2005.

[8] Y. H. Song, Y. S. Shin, and J. W. Chang, "Design and Implementation of HDFS Data Encryption Scheme Using ARIA Algorithms on Hadoop," *KIPS Transactions on Computer and Communication Systems*, Vol.5, No.2, pp.33-40, Feb. 2016.

[9] F. Ahmad and Y. C. Jung, "Per-transaction Shared Key Scheme to Improve Security on Smart Payment System,"

*International Journal of Internet, Broadcasting and Communication*, Vol.8, No.1, pp.7-18, 2016.
[10] NSRI, ARIA Test Vectors, 2004.
[11] S. J. Ha, and C. H. Lee, "Design of High Speed Encryption/Decryption Hardware for Block Cipher ARIA," *The Transactions of The Korean Institute of Electrical Engineers*, Vol.57, No.9, pp.1652-1659, 2008.

**강 민 섭**

https://orcid.org/0000-0001-9782-5276
e-mail : mskang@anyang.ac.kr
1979년 광운대학교 전자통신공학과(학사)
1984년 한양대학교 전자공학과(공학석사)
1992년 (일)오사카대학교 전자공학과 (공학박사)
1984년~1993년 한국전자통신연구원 선임연구원
2001년~2002년 University of California, Irvine 전기전자공학과 객원교수
1993년~현 재 안양대학교 컴퓨터공학과 교수
관심분야 : 임베디드 시스템, 네트워크 보안, 영상 보안시스템 설계, ASIC 설계, IOT