[4-5]

다수의 원시다항식을 이용한 비선형 스트림 암호기와 오류제어에 관한 연구

진양규 임환주 김창규 이만영

한양대학교 전자통신공학과 동의대학교 전자통신공학과

A study on nonlinear stream cipher using distinct primitive polynominal and Error Control

Jin Yang-Kyu, Yim Hoan-Ju, Kim Chang-Kyu, Rhee Man-Young

Dept. of Electronic Comm. Engineering, Hanyang Univ.

*Dept. of Electronic Comm. Engineering, DongEui Univ.

Abstract

For producing a running key generator, a nonlinear combined PN sequence generator can devise using distinct primitive polynomials. The periodicity and linear complexity of the key stream are discussed. Error control codes are used in cryptographic communication to correct trasmission errors. Depending on where the codec device are attached to the cipher system, cryptographic analysis will produce different results under the different feedback schemes. In this study, a stream cipher system incoporated with the error control coding are proposed for error correction. It is found that error correction is achieved by using the external error control coding.

1. 서론.

정보화 사회로 진입함에 따라 현대 컴퓨터나 통신시스템상에서 데이타 보안이나 인증문제는 필수적인 문제가 되고 있다.[7-8] 이러한 정보의 불법적 사용을 막기위해서는 데이타 보호와 통신 상대에 대한 인증(확인)을 위해 컴퓨터및 통신망에서 암호(Cryptography)가 크게 주목을 받고있다. 암호는 비밀을 유지해야하는 정보(Information)가 담긴 비보호 메시지 즉 평문(Plaintext)를 저장하거나 전송할때, 불법적인 사용자들(Opponents)이 메시지를 도청하거나 변형시켜 합법적인 사용자들이 불이익을 받지않도록 변환하는 기술이다. 이때 변환된 메시지를 암호문(Ciphertext)이라한다. 그림 1의 암호변환과 역변환을 결정하는 매개변수를 키(Key)라 한다. 암호알고리즘은 관용 키 알고리즘(Conventional Key algorithm)과 공개 키 알고리즘(Public-key algorithm)으로 대별된다. 공개 키 알고리즘은 서로 다른 비밀키와 공개키를 이용해 키 전송이 필요치 않는 암호 방식으로 Diffie-Hellman에 의해 제안된 이래 발전한 알고리즘으로 RSA암호계, 초 증가성을 이용한 Kanpsack류의 암호계등이 있다. 관용키암호방식은 암호화키와 복호화키가 동일한 암호방식으로 DES가 대표적인 블럭암호(Block Cipher)와 스트림 암호(Stream Cipher)가 있다 (그림 2). 키 스트림은 키와 메시지의 관계에 따라서 동기스트림(Synchronous stream)와 자기동기스트림(Self-synchronous stream)으로 나눌 수 있다. 자기동기스트림에는 평문귀환법(Plaintext feedback mode)과 암호문귀환법(Ciphertext feedback mode)이 있다. 통신채널상의 자연발생 잡음을 제어하기 위하여 부호이론을 도입하는데 암호와 오류제어기중에서 어떤것을 먼저 사용하는가에 따라서 그 결과가 달라진다. 본 논문에서는 여러개의 원시다항식을 사용해서 실 멋셈한 Rueppel의 키 스트림을 주기와 복잡도, 그리고 비밀키 측면에서 향상된 새로운 키 스트림을 제안하고 그키의 랜덤성(randomness)과 상관면역 (correlation immunity)을 증명했다. 일반적인 키 스트림의 오류 확산(error propagation)에 대해서 예를 보이고 오류확산에 대한 대책으로 (15,11)2중오류 RS 부호기를 더한 효과를 컴퓨터 시뮬레이션으로 분석했다.[1-2]

2.스트림 암호기

1) 스트림 암호

스트림 암호는 평문 X를 연속된 비트열 $x_1, x_2, \cdots$로 나누어 키 스트림 Z의 각 요소 $z_1, z_2, \cdots$로 각각 EX-OR해 암호화 한다. 키 스트림 발생기는 Vernam이 제시한 one-time-pad 가 가장 이상적이나 현실성이 없기 때문에 의사무작위(psudo-random)비트열을 고려한다. 안전한 스트림 암호는 키 스트림을 예측할수 없어야 한다. 즉, 후속 키는 앞선 키 스트림으로부터 예측 할수 없어야 한다. 이러한 키 스트림의 불 예측성을 위해서는 다음 조건이 만족해야 한다.

1)긴주기: 키 스트림은 긴 주기를 가져야 한다.
2)선형복잡도: 키 스트림을 발생하는 등가 LFSR를 찾는 일이 불가능 할 정도로 선형복잡도가 커야 한다.
3)랜덤성: 큰 선형복잡도가 랜덤성을 의미하지는 않는다. 키 스트림의 통계적 특성이 이상적 랜덤소스와 같아야 한다.

2)LFSR와 열

키 스트림의 가장 일반적인 유용한 유도부 장치는 선형 귀환레지스터(LFSR)이다. 그것의 일반적인 형태는 그림 3과 같다. 여기서 $c_0, c_1, \cdots, c_{n-1}$를 피드백 계수 $s_0, s_1, s_2 \ldots$를 출력이라한다. 초기 n단의 내용 $s_0, s_1, \cdots, s_{n-1}$를 LFSR의 초기상태라 한다. 동일일차함수(homogeneous linear function)

$$f : GF(2)^n \rightarrow GF(2)$$

$$f(x_0, \cdots, x_{n-1}) = c_0 s_0 + c_1 s_1 + \cdots + c_{n-1} s_{n-1} \quad (1)$$
$$(c_i \in GF(2))$$

를 피드백함수로 가지는 n단 LFSR에 의하여 생성된 수열 s에 대하여 다음 선형점화식(linear recurence relation)이 성립한다.

$$s_{t+n} = c_0 s_0 + c_1 s_{t+1} + \cdots + c_{n-1} s_{t+n-1} = \sum_{i=0}^{n-1} c_i s_{t+i} \quad (2)$$

이러한 의미에서 GF(2)의 원소 $c_0, c_1, \cdots, c_{n-1}$를 계수로 가지는 n차 다항식

$$f(x) = c_0 + c_1 x + c_2 x^2 + \cdots + c_{n-1} x^{n-1} + x^n \quad (3)$$

를 n단 LFSR 및 s의 특성다항식(characteristic polynomial)이라하고 수열 s를 특성다항식 f(x)에 의하여 생성된 수열이라고 한다.

(예) 선형점화식 $s_{t+7} = s_{t+4} + s_{t+3} + s_{t+2} + s_t$ 일때 이에 대응하는 특성다항식은 $f(x) = 1 + x^2 + x^3 + x^4 + x^7$이다. LFSR는 그림 4와 같고 출력은 0000001001111111100101010111100110010100010001100 0 ··· 이다.

다항식 f(x)에서 $\deg(f(x)) = n \geq 1$인 f(x)에 대하여

$$f(x) \mid x^e - 1$$

인 양의 정수 e가 존재한다. 이러한 양의 정수중에서 가장 작은 양의 정수를 다항식 f(x)의 위수(order)라 하고, 이것을 ord(f(x))로 나타낸다.

정리1. f(x)가 기약(irreducible)다항식이면, 영이 아닌 수열 s은 모두 주기가 ord(f(x))인 순환수열이고, 또 $ord(f(x)) \mid 2^L - 1$이다. 특히 $ord(f(x)) = 2^L - 1$일때, f(x)를 GF(2)위의 n차의 원시다항식(primitive polynomial)이라 한다.

정리2. GF(2)상의 원시다항식에 대하여 생성되는 영이 아닌 수열은 순환수열이다.

(예) 기약다항식 $f(x) = 1 + x + x^4$는 가장 낮은 차수의 다항식 $1 + x^{15}$를 나눈다. 그래서 ord(f(x))가 15인 원시다항식으로 주기가 15이고 초기값{0,0,0,1}일때 수열은 000100110101111이다. 기약다항식 $f(x) = 1 + x + x^2 + x^3 + x^4$ 은 가장 낮은 차수의 다항식 $1 + x^5$를 나눈다. 그래서 ord(f(x))가 5로 주기가 5이고 초기값 {0,0,0,1}일때 순열은 00011이다. 이 다항식은 원시다항식이 아니다. 다항식이 원시다항식일때 생성되는 영이 아닌 수열은 주기 $2^L - 1$의 최대주기수열(Maximal period sequence)또는 PN수열(psudo-noise sequence), 또는 m열이라 한다.

정리3. 수열 s가 n차 다항식에 의하여 생성된 수열이면 기약다항식 f(x)와 수열 s는 이 수열의 연이은 임의의 2n개의 항 $s_k, s_{k+1}, \cdots, s_{k+2n-1}$에 의하여 완전히 결정된다.

$$S = \begin{bmatrix} s_k s_{k+1} & \cdots & s_{k+n-1} \\ s_{k+1} s_{k+2} & \cdots & s_{k+n} \\ \vdots & & \vdots \\ s_{k+n-1} s_{k+n} & & s_{k-2n-2} \end{bmatrix}$$

라 하면 식 (2)에 의하여

$$s_{k+n} = [s_{k+n} s_{k+n+1} \cdots s_{k+2n-1}]$$
$$= [c_0, c_1, \cdots, c_{n-1}] \cdot S$$

즉

$$[c_0, c_1, \cdots, c_{n-1}] = s_{k+n} S^{-1} \quad (4)$$

이다.

(예) n = 4 인 경우 평문 10101110이고 암호문 00110100이면 키 스트림이 10011010이다.

식 (4)에 의하여

$$[c_0, c_1, c_2, c_3] = [1,0,1,0] \cdot \begin{bmatrix} 0111 \\ 1001 \\ 1011 \\ 1111 \end{bmatrix}$$

$$= [1,1,0,0]$$이다.

따라서 기약다항식 $f(x) = 1 + x + x^4$이고 키스트림은 1001101011111000이다. 정리3에 의하여 n이 작으면 2n개 만으로 쉽게 수열 s와 특성다항식을 알수가 있다.[14] 이러한 위험을 막기위해서 생성된 수열의 주기와 선형복잡도를 크게 해야한다. 크기가 큰 LFSR를 사용한다는것은 현실적으로 어렵기때문에 그림 5 처럼 LFSR를 기본블럭으로 비선형 결합한다.[5],[11-13]

3)선형복잡도

선형복잡도란 어떤 이진열을 발생할 수 있는 가장 짧은 LFSR에 대한 특성다항식의 차수를 말한다.

정리4. 두 이진 수열 $s = \{s_t\}$, $u = \{u_t\}$의 최소다항식을 $m_s(x), m_u(x)$라 하고 수열 s, u, s+u, 의 선형복잡도를 LC(s), LC(u), LC(s+u)라고하면 다음이 성립한다.

a. $LC(s+u) \leq LC(s) + LC(u)$

b. $GCD(m_s(x), m_u(x)) = 1$이면 $LC(s+u) = LC(s) + LC(u)$

정리5. 두 이진 수열 $s = \{s_t\}$, $u = \{u_t\}$의 최소다항식을 $m_s(x)$, $m_u(x)$라하고 수열 s, u, s+u, 의 선형복잡도를 LC(s), LC(u), LC(s+u)라고하면 다음이 성립한다.

a. $LC(s \cdot u) \leq LC(s) \cdot LC(u)$

b. $m_s \cdot m_u$가 기약인 동시에 서로소이면 $LC(s \cdot u) = LC(s) \cdot LC(u)$

3. Rueppel 키 발생기.

Rueppel은 그림6의 키 발생기를 제안했다. 이것은 N개의 유도 LFSR와 모듈러연산과 메모리로 이루어진 비선형 결합 함수로 이루어 졌다. 각 LFSR의 특성다항식은 원시다항식으로 선택되고 각 차수는 서로소이다. N개의 유도 LFSR로 부터 N개의 입력디지트와 현재의 입력값과는 전혀 관련이 없는피드백 값 C(j-1)과 더해진다. 정수 S(j)가 발생되고, 이 값이 정수 이진 변환기에 입력되어 LSB가

키로 발생된다. 그리고 정수 S(j)의 다른 실 덧셈된 비트가 피드백 되어 메모리에 저장되었다가 피드백값 C(j)를 유도한다. 실 덧셈은 쉽게 장치화가 된다.

Rueppel의 키 발생기는 다음 조건을 만족한다.

(a) 긴주기 : GCD(L_i,L_j)이면, $T = \prod(2^{L_i}-1)$이다.

(b) 선형복잡도 : 근사적으로 $\prod(2^{L_i}-1)$이다.

(c) 랜덤성 : 0비트와 1비트가 거의 같게 나온다.

(d) 상관면역 : (N-1)차가 이루어 진다.

Rueppel의 키 발생기의 가장 중요한 점은 비 선형함수에 메모리를 도입하여 최대 선형복잡도와 최대 상관면역을 동시에 만족한다.[3]

1) 실 덧셈

Rueppel의 키 발생기뿐만 아니라 다른 여러 키 발생기에서 실 덧셈이 이용되는데 그 원리는 다음과 같다.

N개의 이진수 $b_1, b_2, \cdots, b_N$의 정수합을 S라 할때 즉

$$S = \sum_{i=1}^{N} b_i$$

$$S = s_0 + s_1 2 + s_2 2^2 + \cdots + s_r 2^r$$

를 S의 이진 표현이라 한다. 그때

$$s_0 = \Pi^{2^1}(b_1, b_2, \cdots b_N) \qquad (5)$$

이고 여기서

$$\Pi^k(b_1, b_2, \cdots, b_N) = \sum_{1 \le i_1 \cdots i_k \le N} b_{i_1} \cdots b_{i_k}$$

인 N개의 이진열의 곱의 합을 나타낸다. N개의 이진열의 정수합의 이진표현이 덧셈과 곱셈에 이용된다. N = 4 일때 4비트의 합을 나타내는데 3비트면 충분하며 그림 7에 나타나 있다.

2)Control레지스터를 사용한 새로운 Rueppel의 키 발생기

그림 8는 제안된 키 발생기의 블럭이다. CTLREG는 N비트를 가지고 CTLREG의 i번째의 비트는 i번째 LFSR의 열 $\{x_i\}$의 출력과 EX-OR한 결과 열 $\{y_i\}$가 생긴다. CTLREG의 N비트는 전송중에 일정하다. 이 구조에서 CTLREG의 초기값은 2^N개의 다른열을 발생 시킨다. 제안한 키발생기는 다음조건을 만족한다.

(a)주기 : 출력 z의 주기는 $T=(2^{L_1}-1)(2^{L_2}-1)\cdots(2^{L_N}-1)$. 제안된 구조는 단지 EX-OR의 존재에 의해서 Rueppel의 것과 다르다. EX-OR된 열은 EX-OR되지 않은 열과 주기가 같다.

(b) 선형복잡도 : 주기 T와 같다. Rueppel의 선형복잡도는 $LC_0 < T$ 이다. 주기는 T이다. 제안된 구조에서 더해지는 열이 0일때 M열이고 1일때 M열의 보수가 된다. 그러므로 새로 제안한 열의 선형복잡도는 $LC > LC_0$이다. 한편 선형복잡도는 주기보다 크지 않다 즉 $LC < T$ 이다. 위 식들로 부터

$$LC_0 < LC < T$$

LFSR의 차수가 크면 LC_0가 T보다 작을 확률은 0에 기깝다. 그래서 LFSR의 차수가 크면 LC가 T가 된다.

(c) 랜덤성 : LFSR는 PN열을 발생하기 때문에 열 $\{x_i\}$은 균형을 이룬다. 열 $\{y_i\}$도 균형된 열 0과 1를 갖는다. 왜냐하면 열$\{y_i\}$는 $\{x_i\}$ 그 자체나 또는 이것의 보수이기때문이다. $u_k(j)$를 j번째 y_k를 제외한 y_i와 캐리의 합의 LSB라 하면,

$$u_k(j) = \sum_{i=1}^{N} y_i(j) + c(j)$$

z는 모든 캐리의 합의 LSB라 하며,

$$z_k(j) = y_k(j) + u_k(j) \qquad (6)$$

$$p(z=1) = p(y_k=1)p(u_k=0) + p(y_k=0)p(u_k=1)$$
$$= 1/2 \cdot p(u_k=1) + 1/2 \cdot p(u_k=1)$$
$$= 1/2$$

$$p(z=0) = p(y_k=0)p(u_k=0) + p(y_k=1)p(u_k=1)$$
$$= 1/2 \cdot p(u_k=0) + 1/2 \cdot p(u_k=1)$$
$$= 1/2$$

(d) 상관면역 : 출력 z는 (N-1)차 상관면역이다.

식 (6)으로부터

$$p(z=u_k) = p(y_k=0) = 1/2$$
$$p(z \neq u_k) = p(y_k=1) = 1/2$$

z는 u_k와 상관관계가 없다. u_k는 N-1개의 변수를 가지기 때문에 z는 (N-1)차 상관면역이다.[3],[12-13]

4.스트림 암호기의 오류전파와 RS부호

1) 오류전파

스트림 암호기의 3가지 종류가 그림 9에 있다. 여기에서 키 스트림발생기로 LFSR만을 사용했다. (a)는 키 비트가 평문이나 암호문에 무관하게 생성되는 동기스트림 암호기고 (b),(c)는 키 비트가 일정수의 평문과 암호문에 지배를 받는 자가동기스트림 암호기다. 이 때 동기스트림에서 평문,암호문,키 사이의 관계식은 식 (7)와 같다. 여기서 식 우변에 평문귀환법에서는 x,암호문귀환법에서는 y를 대입하면 된다.

$$y_i = \begin{cases} x_i + \sum_{j=0}^{i-1} c_j z_{i-1-j} + \sum_{j=i}^{n-1} c_j z_{j-i} & 0 \le i \le n-1 \\ x_i + \sum_{j=0}^{n-1} c_j z_{i-1-j} & i \ge n \end{cases} \qquad (7)$$

위 식(7)에서 deciphering시에 한 비트오류가 발생하면 동기스트림 암호기에서는 평문 한 비트만 잘못 deciphering될 뿐 다른 비트에는 영향이 없다. 암호문 귀환법에서는 암호문 한 비트가 n+1회 관여 한다. 따라서 deciphering시 n+1비트의 오류가 생긴다. 평문귀환법에서는 deciphering된 평문 한 비트x_i는 암호문 y_i뿐만 아니라 이전에 deciphering된 평문 비트에 의존 한다. 따라서 오류비트 y_k에 대응하는 평문비트 x_k가 잘못deciphering되는것은 물론 그 이후로 오류를 포함하지 않는 암호문이 입력되어도 모든 평문이 오류를 포함한다.

(예) 원시다항식이 $1 + x^2 + x^5$이고 초기값이 {0,0,0,0,1} 일때 오류확산을 생각하자. 평문 'Cryptology '를 ASCII 부호화 하면 0100 0011 0111 0010 0111 1001 0111 0000 0111 0100 0110 1111 0110 1100 0110 1111 0110 0111 0111 1001 0010 0000이고 암호화 하면

00001001011001111100011011101010000100101100111110001 10111···이다. 이때 17번째 암호문 비트에 오류발생시 deciphering후 평문비트는 다음과 같다. 먼저 동기스트림 경우에는 한 비트 오류가 발생한다:0100 0011 0111 0010

1111 1001 0111 0000 0111 0100 0110 1111 0110 1100 0110 1111 0110 0111 0111 1001 0010 0000. 암호문 귀환법에서는 6비트 오류가 발생한다:0100 0011 0111 0010 1010 1001 0111 0000 0111 0100 0110 1111 0110 1100 0110 1111 0110 0111 0111 1001 0010 0000. 평문귀환법에서는 오류발생위치에서 끝까지 오류가 발생한다:0100 0011 0111 0010 1001 0000 1010 0011 1101 0011 0010 0001 1111 0001 0101 0101 0001 0011 1001 0000 1111 0011.

2)RS 부호

RS(Reed-Solomon)부호는 비 2원 BCH부호로 심벌 단위로 부호화 복호화 되기 때문에 산발오류와 연집오류를 모두 정정 할수있다. 유한체 GF(2^m)상의 RS부호는 다음과 같은 제원을 갖는다.

부호장(Code lenth):$n = 2^m-1$

정보장(Information lenth):$k = n-2t$

검사심벌(Parity check sysbol):$n-k = 2t$

최소거리(minimum distance):$d_{min} = 2t + 1$

t중 오류정정 (n,k) RS 부호의 생성다항식은

$$g(x) = (x + \alpha)(x + \alpha^2)(x + \alpha^3) \cdots (x + \alpha^{2t}) \quad (8)$$

이다. c(x)를 전송 하였을때 수신다항식은 r(x) = c(x) + e(x)이다. 복호는 e(x)를 계산하여 올바른 부호어를 얻게 된다. e(x)를 얻는절차는 다음과 같다.

(a)수신어 r(x)에서 신드롬 계산. $s_i = r(\alpha^i)$ $1 \le i \le 2t$

(b)오류위치다항식에서 오류위치를 구한다(Chein의탐지법)

$$\sigma(x) = 1 + \sigma_1 x + \cdots + \sigma_{\nu-1} x^{\nu-1} + \sigma_\nu x^\nu$$

(c)오류위치다항식과 오증요소로 부터 오류치 계산.

(d)오류정정

위의 과정을 Peterson-Gorenstein-Zierler(PGZ)알고리즘을 이용하여 컴퓨터 시뮬레이션 했다(그림 10).

3)외부오류제어기

그림 11처럼 우선 암호한 후 부호화하는것이다. 즉, 수신측에서 복호화 한후 deciphering을 해야 한다. 채널상에서 불가피하게 발생하는 잡음들은 오류정정범위내에서 쉽게 정정된다. 그러나 다른 부호어로 메시지 변경시 오류검출이 되지 않는다.

4)내부오류제어기

그림 12처럼 메시지를 부호화 한후 암호화하여 전송하며 ,수신측에서는 deciphering한후 복호 한다. 동기스트림 암호기에서는 외부오류제어기 사용시와 결과가 같다. 평문귀환법과 암호문귀환법에서는 오류전파가 생긴후 오류제어를 함으로 자연발생잡음도 오류정정이 되지 않는다.

(예) GF(2^m)상의 2중오류정정 (15,11)RS부호를 사용했다. 이때 생성다항식은 $g(x) = x^4 + \alpha^{13} x^3 + \alpha^6 x^2 + \alpha^3 x + \alpha^{10}$ 이다. 키 발생기의 특성다항식 은 $1 + x + x^5$이고 초기값은 {0,1,1,0,1}이다. 표 1.에 6가지 경우의 결과가 있다.

5)실행예

원시다항식 11개

LFSR1: $1 + x^3 + x^{17}$

LFSR2: $1 + x + x^2 + x^5 + x^{19}$

LFSR3: $1 + x^2 + x^{21}$

LFSR4: $1 + x^5 + x^{23}$

LFSR5: $1 + x^3 + x^{25}$

LFSR6: $1 + x^2 + x^{29}$

LFSR7: $1 + x^3 + x^{31}$

LFSR8: $1 + x + x^2 + x^{22} + x^{32}$

LFSR9: $1 + x^3 + x^{41}$

LFSR10: $1 + x^5 + x^{47}$

LFSR11: $1 + x^6 + x^{71}$

초기값은 첫번째 원소를 1로 하고 나머지 모두를 0으로 한다. CTLREG의 초기값도 마찬가지다. 그림 13과 같다. 이 키 발생기는 2^N의 비밀키가 생긴다. 그림 13와 같은 새로 제안한 키 스트림을 사용하여 오류전파가 없는 동기스트림법에 평문 'computer communication'를 예를들어 산발오류와 연집오류의 경우에 내부오류제어기와 외부오류제어기가 같은 결과를 얻는다. 표 2에 외부오류제어기 사용한 경우, 표 3에 내부오류 제어기 사용한 경우의 예이다

5. 결론

PN열을 발생하는 원시다항식을 최고 차수가 서로소가 되게 선형적으로 결합하면 최대의 주기와 복잡도를 갖는다. 실 덧셈을 이용한 Rueppel의 키 발생기에 제어레지스터를 도입하여 같은 초기값에 대해서 2^n개의 다른열을 발생하므로 2^n개의 비밀키가 새로 생겨 Rueppel의 키 발생기보다 좋다. 이 키 발생기의 주기와 복잡도, 랜덤성, 상관면역을 증명했다. 스트림 암호기에서의 오류전파를 실제 예를 들어 설명했고 이러한 오류전파를 제어하기 위하여 부호기를 도입하는데 외부오류제어기를 사용하면 채널상의 자연발생잡음은 간단히 복호된다. 동기스트림 암호기에서는 내부오류제어기와 외부오류제어기의 효과가 같다.

참고문헌.

[1] Rhee, M. Y., Error Correcting Coding Theory, McGraw-Hill, New York, 1989.

[2] 이 만 영, BCH부호와 Reed-Solomon부호, 민음사, 1990.

[3] R.A Rueppel, Analysis and Design of Stream Ciphers, Springer Verlag, New York, 1986.

[4] F.Jessie MacWilliams and Neil J.A.Sloane, Pseudo-Random Sequences and Arrays, Proceeding of The IEE, Vol.64, No.12, Dec 1976.

[5] Geffe, P.R. How to Protect Data with Ciphers that are Really Hard to Break, Electronic, Jan.4, 1973.

[6] Massey, J.L. Shift Register Synthesis and BCH Decoding, IEEE Trans.on IT.VOL.IT-15, Jan 1969.

[7] D.E.R.Denning, Cryptography and Data Security, Addision-Wesley Publishing Company, 1982.

[8] C.H.Meyer, S.M Matyas, Cryptography: a New Dimentionin Computer Date Security, Joh Wiley & Sons, Inc., New York, 1982.

[9] Lidl Niederreiter, Encyclopedia of Mathmatics and Its Applications, 20. Algebra, 1983.

[10] W.Diffie, M.E.Hellman, Privacy and Authentication: An Introductin to Cryptogaphy, Proc. IEEE Vol 67 Mar, 1979.

[11] Pless, V.S. Encrypting Schemes for Computer Confidentiality, IEEE Trans.on Computer, Vol.C-26, No.11, 1977.

[12] Siegenthaler, T. Correlation-Immunity of Nonlinear Combining Functions for Cryptographic Applications, IEEE Trans. on IT, Vol.IT-30, No.5, 1984.

[13] Siegenthaler, T. Decrypting a Class of Stream Ciphers Using Ciphertext Only, IEEE Trans. on Computer, Vol. C-34, No. 1, 1985.

[14] 박 승 안, 이진 수열의 선형복잡도, 제1회정보보호와 암호에 관한 워크샵 논문집.

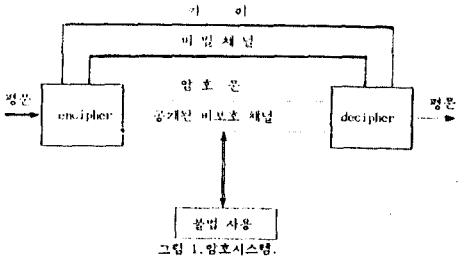


그림 1. 암호시스템.

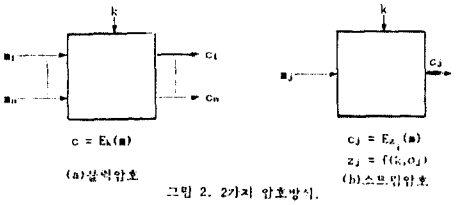


그림 2. 2가지 암호방식.

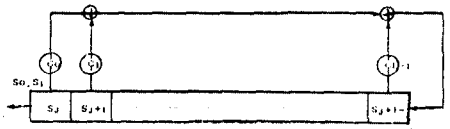


그림 3. 일반적인 LFSR.

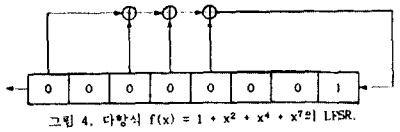


그림 4. 다항식 $f(x) = 1 + x^2 + x^4 + x^7$의 LFSR.

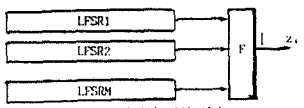


그림 5. 비선형 결합 함수

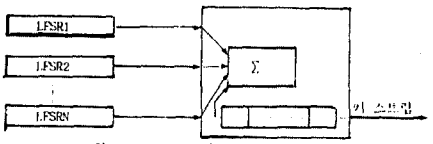


그림 6. Rueppel의 키스트림 발생기

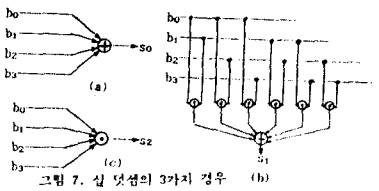


그림 7. 십 덧셈의 3가지 경우

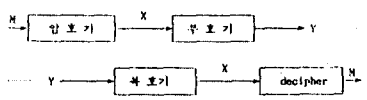


그림 11. 외부오류제어.

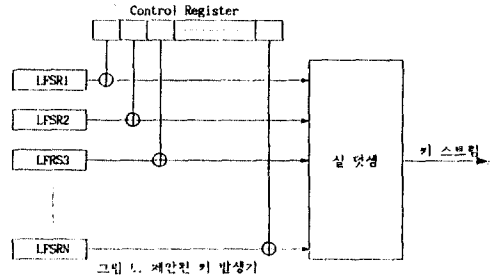
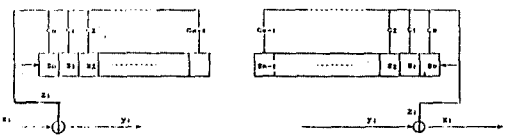
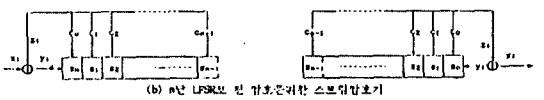


그림 8. 제안된 키 발생기


(a) n단 LFSR 로 된 이진귀환이며 스트림암호기


(b) n단 LFSR로 된 암호문귀환 스트림암호기

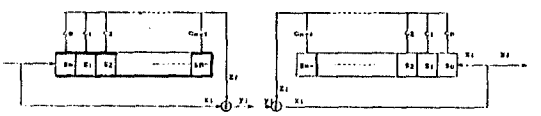

(c) n단 LFSR로 된 평문귀환 스트림암호기

그림 9. 스트림 암호기

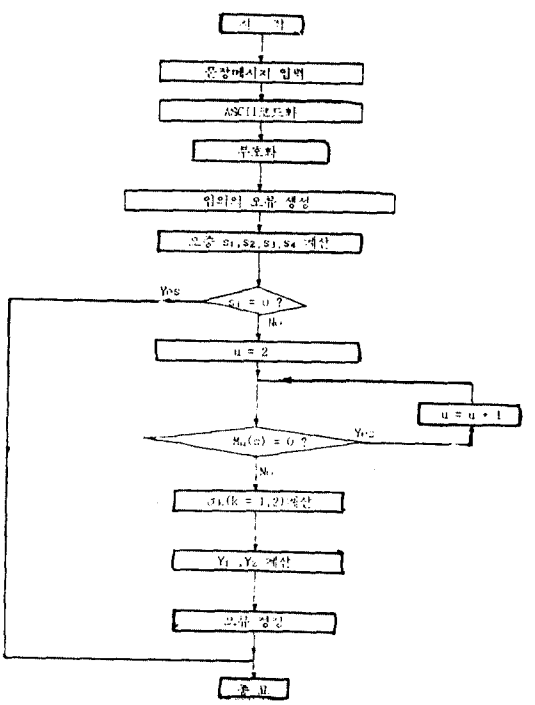


그림 10. (15,11) RS부호시스템의 컴퓨터 시뮬레이션을 위한 신호 흐름도.

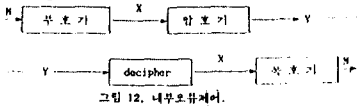


그림 12. 내부오류제어.

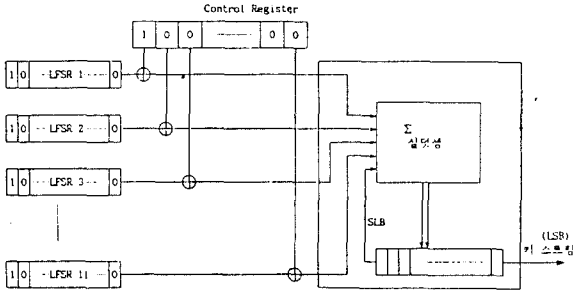


그림 13. 11개의 LFSR를 이용한 키 발생기.

평문: Stream ciphers may be further subdivided into synchronous and self-synchronizing system .

ASCII값:

0101 0011 0111 0000 0111 0010 0110 0101 0110 0001 0110
1101 0010 0000 0100 0011 0110 1001 0111 0000 0110 1000
0110 0101 0111 0010 0111 0011 0010 0000 0110 1101 0110
0001 0111 1001 0010 0000 0110 0010 0110 0101 0010 0000
0110 0110 0111 0101 0111 0010 0111 0100 0110 1000 0110
0101 0111 0010 0010 0000 0111 0011 0111 0101 0110 0010
0110 0100 0110 1001 0111 0110 0110 1001 0110 0100 0110
0101 0110 0100 0010 0000 0110 1001 0110 1110 0111 0100
0110 1111 0111 0011 0111 1001 0110 1110 0110 0011 0110
1000 0111 0010 0110 1111 0110 1110 0110 1111 0111 0101
0111 0011 0010 0000 0110 0001 0110 1110 0110 0100 0010
0000 0111 0011 0110 0101 0110 1100 0110 0110 0010 1101
0111 0011 0111 1001 0110 1110 0110 0011 0110 1000 0111
0010 0110 1111 0110 1110 0110 1001 0111 1010 0110 1001
0110 1110 0110 0111 0010 0000 0111 0011 0111 1001 0111
0011 0111 0100 0110 0101 0110 1101 0010 0000 0010 1110

키이지동키이:

외부오류제어기: Stream ciphers may be further subdivided into synchronous and self-synchronizing system .

내부오류제어기: Stream ciphers may be further subdivided into synchronous and self-synchronizing system .

암호문귀환법:

외부오류제어기: Stream ciphers may be further subdivided into synchronous and self-synchronizing system .

내부오류제어기: Streaf+ulphers may be further subdivided into synchronous and self-synchronizing system .

평문귀환법:

외부오류제어기: Stream ciphers may be further subdivided into synchronous and self-synchronizing system .

내부오류제어기: [illegible]

표 1. 6가지 경우의 예.

```
****Key Auto-Key Mode****
*External error control*
Plaintext:computer communication

ASCII of Plaintext:
0110 0011 0110 1111 0110 1101 0111 0000 0111 0101 0111
0100 0110 0101 0111 0010 0010 0000 0110 0011 0110 1111
0110 1101 0110 1101 0111 0101 0110 1110 0110 1001 0110
0011 0110 0001 0111 0100 0110 1001 0110 1111 0110 1110
Encrypted plaintext: *Ciphering*
0111 0111 0010 1100 1110 1110 1101 1011 1110 1110 0001
0000 0110 0011 1101 1110 1010 0100 0011 0010 1110 1001
0100 1101 1110 1000 1000 0000 0001 0110 0100 0100 0110
1010 1010 0001 1000 0001 1000 0011 0101 0000 0100 1000
Encoding of Plaintext:
0111 0111 0010 1100 1110 1110 1101 1011 1110 1110 0001 1000 0010 0000 0010
0000 0110 0011 1101 1110 1010 0100 0011 0010 1110 1001 1100 1000 0101 1001
0100 1101 1110 1000 1000 0000 0001 0110 0100 0100 0110 0001 1101 1100 1001
1010 1010 0001 1000 0001 1000 0011 0101 0000 0100 1000 0010 1110 1010 0001
Plaintext with error:
0111 0111 0010 1100 1110 1110 1101 1011 1110 1110 0001 1000 0010 0000 0010
0000 0110 0011 1101 1110 1010 0100 0011 0010 1110 1001 1100 1000 0101 1001
0100 1101 1110 1000 1000 0000 0001 0110 0100 0100 0110 0001 1101 1100 1001
1010 1010 0001 1000 0001 1000 0011 0101 0000 0100 1000 0010 1110 1010 0001
Decodered Plaintext: **Decoding**
0111 0111 0010 1100 1110 1110 1101 1011 1110 1110 0001
0000 0110 0011 1101 1110 1010 0100 0011 0010 1110 1001
0100 1101 1110 1000 1000 0000 0001 0110 0100 0100 0110
1010 1010 0001 1000 0001 1000 0011 0101 0000 0100 1000
Deciphered Plaintext: **Deciphering**
0110 0011 0110 1111 0110 1101 0111 0000 0111 0101 0111
0100 0110 0101 0111 0010 0010 0000 0110 0011 0110 1111
0110 1101 0110 1101 0111 0101 0110 1110 0110 1001 0110
0011 0110 0001 0111 0100 0110 1001 0110 1111 0110 1110
Plaintext:
computer communication
```

표 2. 키보오딩에이기.

```
****Key Auto-Key Mode****
*Internal error control*
Plaintext:computer communication

ASCII of Plaintext:
0110 0011 0110 1111 0110 1101 0111 0000 0111 0101 0111
0100 0110 0101 0111 0010 0010 0000 0110 0011 0110 1111
0110 1101 0110 1101 0111 0101 0110 1110 0110 1001 0110
0011 0110 0001 0111 0100 0110 1001 0110 1111 0110 1110
Encoding of Plaintext: **Encoding**
0110 0011 0110 1111 0110 1101 0111 0000 0111 0101 0111 0100 0111 0001 0000
0100 0110 0101 0111 0010 0010 0000 0110 0011 0110 1111 1001 0010 1010 1110
0110 1101 0110 1101 0111 0101 0110 1110 0110 1001 0110 1111 0110 1001 0001
0011 0110 0001 0111 0100 0110 1001 0110 1111 0110 1110 0011 0010 0001 1101
Encrypted plaintext: **Ciphering**
0111 0111 0010 1100 1110 1110 1101 1011 1110 1110 0001 0000 0111 0111 1010
1000 1110 0001 0010 0011 1010 0110 0100 0011 1110 1010 0110 0111 1101 0110
0100 0000 0110 0100 1011 0101 1001 1011 1000 0011 0101 0000 0100 1111 0000
1010 1011 0111 0110 0100 1111 1110 0011 0011 0000 1001 0010 0000 0110 1111
Plaintext with error:
0111 0111 0010 1100 1110 1110 1101 1011 1110 1110 0001 0000 0111 0111 1010
1000 1110 0001 1101 1100 1010 0110 0100 0011 1110 1010 0110 0111 1101 0110
1000 0000 0110 0100 1011 0101 1001 1011 1000 0011 0101 0000 0100 1111 0000
1010 0100 0111 0110 0100 0000 1110 0011 0011 0000 1001 0010 0000 0110 1111
Deciphered Plaintext: **Deciphering**
0110 0011 0111 1111 0110 1101 0111 0000 0111 0101 0111 0100 0111 0001 0000
0100 0110 0101 1000 1101 0010 0000 0110 0011 0110 1111 1001 0010 1010 1110
1010 1101 0110 1101 0111 0101 0110 1110 0110 1001 0110 1111 0110 1001 0001
0011 1001 0001 0111 0100 1001 1001 0110 1111 0110 1110 0011 0010 0001 1101
Decodered Plaintext: **Decoding**
0110 0011 0110 1111 0110 1101 0111 0000 0111 0101 0111
0100 0110 0101 0111 0010 0010 0000 0110 0011 0110 1111
0110 1101 0110 1101 0111 0101 0110 1110 0110 1001 0110
0011 0110 0001 0111 0100 0110 1001 0110 1111 0110 1110
Plaintext:
computer communication
```

표 3. 내부오류제어기.