ISSN: 1226-7244 (Print)
ISSN: 2288-243X (Online)
논문번호 14-02-04
20

j.inst.Korean.electr.electron.eng.Vol.18,No.2,198~205,June 2014
http://dx.doi.org/10.7471/ikeee.2014.18.2.198

소형 애플리케이션에 적합한 AES-128 기반 저면적 암호화 회로 설계

Design of Low-area Encryption Circuit Based on AES-128 Suitable for Tiny Applications

김호진*, 김수진*, 조경순*★

Hojin Kim*, Soojin Kim*, Kyeongsoon Cho*★

Abstract

As the development of information technology, the interests in tiny applications such as wearable devices, portable devices and RFID are increased and the importance of low-area encryption circuit is emphasized. This paper proposes a compact architecture of AES-based encryption circuit suitable for tiny applications. The circuit area is reduced by minimizing storage space and sharing computation resources. The synthesized gate-level circuit using 65nm standard cell library consists of 2,241 gates and two 8x16-bit SRAMs. It can process data at a rate of 50.57Mbits per second. Therefore, the proposed encryption circuit is suitable for various applications requiring very small encryption circuit.

요 약

정보화 기술의 발전에 따라 웨어러블 장치, 휴대용 장치, RFID와 같은 소형 애플리케이션에 대한 관심이 증가하고 있고, 여기에 적용하기 위한 소형 암호화 회로의 중요성이 강조되고 있다. 본 논문에서는 소형 애플리케이션에 적합한 AES 기반 암호화 회로를 제안한다. 제안하는 회로에서는 저장 공간의 최소화, 연산 자원의 공유를 통해서 크기를 최소화 하였다. 제안하는 회로는 8x16 비트 크기의 SRAM 두 개를 사용하였으며, 65nm 표준 셀 라이브러리를 이용하여 합성한 결과 2,241 개의 게이트로 구현되었고, 처리 속도는 초당 50.57M 비트이다. 따라서 저면적 암호화 회로를 필요로 하는 다양한 애플리케이션에 적용하여 사용할 수 있다.

Key words : low-area design, AES, security, encryption, cryptography

Ⅰ. 서론

시대가 발전함에 따라 다양한 애플리케이션을 통하여 금융정보 및 개인 정보를 주고받게 되었다. 타인에게 알려지지 말아야 할 중요한 정보들이 데이터 송수신 과정에서 해킹과 같은 사이버 범죄에 노출되어 악용될 수 있으므로, 이러한 피해를 예방하려면 암호화 알고리즘을 이용하여 정보를 암호화하는 과정이 반드시 필요하다. 이러한 암호화 과정에서 많이 사용되는 AES (Advanced Encryption Standard)[1] 알고리즘은 오랜 기간 동안 다양한 분야의 암호화 알고리즘으로 사용되어 왔던 DES (Data Encryption

* Dept. of Electronics Engineering, Hankuk University of Foreign Studies
ekkd1013@hufs.ac.kr 010-8974-0702
★ Corresponding author

※ Acknowledgment
This work was supported by Hankuk University of Foreign Studies Research Fund of 2014.

Manuscript received Apr. 10, 2014; revised Jun. 2, 2014; accepted Jun. 2. 2014

Standard)의 단점을 극복하기 위하여 미국 NIST (National Institute of Standard and Technology)가 채택한 새로운 암호화 표준으로 128, 192, 256 비트의 키를 지원한다. 일반적인 상용 환경에서 128 비트의 키를 이용하여도 안전하다고 알려져 있으며, 논문 [2-4]에서는 각각 외장 하드, CPU 코어, RFID 및 스마트 카드를 위한 AES 회로를 제안하는 논문으로 128 비트의 키를 사용한다. 논문 [2-4] 이외에도 다양한 분야에서 128 비트의 키를 사용하며, 본 논문에서도 128 비트의 키를 사용하는 회로를 제안한다.

최근 반도체 기술의 발전으로 HDD (Hard Disk Drive), SSD (Solid State Disk), 플래쉬 메모리와 같은 다양한 종류의 저장매체의 읽기와 쓰기 속도가 증가하고 있고, 각 장치들과 연결된 인터페이스의 속도 또한 증가하고 있다. 이러한 고속의 동작을 지원하기 위해서 고성능의 AES 회로에 대한 연구가 진행되었다[5-6]. 뿐만 아니라 휴대용 장치, RFID (Radio Frequency IDentification), 웨어러블 장치와 같은 인간 친화적인 소형 애플리케이션에 대한 개발이 활발히 진행됨에 따라 이에 적합한 저면적 AES 암호화 회로에 대한 관심 또한 증가하고 있다. 이와 같은 소형 애플리케이션에 암호화 회로를 적용하기 위해서는 저면적 회로 설계가 필수적이다[4, 8]. 고속의 동작을 목표로 설계한 [5-6]와 같은 회로들은 회로의 크기 측면에서 소형 애플리케이션에 적합하지 않다.

일반적으로 헬스 케어장치, RFID, 휴대용 장치와 같은 소형 애플리케이션은 송신단에서의 암호화 과정은 필수적이지만 복호화 과정은 필수적이지 않다. 논문 [7-9]에서는 각각 헬스 케어장치, RFID, 휴대용 장치를 위한 보안 회로를 제안하는 논문으로 암호화 회로만을 설계하였다. 논문 [7-9] 이외에도 다양한 분야에서 암호화 회로만을 설계하고 있다. 이에 따라 본 논문에서는 소형 애플리케이션에 적합한 AES-128 알고리즘 기반의 저면적 암호화 회로를 제안한다.

본 논문의 Ⅱ장에서는 AES-128 알고리즘에 대하여 서술하고, Ⅲ장에서는 AES 회로를 저면적으로 설계하기 위한 회로 구조에 대해서 서술한다. 마지막으로 Ⅳ장과 Ⅴ장에서는 실험 결과와 결론을 서술한다.

Ⅱ. AES-128 알고리즘

(그림 1) (a)의 SubByte 연산은 state의 각 데이터를 비선형으로 치환하는 연산으로서 S-box라 부르는 치환 테이블을 사용하여 연산을 수행한다. S-box는 유한체 GF(2^8)의 곱셈에 대한 역원과 GF(2)와의 affine 변환 연산을 통하여 생성된다.

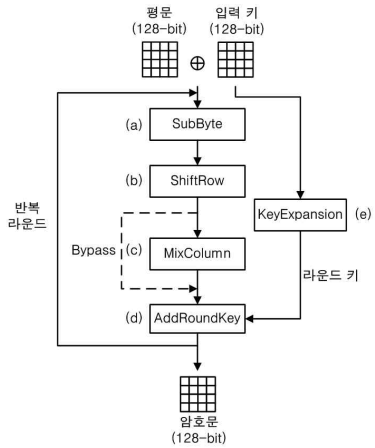


Fig 1. AES-128 encryption algorithm
그림 1. AES-128 암호화 알고리즘

(그림 2)는 이와 같이 생성된 S-box 테이블을 보여주고 있다. SubByte의 연산의 예를 들면, 입력으로 $17_{(16)}$ 값이 들어오는 경우 이 값은 (그림 2)의 x축 1 위치와 y축 7 위치에 존재하는 $f0_{(16)}$로 치환되어 출력된다.

x \ y	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Fig 2. S-box table generated by GF(2^8) and affine transformation
그림 2. GF(2^8)과 affine 변환으로 생성된 S-box 테이블

(그림 1) (b)의 ShiftRow 연산은 state의 각 행을 왼쪽으로 순환 이동하는 연산이다. (그림 3)과 같이 state의 첫 번째 행에 대해서는 순환이동을 하지 않고, 두 번째 행에 대해서는 왼쪽으로 1 바이트, 세 번째 행에 대해서는 왼쪽으로 2 바이트, 네 번째 행에 대해서는 왼쪽으로 3 바이트 순환 이동한다.

(그림 1) (c)의 MixColumn 연산은 state의 각 열을 유한체 GF(2^8) 다항식의 곱셈을 사용하여 연산을 수행하며, state를 구성하고 있는 4 바이트의 각 열을 이용한다. 식 (1-4)는 MixColumn 연산에 사용되는 연산 식으로 '•' 연산은 유한체 GF(2^8)의 곱셈 연산을

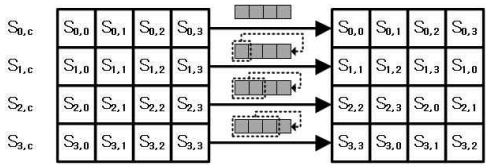


Fig 3. ShiftRow operation
그림 3. ShiftRow 연산

의미한다[1]. 식의 $S_{0,c}$, $S_{1,c}$, $S_{2,c}$, $S_{3,c}$는 state의 열 데이터들을 의미한다.

$$s'_{0,c} = (\{02\} \bullet s_{0,c}) \oplus (\{03\} \bullet s_{1,c}) \oplus s_{2,c} \oplus s_{3,c} \quad (1)$$

$$s'_{1,c} = s_{0,c} \oplus (\{02\} \bullet s_{1,c}) \oplus (\{03\} \bullet s_{2,c}) \oplus s_{3,c} \quad (2)$$

$$s'_{2,c} = s_{0,c} \oplus s_{1,c} \oplus (\{02\} \bullet s_{2,c}) \oplus (\{03\} \bullet s_{3,c}) \quad (3)$$

$$s'_{3,c} = (\{03\} \bullet s_{0,c}) \oplus s_{1,c} \oplus s_{2,c} \oplus (\{02\} \bullet s_{3,c}) \quad (4)$$

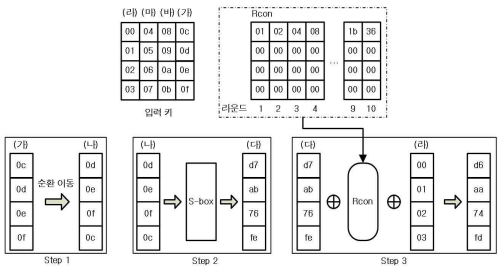


Fig 4. Example of KeyExpansion operation
그림 4. KeyExpansion 연산의 예

(그림 1) (d)에 나타나 있는 AddRoundKey 연산은 KeyExpansion 연산을 통해서 생성된 라운드 키와 MixColumn 데이터와의 XOR 연산을 통하여 수행한다. (그림 1) (e)의 KeyExpansion 연산은 (그림 4)에서 보여주는 과정을 통하여 각 라운드의 라운드 키를 생성한다. 첫 번째 단계로 입력 키의 마지막 열인 (가)를 순환 이동하여 (나)를 생성한다. 두 번째 단계에서는 S-box를 이용하여 (나)의 각 바이트에 대하여 치환을 통해 (다)를 생성한다. 세 번째 단계에서는 (다)값과 각 라운드에 따른 참조 값인 Rcon 값[1], 입력 키의 첫 번째 열인 (라)와 XOR 연산을 수행하여 라운드 키의 첫 번째 열을 생성한다. 라운드 키의 두 번째 열은 앞서 생성한 라운드 키의 첫 번째 열과 (마)와의 XOR 연산으로 생성한다. 라운드 키의 세 번째 열은 앞서 생성한 라운드 키의 두 번째 열과 (바)와의 XOR 연산으로 생성한다. 라운드 키의 네 번째 열은 앞서 생성한 라운드 키의 세 번째 열과 (가)와의 XOR 연산으로 수행한다. 다음 라운드의 라운드 키에 사용하는 입력 키는 이전 라운드의 라운드 키를 사용하여 위의 과정을 반복하게 된다.

III. 제안하는 회로 구조

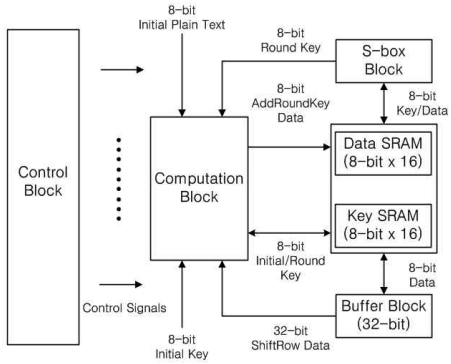


Fig 5. Proposed architecture of AES circuit
그림 5. 제안하는 AES 회로의 구조

(그림 5)는 본 논문에서 제안하는 AES 회로의 구조를 보여주고 있다. AES-128 회로는 연산을 처리하는 비트 수에 따라서 회로의 성능과 크기에 지대한 영향을 받는다. 예를 들어서 한 라운드를 128 비트 단위로 처리하는 경우, 연산에 사용되는 XOR 연산기의 개수는 최소 84개가 필요하고, 연산에 이용되는 S-box는 최소 20개가 필요하게 되어 회로의 동작 속도는 빨라지지만 회로의 크기는 커지게 된다. 반면 연산을 8 비트 단위로 처리하는 경우, 연산에 사용되는 XOR 연산기의 개수는 최소 6개만 필요하게 되고, 연산에 이용되는 S-box는 최소 1개만을 필요하게 되어 회로의 동작 속도는 낮아지지만 회로를 구성하는 연산기의 개수가 줄기 때문에 회로의 크기가 작아진다. 연산기 이외에도 중간 연산 결과를 저장하기 위하여 최소 128 비트 크기의 저장 공간이 1개 이상, 키를 저장하기 위하여 최소 128 비트 크기의 저장 공간이 1개 이상 필요하다. 따라서 회로의 면적을 줄이기 위해서 저장 공간의 최소화도 반드시 필요하다. 이에 따라 본 논문에서 제안하는 회로에서는 회로의 크기 최소화를 최우선 목표로 하여 우선 (그림 1)의 AddRoundKey, KeyExpansion, SubByte, ShiftRow, MixColumn 연산을 8 비트 단위로 처리하고, 또한 중간 연산 결과 및 키의 저장에 필요한 공간을 8x16 비트 SRAM (Static Random Access Memory) 두 개만을 사용하였다. 이와 같이 구성된 회로는 8 비트 크기의 입력 키와 평문을 입력으로 사용하고, 회로의 연산 결과는 8 비트 단위로 출력하여 준다.

(그림 5)에 나타나 있는 바와 같이, 제안하는 회로는 회로 전체를 제어해주기 위한 'Control Block',

SubByte, KeyExpansion 연산을 위한 'S-box Block', (그림 1)에 나타나있는 암호화 과정의 AddroundKey, KeyExpansion, MixColumn 연산을 수행하기 위한 통합 연산 블록인 'Computation Block', 키와 state의 중간 결과를 저장하기 위한 두 개의 8x16 비트 SRAM ('Key SRAM', 'Data SRAM'), ShiftRow, MixColumn 연산을 수행을 위한 32 비트 'Buffer Block'으로 구성되어 있다.

암호화 회로에 입력된 8 비트 크기의 평문과 입력 키는 (그림 5)의 'Computation Block'을 통하여 AddRoundKey 연산이 수행된 다음 8 비트 단위로 'Data SRAM'에 저장되고, 입력 받은 암호 키는 8 비트 단위로 'Key SRAM'에 저장된다. 'Data SRAM'에 저장된 데이터는 (그림 2)의 S-box 치환 테이블로 구성된 'S-box Block'을 통해 8 비트 단위로 SubByte 과정이 수행된 다음 다시 'Data SRAM'에 8 비트 단위로 저장된다. 저장된 'Data SRAM'의 데이터는 ShiftRow 연산을 위해 32 비트 크기의 버퍼인 'Buffer Block'에 저장하고, 'Buffer Block'에서 데이터를 ShiftRow 연산에 맞게 가져오게 된다.

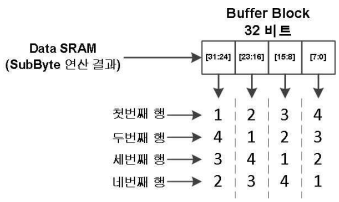


Fig 6. ShiftRow operation flow
그림 6. ShiftRow 연산 과정

(그림 6)과 같이 'Data SRAM'으로부터 SubByte 연산 결과가 'Buffer Block'에 저장된다. 'Buffer Block'에 SubByte 연산 결과의 첫 번째 행이 저장되면 (그림 6)과 같이 첫 번째 행에 해당하는 순서인 1-2-3-4 순서로 데이터를 읽어 ShiftRow 연산을 수행하고, 'Data SRAM'에 8 비트 단위로 다시 저장한다. SubByte 연산의 나머지 행에 대해서도 (그림 6)과 같은 순서로 데이터를 읽어 ShiftRow 연산을 수행하고, 'Data SRAM'에 8비트 단위로 다시 저장한다.

MixColumn 과정에서는 8 비트의 연산 결과를 구하기 위해 'Data SRAM'에 저장되어 있는 state의 열인 4 바이트 데이터를 'Buffer Block'에 임시로 저장한 후 통합 연산 블록인 'Computation Block'으로 보내준다. 'Computation Block'을 통하여 연산이 수행된 MixColumn의 결과에 대해서 KeyExpansion 과정에서 생성한 라운드 키와 8비트 단위로 XOR 연산을 수행함으로써 AddRoundKey 과정이 수행된다.

KeyExpansion 과정은 앞서 'Key SRAM'에 저장된 입력 키를 통합 연산 블록인 'Computation Block'과 치환 테이블이 저장된 'S-box Block'을 사용하여 (그림 4)의 과정을 거쳐 라운드 키를 8 비트 단위로 생성해준다. 생성된 라운드 키는 다음 라운드의 키 생성을 위해 'Key SRAM'에 다시 저장된다. 저면적 AES 회로를 구현하기 위해 이와 같이 8x16 비트 SRAM 두 개만을 사용해 저장 공간을 최소화하였다. 한 개의 SRAM에는 암호문의 중간 데이터를 저장하고, 다른 한 개의 SRAM에는 입력 키 및 라운드 키를 저장하였다. 또한, (그림 1)의 SubByte 연산과 KeyExpansion 연산에서 사용하는 S-box 테이블을 공유하였으며, MixColumn 연산에 사용되는 XOR 게이트들을 공유하였다. 제안하는 회로는 AES-128의 연산을 8 비트 단위로 처리하기 때문에 연산을 128 비트 단위로 처리하는 경우에 비해 연산 속도가 약 1/16 정도로 저하되게 된다. 하지만 제안하는 회로는 Mbps(bit per second)급의 속도를 지원하며, RFID와 같은 소형 애플리케이션에 사용하기에 충분한 성능을 갖는다.

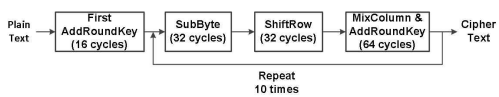


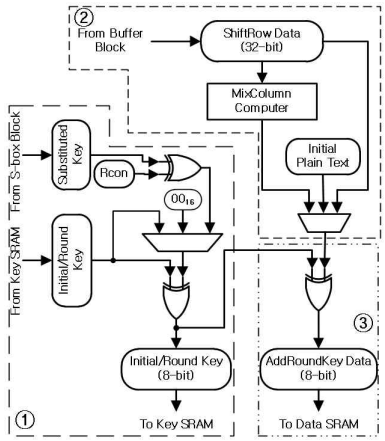
Fig 7. Timing diagram of AES circuit
그림 7. AES 동작 타이밍도

(그림 7)은 제안하는 회로의 타이밍도이다. 연산은 첫 AddRoundKey 과정에서 16 사이클이 소비되고, SubByte 연산에 32 사이클, ShiftRow 연산에 32 사이클, MixColumn과 AddroundKey 연산을 동시에 처리하는데 64 사이클이 소비된다. 또한 SubByte 과정에서부터 MixColumn 연산 과정까지의 연산을 10번의 반복 연산을 하게 되며, 평문 데이터를 암호화하는데 총 1,296 사이클이 소비된다.

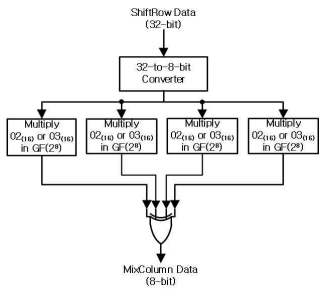
(그림 8) (a)는 'Computation Block'의 구조를 나타내고 있다. 'Computation Block'은 AddRoundKey, KeyExpansion, MixColumn 연산을 모두 처리할 수 있도록 통합 구성하였고, 8 비트 단위로 연산을 처리해 줌으로써 회로의 크기를 최소화 하였다. (그림 8) (a)의 ①은 KeyExpansion 연산을 위한 블록이다. 제안하는 회로의 구조에서는 KeyExpansion 연산 블록의 크기를 최소화하기 위하여 KeyExpansion 과정을 8 비트 단위로 처리하도록 회로를 구성하였다. 라운드 키 생성을 위하여 순환 이동시킨 8 비트 크기의 입력이나 현재 라운드 키를 (그림 5)의 'S-box Block'을 이용하여 8비트 단위로 치환 과정을 수행한 결과

와 (그림 4)의 8 비트 크기의 Rcon 데이터, 8 비트 입력 이나 현재 라운드 키를 입력으로 사용하여 다음 라운드 키를 8 비트 단위로 생성한다.

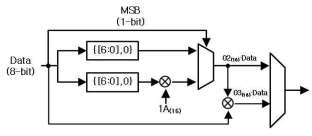
(그림 8) (a)의 ②는 MixColumn 연산을 위한 블록이다. MixColumn 연산은 식 (1-4)가 x = a⊕b⊕c⊕d 과 같은 4개의 입력 간의 XOR 연산과 같은 비슷한 구조를 취하고 있기 때문에 공통 연산 부분을 GF(2^8) 곱셈과 XOR 게이트로 구성된 새로운 연산 구조를 (그림 8) (b)와 같이 구성하였다. 이를 통해서 기존의 식 (1-4)를 일반적으로 구성하였을 때에 비해 연산기



(a) Computation block



(b) MixColumn calculator



(c) Multiplier in GF(2^8)

Fig 8. Architecture of main blocks of AES circuit
그림 8. AES 회로의 주요 블록 구조

의 개수를 1/4로 줄여 회로의 크기를 줄일 수 있었다. 연산기를 1/4로 줄였기 때문에 MixColumn 연산에 한해서 32 비트 단위로 구성하였을 때에 비해 속도가 1/4로 저하 된다. 하지만 제안하는 회로의 암호화 성능은 Mbps급 성능을 보이기 때문에 소형 애플리케이션에 적용하기에 충분하다. (그림 8) (c)는 GF(2^8) 곱셈 연산기의 내부 구조를 표현하고 있다. 입력된 데이터와 $02_{(16)}$와의 연산 결과를 보낼지 $03_{(16)}$과의 연산 결과를 보낼지 선택하도록 구성했다. (그림 8) (a)의 ③은 KeyExpansion 연산을 통하여 생성한 키와 MixColumn의 연산 결과를 8 비트 단위로 XOR연산을 하도록 구성하였다.

제안하는 회로의 크기를 최소화하기 위하여 적용한 방법을 요약하면 다음과 같다. 첫째, 8 비트 단위로 모든 연산을 수행하도록 회로를 구성하였다. 둘째, MixColumn 연산 중 공통되는 연산을 찾아서 중복되지 않도록 연산기를 공유하였다. 셋째, SubByte 연산과 KeyExpansion 연산에서 필요한 S-box를 여러 개 사용하지 않도록 공유하였다. 넷째, ShiftRow, MixColumn, KeyExpansion 연산에서 필요로 하는 버퍼를 공유하였다. 다섯째, 중간 연산 결과와 라운드 키를 저장하기 위한 저장 공간을 줄이기 위해서 8x16 비트 SRAM을 두 개만 사용하였다.

Ⅳ. 실험 결과

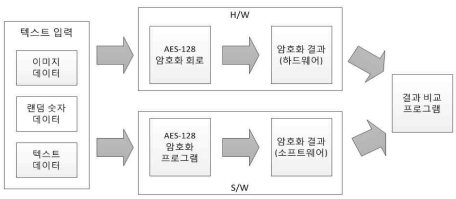


Fig 9. Verification flow of proposed circuit
그림 9. 제안하는 회로의 검증 과정

제안하는 AES 알고리즘 기반 암호화 회로는 Verilog HDL (Hardware Description Language)을 사용하여 RTL (Register Transfer Level) 수준으로 설계하였고, Cadance사의 NC-Verilog를 사용하여 기능을 검증하였다.

암호화 회로는 (그림 9)의 검증 과정을 통하여 회로의 기능을 검증하였다. 암호화 회로의 검증 결과와 암호화 프로그램을 통한 암호화 결과의 비교를 통하여 회로의 기능 검증을 진행하였고, <표 1>은 회로의 입출력을 나타내고 있다. 회로는 클럭, 입력 제어, 입력 키, 평문, 리셋 신호를 입력받고 출력 제어, 암호문

을 출력한다.

Table 1. Input and output port information
표 1. 입력 및 출력 포트 정보

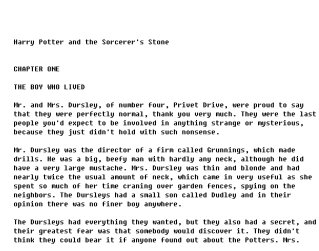
포트이름	입력/출력	비트	설명
clk	입력	1 비트	클럭
en	입력	1 비트	입력 제어
key_in	입력	8 비트	입력 키
plain	입력	8 비트	평문
reset	입력	1 비트	리셋
out_val	출력	1 비트	출력 제어
cipher	출력	8 비트	암호문

기능 검증의 정확도를 높이기 위해 입력 벡터로 (그림 10) (a)의 FIPS (Federal Information Processing Standards)[1]에서 제공하는 128 비트 예제 입력 데이터, (그림 10) (b)의 1.5M 비트 크기의 이미지 데이터, (그림 10) (c)의 1M 비트 크기의 랜덤 숫자 데이터, 그리고 (그림 10) (d)의 0.5M 비트 크기의 텍스트 데이터를 사용하였다.

(a) 128-bit example data[1]

(b) 1.5M-bit image data

(c) 1M-bit random number data



(d) 0.5M-bit text data

Fig 10. Examples of input data
그림 10. 입력 데이터의 예제

(그림 9)의 과정을 통하여 기능 검증을 완료한 RTL 수준의 회로는 Synopsys사의 Design Compiler를 사용하여 65nm 표준 셀 라이브러리를 이용하여 합성하였다. 합성 결과 제안하는 회로의 게이트는 2,241이고, 최대 동작 주파수는 512MHz이다. 제안하는 회로가 128 비트 크기의 데이터를 암호화 하는데 필요로 하는 연산 시간은 1,296 클럭 사이클이며, 이는 참고문헌 [8]에서 제시하는 RFID를 위한 성능 조건을 만족시킨다. 데이터 처리 속도는 50.57Mbps이며, 암호화 데이터와 키 데이터의 중간 연산 값을 저장하기 위해서 8x16 비트 SRAM이 두 개 사용되었다.

<표 2>는 제안하는 암호화 회로와 AES 알고리즘을 기반으로 다른 논문에서 제안한 회로들과의 비교표이다. 논문 [8]과 [9]는 각각 RFID를 위한 저면적 AES 회로와 휴대용 기기를 위한 저면적 AES 회로에 대하여 기술하고 있다.

논문 [8]에서는 회로의 크기를 줄이기 위하여 본 논문의 연산 단위와 동일한 8 비트 단위로 연산을 하도록 구성 하였고, SubByte, MixColumn 연산을 위한 S-box를 공유하여 한 개만 사용하였다. 또한 MixColumn 연산도 본 논문에서 제시하는 8 비트 단위로 계산 값이 나오도록 구성하였다. 하지만 MixColumn 내부의 연산을 수행하여주는 부분에서 추가적인 GF(2^8)연산을 수행하기 때문에 회로의 크기가 제안하는 회로에 비해 크다.

Table 2. Performance comparison
표 2. 성능 비교

	제안한 회로	[8]	[9]	[10]
최대 동작 주파수 (MHz)	512	-	152	417
게이트 수	2,241	3,595	3,100	59,437
클락 사이클 수	1,296	992	160	-
최대 지연 시간	1.95ns	-	6.57ns	2.39ns
속도 (Mbps)	50.57	-	121	10,667
SRAM 크기	128bit x 2	128bit x 2	-	-
제조공정	65nm	350nm	130nm	130nm

논문 [9]에서는 회로의 크기를 줄이기 위하여 본 논문의 연산 단위와 동일한 8 비트 단위로 연산을 처리하도록 구성하였다. 본 논문에서는 회로의 크기를 최소화하기 위해서 SubByte, KeyExpansion 연산을 위한 S-box를 공유하여 한 개만을 사용하였다. 하지만 논문 [9]에서는 SubByte, KeyExpansion 연산을 위한 S-box를 개별적으로 두 개를 구성하였기 때문에 제안하는 회로에 비해 성능은 올라가지만 회로의 크기가 크다. 또한 제안하는 회로에서의 MixColumn 연산은 회로의 크기를 최소화하기 위해 결과 값을 8

비트 단위로 나오도록 구성하였지만 논문 [9]에서는 결과 값을 32 비트 단위로 만들어주기 때문에 회로의 성능은 향상되었지만 회로의 크기가 증가하였다. 연산 사이클 수와 지연시간을 이용하여 속도를 bps단위로 환산하여 비교한 결과 제안하는 회로에 비해 논문 [9]에서 제안하는 회로의 연산 속도가 빠르다. 하지만 회로의 크기를 비교한 결과 논문 [9]에서 제안하는 회로가 본 논문에서 제안하는 회로보다 크기가 더 크다. 따라서 본 논문에서 제안하는 회로가 논문 [8], [9]에 비해 회로의 크기가 작기 때문에 회로의 저면적이 최우선적 목표인 RFID, 웨어러블 장치와 같은 소형 애플리케이션에 더 적합하다. 논문 [10]은 10Gbps의 성능을 지원하는 AES 회로에 대하여 기술하고 있다. 논문 [10]에서는 본 논문의 연산 단위와 다르게 128 비트 단위로 연산을 처리하도록 구성하였다. 이에 따라서 사용되는 연산기의 개수가 증가하였고, 사용되는 S-box의 개수 또한 증가하였다. 이에 따라 논문 [10] 회로의 속도는 높지만 회로의 크기 측면에서 저면적이 요구되는 소형 애플리케이션에는 적합하지 않다.

Ⅴ. 결론

제안하는 저면적 AES 회로는 RFID와 휴대용 장치와 같이 저면적이 요구되는 소형 애플리케이션에 적용이 가능하도록 설계하였다. 회로의 면적을 최소화하기 위해서 8 비트 단위로 연산을 처리하도록 회로를 구성하였다. MixColumn 연산에서 사용하는 XOR 게이트 연산 자원을 공유하여 회로의 크기를 최소화하였고, SubByte 과정과 KeyExpansion 과정에서 사용되는 S-box를 공유하여 회로의 크기를 줄였다. 또한 AES-128 연산에 필요한 저장 공간을 8x16 비트 크기의 SRAM 두 개만을 사용하여 최소화 하였고, KeyExpanstion, ShiftRow, MixColumn 연산에서 사용하는 버퍼를 공유하여 회로의 크기를 최소화하였다. 이와 같은 과정을 통하여 회로의 크기를 최우선적으로 고려하는 RFID, 웨어러블 장치와 같은 소형 애플리케이션에 적합한 AES 기반 암호화 회로를 설계하였다.

References

[1] FIPS Publication 197, Advanced Encryption Standard (AES), *U.S. Doc/NIST.*

[2] K. Thongkhome, C. Thanavijitpun and S. Choomchuay, “A FPGA Design of AES Core Architecture for Portable Hard Disk,” *2011 Eighth International Joint Conference on Computer Science and Software Engineering*, pp.223-228, May 2011.

[3] Tsutsumi. D, Ohmura. I, Abe. T, Yoshimura. H, Inagawa. K, “An AES Processing System with a Compact CPU Core for Secure Communication in Embedded Systems,” *TENCON 2012 IEEE Region 10 Conference*, pp.1-5, Nov. 2012.

[4] Tuan Anh Pham, Hasan, M.S, Hongnian Yu, “Area and Power Optimisation for AES Encryption Module Implementation on FPGA,” *2012 18th International Conference on Automation and Computing (ICAC)*, pp.1-6, Sep. 2012.

[5] X. Zhang, H. Li, S. Yang and S. Han, “On a High-Performance and Balanced Method of Hardware Implementation for AES,” *2013 IEEE 7th International Conference on Software Security and Reliability-Companion*, pp.16-20, Jun. 2013.

[6] X. Cai, R. Sun and J. Liu, “An Ultrahigh Speed AES Processor Method Based on FPGA,” *2013 5th International Conference on Intelligent Networking and Collaborative Systems,* pp.633-636, Sep. 2013.

[7] Doukas. C, Maglogiannis. I, Koufi. V, Malamateniou. F, Vassilacopoulos. G, “Enabling Data Protection through PKI Encryption in IoT m-Health Devices,” *2012 IEEE 12th International Conference on Bioinformatics & Bioengineering(BIBE),* pp. 25-29, Nov. 2012.

[8] M. Feldhofer, S. Dominikus and J. Wolkerstorfer, “Strong Authentication for RFID Systems Using the AES Algorithm,” *Proceedings of the Workshop on Cryptographic Hardware and Embedded Systems*, pp.357 - 370, Aug. 2004.

[9] P. Hamalainen, T. Alho, M. Hannikainen and T. D. Hamalainen, “Design and Implementation of Low-area and Low-power AES Encryption Hardware Core,” *9th EUROMICRO Conference on Digital System Design 2006,* pp.577-583, Sep. 2006.

[10] P. Maistri and R. Leveugle, “10-Gigabit Throughput and Low Area for a Hardware Implementation of the Advanced Encryption Standard,” *2011 14th Euromicro Conference on Digital System Design*, pp.266-269, Aug. 2011.

BIOGRAPHY

Hojin Kim (Member)

2012: B.S. degree in Electronics Engineering, Hankuk University of Foreign Studies, Korea.
2014: M.S. degree in Electronics and Information Engineering, Hankuk University of Foreign Studies, Korea.

Soojin Kim (Member)

2007: B.S. degree in Electronics Engineering, Hankuk University of Foreign Studies, Korea.
2009: M.S. degree in Electronics and Information Engineering, Hankuk University of Foreign Studies, Korea.
2009~Present: Pursuing a Ph.D. degree in Electronics and Information Engineering, Hankuk University of Foreign Studies, Korea.
2010~2013: Researcher at the SoC Platform Research Center at Korea Electronics Technology Institute, Korea.

Kyeongsoon Cho (Member)

1982: B.S. degree in Electronics Engineering, Seoul National University, Korea.
1984: M.S. degree in Electronics Engineering, Seoul National University, Korea.
1988: Ph.D. degree in Electrical and Computer Engineering, Carnegie Mellon University, U.S.A.
1988~1994: Senior researcher at the Semiconductor ASIC Division of the Samsung Electronics Company.
1994~Present: Professor at the Department of Electronics and Information Engineering at Hankuk University of Foreign Studies.
1999~2003: Senior director at Enhanced Chip Technology.
2003~2004: Head of the CoAsia Korea Research and Development Center.
2005~2011: Vice director of the Collaborative Project for Excellence in System IC Technology.
2005~2012: Technical advisor of Dongbu HiTek.
2012~Present: Technical advisor of DawinTech.