128 비트 LEA 암호화 블록 하드웨어 구현 연구

(A Study on Hardware Implementation of 128-bit LEA Encryption Block)

윤기하*, 박성모**
(Gi Ha Yoon, Seong Mo Park)

요 약

본 논문은 사물인터넷 보안용 경량 암호 알고리듬 중, '128비트 블록 암호 LEA'의 암호화 블록 하드웨어 구현에 대해 기술한다. 라운드 함수 블록과 키 스케줄 블록은 높은 처리성능을 위하여 병렬회로로 설계되었다. 암호화 블록은 128비트의 비밀키를 지원하며, FSM 방식과 24/n단계(n = 1, 2, 3, 4, 8, 12) 파이프라인 방식으로 설계되었다. LEA-128 암호화 블록을 Verilog-HDL로 모델링하여 FPGA 상에서 구현하고, 합성결과로부터 최소면적 및 최대 처리성능을 제시한다.

■ 중심어 : 사물인터넷 보안 ; 경량 암호 알고리듬 ; FSM ; 파이프라인 ; FPGA 합성 ;

Abstract

This paper describes hardware implementation of the encryption block of the '128 bit block cipher LEA' among various lightweight encryption algorithms for IoT (Internet of Things) security. Round function blocks and key-schedule blocks are designed by parallel circuits for high throughput. The encryption blocks support secret-key of 128 bits, and are designed by FSM method and 24/n stage(n=1, 2, 3, 4, 8, 12) pipeline methods. The LEA-128 encryption blocks are modeled using Verilog-HDL and implemented on FPGA, and according to the synthesis results, minimum area and maximum throughput are provided.

■ keywords : IoT security ; Lightweight Encryption Algorithm(LEA) ; FSM ; Pipeline ; FPGA synthesis ;

Ⅰ. 서 론

사물인터넷은 인터넷에서 다양한 형태의 데이터를 처리하고 전송하므로, 제 3자의 악의적인 목적에 의한 정보 유출이나 조작과 같은 정보보안 위험에 노출된다. 스마트기기를 이용한 개인 활동정보 패턴을 분석하여 제공하는 서비스[1]나, 유전체 정보를 통해 질병 예측 및 치료와 같은 개인 맞춤의학[2] 등, 개인정보를 활용하는 분야가 점진적으로 증가함에 따라 사물인터넷 환경에서 정보보호는 필수 요소이다. 사용자가 얻고자하는 정보가 점차 늘어감에 따라, 하나의 사물인터넷 단말이 수집, 처리 및 송수신해야 하는 데이터가 점차 증가할 것으로 보인다. 데이터의 암호화는 제 3자에 의해 내용이 노출되거나 의도적으로 손상, 조작하는 행위로부터 정보를 보호할 수 있다[3]. 사물인터넷 환경에서 정보보안은 각종 센서 네트워크, RFID 태그나 스마트카드 활용 분야 등 여러 하드웨어 또는 소프트웨어 환경의 네트워크 및 단말기에 사용되므로, 적은 자원으로 구현가능하고 저전력으로 동작할 수 있는 경량 암호 알고리듬이 요구된다[4].

경량 암호화는 SPN(Substitution-Permutation Network)과 ARX(Addition, Rotation, XOR) 구조 구분으로 나눌 수 있다. SPN 구조의 대표적인 암호화 블록 기술은 AES(Advanced Encryption Standard)로, 128비트 블록 사이즈를 128비트, 192비트 및 256비트의 비밀키를 통

* 학생회원, 전남대학교 전자컴퓨터공학과
** 종신회원, 전남대학교 전자컴퓨터공학과

접수일자 : 2015년 11월 25일
수정일자 : 2015년 12월 10일
게재확정일 : 2015년 12월 11일
교신저자 : 박성모 e-mail : smpark@jnu.ac.kr

해 암호화한다. 저사양 컴퓨팅 환경에서 개발된 AES는 8비트 단위의 연산처리를 기반으로 한다. 이외에도 PRESENT, CLEFIA, HummingBird-2 등 SPN 구조의 다양한 암호화 알고리듬이 존재한다. ARX 구조의 암호화 기술은 국내 표준인 SEED, HIGHT, LEA 및 KATAN, IDEA 등이 있다.

본 논문에서는 지금까지 연구되어온 경량 암호 알고리듬[5] 중, 최근 국내 표준으로 발표된 '128 비트 블록 암호 LEA'를 FPGA 상에서 다양한 구성으로 구현하여 LEA 암호화 알고리듬의 저면적 구현과 높은 처리성능 구현 등, LEA 알고리듬의 다양한 하드웨어 구성에 따른 구현 결과를 제시한다.

Ⅱ. 관련연구

블록암호 LEA(Lightweight Encryption Algorithm)는 128비트 데이터 블록을 암호화하는 알고리듬으로 128, 192, 256 비트의 비밀키(S-Key)를 사용할 수 있으며, 요구되는 안전성 기준에 따라 용도가 구분될 수 있다. LEA는 각 라운드 함수와 라운드 키 함수에서 ARX(Addition, Rotation, XOR) 연산을 기반으로 한 Feistel 유사 구조이며, 충분한 안전성 보장과 동시에 S-Box 사용을 배제하여 경량 구현이 가능하도록 제안되었다[6]. 비밀키는 스케줄함수를 통해 각 라운드에서 사용되는 라운드 키를 생성하며, 각 라운드는 사용되는 비밀키의 크기에 따라 24, 28, 32 라운드의 연산을 수행한다. 전반적인 LEA 동작은 그림 1과 같다.

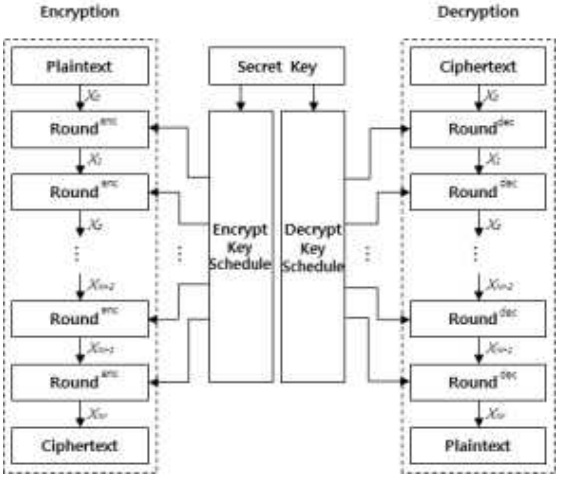

그림 1. LEA 암호 알고리듬

동일한 라운드 연산과 키 스케줄 연산을 정의된 수의 라운드에 거쳐 데이터를 암호화하는데 128비트 비밀키를 사용하는 LEA의 경우 키 스케줄 함수는 그림 2의 연산을 통해 얻을 수 있다.

Key Schedule function

Input : 128bits Secret key

Output : 192bits Round key (RK_0^{enc} to RK_{24}^{enc})

1: T ← K

2: **for** i = 0 **to** 23 **do**

3: T[0] ← ROL_1(T[0] + ROL_i(δ[i **mod** 4]))

4: T[1] ← ROL_3(T[1] + ROL_{i+1}(δ[i **mod** 4]))

5: T[2] ← ROL_6(T[2] + ROL_{i+2}(δ[i **mod** 4]))

6: T[3] ← ROL_{11}(T[3] + ROL_{i+3}(δ[i **mod** 4]))

7: RK_i^{enc} ← (T[0], T[1], T[2], T[1], T[3], T[1])

8: **end for**

그림 2. 128비트 키 스케줄 슈도코드

128비트 크기를 갖는 변수 T는 32비트 단위로 나누어 T[0]부터 T[3]까지 구분되고, ROL(Rotate bit Left)연산과 덧셈연산을 통해 라운드 키를 생성한다. 128비트 비밀키로부터 라운드 키를 생성할 때, 32비트 고유상수 δ[0]~δ[3]값이 사용되는데, 이 값은 순서대로 16진수 "c3efe9db", "44525b02", "79e27c8a", "78df30ec" 값으로 정의되어 있다[6]. 각 라운드 연산은 비밀키의 크기와 무관하게 동일한 연산이 수행되도록 그림 3과 같이 구성되어 있다.

Encryption Round function

Input : Round X_i, 192bits Round key(RK_i^{enc}),
Round Number i

Output : Round X_{i+1}

1: $X_{i+1}[0] \leftarrow ROL_9((X_i[0]$ **xor** $RK_i^{enc}[0]) + (X_i[1]$ **xor** $RK_i^{enc}[1]))$

2: $X_{i+1}[1] \leftarrow ROR_5((X_i[1]$ **xor** $RK_i^{enc}[2]) + (X_i[2]$ **xor** $RK_i^{enc}[3]))$

3: $X_{i+1}[2] \leftarrow ROR_3((X_i[2]$ **xor** $RK_i^{enc}[4]) + (X_i[3]$ **xor** $RK_i^{enc}[5]))$

4: $X_{i+1}[3] \leftarrow X_i[0]$

그림 3. LEA 암호화 라운드 연산 슈도코드

라운드 함수는 키 스케줄을 통해 생성된 라운드 키와 평문(또는 라운드 연산 값)을 32비트 단위 ARX 연산한다. 그림 2와 그림 3에서 볼 수 있듯이, i번째 라운드 연산을 수행하기 위해서는 i번째 라운드 키 연산이 선행되어야 하드웨어 고속 동작에 유리한 것을 알 수 있다.

사물인터넷에서 경량 암호 알고리듬이 요구되고, 국내에서 LEA가 국내표준으로 발표됨으로써 소프트웨어 및 하드웨어 분야에서 경량 암호화 구현 연구가 진행되고 있다. [7]에서는 128비트 LEA 암호화 및 복호화 블록 설계에서 각 라운드 함수와 라운드 키 구현을 32비트 단위의 순차연산이 되도록 구현하였다. 이에 따라, 각각의 라운드 값, 라운드 키를 얻는데 3클럭 주기가 소요되며 최종적으로 단위시간당 처리성능이 약 216.24Mbps를 보이는 것으로 평가하였다. [8]에서는 128비트 LEA 암호화 블록을 6단계 파이프라인으로 설계하고 32비트 단위 입출력 블록을 추가로 구성한 형태로 구현하였다. 이것은 데이터 입력이 32비트 단위로 제한되는 조건에서 최대성능으로 구현결과는 약 6613Mbps의 시간당 처리성능을 제시한다.

Ⅲ. LEA 설계

1. 키 스케줄 및 라운드 연산 블록 설계

[7]에서처럼 키 스케줄 및 각 라운드 연산을 32비드 단위로 구분하고 연산 블록을 공유(Resource Sharing)하여 구현하는 방법은, 하나의 ARX 연산 블록을 재사용하여 결과 값을 도출하므로 저면적 구현에 용이한 반면, 라운드 키 및 각 라운드 연산을 수행하는데 3 클럭 주기가 소요되어, 병렬 처리대비 이론적으로 3배 느린 처리속도를 갖는다. 본 연구는 고속화 구현이 용이하도록 키 스케줄 블록 및 라운드 연산 블록을 병렬로 구성하였다.

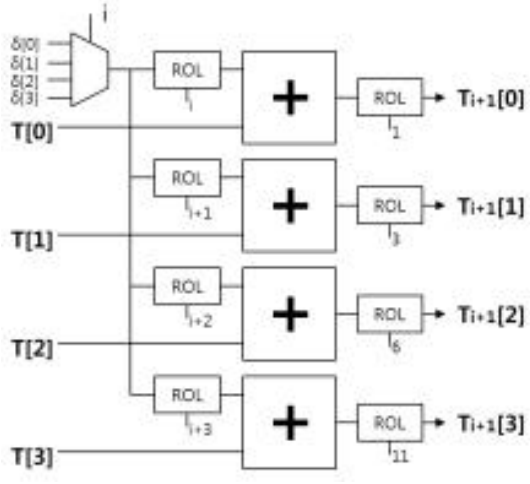

그림 91. LEA-128 암호화 키 스케줄 블록

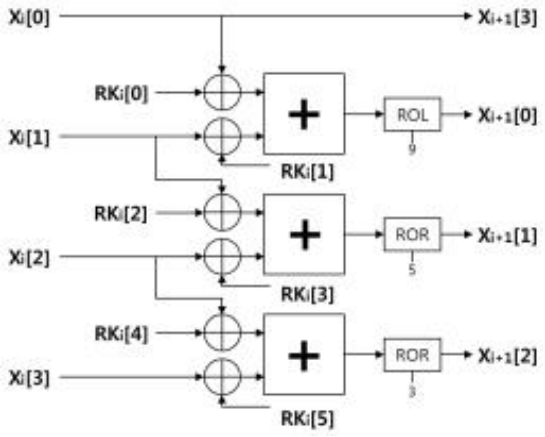

그림 5. LEA-128 암호화 라운드 연산 블록

암호화 키 스케줄 블록은 그림 4에서 보이는 것처럼 입력 데이터를 요구 값만큼 비트순환 후, 출력하는 ROL 블록을 추가 구성하였다. 키 스케줄은 라운드 회차에 따라 입력되는 상수(δ)값이 변경되어야한다. 따라서, 키 스케줄 블록 외부로부터 라운드 회차 값(i)을 입력받아 하위 2비트로 상수 값을 선택적으로 사용하도록 설계하였다. 암호화 라운드 연산 블록은 그림 5과 같은 병렬회로로 설계하였다.

2. 암호화 알고리듬 설계

LEA암호화 알고리듬 전체 블록의 구현은 키 스케줄 블록과 라운드 연산 블록을 구성하는 방법에 따라서, 사용되는 요구자원 및 처리 성능에 차이를 보인다. FSM 구성을 통한 적은 자원(저면적)구현부터 파이프라인 구성을 통해 고속처리 구현이 가능하다. 파이프라인 구현은 파이프라인의 단계에 따라 처리성능과 소요자원에 차이가 발생한다.

가. FSM 설계

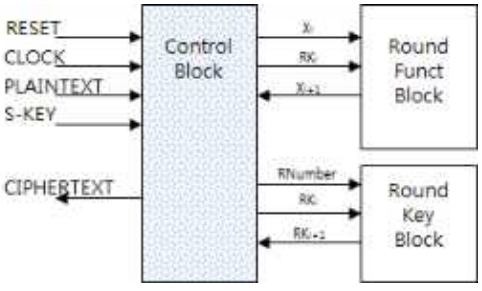

그림 6. FSM 방식의 LEA-128 암호화 블록 설계

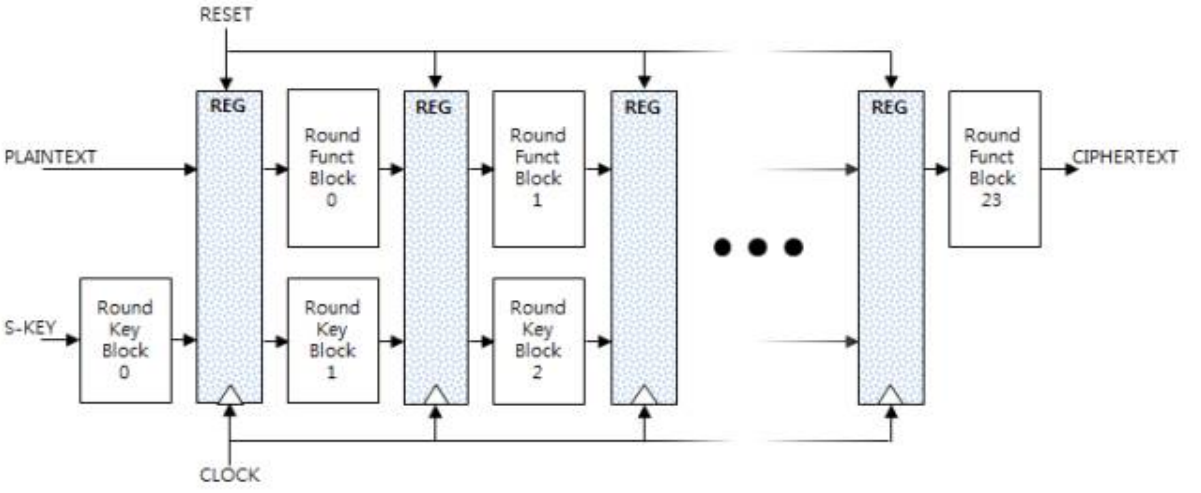

그림 7. 24단계 파이프라인 LEA-128 암호화 블록 설계

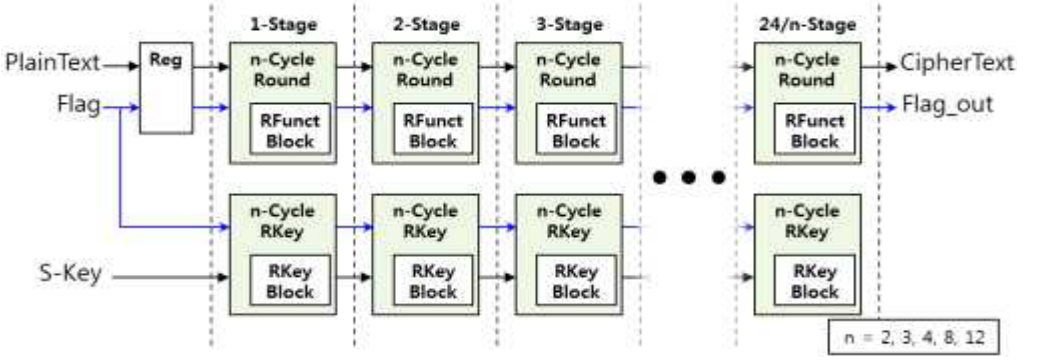

그림 8. 24/n단계(n = 2, 3, 4, 8, 12) 파이프라인 LEA-128 암호화 블록 설계

FSM 설계는 키 스케줄 블록과 라운드 함수 블록을 1개씩 사용하고, 상태 레지스터와 5비트 라운드 카운터, 데이터 레지스터를 추가하여 최소한의 블록을 매회 라운드마다 재사용되도록 설계하였다. 라운드 키가 라운드 함수 연산보다 1클럭 주기 먼저 연산되어 결과 값을 레지스터에 저장하고 있어야 전체 동작속도가 향상되므로 라운드 연산블록이 1클럭 주기 후에 동작되도록 설계하였다.

그림 6과 같이 제어 블록(Control Block)에서 키 스케줄 블록(Round Key Block)에 라운드 회차 값과 비밀키 또는 이전 라운드 키 값을 입력으로 인가하면 해당 라운드에 필요한 라운드 키가 생성되고 이것을 제어 블록에서 레지스터에 저장한 후, 다음 클럭에 라운드 연산 블록(Round Funct Block)에 값을 인가하여 라운드 연산을 수행한다. 이러한 설계 방법은 하나의 평문을 암호화 하는데 매번 24클럭 주기가 요구되며, 암호화 중에 새로운 평문을 입력받아 암호화하는 것이 불가능하다.

나. 파이프라인 설계

전체 라운드 파이프라인 설계는 키 스케줄과 라운드 연산이 고속 처리에 용이하도록 키 스케줄 블록 및 라운드 연산 블록을 라운드 회수만큼 나열하고 블록 간, 레지스터를 거치는 형태로 구성한다. FSM 설계와 마찬가지로 라운드 연산 블록이 1클럭 주기 후, 동작하도록 구성하였다. LEA 암호화 블록을 24단계 파이프라인으로 구성하면 입력받은 평문이 암호화되어 출력되기까지는 24 클럭 주기가 소요되고, 매 클럭 주기마다 새로운 평문을 입력받아 모두 암호화하여, 출력하는 것이 가능하다.

모든 라운드 연산을 파이프라인으로 구성하면 FSM 구현결과 대비 매우 빠른 동작속도 및 처리성능을 기대할 수 있지만, 많은 자원소요가 예상된다. 이에 따라, 본

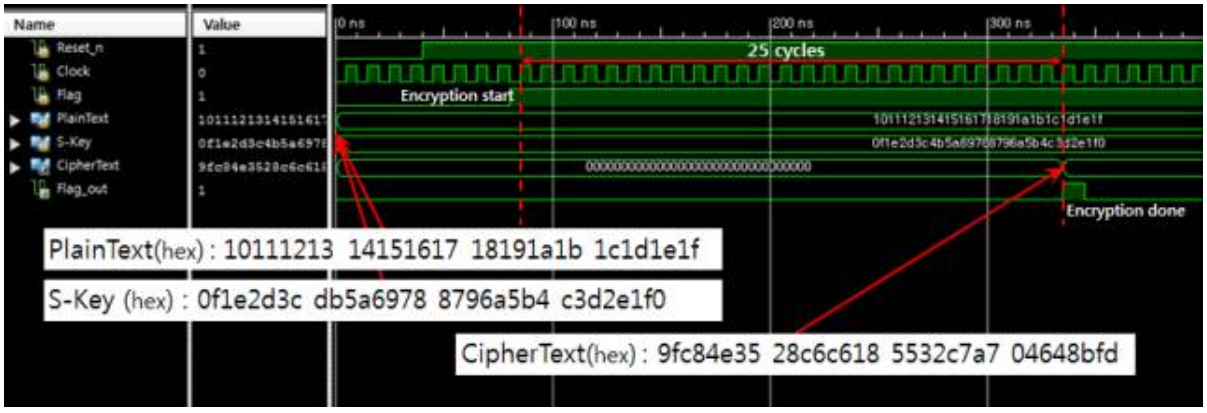

그림 9. FSM 방식의 LEA-128 암호화 블록 시뮬레이션 결과

표 1. 파이프라인 LEA 검증을 위한 Test-vector

Group	Type	Value(Hexadecimal)
Vector-0[6]	PlainText	0111213 14151617 18191a1b 1c1d1e1f
	Secret-Key	0f1e2d3c db5a6978 8796a5b4 c3d2e1f0
	CipherText	**9fc84e35 28c6c618 5532c7a7 04648bfd**
Vector-1	PlainText	f1e1d1c1 b1a19181 71615141 31211101
	Secret-Key	0f1e2d3c db5a6978 8796a5b4 c3d2e1f0
	CipherText	**49c16b50 2603c0d5 b4fedc47 6537c38d**
Vector-2	PlainText	f1e1d1c1 b1a19181 71615141 31211101
	Secret-Key	10111213 14151617 18191a1b 1c1d1e1f
	CipherText	**38bc0a14 3f526b49 d4be7061 9d05d5b9**
Vector-3	PlainText	10111213 14151617 18191a1b 1c1d1e1f
	Secret-Key	1a2b3c4d 5f607182 9304a5b6 c7d8e9f0
	CipherText	**6422356d 934ad860 a0d060ce 6d3fc162**

연구에서는 FSM 방식과 24단계 파이프라인 방식의 LEA 암호화 블록 설계와 더불어, 각 파이프라인 단계마다 처리하는 라운드 수를 증가시킴으로써 파이프라인 단계를 줄인 단계별 파이프라인 설계를 수행하였다. 24단계 파이프라인 LEA 암호화 블록은 그림 7과 같은 구성으로 설계하였다.

24/n단계(n=2, 3, 4, 8, 12) 파이프라인 LEA 설계는 그림 8과 같이 구성되며, 매 키 스케줄 및 라운드 연산 블록에서 n회 라운드를 연산하도록 설계하였다. 이렇게 설계된 24/n단계 파이프라인 LEA는 클럭 주기 24/n회마다 입력되는 데이터 및 비밀키에 의한 암호문을 출력할 수 있다.

하나의 키 스케줄 및 라운드 연산 블록을 레지스터를 통해 2, 3, 4, 8, 12회 재사용하도록 제어하는 중간블록 'n-Cycle RKey'와 'n-Cycle Round'을 설계하고, 이것을 순차적으로 배치하여 각각 12, 8, 6, 3, 2단계 파이프라인 LEA 암호화 알고리듬 블록을 구성할 수 있다. 24단계 파이프라인에서는 키 스케줄 블록의 라운드 회차 값을 매 키 스케줄 블록마다 고정된 정수를 인가하여 생성하지만, 24/n 단계 파이프라인에서는 하나의 키 스케줄 블록이 의도한 라운드 수만큼 연산되도록 제어하는 회로가 추가된다. 라운드 연산블록은 해당 단계에서 생성되는 라운드 키와 평문 또는 암호화 값을 입력 받아 라운드 연산을 수행한다. 정확한 시간축 상에서 라운드 키 생성 및 연산이 수행 될 수 있도록, 각각의 키 스케줄 및 라운드 연산 중간블록에 평문 및 비밀키 입력 외, 입력 유효 신호(Flag)와 출력 유효 신호(Flag_out)를 추가하였으며, 각 중간블록의 입력 유효 신호는 연산이 완

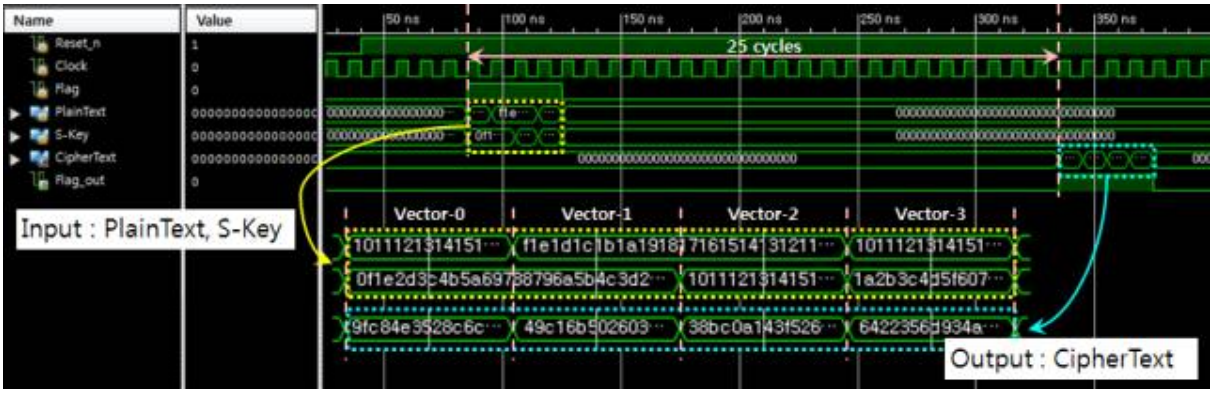

그림 10. 24단계 파이프라인 LEA-128 암호화 블록 시뮬레이션 결과

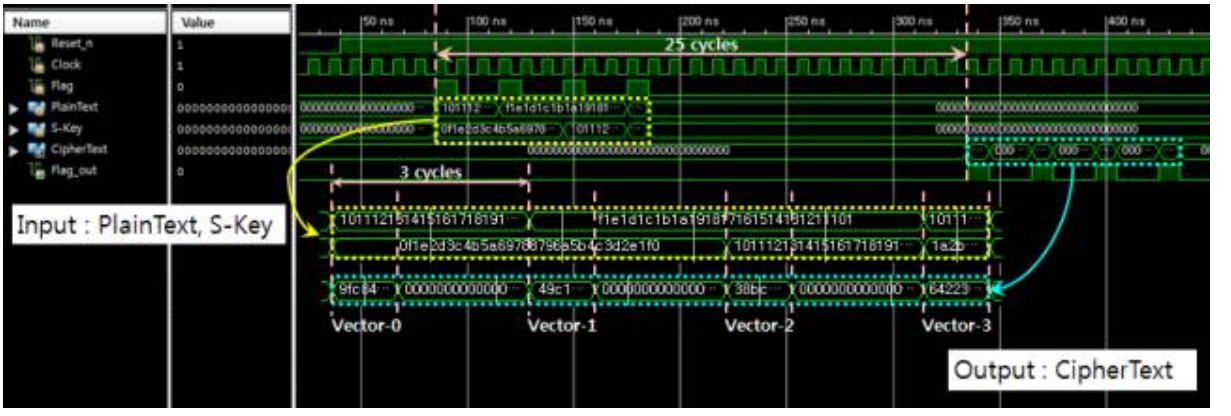

그림 11. 8단계 파이프라인 LEA-128 암호화 블록 시뮬레이션 결과

료되는 시점에 출력 유효 신호가 생성되도록 하였다. 이 유효 신호를 이용하여 각 단계 중간블록의 구동 여부를 결정하도록 설계하였다.

라운드 키 생성은 매회 라운드 적용되는 고정 상수 값이 선택적으로 적용된다. 따라서 키 스케줄 중간블록을 구성할 때, 구성되는 파이프라인 순서에 따라 라운드 회차 정보를 일부 고정 값으로 주고, 중간블록 내부에서 수행되는 라운드 횟수에 따른 변화 값이 반영되도록 설계하였다.

Ⅳ. 합성 및 실험

1. 실험

모든 회로는 Verilog-HDL을 이용하여 Structural Modeling과 Data Flow Modeling 기법을 혼합하여 계층적 구조로 설계하였다. 설계된 LEA-128 암호화 블록의 기능 검증은 한국정보통신기술협회(TTA)에서 제공하는 표준문서[6]에서 16진수 값으로 제시된 128비트 평문 "10111213 14151617 18191a1b 1c1d1e1f"와 비밀키 "0f1e2d3c db5a6978 8796a5b4 c3d2e1f0"을 사용하였으며, 결과인 16진수 암호문 "9fc84e35 28c6c618 5532c7a7 04648bfd"이 그림 9와 같이 출력되어 FSM으로 설계한 LEA-128 암호화 블록의 기능이 정상적으로 동작함을 확인하였다.

24단계 파이프라인 LEA-128 암호화 블록은 매 클럭 주기마다 입력받은 평문과 비밀키를 순차적으로 암호화하여 출력할 수 있으며, 24/n단계(n = 2, 3, 4, 8, 12) 파이프라인 LEA-128 암호화 블록은 키 스케줄 중간블록과 라운드 연산 중간블록에서 입력받은 데이터를 처리하는데 n회 클럭 주기가 소요되므로, n회 클럭 주기마다 입력되는 데이터를 순차적으로 암호화하여 출력할 수 있다. 이에 따라, 각 단계별 파이프라인 LEA-128 암호화 블록의 순차적 암호화기능 검증을 위한 Test-vector를 표 1과 같이 생성하였다.

파이프라인 LEA의 검증을 위한 데이터는 표 1에서 제시한 'Vector 0'부터 'Vector 3' 값을 순차적으로 입력하여, 최초 입력된 평문과 비밀키에서 "평문 변경 → 비

밀키 변경 → 평문&비밀키 변경" 순서로 입력 값에 변화를 주어, 암호문 값을 확인하는 방법으로 시뮬레이션을 수행하였다.

각 단계별 파이프라인으로 설계한 LEA-128 암호화 블록의 최대 처리성능 조건에서 평문과 비밀키가 순차적으로 입력되도록 최초 평문과 비밀키를 입력한 후, 각 단계 라운드 연산 중간블록의 연산 주기마다 입력 값을 변경하는 방법으로 Testbench를 작성하여, 각 단계별 파이프라인 LEA-128 암호화 블록이 최대성능으로 동작되도록 시뮬레이션 환경을 구성하였다.

그림 10에서 보이는 것처럼 24단계 파이프라인 LEA는 매 클럭마다 입력되는 Vector-0~3의 평문과 비밀키 값에 따라 암호문 생성에 필요한(latency) 25클럭 주기 이후, 매 클럭 주기마다 Vector-0부터 Vector-3에 해당하는 암호문이 출력됨을 확인하였다. 8단계 파이프라인의 경우 라운드 연산 중간블록 및 키 스케줄 중간블록 하나당 3회의 라운드를 처리한다. Test-vector는 3클럭 주기마다 입력되고, 그림 11과 같이 25클럭 주기 이후에 각 입력 값에 해당하는 암호문이 순차적으로 출력됨을 확인하였다. 12, 6, 4, 3단계 파이프라인 LEA-128 암호화 블록도 표준문서[6]와 표 1에 제시한 Test-vector를 바탕으로 시뮬레이션을 수행한 결과, 정상적으로 동작함을 확인하였다.

2. FPGA 합성

Verilog-HDL을 이용한 FSM방식의 LEA-128 암호화 블록 및 각 단계별 파이프라인 LEA-128 암호화 블록의 하드웨어는 Xilinx의 합성 소프트웨어 ISE Design Suit (Ver 14.7, WebPack)에서 Xilinx Virtex 5 series(XC5VLX50T)를 타깃으로 구현하였다.

표 2. 구현방법에 따른 LEA-128 암호화 블록 합성결과

Design	Max.Freq (MHz)	Clock cycles	Resource			Throughput (Mbps)
			FFs	LUTs	Slices	
FSM	323.55	25	552	616	775	1656.57
Pipeline2	274.77	12	679	1361	1445	2930.88
Pipeline3	310.04	8	936	1867	1948	4960.64
Pipeline6	317.15	4	1713	3529	3568	10148.8
Pipeline8	323.95	3	2257	4802	4837	13821.87
Pipeline12	333.30	2	3283	6612	6655	21331.2
Pipeline24	418.69	1	6426	7737	8819	53592.32

* Target device : Virtex5 XC5VLX50T

표 2와 같이 각각 하나의 라운드 연산 블록과 키 스케줄 블록을 재사용하도록 구현하는 FSM 방식의 LEA-128 암호화 블록이 모든 라운드마다 각각 라운드 연산 블록과 키 스케줄 블록을 사용하는 24단계 파이프라인 LEA-128 암호화 블록 대비, 소요자원(Slices 기준)은 약 8.79%, 처리성능(Throughput)은 약 3.22% 수준으로 구현되었다. 대부분의 LEA-128 암호화 블록이 300MHz 이상의 주파수 영역에서 동작되도록 구현되었으며, 2단계 파이프라인 LEA-128 암호화 블록에서 제어회로 중, 키 스케줄 부분에서 일부 지연요소에 의해 275MHz 수준의 합성결과를 보였다.

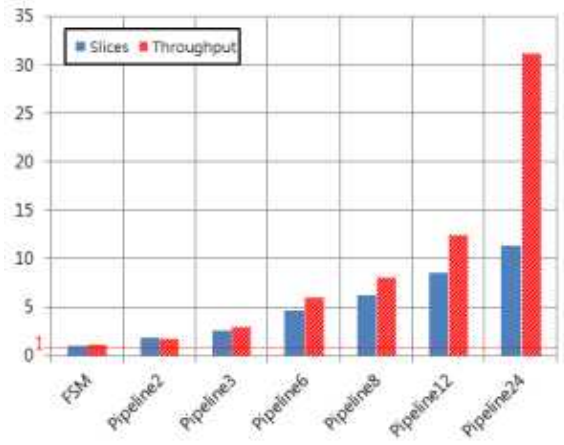

그림 12. LEA-128 암호화 블록별 소요자원 및 처리성능

그림 12는 FSM 구현 방식의 LEA-128 암호화 블록을 기준으로 각 단계별 파이프라인 LEA-128 암호화 블록의 소요자원 및 처리성능 비율을 나타낸 것이다. Xilinx FPGA에서는 LUT(Look Up Table)에 조합회로(Combinational logic)와 순차회로(Sequential logic)가 합쳐진 형태로 구현되며 이것을 슬라이스(Slices)로 표현하고 있어, 그림 12에서 나타낸 소요자원을 슬라이스로 정의하고 구성방식에 따른 소요자원의 비교 값을 도출하였다. 그림 12에서 보이는 것처럼 24단계 파이프라인 LEA-128 암호화 블록이 12단계 파이프라인 LEA-128 암호화 블록보다 소요자원은 약 1.33배 증가한 반면, 처리성능은 약 2.51배 높게 구현되었다. 각 라운드를 제어하는 회로를 생략하고 매회 라운드를 독립된 키 스케줄 블록과 라운드 연산블록으로 처리하여, 24단계 파이프라인 LEA-128 암호화 블록의 성능이 비약적으로 높게 구현된 것으로 보이며, 많은 레지스터가 공통된 LUT에 포함된 형태로 구현되어 소요자원은 적은 차이를 보인 것으로 사료된다.

V. 결 론

본 논문에서는 다양한 경량 암호화 알고리듬 중, 국내 한국정보통신기술협회(TTA) 표준인 128비트 LEA 암호화 알고리듬의 주요 블록인 키 스케줄 및 라운드 연산 블록 설계하고 암호화 블록을 FSM 방식과 2, 3, 6, 8, 12 및 24단계 파이프라인 방식의 다양한 구성으로 설계하였다. 아울러, FPGA상에서 다양한 구성으로 구현된 128비트 LEA 암호화 블록의 최소자원(저면적) 구현 및 최대성능 구현에 대한 성능지표를 제시하였다. FPGA 상에서 LEA-128 암호화 블록은 최소 775 슬라이스의 적은 자원으로 구현하였을 때, 약 1656Mbps의 처리성능을 보이고, 24단계 파이프라인 LEA 암호화 블록은 8819 슬라이스의 자원이 소요되며 최대 53Gbps의 높은 시간당 처리량을 갖는다. 따라서, 50Gbps급의 대용량 데이터를 요구하는 다양한 응용분야에도 활용될 수 있을 것으로 보인다.

'본 논문은 IDEC의 EDA Tool에서 지원하여 수행하였음.'

참고 문헌

[1] M.I. Joo, G.S. Chung and H.C. Kim, "Implementation of a system to analyze user behavior patterns based on vital signs and user locations" *Smart Media Journal*, Vol. 3, No. 4, pp. 35-40, Dec, 2014

[2] D.M. Kim, H.Y. Jeong, I.C. Kim and Y.G. Won, "Individual Genome Sequences and Their Smart Application in Personalized Medicine", *Smart Media Journal*, Vol. 2, No. 4, pp. 34-40, Dec, 2013

[3] 신동희, 정재열, 강성현, "사물인터넷 동향과 전망" *인터넷정보학회지*, 제14권, 제2호, 32-46쪽, 2013년 6월

[4] T. Eisenbarth, C. Paar, A. Poschmann, S. Kumar and L. Uhsadel, "A Survey of Lightweight-Crypography Implementations", *IEEE Design&Test of Computers,* vol. 24, No. 06, pp. 522-533, Nov-Dec, 2007

[5] 서화정, 김호원, "사물인터넷을 위한 경량 암호 알고리즘 구현", *정보보호학회지*, 제25권, 제2호, 12-19쪽, 2015년 4월

[6] Telecommunications Technology Association, "128-Bit Block Cipher LEA", *TTA Standard*, TTAK.KO-12.0223, 2013.

[7] M.J. Sung and K.W. Shin, "A Small-area Hardware Design of 128-bit Lightweight Encryption Algorithm LEA", *Journal of the Korea Institute of Information and Communication Engineering*, Vol. 19, No. 4, pp. 888-894, Apr, 2015

[8] 이철, 고성학, 방능수, "LEA 파이프라인 구조 설계", *한국정보과학회 학술발표논문집*, 739-740쪽, 한국, 2014년 12월.

저 자 소 개

윤기하 (학생회원)

2012년 목포대학교 정보통신공학과 학사 졸업.
2014년 전남대학교 전자컴퓨터공학과 석사 과정.
〈주관심분야 : SoC 설계, 정보보호/통신 반도체 IP 설계〉

박성모 (종신회원)

1977년 서울대학교 전자공학과 학사
1979년 한국과학기술원 전기 및 전자공학과 석사
1988년 노스캐롤라이나 주립대학 전기 및 컴퓨터공학과 공학박사
1979년~1984년 한국전자기술연구소 설계개발부 선임연구원
1988년~1992년 올드도미니언대학교 전기 및 컴퓨터공학과 조교수
1992년~현재 전남대학교 전자컴퓨터공학부 교수
〈주관심분야 : 멀티미디어 프로세서 구조, SoC 설계, 영상압축, 임베디드 시스템 등〉