LEA 암·복호화 블록 파이프라인 구현 연구

(A Study on Pipeline Implementation of LEA Encryption·Decryption Block)

윤기하*, 박성모**

(Gi Ha Yoon, Seong Mo Park)

요 약

본 논문은 사물인터넷 환경의 초소형 기기에서 사용될 수 있는 경량 블록암호 알고리듬인 LEA의 암호화 및 복호화 블록의 하드웨어 구현에 관한 연구이다. 128비트, 192비트 및 256비트 크기의 모든 비밀키를 수용하고 암·복호화 기능을 통합한 구현을 목표로 하며, 성능향상을 위해 파이프라인 기법을 적용한 설계 결과를 제시한다. 복호화 기능을 실행할 때, 라운드키가 암호화 기능의 역순으로 사용되는데, 이때 발생되는 성능저하를 최소화한 효율적인 하드웨어 구현방법을 제시한다. 비밀키 크기에 따라 라운드 횟수가 24, 28 또는 32회 동작함을 고려하여, LEA 파이프라인은 매 파이프라인 단계에서 4번의 라운드 함수 연산이 수행되도록 구현하였다.

■ 중심어 : 경량 블록암호 알고리듬 ; 사물인터넷 ; FPGA ; 키스케줄 ; 파이프라인

Abstract

This paper is a study on the hardware implementation of the encryption and decryption block of the lightweight block cipher algorithm LEA which can be used for tiny devices in IoT environment. It accepts all secret keys with 128 bit, 192 bit, and 256 bit sizes and aims at the integrated implementation of encryption and decryption functions. It describes design results of applying pipeline method for performance enhancement. When a decryption function is executed, round keys are used in reverse order of encryption function. An efficient hardware implementation method for minimizing performance degradation are suggested. Considering the number of rounds are 24, 28, or 32 times according to the size of secret keys, pipeline of LEA is implemented so that 4 round function operations are executed in each pipeline stage.

■ keywords : Lightweight Encryption Algorithm(LEA) ; Internet of Things(IoT) ; FPGA ; Key schedule ; Pipeline

Ⅰ. 서 론

최근에 사물인터넷(IoT: Internet of Things) 산업이 급성장하고 있다. 인터넷 연결을 생각하지 못했던 다양한 사물들이 인터넷과 연결되고 있으며, 사물 간 상호 연결이 심화될수록 사이버보안 위협이 현실세계로 확산될 수 있다. 특히, 인체와 밀접하게 연결되는 IoT 기기들은 보안 사고가 인명 사고로 이어질 수 있어 서비스 활성화와 보안을 동시에 고려해야 한다. 사물인터넷이 적용될 수 있는 모든 제품 및 서비스에 보안이 필요한데, 단말 계층에서의 보안을 위한 저전력 경량암호 알고리듬의 하드웨어 구현에 대한 연구가 활발하게 이루어지고 있다.[1] 다양한 암호화 알고리듬 중, 비밀키 암호방식은 암호화 또는 복호화에 동일한 비밀키를 사용하는 방식으로 대칭키 암호(Symmetric-Key Encipherment)방식이라고도 불리며, 구현 방식이 간단하고 동작이 빠른 장점을 갖는다. 대표적인 알고리듬으로 AES(Advanced Encryption Standard)를 비롯하여 SEED, HIGHT(HIGh Security and light weigHT) 및 LEA 등 다양한 알고리듬이 존재한다.[2]

본 논문은 국내 국가보안기술연구소(NSRI)가 개발하여 사물인터넷 환경에서 적은 메모리와 저효율 CPU의 초소형 기기에서도 데이터를 고속으로 암호화할 수 있는 경량 블록암호 알고리듬인 LEA의 암호화 및 복호화 블록 하드웨어 구현에 관한 연구이다. 다양한 비밀키 크기를 모두 수용하고 암·복호화 기능을 통합한 구현을 목표로 하여, 각 기능블록 단위 설계방법과 성능향상을 위해 파이프라인 기법을 적용한 설계방법을 제시

* 정회원, 한국전자통신연구원 호남권연구센터
** 종신회원, 전남대학교 전자컴퓨터공학부

접수일자 : 2017년 07월 07일
수정일자 : 2017년 08월 08일
게재확정일 : 2017년 09월 13일
교신저자 : 박성모 e-mail : smpark@jnu.ac.kr

한다. 복호화 기능에서 순차적으로 생성되는 라운드 키 중, 마지막 라운드 키를 최초 라운드 연산에 사용하여 발생되는 성능저하를 최소화한 효율적인 하드웨어 구현방법을 제시한다.

Ⅱ. 본 론

1. LEA 암호화 및 복호화 연산 블록

LEA(Lightweight Encryption Algorithm)는 128비트의 평문을 128, 192 및 256비트의 비밀키를 통해 암호화하는 알고리듬으로 요구되는 안전성 기준에 따라 비밀키를 선택적으로 사용할 수 있다. LEA의 모든 연산처리는 32비트 단위의 ARX(Addition, Rotation, XOR) 연산을 기반으로 충분한 안전성을 보장하며 S-Box를 배제하여 경량 구현이 가능하도록 구성되었다.[3] 그림 1은 LEA의 전반적인 동작을 나타낸다.

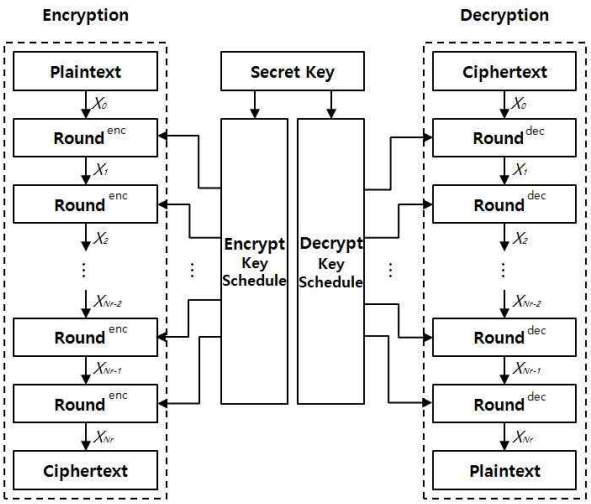

그림 1. LEA 알고리즘

그림 1에서 볼 수 있듯이, LEA의 암·복호화는 키스케줄(Key Schedule)로부터 얻어지는 라운드 키(RK_{Nr})와 입력받은 데이터(X_{Nr})를 정해진 수의 라운드(Round) 연산을 통해 얻을 수 있다. LEA 암호화 및 복호화 라운드 함수는 비밀키(Secret Key)의 크기와 상관없이 동일한 연산을 비밀키의 크기에 따라 정해진 수 만큼 반복한다. 32비트 입·출력 기준의 암호화 라운드 연산은 그림 2와 같다. 저면적 설계를 위하여 하나의 ARX 연산 블록을 재사용하도록 하드웨어를 구현하면 1회 라운드 함수 연산에 3클록 주기가 소요되고, 필요한 연산을 모두 하드웨어로 구현하여 입력된 평문 또는 이전 라운드 함수 연산결과 값을 병렬로 처리하면 1회 라운드 함수 연산을 1클록 주기에 수행할 수 있다.

LEA 복호화 라운드는 암호화 연산의 역산으로 그림 3과 같이 구성되며, 암호화 라운드 함수 기능블록처럼 병렬회로로

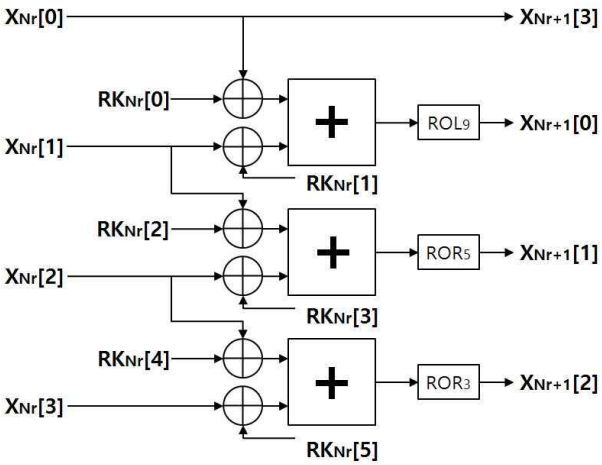

그림 2. LEA 암호화 라운드 연산 블록다이어그램

구현 가능하다. 복호화 라운드 함수 연산은 병렬구성 시, 입력부터 출력까지의 데이터경로가 암호화 라운드 함수 기능블록보다 복잡하여 구현결과의 동작속도는 다소 낮을 것으로 예상된다.

LEA 복호화 동작에서 사용되는 라운드키(RK_{Nd})는 암호화 라운드키의 역순으로 대응되어 마지막 라운드키가 최초 라운드 연산에 사용된다.

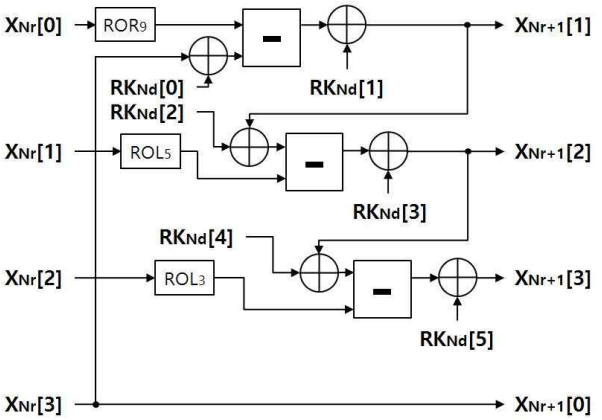

그림 3. LEA 복호화 라운드 연산 블록다이어그램

128비트 비밀키의 LEA 복호화 동작에서 진행라운드(Nr)에 따른 복호화 라운드키(RK_{Nd})는 식(1)과 같이 정의된다.

$$RK_{Nd} = RK_{(23-Nr)} \quad (Nr = 0 \text{ to } 23) \qquad (1)$$

LEA 암·복호화는 비밀키 크기(128비트, 192비트, 256 비트)에 따라 라운드 수가 정해지며, 각각 24회, 28회, 32회 라운드 함수 연산이 반복되며, 비밀키 크기에 따른 키스케줄 연산블록 구성은 그림 4와 같다.

라운드키는 비밀키와 8개의 32비트 상수(δ_i)의 연산을 통해 얻을 수 있다. 비밀키의 크기가 128비트일 때, 4개(δ_0~δ_3)의

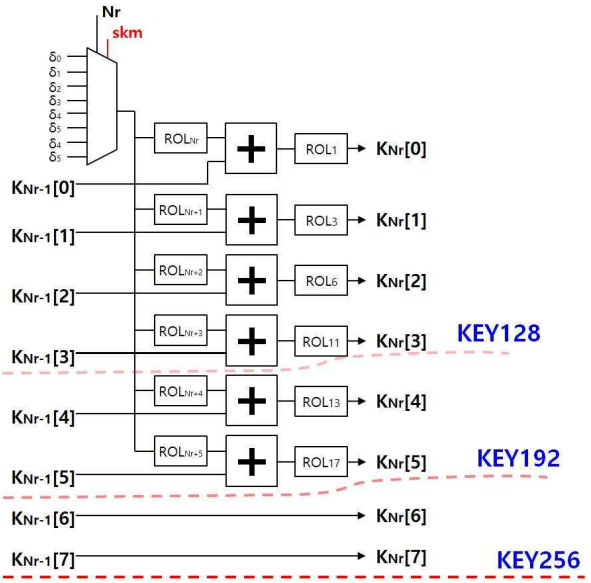

그림 4. LEA 키스케줄 연산 블록다이어그램

상수를 라운드키 연산에 사용하고, 192비트일 때 6개(δ_0~δ_5). 256비트일 때는 모든 상수(δ_0~δ_7)를 사용하며, 키스케줄을 통해 생성된 라운드키는 비밀키의 크기와 상관없이 고정된 192비트의 크기로 생성된다[3]. 그림 4의 LEA 키스케줄 연산 블록다이어그램에서 비밀키의 크기가 192, 256비트인 경우 $K_{Nr}[0 \sim 5]$의 값을 라운드키로 사용하며, 128비트의 비밀키에 의해 생성된 라운드키는 식(2)와 같이 192비트로 확장하여 사용한다.

$$RK_i = \{K_{Nr}[0], K_{Nr}[1], K_{Nr}[2], K_{Nr}[1], K_{Nr}[3], K_{Nr}[1]\} \quad (2)$$

기존의 LEA 하드웨어 구현 및 설계에 대한 연구는 FPGA 기반의 저전력, 저면적 구현연구[4]와, 6단계 파이프라인 구성을 적용하여 6.4Gbps의 처리량(throughput)을 갖는 파이프라인 LEA 암호화 블록 구현연구[5], 나아가 암호화 LEA의 저면적 및 구분된 단계별 파이프라인 구현[6]까지 다양한 연구가 진행되었다. 특히, FPGA기반(Xilinx Virtex5 시리즈)의 LEA 암호화 블록의 설계 및 구현은 128비트 비밀키로 동작하는 LEA 암호화 블록의 경우 파이프라인 단계에 따라 1.6Gbps부터 최대 24단 파이프라인일 때 53Gbps의 처리량을 가질 수 있는 것으로 확인되었다.[6]

2. LEA 기능 블록 설계

가. LEA 라운드 함수

LEA 암호화 라운드 함수 기능블록은 제어블록을 따로 설계할 것을 고려하여, 그림 5와 같이 조합 논리 회로만의 구성으로 설계하였다.

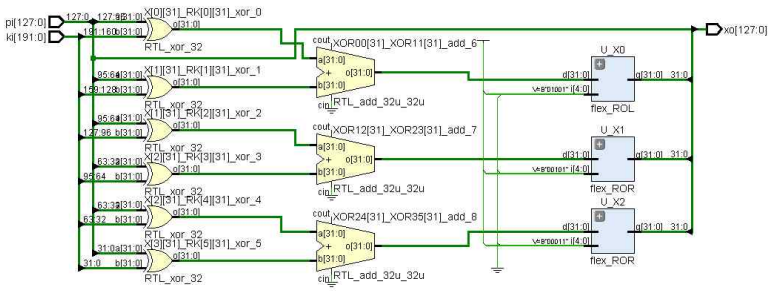

그림 5. LEA 암호화 라운드 함수 기능블록

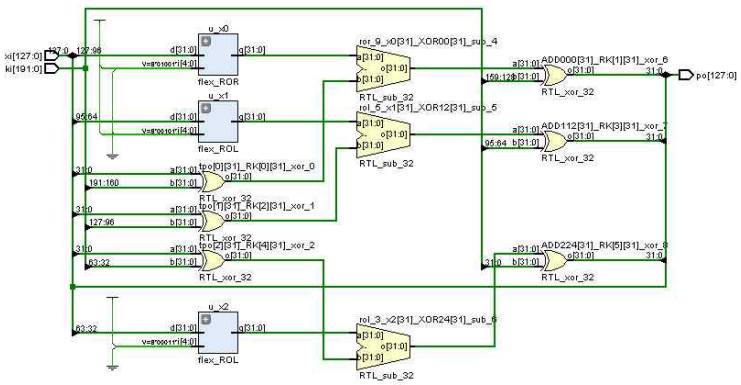

그림 6. LEA 복호화 라운드 함수 기능블록

복호화 라운드 함수 기능블록을 암호화 라운드 함수 기능블록과 같이 병렬회로로 설계하였으며, 그림 6과 같은 구성으로 설계하였다.

LEA의 암호화 라운드 함수 기능블록 및 복호화 라운드 함수 기능블록은 Xilinx사의 ISE 14.7에서 기본설정으로 합성할 때, 소요자원이 유사하게 구현되지만 데이터 경로 구성의 차이로 동작성능이 표 1과 같은 차이를 보인다.

표 1. LEA 라운드 함수 기능블록 합성결과

Block	Slices	Max. Path delay (ns)	Frequency (MHz)
ENC Round	189	5.061	197.589
DEC Round	192	7.509	133.174

LEA 암·복호화 라운드 함수의 통합 설계는 암호화 및 복호화 동작에 따라 연산순서를 변경하는 구조로 설계하였다. 1비트의 모드 선택 신호(암·복호화 구분) 처리를 위한 다수의 멀티플렉서(MUX)를 추가 구성하여 설계하였다. LEA 암호화 및 복호화 라운드 함수 연산 통합 시, 많은 MUX 연산 블록을 사용하여 암호화 및 복호화 라운드 함수 기능블록을 독립적으로 설계했을 때보다 효율은 낮아진다.

나. LEA 키스케줄

LEA 키스케줄은 LEA 비밀키의 크기에 따른 연산을 고려하여, 암호화 키스케줄 블록 및 암·복호화 키스케줄 블록으로 구분하여 설계하였다. 현재 라운드 횟수 정보를 매회 라운드에 사용되는 상수의 순환 횟수를 결정하도록 설계하였고, 비밀 키 크기에 따른 키스케줄 블록 및 암호화 전용 통합 키스케줄 블록 합성 결과는 표 2와 같다.

표 2. 비밀키에 따른 암호화 키스케줄 블록 합성결과

Block	Slices	Max. Path delay (ns)	Frequency (MHz)
ENC-KEY128	142	6.185	161.681
ENC-KEY192	192	5.325	187.793
ENC-KEY256	228	6.266	159.591
ENC-KEY-ALL	818	6.754	148.060

LEA 복호화 라운드키는 암호화 라운드키를 역순으로 사용한다. 이에 따라, 암호화 기준 마지막 라운드키 생성 후, 이전 라운드키를 생성하기 위한 역산이 필요하다. 따라서, LEA 복호화 키스케줄 블록은 LEA 암호화 키스케줄 블록에서 역산에 필요한 우측순환연산(ROR_n) 블록을 추가하여 구성한다. 암호화 또는 복호화 기능을 결정하는 1비트의 신호는 입력되는 비밀키의 크기에 비례하여 더 많은 MUX 블록이 소요되며, 요구되는 소요자원도 증가하게 된다. 암·복호화 키스케줄 블록 설계는 표 3과 같은 합성결과를 보인다.

표 3. 비밀키에 따른 암·복호화 키스케줄 블록 합성결과

Block	Slices	Max. Path delay (ns)	Frequency (MHz)
DEC-KEY128	420	6.774	147.623
DEC-KEY192	602	6.371	156.961
DEC-KEY256	690	6.854	145.900
DEC-KEY-ALL	1616	7.514	133.085

3. 암·복호화 파이프라인 LEA 설계

기존의 파이프라인 설계에서는 복호화 동작을 고려하지 않아, LEA 복호화 동작이 불가한 형태로 설계되었으며[5, 6], 128, 192 및 256비트의 비밀키 크기를 모두 수용한 암·복호화 LEA에 대한 연구는 진행되지 않았다.

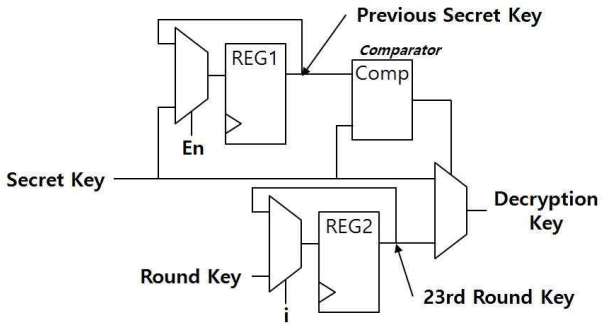

그림 7. 비밀키 변경 감지회로

LEA 복호화 동작은 첫 번째 복호화 라운드키 생성을 위한 추가 연산시간이 요구되며, 파이프라인을 사용하지 않고 FSM(Finite State Machine)기법을 통한 설계 기준의 암호화 동작대비 약 2배의 클록주기가 소요된다. 일반적으로 단일 비밀키를 사용하여 복수의 암호화된 데이터를 복호화하는 경우가 대부분임을 고려할 때, 이와 같은 복호화 동작의 성능저하 요인은 그림 7과 같은 비밀키 변경감지 회로를 적용하여 개선할 수 있다. 그림 7은 128비트 비밀키를 사용하는 경우로, 순방향 라운드키 생성시간의 최소화를 위해, 비밀키 변경 시 24번째 생성되는 라운드키를 레지스터에 저장하는 회로를 추가하고, 복호화 동작 시에 비밀키 변경 감지회로를 활용하여 비밀키가 유지되는 동안에는 저장된 라운드키를 사용하여 첫회 복호화 라운드연산에 필요한 라운드키 연산 시간(선 처리 시간)을 줄일 수 있다.

암호화 및 복호화 라운드 연산 블록을 동일한 단계를 거치는 파이프라인으로 구성해도 복호화 동작은 라운드키의 선처리 단계에 의해 더 낮은 처리성능을 보일 것이다. 이와 같은 단점을 보완하기 위해 FSM 기법 설계에서 도입했던 비밀키 변경 감지 블록을 추가하여 복호화 동작 중, 라운드키 선처리 과정은 비밀키가 유지되면 기존 비밀키에 의해 생성된 복호화용 첫 번째 라운드키를 사용하도록 구성하였다. 비밀키의 크기에 따라 라운드 횟수가 달라지므로 하나의 LEA 파이프라인 단계(LPS)에서 4번의 라운드 함수 연산을 수행하도록 설계하면 비교적 단순하게 통합된 암·복호화 파이프라인 LEA 블록을 그림 8과 같이 구성할 수 있다. 이와 같이 파이프라인 단계를 구분하면 128비트 비밀키 동작에서는 암·복호 데이터를 6번째 LEA 파이프라인 단계에서 얻고 192비트 및 256비트 비밀키 동작은 각각 7번째, 8번째 파이프라인 단계에서 암호화 또는 복호화 데이터를 얻을 수 있다. 이와 같은 구성의 암호화 파이프라인 LEA 블록은 비밀키의 크기에 따라 암호화된 데이터를 24, 28 또는 32클록주기 이후 출력하며, 연속된 데이터는 4클록주기마다 처리할 수 있다.

적용된 LEA 비밀키 변경감지회로(SKD_PIPE)는 파이프라인 LEA의 앞단에서 비밀키가 변경될 때, FSM으로 설계한 독립적인 키스케줄 블록을 통해 첫 번째 복호화 라운드키를 생성한

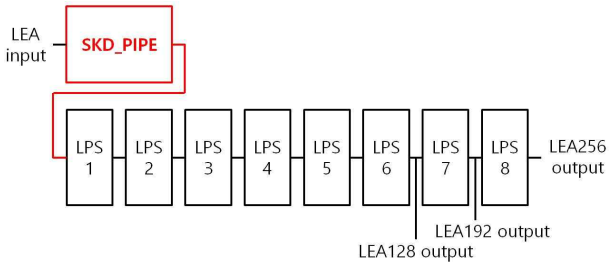

그림 8. 암·복호화 파이프라인 LEA 블록

뒤, 암·복호화 파이프라인 LEA를 구동시킨다. 비밀키 변경 시, 마지막 라운드키 생성은 비밀키 크기에 따라 128비트인 경우 25클록 주기, 192비트 및 256비트인 경우 각각 29, 33클록주기에 거쳐 동작된다. 따라서, 비밀키가 변경되지 않는 상황에서는 첫 번째 복호화 라운드키 생성을 위한 수십 클록주기의 연산을 생략하는 효과를 기대할 수 있다.

본 논문에서 설계한 암·복호화 파이프라인 LEA 블록의 동작은 키변경 감지, 마지막 라운드키 생성 및 암·복호화 라운드 연산의 과정(NR_{func})이 수행되어, 동일한 비밀키를 사용하는 경우에 NR_{func} +1 클록주기가 소요되고, 비밀키를 변경했을 경우는 최대 2배의 클록주기 $2*NR_{func}$ + 2가 소요된다.

위와 같은 구성의 암·복호화 파이프라인 LEA는 FPGA(Xilinx Virtex5 시리즈)에서 합성 시, 표 4와 같은 결과를 보인다.

표 4. 암·복호화 파이프라인 LEA 합성결과

Block	Frequency (MHz)	Slices	Max. Throughput (Mbps)
$LEA_{PIPE}128$	133.223	7651	4263.456
$LEA_{PIPE}192$	133.209	11581	4262.688
$LEA_{PIPE}256$	124.900	16204	3996.800
$LEA_{PIPE}ALL$	122.773	16033	3928.740

$LEA_{PIPE}128/192/256$은 그림 8의 구성에서 비밀키 크기에 맞추어 비밀키 설정 값(key-mode)을 고정한 형태로 합성한 결과이며 $LEA_{PIPE}ALL$은 비밀키의 3가지 크기에 대한 동작을 모두 포함하는 형태로 합성한 결과이다. 대체적으로 비밀키 관련 연산 블록이 많아짐에 따라, 소요 면적(Slices)은 증가하고 동작속도는 감소하는 형태를 보인다. 현재 합성결과는 $LEA_{PIPE}ALL$을 기준으로 최적화되었다는 점을 감안하면, 각 비밀키 크기별로 설계를 분리하여 합성할 경우 소요자원 부분에서 보다 개선된 결과를 얻을 수 있을 것으로 추정된다.

4. 시뮬레이션 및 검증

본 연구에서 제안한 LEA 비밀키 변경감지회로를 적용한 암·복호화 파이프라인 LEA는 비밀키 크기가 연속적으로 변경되는 암호화를 그림 9와 같이 정상적으로 수행하는 것을 시뮬레이션을 통하여 확인하였다.

암·복호화 파이프라인 LEA 블록의 암호화 동작은 파이프라인 블록을 구동함과 더불어 독립적으로 설계된 비밀키 변경 감지 블록에서 마지막 라운드키 생성이 동시에 동작되도록 하였으며 암호화 수행 직 후, 복호화를 요구하는 동작을 수행하도록 시뮬레이션을 구성하였다. 암·복호화 파이프라인 LEA의 128비트 비밀키 동작에서 동일한 비밀키로 고정한 복호화는 그림 10과 같이 정상적으로 동작하는 것을 확인하였다.

암·복호화 파이프라인 LEA 복호화 동작은 최초 비밀키 인가 또는 비밀키 변경에 의한 첫 번째 복호화 라운드키 생성과정이 후에는, 암호화 동작대비 비밀키 변경 감지를 위한 1클록주기만 더 소요되는 것을 그림 10에서 확인할 수 있다.

그림 11은 비밀키 크기에 따른 LEA 암·복호화 복합 동작에

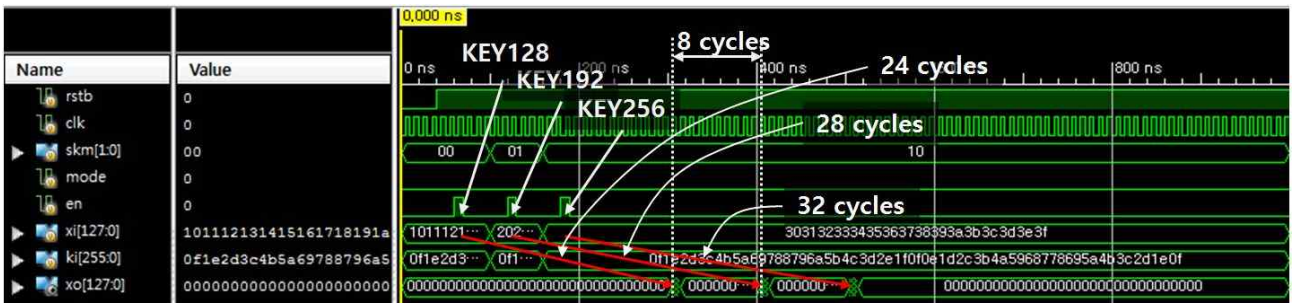

그림 9. 암·복호화 파이프라인 LEA의 연속 데이터 암호화 동작 시뮬레이션

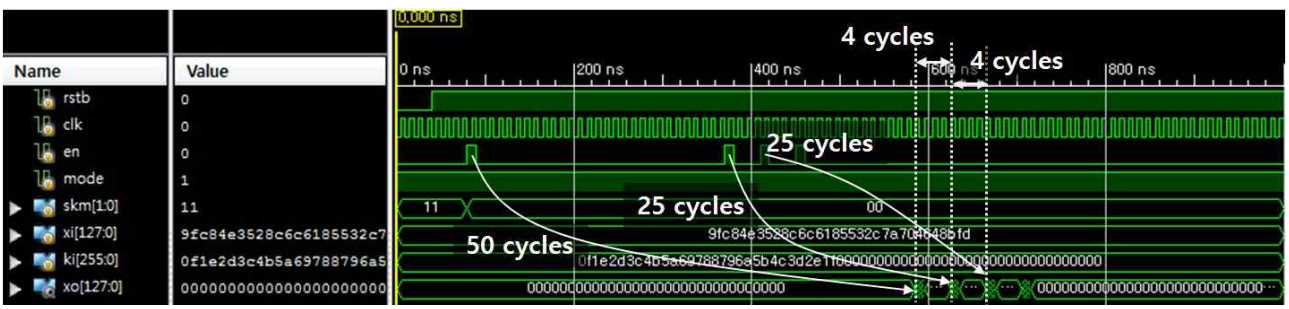

그림 10. 암·복호화 파이프라인 LEA의 128비트 비밀키 복호화 동작 시뮬레이션

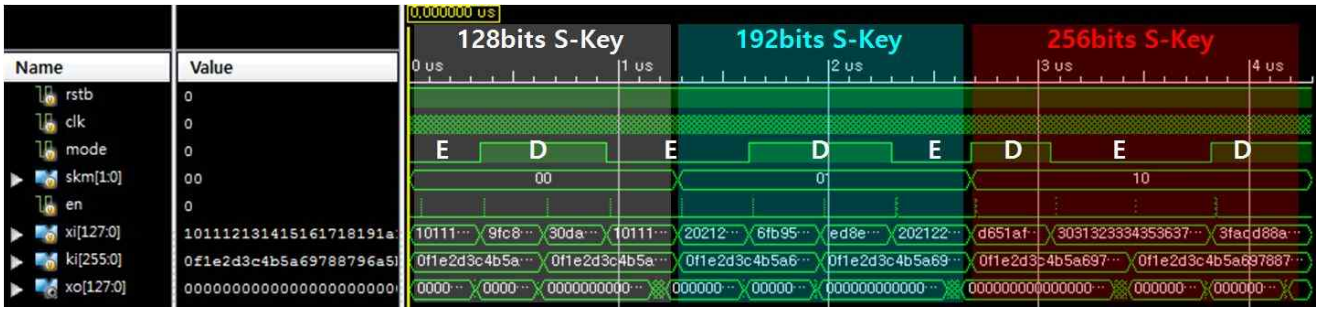

그림 11. 암·복호화 파이프라인 LEA의 복합 동작 시뮬레이션

대한 시뮬레이션을 나타내며, 비밀키 변경 감지회로의 동작을 포함한 모든 LEA 암·복호화 동작이 정상적으로 동작됨을 확인하였다.

III. 결론

본 논문에서는 경량 블록암호 알고리듬인 LEA를 비밀키 크기별 암호화 및 암·복호화 기능으로 구분하여 하드웨어로 설계하였다. 128비트, 192비트, 256비트 비밀키 크기를 모두 수용하고 암·복호화 기능을 통합하여 설계하고 성능향상을 위해 파이프라인 기법을 적용하여 구현하였다. 마지막 라운드키를 생성하는 선처리 과정이 필요한 복호화 동작은 비밀키 변경 감지 회로를 적용하여 성능저하를 개선하였다. 연속된 복호화 연산처리는 암호화 처리성능과 동일한 수준의 결과를 보였으며, 해당 기능을 적용한 LEA 암·복호화 블록이 정상동작 되는 것을 시뮬레이션을 통하여 검증하였다. 파이프라인 LEA 구현은 비밀키 크기에 따라 라운드 횟수가 24, 28 또는 32회 동작함을 고려하여 매 파이프라인 단계에서 4번의 라운드 함수 연산이 수행되도록 구현하는 것이 모든 비밀키 크기를 수용한 파이프라인 LEA를 설계하는데 효율적인 방안으로 보인다. 다만, 복호화 과정을 고려하면 라운드키에 대한 선처리 과정 블록이 추가로 요구되므로 암호화 동작만 구현된 LEA와 비교 시, 동작속도 저하 및 자원소요량 증가의 요인이 된다. 파이프라인 LEA 설계는 1Gbps 이상의 높은 처리성능의 암호화 기능을 요구하는 응용분야에 활용하는 것이 바람직하다. 본 논문에서 기술한 LEA 암호화 및 암·복호화 블록의 하드웨어 구현결과는 다양한 환경에 대한 LEA 하드웨어 설계 및 구현에 성능지표로 활용될 수 있을 것이다.

REFERENCES

[1] 미래창조과학부·한국인터넷진흥원, "사물인터넷(IoT) 환경에서의 암호·인증기술 이용 안내서", 2016. 04.

[2] 서화정, 김호원. "사물인터넷을 위한 경량 암호 알고리즘 구현", *한국정보보호학회 정보보호학회지*, 제25권 제2호, pp.12-19, 2015. 04.

[3] Telecommunications Technology Association, "128-Bit Block Cipher LEA", TTA Standard, TTAK.KO-12.0223, 2013. 12.

[4] 성미지, 신경욱 "128비트 경량 블록암호 LEA의 저면적 하드웨어 설계", *한국정보보호학회 정보보호학회지*, 제19권 제4호, pp.888-894, 2015. 04.

[5] 이철, 박능수 "고처리율 파이프라인 LEA 설계", *대한전기학회 전기학회논문지*, 제64권 제10호, pp.1460-1468, 2015. 10.

[6] 윤기하, 박성모, "128비트 LEA 암호화 블록 하드웨어 구현 연구", *한국스마트미디어학회 스마트미디어저널*, 제4권 제4호, pp.39-46, 2015. 12.

저 자 소 개

윤기하(정회원)

2012년 목포대학교 정보통신공학과 학사
2017년 전남대학교 전자컴퓨터공학과 석사
2017년~현재 한국전자통신연구원 호남권연구센터 연구원

<주관심분야 : 실시간 이더넷 MAC, 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, 사물인터넷 단말, 임베디드 시스템>