한국산학기술학회논문지
Vol. 11, No. 12 pp. 4962-4967, 2010

스마트 카드용 내장형 키 스케쥴러 블록 설계

송제호[1*]
[1]전북대학교 IT응용시스템공학과

Design of Inner Key scheduler block for Smart Card

Je-Ho Song[1*]
[1]**Dept. of IT Applied System Eng. Chonbuk National University**

요 약 스마트 카드는 암호알고리즘의 개발과 더불어 전자상거래 환경이 구축되면서 가치이전의 수단 및 활용분야가 다양하기 때문에 정보 통신망 환경에서 중요한 보안 장치로 수요나 활용면에서 급격한 증가율을 보이고 있다.
따라서 본 논문에서 제안한 스마트 카드용 키 스케쥴러 블록은 다양한 멀티미디어 및 실시간 통신의 암호시스템에 적용할 경우 기존시스템과 호환성이 용이하여 하드웨어 설계와 비도 및 처리속도를 향상시킬 수 있다고 사료된다.

Abstract Security of the electronic commercial transaction especially through the information communication network is gaining its significance due to rapid development of information and communication related fields. For that, some kind of cryptographic algorithm is already in use for the smart card. However, the growing needs of handling multimedia and real time communication bring the smart card into more stringent use of its resources. Therefore, we proposed a key scheduler block of the smart card to facilitate multimedia communication and real time communication.

Key Words : Smart card, Key scheduler block, Symmetric cryptography, Asymmetric cryptography, Security level

1. 서론

1990년대까지 사용된 대부분의 스트림 암호알고리즘은 LFSR(Linear Feed-back Shift Register)을 기본으로 하여 여러가지 비선형적 결합을 통해 주기가 긴 키 스트림을 발생하는 형태가 일반적이었다. 이런 형태는 하드웨어 구현에 용이하고 안전도를 수식적으로 표현할 수 있는 장점이 있지만 소프트웨어의 구현성이 문제가 되어 최근에는 RC4, SEAL3.0, ISAAC, TWOPRIME 등의 알고리즘과 부분별 혼합형 알고리즘이 제안되고 있다[1,2].

제안된 암호알고리즘은 스마트 카드에 대한 구조, 특성, 안전성, 관련 표준, 기술 및 다양한 응용부분을 고려하여 국가간 스마트 카드 표준화 기구인 ISO /IEC JTC1 SC17의 기준을 적용하였다. 본 논문에서는 스마트 카드의 암호시스템에 적합하도록 블록 및 스트림 암호알고리즘에 기반을 둔 암호알고리즘을 제안하였다[3,4].

대칭형 암호방식에(symmetric cryptography) 비대칭형 암호방식(asymmetric cryptography)개념을 적용한 새로운 암호알고리즘은 데이터 재배열, 치환, 데이터 암호블록, 키 스케쥴러로 구성되어 있고, 데이터 암호화 과정은 128 비트 평문 블록을 64 비트씩 2개의 블록으로 분할하고 확장을 거친 후 80 비트 크기를 가지는 혼합형 키를 사용하여 암호화하였다[5,6]. 또한, 단일 라운드(round)만을 사용하고도 기존 16 라운드에 해당하는 비도(security level)를 얻도록 혼합형 키(hybrid key)와 블록 암호시스템의 비선형 부분인 F 암호함수부분을 더욱 더 비선형화 시켰다. 입력 신호를 평문 및 키 데이터로 사용할 수 있고 암호문은 비인가자에게 인증용으로 적용되며 기존 시스템과 호환성이 용이하도록 하였다[7].

제안된 키 스케쥴러 알고리즘을 이용하여 Synopsys ver 1999.10으로 설계하였고 40MHz의 시스템 속도환경에서 확인하였다.

*교신저자 : 송제호(songjh@jbnu.ac.kr)
접수일 10년 10월 29일 수정일 10년 12월 16일 게재확정일 10년 12월 17일

2. 기존 암호알고리즘 구조

블록 암호의 동작모드는 크게 4가지로 분류된다. ECB 모드(Electronic Code Book)는 각각의 평문을 블록 단위로 독립적인 암·복호화 과정을 수행한다. CBC 모드(Cipher Block Chaining)는 일반적인 암호 변환방식으로서 가장 많이 사용하는 방식으로서 IV(Initialization Vector)값을 사용한다. CFB 모드(Cipher Feedback)는 블록 암호를 스트림 암호로 변환하여 암호화를 수행하는 방식으로서 블록 암호화는 의사난수데이터(pseudo random data)를 생성하기 위해 사용되며 암호문을 생성하기 위하여 평문(M_i)과 XOR 연산하고 출력된 암호문은 다음 블록에 대한 의사난수데이터를 만들기 위하여 블록 암호화로 귀환된다. OFB 모드(Output Feedback)는 독립적인 수열 데이터 블록(sequence data block) S를 자체 동기(self synchronizing) 스트림 암호문으로 변환하는 과정을 거쳐 수행된다.

블록 암호알고리즘은 DES와 같은 형태인 Feistel 구조와 치환(substitution)과 재배열(permutation)을 반복하여 사용하는 S-P 네트웍 구조등으로 구성된다. Feistel 방식은 한 라운드에 평문의 일부만 처리하여 병렬처리 효율이 낮은 반면 라운드 함수 설계의 융통성과 암·복호화 과정이 동일하다는 장점을 가진다. S-P 네트웍 방식은 한 라운드에서 전체 평문을 암호화하므로 병렬처리가 가능하여 속도가 빠르지만 복호화를 고려하여 암호화 과정을 설계하므로 설계의 폭이 좁은 단점을 가진다.

Feistel 방식의 암호화는 반복되는 블록 암호화의 특별한 형태로서 S-P 네트워크 또는 변환과 라운드 함수를 이용하며 구조는 식 1과 같다. 암호화 과정은 먼저, 평문을 우측과 좌측 반씩 두 개(L_0, R_0)로 나누고 라운드 함수 F는 서브키(K_i)를 우측에만 적용하고, F 출력은 좌측의 반과 XOR 연산을 한 후 우측으로 위치가 교환된다.

$$\begin{aligned} L_i &= R_{i-1} \\ R_i &= L_{i-1} \oplus F(R_{i-1}, K_i) \\ C &= R_i \| L_i = L_{i-1} \oplus F(R_{i-1}, K_i) \| R_{i-1} \end{aligned} \quad (1)$$

스트림 암호방식은 데이터를 비트 단위로 처리하는 암호방식으로서 주로 유럽을 중심으로 발전되었으며 블록 암호에 비하여 암호화의 속도는 빠르고 암호문을 해독하기 힘든 장점이 있다. 스트림 암호는 주로 하드웨어로 구현되며 군용통신과 같이 데이터 통신 내용이 매우 민감한 부분에 적용된다. 스트림 암호는 보통 긴 주기를 가지는 수열을 발생하여 문자열과 비트별 논리합을 하여 암호문을 발생하는 방법을 사용하기 때문에 암호의 안전성은 전적으로 키 수열의 안전성에 기인한다. 블록암호에 비하여 긴 역사를 가지고 있는 스트림암호는 키 스트림 생성기로 사용되는 LFSR, 키 생성방법에 따라 OTP(one time pad), 키 생성이 평문과 암호문에 종속되지 않는 동기 스트림 암호화(synchronous stream cipher) 혹은 KAK(key auto key), 키 생성이 평문과 암호문에 종속되는 자체 동기 스트림 암호화(self synchronous stream cipher, asynchronous stream cipher) 혹은 CTAK(cipher-text auto key)을 근간으로 하지만 최근에는 소프트웨어 구현에 적합한 여러가지 방식이 제안되고 있다. LFSR은 스트림암호에서 의사난수발생기(PRG : pseudo random generator)를 이용하여 수학적으로 분석이 가능한 이진수열을 효율적으로 발생할 수 있는 장치로 유한체 위에 정의된 선형점화식 수열로 모델링할 수 있으며 수열의 특성은 점화식에 의해 유도되는 특성다항식에 의하여 결정된다. LFSR은 암호뿐아니라 대역확산통신 등에도 많이 활용되고 구현 복잡도가 작아 빠른 속도가 요구되는 곳에 적용된다[3].

유한체 GF(2)={0,1}위에 다음과 같이 정의된 수열을 식 (2)로 나타낸다.

$$S_{j+n} = (C_0 S_{j+n-1} + C_1 S_{j+n-2} + \cdots + C_{n-1} S_j) \bmod 2, \quad j \geq 0 \quad (2)$$

여기서, $S_0, S_1, \cdots, S_{n-1}$은 초기치로 정의 된다.

이때 식 (2)의 특성다항식(characteristic polynomial) $f(x)$는 식 (3)과 같다.

$$f(x) = s^n + C_0 x^{n-1} + C_1 x^{n-2} + \cdots + C_{n-2} x + C_{n-1} \quad (3)$$

이때 C_{n-1}은 반드시 1이어야 하며 초기 값이 모두 0인 경우는 배제한다. LFSR에서는 초기 값으로부터 나머지 수열을 모두 생성할 수 있기 때문에 특성다항식 $f(x)$의 특성이 중요하다. 따라서, n개 단을 갖는 LFSR에 의하여 생성되는 수열의 특성은 n차 특성다항식에 의하여 결정된다.

유한체위에 정의된 다항식의 특성은 그 다항식의 차수(degree)와 위수(order)에 의하여 결정되며 특히 위수가 정확히 $2^n - 1$인 다항식을 원시다항식(primitive

polynomial)이라 부르며 이러한 다항식에 의하여 결정되는 LFSR을 최대주기 수열이라 한다.

수열의 비선형성의 정도를 나타내는 척도로서 가장 중요한 개념은 선형복잡도(LC : Linear Complexity)이다. 어느 수열의 선형복잡도는 그 수열을 생성하는 최소다항식(minimal polynomial)의 차수를 의미하는데 여기서 최소다항식은 수열의 특성다항식 중 가장 작은 다항식을 의미한다[8].

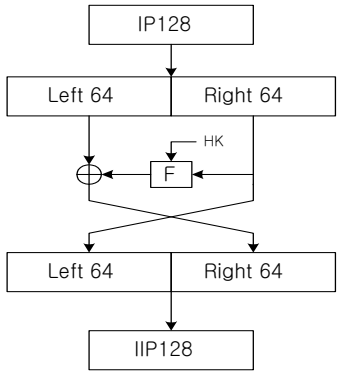
LFSR을 이용한 스트림암호 구성에 있어서 선형적인 특성은 암호시스템에서 해결해야하는 과제이다. 해결 방안으로 LFSR 단의 내용을 비선형 결합시켜 병렬 처리로 수열을 발생하는 경우 선형복잡도를 증가시킬 수 있게 두 개 이상의 LFSR을 선형 혹은 비선형으로 결합하는 것이다. 다른 방법은 시각제어 논리를 이용하는 것으로 보통 2개 이상의 LFSR로 구성되며 1개의 LFSR의 출력으로부터 다른 LFSR의 출력을 제어 하여 출력을 발생하는 기법으로 Stop & Go generator, alternating step generator, binary rated multiplier 등 많은 수열 발생기가 제안되었다. 마지막으로 메모리를 이용하는 방법으로 Summation Generator 등이 대표적인 경우이며 최근에는 FCSR(Feedback with Carry Shift Register)등 많은 이론이 발전되고 있는 단계로 이때 발생되는 수열도 특정한 조건하에서 주기 및 선형 복잡도를 매우 크게 할 수 있다.

3. 제안된 키 스케쥴러 알고리즘

암호알고리즘의 구현론에서 소프트웨어 방식은 융통성이 크며 업데이트가 용이하지만 크랙 및 해킹에 취약성을 보인다. 그러므로 본 논문에서는 정보누출 및 변조를 막고 정보보호 차원에서 하드웨어에 의한 구현을 선택하여 플랫폼의 유·출입부분에서 동작하도록 암호알고리즘을 제안하였으며 설계된 암호시스템은 대칭형 암호시스템을 기본으로 비대칭형 암호시스템 특징을 가질 수 있도록 설계하였다. 그림 1은 제안된 혼합형 암호시스템의 데이터 암호블록으로서 기존 Feistel 구조와 동일한 형태를 가진다.

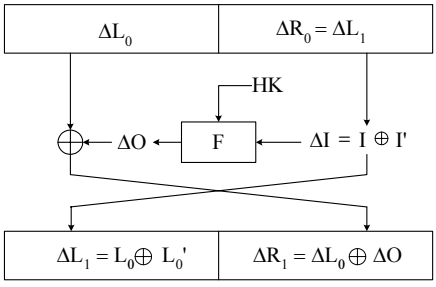
$$L_1 = R_0$$
$$R_1 = L_0 \oplus f(R_0, HK) \quad (4)$$

식 (4)는 기본 Feistel 구조와 유사하지만 키생성 및 생성된 키 정보의 형태는 다른 형태를 취하게 된다. 또한 반복에 관한 i 파라미터가 없는 대신 이전과 이후에 관한 방정식만 존재한다. 입력 128 비트는 IP를 거친 후 좌우 64비트씩 분리된다. 좌우로 분리된 64 비트는 그림 4와 같이 오른쪽 64 비트는 왼쪽으로 이동하며 왼쪽 64 비트는 혼합형 키와 F 암호함수의 연산과정 후 XOR 연산을 수행한 후 오른쪽으로 이동하게된다. 이러한 연산을 수행한 후 역초기치환(IIP : Inverse Initial Permutation)을 수행하여 암호화된 데이터를 출력하게 된다.



[그림 1] Feistel 구조와 SPN 구조

블록 암호시스템은 반복을 최소 16회 정도 반복 수행하지만 본 논문에서 사용한 Feistel 구조는 그림 2와 같이 단지 1회의 라운드에 대해서만 수행하도록 한다. 이러한 기능은 F 암호함수에 있다. F 암호함수는 일반적으로 사용되는 키 스케쥴에 의해 발생된 키를 사용하는 것이 아니고 단순 키 기능과 인증 및 비대칭형 개념을 가진 혼합된 키 값을 사용하여 비도를 더 높일 수 있다.

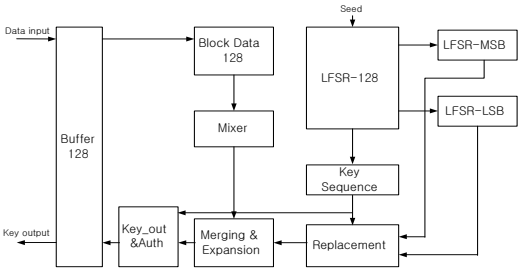


[그림 2] 단일 라운드에 대한 Feistel 구조 특성

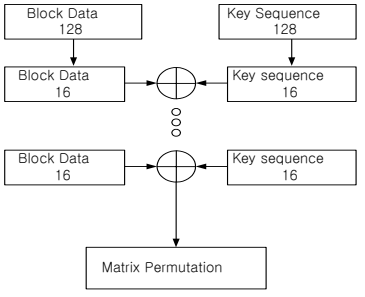
키 스케쥴러는 스트림 암호시스템으로 구성되어 있다. 그림 3은 스트림 암호방식을 적용한 키 스케쥴러 블록이다.

키 스케쥴러 입력으로는 그림 3과 같이 키 데이터가 별도로 존재하는 것이 아니고 암호화 대상 데이터가 키

스케쥴러의 입력이 되므로 그림 4와 같이 Mixer의 입력은 128 비트의 입력데이터와 LFSR로부터 생성된 키 수열이 된다.



[그림 3] 스트림 암호알고리즘의 키 스케쥴러 블록도



[그림 4] Mixer 블록

이 두가지의 데이터는 16 비트씩 8개의 블록으로 분리되며 각각 XOR 연산을 수행하게된다. 16 비트의 데이터 8 블록들은 XOR 연산수행 후 메트릭스 치환을 Merging & Expansion 블록의 입력으로 동시에 사용된다. 메트릭스 치환은 16 비트씩 8 블록이 독립적으로 XOR 연산을 수행하게 되며 행 데이터들에 대한 연산결과는 열 데이터의 형태로 출력되어진다. 입력데이터의 집합을 I, 키 수열의 집합을 K라 하면 I, K에 대한 표현은 식 (5)와 같다.

식 (5)에서 i 와 k는 각각 16 비트씩으로 구성된 스트림 데이터들이다.

$$I = \{i_0, i_1, i_2, i_3, i_4, i_5, i_6, i_7\}$$
$$K = \{k_0, k_1, k_2, k_3, k_4, k_5, k_6, k_7\} \quad (5)$$

XOR 연산을 수행한 결과를 M이라 하였을 경우 XOR 연산을 수행한 입력 데이터들은 식 (6)과 같이 표현된다.

$$M = I \oplus K \quad (6)$$

식 (6)에서 M 역시 16 비트 8 블록이므로 식 (7)과 같은 수열을 가진다.

$$M = \{m_0, m_1, m_2, m_3, m_4, m_5, m_6, m_7\} \quad (7)$$

식 (7)에 식 (5)를 대입하고 연산 수행을 행렬로 표현하여 정리하면 식 (8)과 같다.

$$M = \begin{bmatrix} ik_{00} & ik_{01} & ik_{02} & ik_{03} & ik_{04} & ik_{05} & ik_{06} & ik_{07} \\ ik_{10} & ik_{11} & ik_{12} & ik_{13} & ik_{14} & ik_{15} & ik_{16} & ik_{17} \\ ik_{20} & ik_{21} & ik_{22} & ik_{23} & ik_{24} & ik_{25} & ik_{26} & ik_{27} \\ ik_{30} & ik_{31} & ik_{32} & ik_{33} & ik_{34} & ik_{35} & ik_{36} & ik_{37} \\ ik_{40} & ik_{41} & ik_{42} & ik_{43} & ik_{44} & ik_{45} & ik_{46} & ik_{47} \\ ik_{50} & ik_{51} & ik_{52} & ik_{53} & ik_{54} & ik_{55} & ik_{56} & ik_{57} \\ ik_{60} & ik_{61} & ik_{62} & ik_{63} & ik_{64} & ik_{65} & ik_{66} & ik_{67} \\ ik_{70} & ik_{71} & ik_{72} & ik_{73} & ik_{74} & ik_{75} & ik_{76} & ik_{77} \end{bmatrix} \quad (8)$$

여기에서 ik_{32}는 $i_{32} \bigoplus k_{32}$의 연산결과임을 나타낸다.

식 (8)에 대한 매트릭스 치환은 식 (9)와 같이 전치행렬로 표시되고 연산을 통하여 생성된 64 비트는 Merging & Expansion 모듈의 입력으로 사용된다.

$$M^T = MP \quad (9)$$

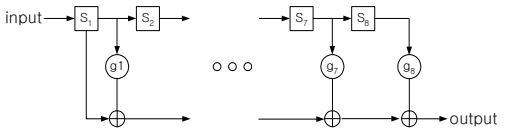
$$= \begin{bmatrix} ik_{00} & ik_{10} & ik_{20} & ik_{30} & ik_{40} & ik_{50} & ik_{60} & ik_{70} \\ ik_{01} & ik_{11} & ik_{21} & ik_{31} & ik_{41} & ik_{51} & ik_{61} & ik_{71} \\ ik_{02} & ik_{12} & ik_{22} & ik_{32} & ik_{42} & ik_{52} & ik_{62} & ik_{72} \\ ik_{03} & ik_{13} & ik_{23} & ik_{33} & ik_{43} & ik_{53} & ik_{63} & ik_{73} \\ ik_{04} & ik_{14} & ik_{24} & ik_{34} & ik_{44} & ik_{54} & ik_{64} & ik_{74} \\ ik_{05} & ik_{15} & ik_{25} & ik_{35} & ik_{45} & ik_{55} & ik_{65} & ik_{75} \\ ik_{06} & ik_{16} & ik_{26} & ik_{36} & ik_{46} & ik_{56} & ik_{66} & ik_{76} \\ ik_{07} & ik_{17} & ik_{27} & ik_{37} & ik_{47} & ik_{57} & ik_{67} & ik_{77} \end{bmatrix}$$

LFSR-128은 8단으로 구성된 LFSR이 16개 포함된 의사난수발생기로 그림 8과 같다. LFSR의 탭 계수는 $g_1, g_2, g_3, g_4, g_5, g_6, g_7, g_8$으로 설정하고 키 부호기의 초기내용은 $s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8$로 되어 있다. $2m = 16$이므로 입력 키 수열은 $K = (k_0, k_1, ..., k_{15})$이며 부호화된 출력 키 수열 $Z = (z_0, z_1, ..., z_{15})$다. 모듈러-2 연산에서 $k_j + s_j = z_j$ 가 $z_j + k_j = s_j$와 같이 교환법칙이 성립되므로 8단 스트림 키 부호 생성기를 해석하기 위하여 초기상태조건을 적용하여 정리하면 식 (10)과 같이 된다.

$$\begin{bmatrix} z_8 + k_8 \\ z_9 + k_9 \\ z_{10} + k_{10} \\ z_{11} + k_{11} \end{bmatrix} = \begin{bmatrix} k_7 & k_6 & k_5 & k_4 & k_3 & k_2 & k_1 & k_0 \\ k_8 & k_7 & k_6 & k_5 & k_4 & k_3 & k_2 & k_1 \\ k_9 & k_8 & k_7 & k_6 & k_5 & k_4 & k_3 & k_2 \\ k_{10} & k_9 & k_8 & k_7 & k_6 & k_5 & k_4 & k_3 \end{bmatrix} \begin{bmatrix} g_1 \\ g_2 \\ g_3 \\ g_4 \end{bmatrix} \quad (10)$$

$$\begin{bmatrix} z_{12} + k_{12} \\ z_{13} + k_{13} \\ z_{14} + k_{14} \\ z_{15} + k_{15} \end{bmatrix} = \begin{bmatrix} k_{11} & k_{10} & k_9 & k_8 & k_7 & k_6 & k_5 & k_4 \\ k_{12} & k_{11} & k_{10} & k_9 & k_8 & k_7 & k_6 & k_5 \\ k_{13} & k_{12} & k_{11} & k_{10} & k_9 & k_8 & k_7 & k_6 \\ k_{14} & k_{13} & k_{12} & k_{11} & k_{10} & k_9 & k_8 & k_7 \end{bmatrix} \begin{bmatrix} g_5 \\ g_6 \\ g_7 \\ g_8 \end{bmatrix}$$

$$G(x) = x^7 + x^3 + 1 \quad (11)$$



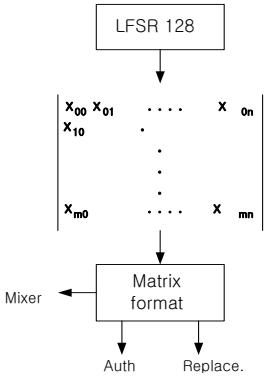
[그림 5] 8단 LFSR 의사난수발생기

이때 그림 5와 식 (10)을 식 (11) LFSR 생성다항식에 적용하여 정리하면 식 (12)가 된다.

$$
\begin{aligned}
z_0 + k_0 &= g_1 s_1 + g_4 s_6 + g_8 s_8 \\
z_1 + k_1 &= g_1 k_0 + g_4 s_5 + g_8 s_7 \\
z_2 + k_2 &= g_1 k_1 + g_4 s_4 + g_8 s_6 \\
z_3 + k_3 &= g_1 k_2 + g_4 s_3 + g_8 s_5 \\
\\
z_4 + z_4 &= g_1 k_3 + g_4 s_2 + g_8 s_4 \\
z_5 + k_5 &= g_1 k_4 + g_4 s_1 + g_8 s_3 \\
z_6 + k_6 &= g_1 k_5 + g_4 k_0 + g_8 s_2 \\
z_7 + k_7 &= g_1 k_6 + g_4 k_1 + g_8 s_1 \\
\\
z_8 + z_8 &= g_1 k_7 + g_4 k_2 + g_8 k_0 \\
z_9 + k_9 &= g_1 k_8 + g_4 k_3 + g_8 k_1 \\
z_{10} + k_{10} &= g_1 k_9 + g_4 k_4 + g_8 k_2 \\
z_{11} + k_{11} &= g_1 k_{10} + g_4 k_5 + g_8 k_3 \\
\\
z_{12} + z_{12} &= g_1 k_{11} + g_4 k_6 + g_8 k_4 \\
z_{13} + k_{13} &= g_1 k_{12} + g_4 k_7 + g_8 k_5 \\
z_{14} + k_{14} &= g_1 k_{13} + g_4 k_8 + g_8 k_6 \\
z_{15} + k_{15} &= g_1 k_{14} + g_4 k_9 + g_8 k_7
\end{aligned}
\qquad (12)
$$

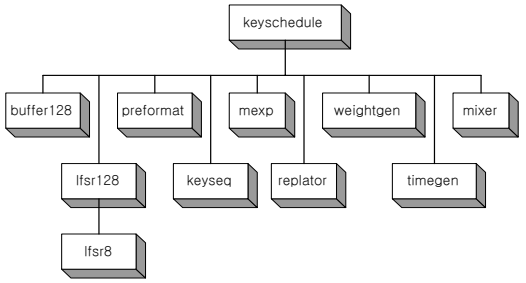
그림 6은 식 (12)를 바탕으로 키 수열을 생성시키기 위한 키 수열 생성 블록이다.

키 스케쥴러 블록은 데이터를 암호화하기 위한 키 생성블록으로 Synopsys ver 1999.10으로 설계하였다. 암호화에 필요한 키값을 생성하는데 사용되는 하부블록은 그림 7과 같이 9개의 기능블록으로 구성하였다.



[그림 6] 키 수열 생성기

데이터 암호를 위한 키 스케쥴러 블록은 스트림 암호 시스템을 기본으로 하여 LFSR의 의사난수발생 및 SPN을 동시에 사용하고 있으며 암호화를 수행하기 위한 키 생성뿐만이 아니라 송신자에 대한 인증용 데이터까지 출력한다.



[그림 7] 키 스케쥴러 블록도

buffer128 블록은 입출력 포트가 구성되어 있어서 암호화 및 정보 누출방지에 대한 게이트 역할을 수행하는 기능블록이다. buffer128 기능블록은 첫번째 관문으로써 암호화를 수행중이거나 수행한 후에 유입데이터가 비인가자에 의한 신호임이 밝혀지면 system_en 단자에 의하여 buffer128은 암호화에 관련된 모든 동작은 정지되며 비인가 신호를 차단한다. preformat 기능블록은 키 데이터를 처리하기 전에 미리 데이터에 대한 포맷을 정리하는 기능블록으로써 mixer의 입력으로 사용할 데이터 16비트씩 8개의 블록으로 구별하는 동작을 수행하도록 설계하였다.

lfsr128 블록으로 내부에는 8단으로 구성된 lfsr8 블록이 존재한다. lfsr8에 대한 생성다항식을 이용하여 구현한 lfsr8은 주기가 2^8-1의 길이를 가지지만 본 논문에서는 8T만을 사용하므로 LFSR의 주기는 크게 비도를 좌우하지 않는다. lfsr8 16개의 모듈은 각각 8T 동안 동일하게 출력된 값들은 weight & time generator, 키 수열의 입력으로 사용된다.

keyseq 기능블록은 키 수열 블록으로써 키 생성 수열을 수행하는 기능블록이다. 8 비트 16개의 블록 데이터를 입력으로 내부의 key matrix 연산을 수행한 후 mixer와 replacement 블록으로 데이터를 전송한다. 이때 인증용 데이터 16 비트를 생성하여 키 스케쥴러 블록의 최종 출력값에 더해져 인증용으로 사용된다.

weightgen 블록은 lfsr128로부터 출력되는 데이터를 재포맷하고 MSB와 LSB 데이터를 분리 및 변화시키는 기능을 수행하도록 설계하였다.

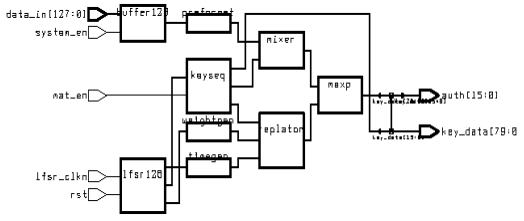
timegen 블록은 lfsr128로부터 출력되는 데이터 재포맷 기능으로 키 수열 출력에서 비밀키와 공개키로 분리하기 위하여 time slot을 생성한다. replator 기능블록은 키 수열, weight & time generator의 데이터를 받아서 새로운 데이터 포맷을 결정함으로써 time slot에 의한 암호용 키

수열 생성을 수행하도록 한다. mexp 기능블록으로써 데이터에 대한 merging과 expansion 기능을 수행한다. interleaving된 mixer 블록의 데이터와 replator 블록에서 생성된 weight &time slot을 이용하여 64 비트 크기를 가지는 새로운 블록데이터를 생성하도록 설계하였다.

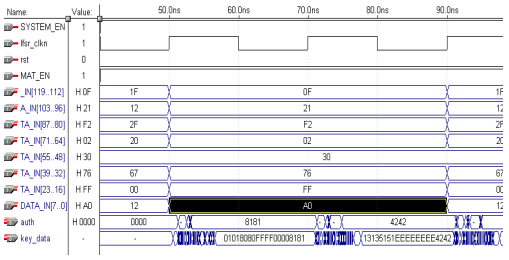
4. 모의실험 및 검증

그림 8의 키 스케쥴러 기능블록은 평문을 입력으로 받아 평문과의 상관성을 배제하고 평문을 암호화할 수 있는 키 수열을 생성하는 부분과 16 비트의 인증용 데이터를 생성하는 부분으로써 대칭형과 비대칭형 암호시스템의 기능을 모두 수행할 수 있도록 설계하였다. 키 스케쥴러 블록의 입력은 암호화하려는 데이터를 이용하였으며 제어신호에 의하여 인증용과 키 데이터를 생성하였다.

그림 9는 키 스케쥴러 블록을 Altera MAX+Plus Ⅱ툴로 모의실험 결과이다.



[그림 8] 키 스케쥴러의 기능 블록도



[그림 9] 키 스케쥴러의 모의실험 결과

5. 결론

본 논문에서는 이동성 및 네트워크 환경이 강조되는 스마트 카드의 암호시스템에 적합한 스트림 암호알고리즘에 대하여 고찰하였다. 그리고 제안한 암호알고리즘을 스마트 카드에 대한 구조, 특성, 안전성, 관련 표준, 기술 및 다양한 응용부분을 고려하여 국가간 스마트카드 표준화 기구인 ISO/IEC JTC1 SC17의 기준을 적용하였다.

대칭형 암호방식에 비대칭형 암호개념을 적용한 키 스케쥴러로를 설계하여 입력 신호를 평문 및 키 데이터로 사용할 수 있고 암호문은 비인가자에게 인증용으로 적용된다. 제안된 키 스케쥴러 알고리즘을 이용하여 스마트 카드의 암호시스템을 Synopsys ver 1999.10으로 설계하였고 40MHz의 시스템 속도환경에서 확인하였다. 제안된 알고리즘을 스마트 카드의 암호시스템에 적용할 경우 기존시스템과 호환성이 용이하여 하드웨어 설계와 비도 및 처리속도를 향상시킬 수 있다고 사료된다.

참고문헌

[1] W. Stallings, Cryptography and Network Security, Prentice Hall, 1998.

[2] B. Schneier, Applied Cryptography : Protocols, Algorithms, and Source Code in C, John Wiley & Sons, Inc., New York, USA, 1994.

[3] D. R. Stinson, Cryptography Theory and Practice, Chapman & Hall/CRC, 2002.

[4] H. Miyano, AMethod to Estimate the Number of Ciphertext Pairs for Differential Cryptanalysis, Abstracts of ASIACRYPT91, 1991.

[5] 서광석, 김창한, 암호학과 대수학, 북스힐, 1999.

[6] T. Siegenthaler, "Decrypting a Class of Stream Ciphers Using Ciphertext Only," IEEE Trans. on Computer, Vol. C-34, No. 1, pp. 81-85, Jan. 1985.

[7] R. Rueppel, "Stream Ciphers," Contemporary Cryptology: The science of Infor. Integrity, New York, IEEE Pres, pp. 65-134, 1991.

[8] 이병관, 전자상거래 보안, 남두도서, 2002.

송 제 호(Je-Ho Song) **[정회원]**

- 1991년 2월 : 원광대학교 전자공학과 (공학사)
- 1993년 2월 : 원광대학교 전자공학과 (공학석사)
- 2003년 2월 : 원광대학교 전자공학과 (공학박사)
- 1996년 3월 ~ 현재 : 전북대학교 IT응용시스템공학과 교수

<관심분야>
VLSI, 정보통신, 통신망 네트워크 시스템설계