論文2002-39TE-1-7

SEED 암호 알고리즘을 적용한 음성 신호 암호화 칩 설계

(The chip design for the cipher of the voice signal to use the SEED cipher algorithm)

安 寅 秀 *, 崔 太 燮 *, 林 承 河 **, 司 空 石 鎭 *

(In-Soo Ahn, Tae-Sup Choi, Seung-Ha Lim, and Sug-Chin Sakong)

요 약

정보 통신의 급속한 발전과 확산으로 전세계가 통신망으로 개방화되고, 정보 자체가 국가 경제 발전을 좌우하는 중요한 변수로 작용하게 되었다. 정보 보안은 특성상 각 국가마다 독단적인 정보보호시스템을 개발하여 독립적으로 그 비밀성을 유지해야 할 필요가 있다. 이에 국내 암호 알고리즘의 활용 확대를 위해 국내 표준인 SEED 암호 알고리즘을 적용하여 Xilinx사 XCV300PQ240 칩을 타겟(target)으로 최대 동작 주파수 47.895㎒이고, 등가게이트는 27,285인 음성 암호화 칩을 설계하였다.

Abstract

The world was opened by communication network because of fast improvement and diffusion of information communication. And information was effected in important factor that control economy improvement of the country. The country should improve the information security system because of necessity to maintain its information security independently. Therefore we have used the SEED cipher algorithm and designed the cipher chip of the voice band signal using the Xilinx Co. XCV300PQ240 chip. At the result we designed the voice signal cipher chip of the maximum fregquency 47.895㎒ and the total equivalent gate 27,285.

Ⅰ. 서 론

컴퓨터와 통신의 발달로 정보 관리와 전송 분야 등에서 정보 보호의 역할은 절대적인 필수 요소이다. 이전부터 암호 기술은 오랜 역사를 갖고 있었지만, 최근까지도 일반인에게 암호를 위한 기술적인 내용이 알려진 것은 거의 없었다. 이러한 상황에서 1990년대 중반에 들어 인터넷의 급속한 보급과 상용화에 따라 정보화의 가치에 커다란 의미를 두고, 그것을 보호하기 위한 정보화의 진전이 필연적으로 대두되었다. 21세기 정보화 사회에서는 정보 통신의 급속한 발전과 확산으로 전세계의 정보가 통신망으로 개방화됨에 따라 정보 자체가 국가 경제의 발전을 좌우하는 중요한 변수로 작용하게 될 것이다.

특정 시스템에 암호화를 적용할 경우 암호화 처리는 각종 컴퓨터 시스템이나 통신시스템에서 느린 암호화 처리 속도로 인해 시스템의 속도가 지연된다[1]. 최근 다양한 정보통신기기들의 고속화 열풍에 따라 빠른 암호화 속도를 갖는 여러 상업용 암호화 소프트웨어와 하드

* 正會員, 國民大學校 電子情報通信工學部
(School of Electrical Engineering, Kookmin University)
** 正會員, 富川大學 電子科
(Dept. of Electronics Engineering, Bucheon College)
接受日字:2002年1月7日, 수정완료일:2002年3月14日

웨어 제품들이 개발되고 있고, 현재 암호화 알고리즘을 적용한 암호화 칩(chip)의 구현이 활발히 진행되고 있다. 반면, 상대적으로 고속화에 따른 암호 알고리즘의 공격 가능성이 높아지고, 공격 방법도 다양해지고 있다. 가장 범용화된 DES 알고리즘을 적용하여 소프트웨어로 구현하게 되면 계산의 복잡성으로 인해 10~60Mbps 정도의 암호화 속도를 갖는다[2 4]. 실시간 통신이나 100Mbps 이상의 초고속 통신망에서 암·복호화 동작이 시스템에 미치는 부하 등을 최소화하기 위해서는 암호화 시스템 또한 100Mbps 이상으로 동작하여야 하며, 이를 위해서는 소프트웨어보다는 하드웨어로 구현하는 것이 훨씬 효율적이다[5-6]. 현재 대중화되어 있는 유선이나 무선 통신에서 주고받는 사적인 또는 공적인 통화 내용은 공중에서 무방비 상태로 노출되어 있어 정보의 비밀성 보장에 문제가 있다. 따라서 이러한 문제점을 미연에 방지하고자 통화 내용을 암호화 할 수 있도록 국내 표준 블록 암호 알고리즘으로 공표된 SEED를 적용하여 음성 신호 암호화 칩을 설계하였다.

제안한 방식은 암호화 칩으로 구현하기 위해서 탑-다운(Top-down) 방식으로 설계하였다. 우선 크게 칩의 데이터 입·출력 제어를 담당하는 제어부(Control Block), 데이터 암호화시 각 라운드에 필요한 키 값을 생성하는 키 생성부(Key Generation Block), 암·복호화 처리를 담당하는 블록 단위로 데이터의 암·복호화를 담당하는 데이터패스(Datapath Block)로 나눌 수 있다. 이들을 하드웨어의 자원을 축소할 수 있는 방식의 단일 라운드 방식을 적용하여 하나의 모듈(module)로 설계하였으며, 실제 음성을 대상으로 확인할 결과 암호화된 신호와 복호화된 신호를 확인할 수 있었다.

Ⅱ. SEED 블록 암호 알고리즘

SEED는 대칭 키 구조의 블록 단위로 메시지를 처리하는 알고리즘으로서 1999년 국내 128비트 블록 암호 알고리즘 표준안으로 제안되었다. SEED 알고리즘은 Feistel 구조의 블록 암호 방식으로 데이터 처리 단위는 8, 16, 32비트 모두 가능하다. 입·출력문과 입력키의 크기는 128비트이므로 알고리즘을 분석할 때 전수조사 공격을 피하기 위한 충분한 안전도를 제공하며, 또한 현재 블록 암호 알고리즘의 구조상의 취약성을 분석하는 가장 강력한 수단으로 알려져 있는 차분해독법(DC:Differential Cryptanalysis)[7-10]과 선형해독법(LC:Linear Cryptanalysis)[11 12]에 대하여 안전하다. 또한, 키 생성 알고리즘에 특정한 취약성이 없도록 설계되었고, 소프트웨어로 구현시 암·복호화 속도는 3중 DES보다 효율적이다[13].

1. 알고리즘 전체 구성도

그림 1은 SEED 암호 알고리즘의 전체 구조를 나타낸 것이다. 전체 구조는 Feistel 구조로 이루어져 있으며, 128비트 평문 블록 단위당 128비트 키(Key)로부터 생성된 16개의 64비트 라운드 키(Round Key)를 입력으로 하여 총 16회의 라운드(Round)를 거쳐 128비트의 평문 블록을 128비트 암호문 블록으로 출력한다. SEED는 평문 블록 128비트를 64비트 블록 L_0 와 R_0 로 나누어 F 함수를 적용하여 16회 라운드를 거친 후, 최종 출력 비트 L_{16} 와 R_{16} 을 얻는다. 이때 적용되는 SEED의 F 함수는 수정된 Feistel 암호 알고리즘으로 F 함수에서의 연산은 각 64비트를 32비트의 2개의 블록(CI, DI)으로 입력받아 32비트의 2개의 블록(CO, DO)으로 출력한다. 이때, F 함수 내의 G 함수는 두 개의 8비트 S 함수를 이용하여 32비트를 32비트로 보낸다. G 함수에서 주어진 입력 32비트를 4개의 8비트 블럭(X_3, X_2, X_1, X_0)으로 분할하여 2개의 S 함수를 (S_2, S_1, S_2, S_1) 순서로 적용시켜 (Y_3,

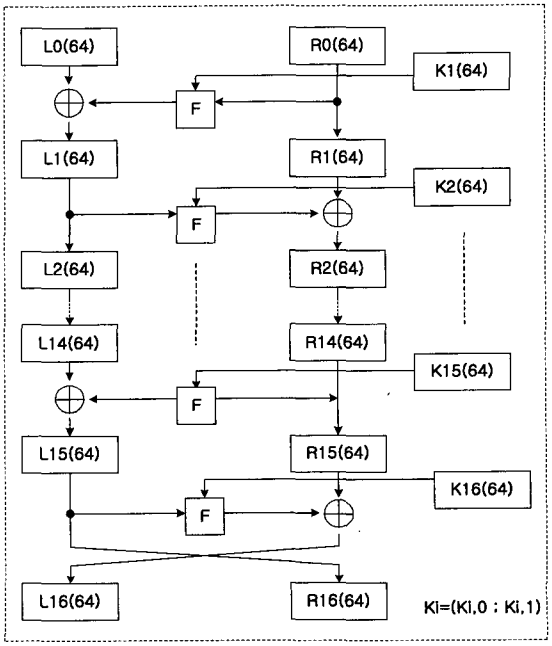


그림 1. SEED 알고리즘 전체 구조도
Fig. 1. The full architecture of SEED algorithm.

Y_2, Y_1, Y_0)를 생성한다. 이 관계를 식 (1)으로 나타낼 수 있다.

$$Y_0 = S_1(X_0), \quad Y_1 = S_2(X_1), \quad Y_2 = S_1(X_2), \quad Y_3 = S_2(X_3) \tag{1}$$

4개의 확장된 4바이트 S 함수들을 XOR 함으로써 식 (2)와 같이 (Z_3, Z_2, Z_1, Z_0)를 구한다.

$$\begin{aligned} Z_0 &= (Y_0 \& m_0) \oplus (Y_1 \& m_1) \oplus (Y_2 \& m_2) \oplus (Y_3 \& m_3) \\ Z_1 &= (Y_0 \& m_1) \oplus (Y_1 \& m_2) \oplus (Y_2 \& m_3) \oplus (Y_3 \& m_0) \\ Z_2 &= (Y_0 \& m_2) \oplus (Y_1 \& m_3) \oplus (Y_0 \& m_2) \oplus (Y_3 \& m_1) \\ Z_3 &= (Y_0 \& m_3) \oplus (Y_1 \& m_0) \oplus (Y_2 \& m_1) \oplus (Y_3 \& m_2) \end{aligned} \tag{2}$$

$$m_0 = 0xfc = 11111100 \qquad m_1 = 0xf3 = 11110011$$
$$m_2 = 0xcf = 11001111 \qquad m_3 = 0x3f = 00111111$$

위의 식 (2)는 식 (3)과 같이 정리할 수 있다. 여기에서의 &은 비트 단위 곱 연산이고, ||은 연접(concatenation)을 나타낸다.

$$\begin{aligned} SS_0 &= S_1(X_0)\&m_3 \| S_1(X_0)\&m_2 \| S_1(X_0)\&m_1 \| S_1(X_0)\&m_0 \\ SS_1 &= S_2(X_1)\&m_0 \| S_2(X_1)\&m_1 \| S_2(X_1)\&m_2 \| S_2(X_1)\&m_3 \\ SS_2 &= S_1(X_2)\&m_1 \| S_1(X_2)\&m_0 \| S_1(X_2)\&m_3 \| S_1(X_2)\&m_2 \\ SS_3 &= S_2(X_3)\&m_2 \| S_2(X_3)\&m_1 \| S_2(X_3)\&m_0 \| S_2(X_3)\&m_3 \end{aligned} \tag{3}$$

$$Z = SS_0(X_0) \oplus SS_1(X_1) \oplus SS_2(X_2) \oplus SS_3(X_3)$$

SEED 알고리즘에서 사용되는 S 함수는 부울(Bool) 함수를 사용하고 있다. 즉, S 함수는 전단사 함수 x^n의 선형 변환으로 식 (4)와 같이 나타낼 수 있다.

$$S(x) = A \cdot x^n \oplus b \quad (S(0) \neq 0,\ S(1) \neq 1) \tag{4}$$

키 생성 과정은 128비트 암호 키를 64비트씩 좌우로 나누어 이들을 교대로 8비트씩 좌우로 회전 이동한 후, 결과의 4워드(word)들에 대한 간단한 산술 연산과 G 함수를 적용하여 라운드 키를 생성한다. 라운드 키 생성은 암호화나 복호화를 할 때, 암호 키로부터 필요한 라운드 키를 간단히 계산할 수 있으며, 사용되는 라운드 키는 다음 과정으로 생성된다.

우선, 1단계로 비트 키를 4개의 비트 블록 (A, B, C, D)으로 나눈다. 2단계, $K_{1,0} = G(A + C - KC_0)$, $K_{1,1} = G(B - D + KC_0)$을 계산하여 1라운드 키를 생성한다. 이때 KC_0는 회전 상수이다. 3단계, $B \| A = (B \| A) \gg 8$ 이동하여 새로운 (A, B, C, D)를 구성한다. 4단계, $K_{2,0} = G(A + C - KC_1)$, $K_{2,1} = G(B - D + KC_1)$을 계산하여 2라운드 키를 생성한다. 이때, KC_1은 회전 상수이다. 5단계, $D \| C = (D \| C) \ll 8$ 이동하여 새로운 (A, B, C, D)를 구성한다. 6단계, 위의 2단계에서 5단계까지의 과정을 16개의 라운드 키를 생성할 때까지 반복한다. 이때, KC_i는 황금비의 소수 부분으로부터 생성된 상수를 나타낸다. $B \| A = (B \| A) \gg 8$는 $B \| A$를 오른쪽으로 8비트 회전시킨 64비트이고, $B \| A = (B \| A) \ll 8$는 $B \| A$를 왼쪽으로 8비트 회전시킨 64비트이다. 실제로 KC_i는 식 (5)와 같다.

$$KC_0 = INT\left(\frac{\sqrt{5}-1}{2} \times 2^{32}\right) = 0x9e3779b9$$
$$KC_i = KC_{i-1}^{\ll 1} \qquad (1 \leq i \leq 16) \tag{5}$$

그림 2는 키 생성 알고리즘을 이용한 라운드 키 생성 과정 구조도이다.

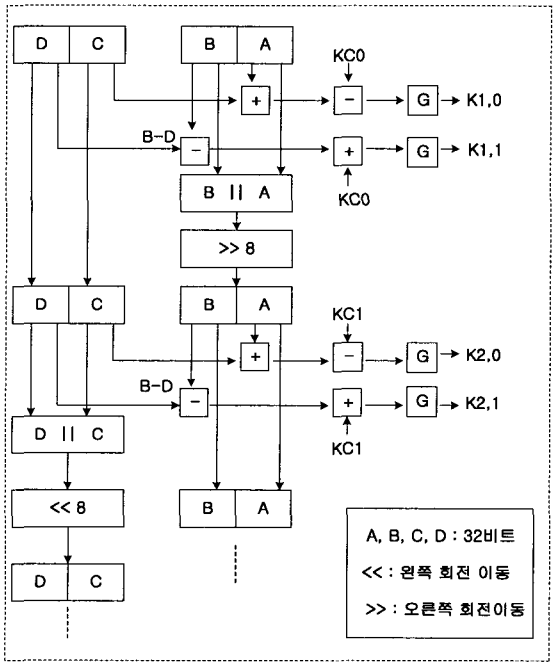


그림 2. 라운드 키 생성 과정 구조도
Fig. 2. The architecture of round key generation process.

Ⅲ. 알고리즘을 적용한 암호화 칩 설계

본 논문에서는 국내 표준 블록 암호 알고리즘 SEED을 적용하여 암호화 칩을 설계하고 그 성능을 검증한다. SEED는 많은 양의 정보를 암호화하게 되므로 빠른 암호화 수행 능력을 가지도록 하는 것이 가장 중요하다. 따라서 효율적인 정보의 암호화 처리 속도와 암호화 칩 구현시 면적의 효율성이 이루어져야 한다.

1. 하드웨어 설계 방향

그림 3은 음성 신호 암·복호화 과정을 나타낸 블록도이다.

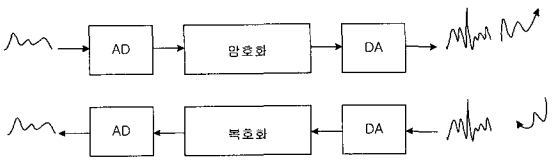


그림 3. 음성 신호 암·복호화 과정 블록도
Fig. 3. The encryption & decryption process block diagram of the voice signal.

우선 AD(Analog to Digital) 컨버터를 통해 아날로그 음성 신호를 디지털 신호를 8비트 단위로 변환하여 암호화 모듈의 입력값으로 전달한다. 이것을 버퍼링을 통해 128비트의 입력 데이터 값으로 전달한다. 전달된 데이터는 암호화 과정을 거친 후, DA(Digital to Analog) 컨버터를 거쳐 암호화된 아날로그 신호로 생성되어 임의의 통신선로를 거치게 된다. 식별할 수 없는 형태의 암호화된 신호는 무선이나 유선의 통로를 거쳐 상대방의 모듈에 전달되고, 이렇게 전달된 신호는 다시 디지털 신호로 변환된 후 복호화 과정을 거쳐 다시 원래의 음성으로 복원되어 상대방에게 전달된다.

암호화 칩 설계의 경우 대부분의 블록 암호화 알고리즘의 자체 특성 때문에 많은 하드웨어 자원이 필요하다. SEED의 경우는 한 번의 암호화 또는 복호화 과정에서 최적화 된 하드웨어 설계가 필요하므로 모든 알고리즘이 하나의 칩에서 부가적인 요소없이 음성 신호의 암호화가 이루어지도록 설계하였다. 대표적인 구현 방식으로 전라운드형, 단일 라운드형, S-box 공유형 등이 있는데, 전라운드형 방식은 일반적으로 파이프라인 방식을 적용하여 설계하며, 하드웨어 자원을 많이 필요로 하지만, 속도가 가장 빠른 장점이 있다. 단일 라운드 방식은 SEED의 16개 라운드를 1개의 단일 라운드 프로세서에 의해서 반복 수행하는 것으로 속도는 전라운드형에 비해 느리지만 많은 하드웨어 자원을 절약할 수 있다. 마지막으로 S-box 공유형으로 S-box의 수에 따라 적용 방식이 다소 차이는 있겠으나, 공통된 것은 서로 다른 S-box를 공유하여 구현하는 방식으로 S-box에 따른 하드웨어의 자원을 절약할 수 있는 장점이 있다. 본 연구에서는 위의 3가지 방식 중에서 단일 라운드형 방식을 적용하여 탑-다운(Top-down) 방식 하드웨어를 구현하였다.

2. 단일 라운드 구현 방식

하나의 라운드만을 사용하여 16회 반복하는 구현 방식이다. 키 생성부에서 생성된 16개의 서브 키(sub key)를 동시에 16개 라운드의 라운드 키 입력값으로 제공하여 16개의 단일 라운드들이 동시에 동작할 수 있는 여건을 마련함으로써 최종 출력값을 생성한다.

(1) 전체 모듈 설계

SEED는 128비트의 암호키와 128비트의 평문(plain text)을 입력 데이터로 하여 128비트의 암호문(cipher text)을 출력한다. 그림 4는 암호화 칩의 전체 모듈을 나타낸다.

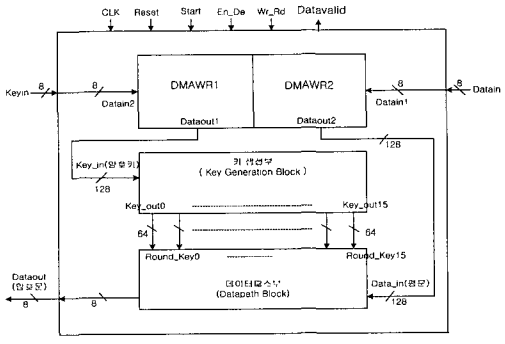
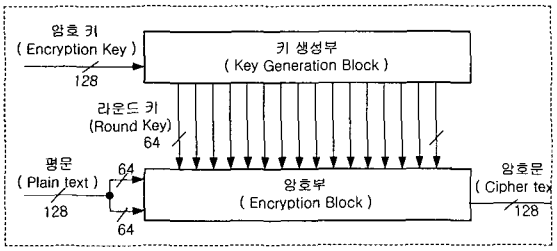


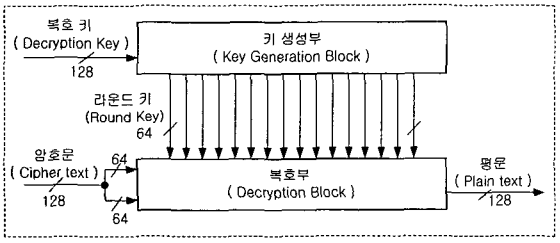
그림 4. 암호화 칩의 전체 모듈
Fig. 4. The full module of the cipher chip.

데이터와 키 입력은 8비트 단위로 입력된다. 8비트씩 입력된 데이터와 키 값은 128비트씩 각각 DMAWR1과 DMAWR2에 저장되었다가 키 생성부와 암호부로 전달된다. 키 생성부에서 16개의 키 값이 출력되어 암호부의 라운드 키 입력값으로 전달된다. 이것은 128비트 데이터 입력값과 함께 암호화 과정을 거쳐 최종으로 128비트의 암호화된 출력값을 생성한다. 그림 5는 키 생성부와 암·복호화부의 관계를 나타낸 것이다. 키 생성부

에서는 128비트 키 입력값을 받아서 64비트의 키 생성값 16개를 출력하여 암호부의 각 라운드의 라운드 키의 입력값으로 전달한다. 16개의 라운드 키와 128비트 평문 데이터를 입력받아 암호부의 F 함수와 G 함수의 연산을 수행한 후, 128비트의 암호문을 8비트 단위로 출력한다. 복호화의 경우에는 평문대신에 암호문을 입력하고, 키의 순서만 역으로 적용하여 복호화된 값을 얻는다.



(a) 키 생성부와 암호화부
(a) The key generation block and encryption block.

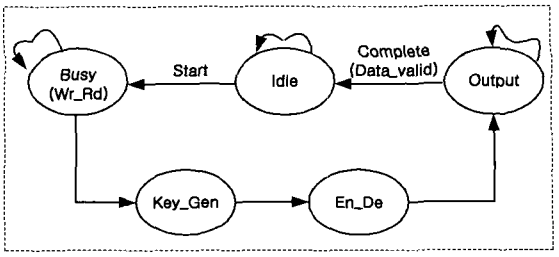


(b) 키 생성부와 복호화부
(b) (a) The key generation block and decryption block.

그림 5. 키 생성부와 암·복호화부
Fig. 5. The key generation block and encryption & decryption block.

(2) 상태 변화도

그림 6은 키 생성부와 데이터 암호부의 처리를 동작 흐름에 따라 상태 흐름도를 나타낸 것이다. Idle은 입력값을 전달받기 위한 대기 상태를 나타내고, Start 신호가 전달되면 키와 데이터값이 입력된다. 키와 데이터 입력부의 신호에 따라 버퍼링(buffering)이 완료되면, 입력키와 입력 데이터를 키 생성부와 암호부에 전달한다. 암호화부에서 16 라운드 수행이 완료되면, 암호화 또는 복호화 된 데이터를 출력부에 전송한다. Datavalid는 출력값의 생성 여부를 담당한다.



Start : 암호화 또는 복호화 시작	En_De : 암호화 또는 복호화
Busy : 진행 상태	Idle : 대기 상태
Wr_Rd : 쓰기 또는 읽기	Key_Gen : 키 생성부
Datavalid : 데이터의 암호화 또는 복호화 종료	Output : 출력

그림 6. 상태 변화도
Fig. 6. The state flow diagram.

(3) 키 생성 과정

그림 7과 같이 키 입력데이터는 128(127 downto 0) 비트 암호키를 32(31downto 0)비트씩 4개의 D, C, B, A로 나누어 연산을 수행한다. 키 생성부에서 생성된 키 결과값은 암호부의 16개의 라운드 키 입력값으로 전달된다.

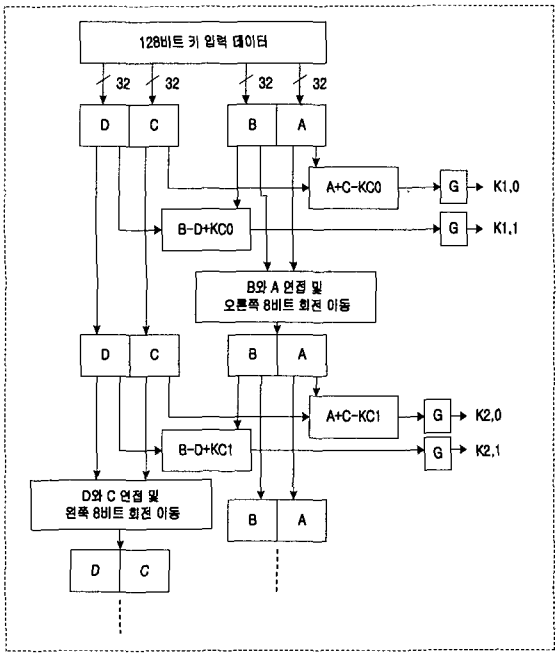


그림 7. 키 생성부의 연산 과정 블록도
Fig. 7. The operation process block diagram of key generation.

키 생성부에서는 덧셈과 뺄셈 그리고 G 함수부의 연산이 수행된다. 여기에서 KC는 회전 상수를 나타낸다. 위의 연산 과정을 식으로 나타내면 식 (9)와 같다.

$$\begin{aligned} K1,0 &= G(A + C - KC_0) \\ K1,1 &= G(B - D + KC_0) \\ K2,0 &= G(A + C - KC_1) \\ K2,1 &= G(B - D + KC_1) \\ &\vdots \\ K15,0 &= G(A + C - KC_{15}) \\ K15,1 &= G(B - D + KC_{15}) \end{aligned} \tag{9}$$

(4) 암호화 과정

암호부 설계는 라운드 블록과 이것에 포함된 F 함수부, G 함수부의 연산이 모두 수행되는 부분으로 라운드는 F 함수부와 레지스터를 포함하고, F 함수부는 G 함수부를 구성 요소로 포함하여 연산을 수행하도록 설계하였다. 그림 8은 암호부의 각 부분별 설계에 대하여 나타낸 것이다.

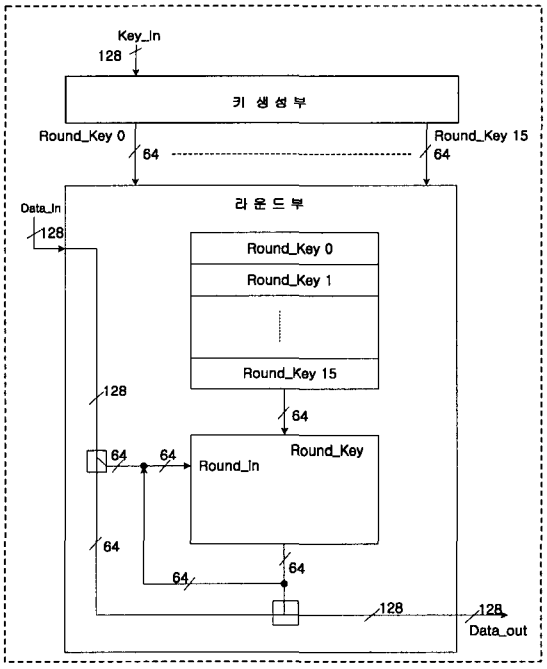


그림 8. 암호화 블록도
Fig. 8. The encryption block diagram.

키 생성 블록에서의 16개의 라운드 키가 항상 병렬적으로 공급되면 암호부에서는 이것은 버퍼에 저장하였다가 라운드부로 전달한다. 평문 입력 데이터와 라운드 키를 전달받은 후, 라운드부 내부의 F 함수 연산을 수행하여 생성된 첫 번째 라운드 출력값은 라운드 입력값으로 다시 들어가 두 번째 라운드 키 값과 함께 라운드부의 연산을 수행한다.

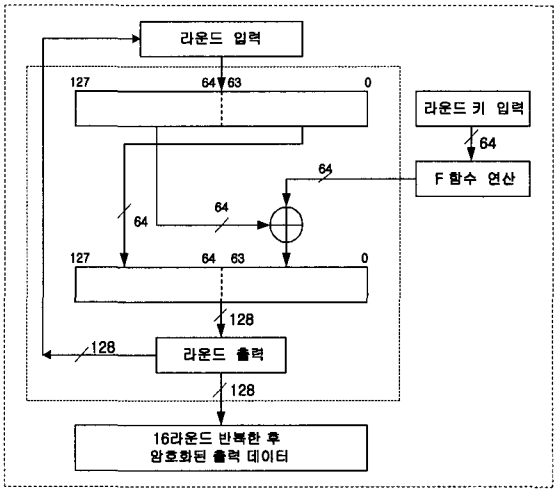


그림 9. 라운드 블록 설계도
Fig. 9. The design diagram of the round block.

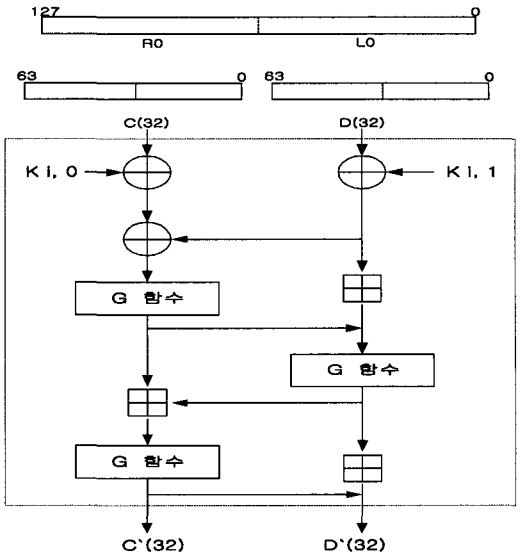


그림 10. F 함수 블록
Fig. 10. The F function block.

그림 9는 라운드 블록도로 128비트 입력 데이터를 1라운드의 라운드 입력 값으로 하고, 키 생성기에서 생성된 라운드 키를 적용하여 생성된 1라운드 출력 값을 다시 처음 라운드의 입력 값으로 반복하여 사용한다.

이렇게 16회를 반복한 후, 생성된 최종 결과 값을 서로의 위치를 변환하여 암호화된 최종 출력 값을 얻는다. 그림 10은 F 함수 블록도를 나타낸 것이다. F 함수부는 G 함수 3개와 가산기(Adder) 연산 3개로 이루어져 있고, 64비트의 데이터와 64비트 라운드 키를 입력값으로 한다. 각 세그먼트(segment)에서 수행된 연산 결과는 반복하여 처음 세그먼트의 입력값으로 전달되어 최종적인 연산 결과를 생성한다.

Ⅳ. 실험 및 고찰

SEED 암호화 칩은 Xilinx사의 XCV300PQ240을 대상으로 탑 다운(Top down) 방식으로 설계하였다. 시뮬레이션 툴(Tool)은 Foundation Express 사용하여 Functional 시뮬레이션과 합성(synthesis)을 통해 칩의 성능을 고찰하였다. 본 실험에서는 음성 신호를 암호화하기 위해 음성 아날로그 신호를 디지털 신호로 변환하는 8비트 분해능의 AD/DA 컨버터(Converter)를 사용하였다. 탑 다운 방식으로 설계된 각 부는 기능별 시뮬레이션과 파형 관측을 통해 성능을 검증하였다. VHDL 코드 작성에서 구성한 요소들은 S-box 관련 요소, F 함수와 G 함수 그리고 라운드 함수는 F 함수와 G 함수가 포함될 수 있도록 구성하였다. 구성된 라운드 함수를 주체로 하여 키 생성기에서 생성된 키 값을 가지고 16라운드를 운영하는 암호화부를 구성하고, 출력된 128비트 암호문을 8비트 단위로 출력할 수 있도록 이 구성 요소들을 하나의 모듈로 설정하여 칩을 설계하였다.

그림 11은 G 함수 연산을 포함한 F 함수의 기능 시뮬레이션 파형을 나타낸다. 단, 각 시뮬레이션에서의 테스트 벡터(test vector)는 임의의 것을 사용하였다.

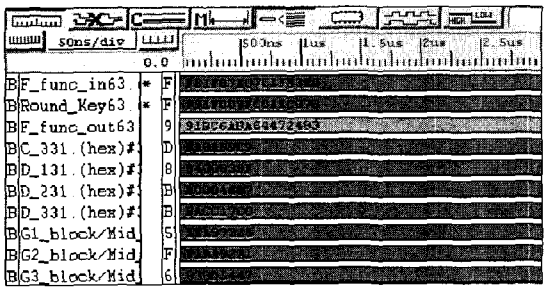


그림 11. F 함수 기능 시뮬레이션 파형
Fig. 11. The functional simulation waveform of the F function.

그림 12는 키 생성기에서 생성된 키 값을 라운드 키(Round_Key) 입력값으로 하고, 128비트 입력 데이터를 라운드 입력값(Round_in)으로 하여 각 라운드에서 생성될 수 있는 라운드 출력(Round_out)값에 대한 결과를 나타낸 것이다. 여기서의 16회의 라운드를 반복 수행하여 최종 생성된 값(Round_out)이 Data_out에 전달되어 암호화된 값이 출력된다.

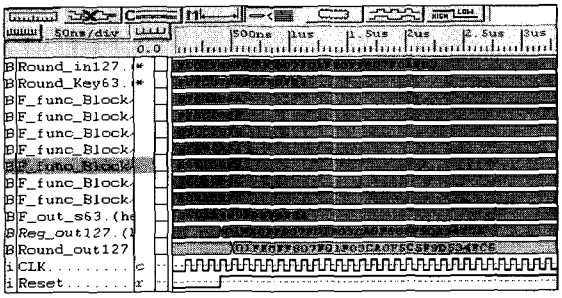


그림 12. 라운드 부의 기능 시뮬레이션 파형
Fig. 12. The functional simulation waveform of the round block.

그림 13은 128비트 키 입력(Key_in) 값을 받아 64비트로 구성된 16개의 키(Key_out) 생성값을 나타내는 시뮬레이션 파형이다. 동시에 생성된 16개의 키 값은 암호화부의 라운드 키(Round_Key) 입력값으로 전달된다.

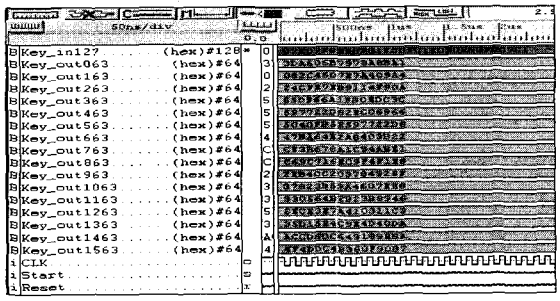


그림 13. 키 생성부의 기능 시뮬레이션 파형
Fig. 13. The functional simulation waveform of key generation block.

그림 14는 데이터 패스(datapath)부를 나타낸 시뮬레이션 파형이다.

기존의 논문에서는 하드웨어 자원 축소를 위해 단일 라운드 방식을 주로 사용하며[14-16], SEED 알고리즘에서 하드웨어 비중을 많이 차지하는 S-box 연산을 간소화하여 구현[16]하는 경우는 131.57㎒의 클록 주파수에 29Mbps의 성능을 보인다. 반면, 본 연구는 S-box1,

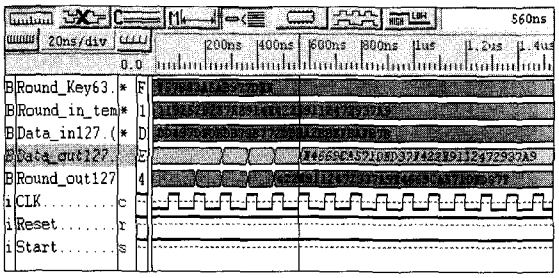


그림 14. 데이터패스부의 기능 시뮬레이션 파형
Fig. 14. The functional simulation waveform of datapath block.

S-box2를 이용한 SS-box 연산을 수행하며, 이에 기존 논문[14]보다 하드웨어 자원을 축소하고, 속도면에서 약 45% 정도 향상됨을 확인하였다. 그림 15와 그림 16은 위의 시뮬레이션과 합성을 거친 후 구현된 칩을 통해 생성된 암호화, 복호화된 파형을 나타낸다. 설계된 칩은 타겟 칩을 옵션 보드(option board)로 하는 엘리시스사

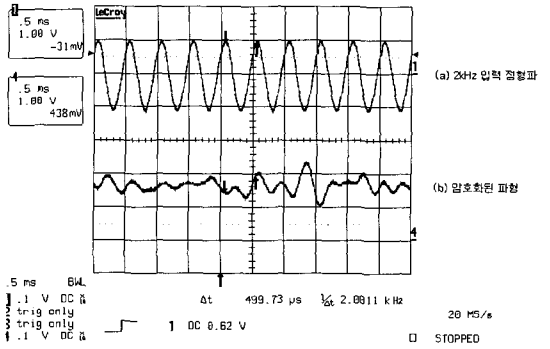


그림 15. 2㎑ 입력 정형파와 암호화된 파형
Fig. 15. The 2㎑ input sine wave and encrypted waveform.

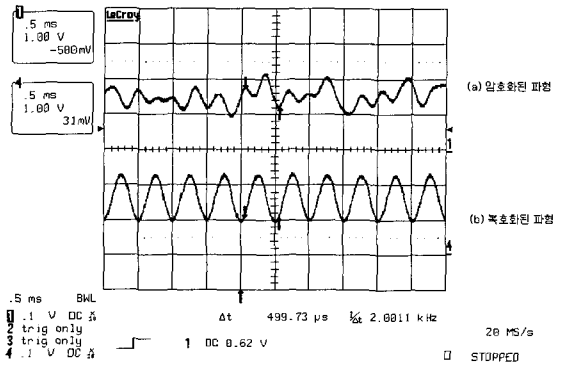


그림 16. 암호화된 파형과 복호화된 파형
Fig. 16. The encrypted waveform and decrypted waveform.

의 에뮬레이션 보드를 사용하여 그 성능을 검증하였다.

그림 15는 2㎑ 입력 정형파를 전달하여 암호화된 파형을 관측한 것이다. (a)는 2㎑ 입력 정형파이고, (b)는 암호화된 파형을 나타낸다.

그림 16은 2㎑ 입력 정형파에 대한 암호화된 파형과 복호화된 파형을 관측한 것으로 (a)암호화된 파형이고, (b)는 복호화된 파형을 나타낸다.

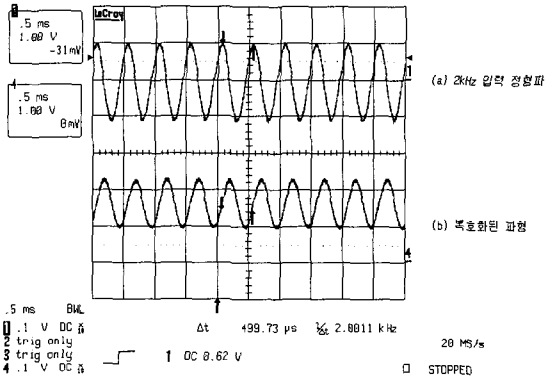


그림 17. 2㎑ 입력 정형파와 복호화된 파형
Fig. 17. The 2㎑ input sine wave and decrypted waveform.

그림 17은 2㎑ 입력 정형파와 암호화된 파형이 복호화된 것으로, (a)는 2㎑ 입력 정형파이고, (b)는 암호화된 파형을 나타낸다.

V. 결 론

본 연구는 무방비 상태로 노출되어 있는 통신상의 음성 신호를 암호화하여 상대방에게 전달함으로써 정보의 노출을 미연에 방지할 수 있도록 음성 신호 암호화 칩을 Xilinx사의 XCV300PQ240을 대상으로 설계하고, 하드웨어를 설계하였다. 설계된 칩은 최대 동작 주파수 47.895㎒이고, 전체 등가 게이트 27,285개이며, 슬라이스(slice)의 자원 활용률은 49%이다. 실제 음성 대역의 아날로그 신호를 AD/DA 컨버터를 이용하여 암호화 또는 복호화를 수행한 후의 파형을 관측함으로써 원신호가 암호화되고, 다시 원래의 신호로 제대로 복호화 됨을 확인할 수 있었다. SEED 암호 알고리즘 구조상 S-box는 설계된 칩에서 상당한 비중의 면적을 차지하고 속도 저하를 초래한다. 따라서, 이것을 ROM의 형태로 설계하면 하드웨어 자원을 줄이고, 효율성을 높일 수 있을 것으로 사료된다. 차후, 음성 신호와 더불어 대

량의 데이터 처리가 불가피한 영상 신호 암호화의 효율적 성과를 기대할 수 있다.

참 고 문 헌

[1] Charles P. Pfleeger. Security in Computing, Prentice Hall. 1989.

[2] Andreas Pfitzmann and Ralf Abmann. "Efficient Software Implementations of (Generalized)DES." SECURICOM '90. 1990.

[3] Ralph C. Merkle, "Fast Software Functions.", CRYPTO '90, 1990.

[4] Deborah Williams and Harvey J Hindin, "Can software do encryption job.", Electronics, 1980. 7.

[5] Zimmermann, A. Curiger, H. Kaeslin, N. Felber, W. Fiehtner, "A 177Mb/s VLSI Implementation of the International Data Encryption Algorithm.", IEEE Journal of Solid State Circuit, Vol. 29. No. 3, March. 1994.

[6] Han Eberle, "A High-speed DES Implementation for Network Applications." CRYPTO '92. 1992.

[7] Schneirer Bruce. Applied Cryptography, John Wiley & Sons, Inc., 1988, pp. 219-296.

[8] E. Biham and A. Shamir. Differential Cryptanalysis of DES-like cryptosystems. In A. J. Menezes and S. A. Vanstone, editors, Proc. CRYPTO 90, pp. 2-21. Springe-Verlag, 1991. Lecture Notes in Computer Science No. 537.

[9] E. Biham and A. Shamir. Differential Cryptanalysis of FEAL and N-Hash. In D. W. Davies, editor, Advances in Cryptology-Eurocrypt'91, pp.1-16, Springer-Verlag, Berlin, 1991.

[10] E. Biham and A. Shamir. Differential Cryptanalysis of the full 16-round DES. In Ernest F. Brickell, editor, Proc. CRYPTO 92, pp. 487-496. Springer-Verlag, 1992. Lecture Notes in Computer Science No. 740.

[11] M. Matsui. Linear Cryptanalysis method for DES cipher. In T. Helleseth, editor, Advanced in Cryptology-Eurocrypt'93, Vol..765 of Lecture Notes in Computer Science, pp. 386-397, Springer-Verlag, Berlin, 1994.

[12] M. Matsui. The first experimental Cryptanalysis of Data Encryption Standard. In Yvo G. Desmedt, editor, Advances in Cryptology-Crypto'94, vol.839 of Lecture Notes in Computer Science, pp. 1-11, Springer-Verlag, Berlin, 1994.

[13] 128비트 블록 암호 알고리즘(SEED) 개발 및 분석 보고서, 한국정보보호센터, 1999

[14] 신혜진, SEED 암호화 프로세서의 하드웨어 설계 및 구현, 한국항공대학교, 2000

[15] 이규원, SEED 블록 암호 알고리즘의 칩 설계 연구, 서울여자대학교, 2000

[16] 서영호, 대한민국 표준 128비트 블록 암호 알고리즘의 하드웨어 구현에 관한 연구, 광운대학교, 2000

저 자 소 개

安 寅 秀(正會員) 第37卷 SD編 第6號 參照

崔 太 燮(正會員) 第37卷 SD編 第6號 參照

林 承 河(正會員) 第36卷 T編 第3號 參照
현재 : 부천대학 전자과 교수

司空石鎭(正會員) 第37卷 SD編 第6號 參照
현재 : 국민대학교 전자정보통신공학부 교수