通信情報保護學會論文誌
第 8 卷 第 2 號, 1998. 6

5중 오류정정 (31, 21) RS 부호의 효율적인 복호 알고리즘과 VHDL 시뮬레이션

강 경 식* , 박 진 수**,

Efficient Decoding Algorithm of 5-error-correcting(31, 21) RS Code and VHDL Simulation

kyung-Sik, Kang, Jin-Soo, Park

요 약

RS 부호의 복호 기법은 전체 통신 시스템의 성능 및 복잡도에 큰 영향을 미친다. 지금까지 RS 부호의 복호 기법은 다양한 방법에 있으나 Euclid 알고리즘과 변환복호기법을 이용한 복호 기법은 오류정정능력이 큰 복호 기법으로 널리 적용되고 있다. 본 논문에서는 오류정정능력이 5이상인 RS 부호의 복호 알고리즘에 적용될 수 있는 효율적인 복호 알고리즘을 제시하고, 이를 이용하여 5중 오류 정정 (31, 21) RS 부호기 및 복호기를 설계하고 VHDL을 사용한 컴퓨터 시뮬레이션을 통해서 그 타당성을 검증하였다.

Abstract

Efficient decoding algorithm which is applicable to the error correcting coding circuit for wireless mobile communication system is presented. By using the proposed decoding algorithm, the pair of encoder and decoder are designed for (31, 21) RS codes with 5-error-correcting capability. And to show the validity of the presented algorithm, computer simulations are performed with VHDL. As a result of simulation, the validity of decoding algorithm is demonstrated.

본 연구는 주성대학 교내지원 연구비에 의하여 수행 되었음.
*주성 대학 공학 1학부
**청주대학교 전자 · 정보통신 · 반도체공학부

제 1 장 서 론

RS 부호의 부호화 및 복호화 알고리즘은 유한체 이론(Field Theory)에 기초를 두고 있다. RS 부호의 부호화 및 복호 회로는 60년도부터 개발되어 왔고 80년도 이후에 많은 통신 시스템의 오류정정로에 활용되고있다.

응용 분야는 Campact Disk, Digital Audio Tape, 디지탈통신망 및 컴퓨터 저장 시스템의 오류정정회로로 활용되고 있다. 이의 응용은 VLSI 기술의 발달과 효율적인 복호 알고리즘의 개발, 정보원의 디지털화, 그리고 전송 장치에서 풍부한 오버헤드 확보 등으로 크게 확산될 예정이다.[1, 2]

RS 부호의 복호 단계는 크게 오증(Syndrome)을 계산하는 단계, 오증으로부터 오류위치다항식을 계산하는 단계, 오류위치다항식의 계수와 오증 요소를 이용한 각 오류위치에 대응되는 오류치를 구하는 단계, 그리고 실제로 오류를 정정하는 단계로 구성된다. 복호기의 복잡도에 크게 영향을 미치는 회로는 유한체상에서는 승산, 제산, 그리고 역원등을 수행하는 연산기 회로, 오증으로부터 오류위치를 계산하는 회로, 오류위치 다항식의 계수와 오증으로부터 오류치를 계산하는 회로 등이다. 따라서 위의 세 부분의 복잡도를 감소시키면 전체 복호기의 복잡도를 감소시킬 수 있다. 유한체상에서의 연산기회로는 RS 부호의 부호기 뿐만아니라 복호기의 복잡도를 감소시키는 매우 중요한 부분이다. 따라서 복잡도를 줄이면서 고속동작이 가능한 연산기 구조가 요구된다. RS부호의 복호알고리즘은 크게 PGZ(Peterson-Gorenstein-Zierler)복호 기법, 오류위치 다항식을 반복적으로 쉽게 구할 수 있는 Berlekamp-Massey의 반복 알고리즘, 오류위치다항식으로부터 오류 위치를 구하기 위한, Chien의 복호 알고리즘, Blahut 등이 제시한 유한체 푸리에 변환을 위한 복호 기법, 오류정정능력이 5 이상인 RS 부호의 복호 알고리즘으로 적합한 Euclid 알고리즘을 이용한 복호법, 유한체상의 방정식의 해 및 Trace 함수의 특성을 이용하여 실현된 Perkinghom 복호기법, 그리고 오증 요소로부터 오류위치와 오류치가 동시에 계산될 수 있는 직접 복호 알고리즘 등의 다양한 알고리즘들이 제안되어 왔다. PGZ 복호 알고리즘은 오류정정능력이 3 이하인 RS 부호의 복호시 매우 적합한 복호기법으로 알려져 있다.[3, 4, 5, 6]

본 논문에서는 복호지연시간을 줄이기위해 유한체의 이산(Discrete) Fourier변환을 이용하여 오류를 정정하는 복호기법을 제시하였고 복잡도를 줄이기 위해 오류위치다항식을 구하는데 역원계산이 필요없는 효율적인 Euclid 알고리즘을 제시하고 이를 이용하여 오류정정능력이 5인 (31,21) RS부호의 부호기 및 복호기를 설계하였다.

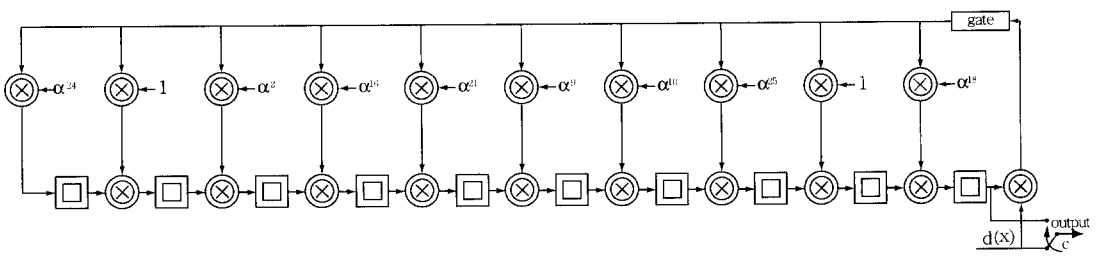


그림 1 (31,21) RS 부호기

제 2 장 Euclid 알고리즘을 이용한 RS 부호의 복호 알고리즘

Euclid 알고리즘을 이용한 RS 부호의 복호 단계는 다음과 같이 6 단계로 구성된다.

(단계 1) 수신 계열로부터 식 (1)과 같은 오증 요소를 계산한다. 여기서, N 은 RS 부호의 부호장이고, r_n $(0 \le n \le N-1)$은 수신된 부호 벡터이며, $E_k = S_k$ $(1 \le k \le 2t)$은 변환된 오류위치 값으로 정의한 일반적인 RS 부호의 오증요소 계산회로는 그림 2와 같다.

$$S_k = \sum_{n=0}^{N-1} r_n \alpha^{nk},\ 1 \le k \le 2t \qquad (1)$$

(단계 2) 오증 다항식 $S(x) = \sum_{k=1}^{2t} S_k x^{2t-k}$와 $A(x) = x^{2t}$를 이용하고 개선된 Euclid 알고리즘을 적용하여 식 (2)와 같은 다항식을 구한다.

$$\lambda(x) = \lambda_0 x^t + \lambda_1 x^{t-1} + \cdots + \lambda_t \qquad (2)$$

(단계 3) 오류 위치 다항식은 식 (2)와 같은 $\lambda(x)$를 이용하면 식 (3)과 같이 구해진다. 여기서 $\sigma_k = \frac{\lambda_k}{\lambda_0}$ 이다.

$$\sigma(x) = x^t + \sigma_1 x^{t-1} + \cdots + \sigma_t \qquad (3)$$

(단계 4) 오류 계열의 변환 오류계열의 나머지 오증 요소 E_k를 식 (4)를 이용하여 계산한다.

$$E_{2t+j} = \sum_{k=1}^{t} (-1)^k \sigma_k E_{2t+j-k} \qquad (4)$$

(단계 5) 실제적인 오류 값을 구하기 위해서 $GF(2^m)$상의 E_l의 역변환 값 e_l을 식 (5)을 이용하여 계산한다.

$$e_l = \sum_{n=0}^{N-1} E_k \alpha^{-lk},\ 0 \le l \le N-1 \qquad (5)$$

(단계 6) 최종적으로 수신 부호 값과 오류값을 더해주면 오류는 정정된다.

제3장 5중 오류 정정 (31, 21) RS 부호기 및 복호기 설계

본 장에서는 (31, 21) RS 부호의 부호기 및 복호기의 세부 회로를 설계한다.

가. (31, 21) RS 부호기

(31, 21) RS 부호의 부호기는 생성 다항식을 이용하여 설계된다. (31, 21) *RS* 부호의 생성 다항식은 식 (6)과 같이 구해진다. 여기서 α의 최소다항식 $P(x) = 1 + x^2 + x^5$ 이다.

$$\begin{aligned} g(x) &= \prod_{i=1}^{10} (x + \alpha^i) \\ &= x^{10} + \alpha^{18} x^9 + x^9 + \alpha^{25} x^7 + \alpha^{10} x^6 + \alpha^9 x^5 + \alpha^{21} x^4 + \alpha^{16} \\ &\quad x^3 + \alpha^2 x^2 + x + \alpha^{24} \end{aligned} \qquad (6)$$

식 (6)을 이용하여 설계된 (31, 21) RS 부호의 부호기는 그림 1과 같다.

나. 오증 요소 계산 회로

오증 요소는 식 (7)을 이용하여 구한다.

$$\begin{aligned} S_i &= r(\alpha^i) \\ &= r_0 + r_1 \alpha^i + r_2 \alpha^{2i} \cdots + r_{n-1} \alpha^{(n-1)i} \\ &= (\cdots (r_{n-1} \alpha^i + r_{n-2}) \alpha^i + r_{n-3}) \alpha^i + r_{n-4}) \alpha^i + \cdots + r_1) \\ &\quad \alpha^i + r_0) \end{aligned} \qquad (7)$$

식 (7)을 이용하여 설계된 (31, 21) RS 부호의 오증요소 계산 회로는 그림 2와 같다.

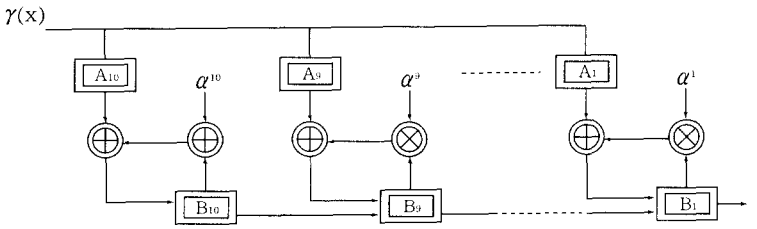


그림 2 (31, 21) RS 부호의 오증요소 계산 회로

모든 수신 계열 $r(x)$가 입력된 후, 레지스터 B_i에 있는 값이 계산된 S_i 값이다. 즉, $B_1 = S_1$, $B_2 = S_2$, ... $B_{10} = S_{10}$이다. 그리고 오증다항식은 식 (8)과 같이 정의한다.

$$S(x) = \sum_{k=1}^{10} S_k \; S^{10-k} \tag{8}$$

다. 오류위치 다항식 계산 회로

식 (9),(10)을 구한 후, 식 (11)의 반복과정을 수행한다.

$$A(x) = x^{10} \tag{9}$$

$$S(x) = \sum_{k=1}^{10} S_k \; x^{10-k} \tag{10}$$

$$\mu_i(x)A(x) + \lambda_i \; (x)S(x) = R_i \; (x) \tag{11}$$

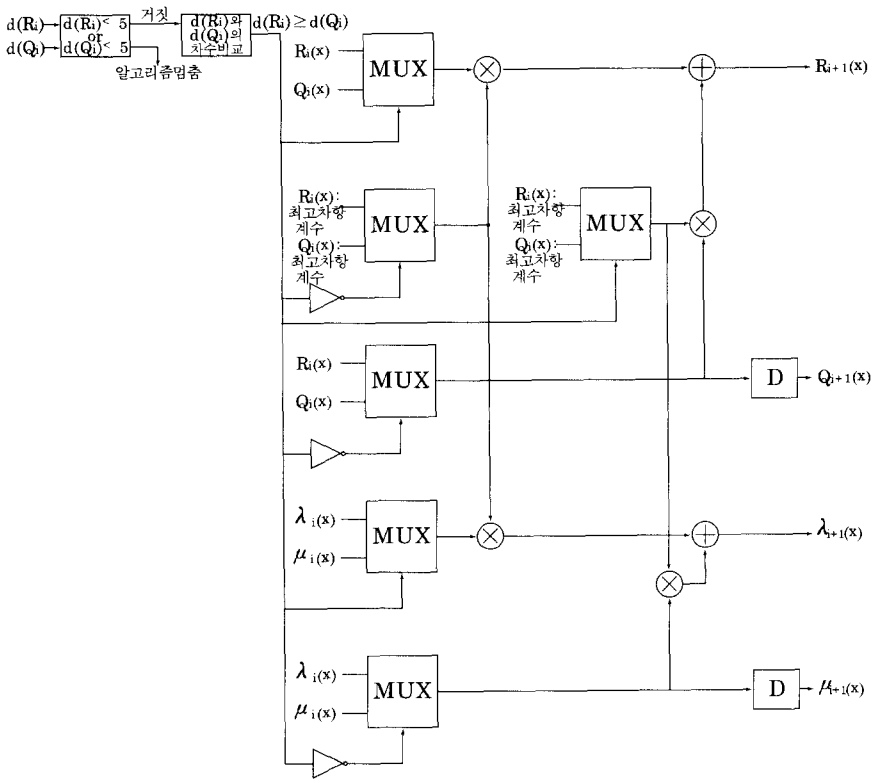


그림 3 $R_i(x)$, $Q_i(x)$, $\lambda_i(x)$, $\mu_i(x)$의 계산을 위한 반복회로

이 과정은 $R_i(x)$ 차수가 5 이하가 될 때까지 수행한다. Euclid 알고리즘의 초기 조건은 식 (12)와 같다. 식 (12)와 같은 초기치를 이용하여 식(13)~식(16)과 같은 $\lambda_i(x)$와 $Ri(x)$를 Ri(x)의 차수가 5차 이하가 될 때까지 반복한다. 알고리즘이 종료될 때 $\lambda_i(x)$는 $\lambda(x)$가 된다.

$$R_0(x) = A(x),\ \theta_0(x) = S(x) \quad (12)$$
$$\lambda_0(x) = 0,\ \mu_0(x) = 1$$
$$R_i(x) = [a_{i-1}\, b_{i-1}\, R_{i-1}(x) + \bar{\sigma}_{i-1}\, a_{i-1}\, \theta_{i-1}(x)] - x^{|l_{i-1}|}[\sigma_{i-1} a_{i-1}\ \theta_{i-1}(x) + \bar{\sigma}_{i-1} b_{i-1}\ R_{i-1}(x)] \quad (13)$$
$$\theta_i(x) = \sigma_{i-1}\theta_{i-1}(x) + \bar{\sigma}_{i-1}\ R_{i-1}(x) \quad (14)$$
$$\lambda_i(x) = [\sigma_{i-1} b_{i-1}\ \lambda_{i-1}(x) + \bar{\sigma}_{i-1}\ a_{i-1}\ \mu_{i-1}(x)] - x^{|l_{i-1}|}[\sigma_{i-1}\ a_{i-1}\ \mu_{i-1}(x) + \bar{\sigma}_{i-1} b_{i-1}\lambda_{i-1}(x)] \quad (15)$$
$$\mu_i(x) = \sigma_{i-1}\ \mu_{i-1}(x) + \bar{\sigma}_{i-1}\ \lambda_{i-1}(x) \quad (16)$$

여기서, a_{i-1}는 $R_{i-1}(x)$의 최고차 항의 계수이고, b_{i-1}은 $\theta_{i-1}(x)$의 최고차 항의 계수이며, $|l_{i-1}| = deg(R_{i-1}(x) - deg(\theta_{i-1}(x)))$이다. 그리고 σ_{i-1}는 식 (17)과 같이 구해진다.

$$\sigma_{i-1} = 1,\ \text{if } |l_{i-1}| \geq 0 \quad (17)$$
$$\bar{\sigma}_{i-1} = 0,\ \text{if } |l_{i-1}| < 0$$

위에서 구한 λ(x)로부터 오류 위치 다항식 σ(x)을 구하는 회로는 식 (3)을 실현한 그림 4와 같다.

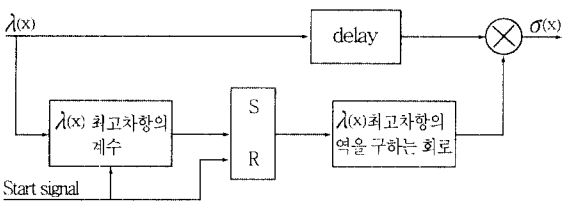


그림 4 l(x)로부터 s(x)를 구하는 회로

라. 오류위치 다항식으로 부터 변환 오류위치 다항식 계산

전체 변환오류는 식 (18)을 이용하여 구한다. 이는 $\sigma(x)$을 이용하여 구한다. 여기서, E1, … , E10은 오증 요소들 S1, … , S10 이다.

$$E_{10+j} = \sum_{k=1}^{t} \sigma_k\ E_{10+j-k} \quad (18)$$

식 (18)을 이용하여 설계된 오류가 5개 발생한 경우의 변환 오류를 구하는 회로는 그림 5와 같다.

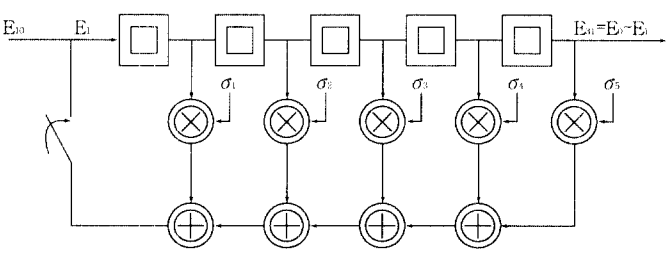


그림 5 변환오류 계열를 구하는 회로

마. 역변환 회로

변환오류 E_k를 역변환하면 오류값 e_k를 구할 수 있다.

$$e_k = \sum_{n=0}^{30} E_n \ \alpha^{nk} \ ,0 \le k \le 30$$

$$= (\cdots(((E_{30}\alpha^k + E_{29})\alpha^k + E_{28})\alpha^k + \cdots)\alpha^k + E_0) \quad (19)$$

식 (19)를 이용하여 설계된 시간 영역 오류값을 계산하는 회로는 그림 6과 같다.

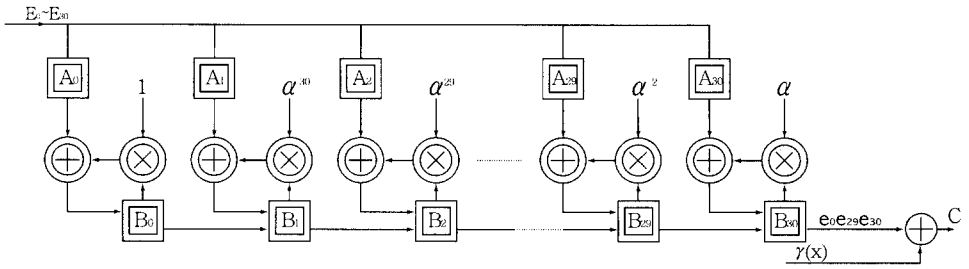


그림 6 Ei를 역변환하여 ei를 구하는 오류정정 회로

바. PGZ(Peterson-Gorenstein-Zierler)복호기법과 효율적인 복호알고리즘과의 비교

PGZ복호알고리즘을 이용한 복호기는 식 (20)과 식 (21)에 근거하여 계산된 결과는 표 1과 같다. (여기서 오류의 갯수는 v, 오류치를 $Y_i = e_i$, 오류위치번호를 $Z_i = \alpha^{l_i}$)

$$\begin{aligned} S_1 &= Y_1Z_1 + Y_2Z_2 + Y_3Z_3 + \cdots + Y_vZ_v \\ S_2 &= Y_1Z_1^2 + Y_2Z_2^2 + Y_3Z_3^2 + \cdots + Y_vZ_v^2 \\ S_3 &= Y_1Z_1^3 + Y_2Z_2^3 + Y_3Z_3^3 + Y_4Z_4^3 + Y_vZ_v^3 \\ &\vdots \\ S_{2t} &= Y_1Z_1^{2t} + Y_2Z_2^{2t} + Y_3Z_3^{2t} + Y_4Z_4^{2t} + Y_vZ_v^{2t} \end{aligned} \quad (20)$$

$$\left|\begin{array}{c|cccccc} S_v & S_0 & S_1 & \cdots & S_{v-2} & S_{v-1} \\ S_{v+1} & S_1 & S_2 & \cdots & S_{v-1} & S_v \\ \vdots & & & \vdots & & \\ S_{2v-1} & S_{2v-2} & S_v & \cdots & S_{2v-3} & S_{2v-2} \end{array}\right| \cdot \left|\begin{array}{c} \sigma_v \\ \sigma_{v-1} \\ \vdots \\ \sigma_1 \end{array}\right| \quad (21)$$

5중오류정정(31,21) RS부호의 복호기를 PGZ복호알고리즘을 이용한 복호기의 소자수와 비교하면 표 2와 같다.

〈표 1〉 PGZ의 복호알고리즘을 이용한 복호기의 복잡도

비교항목 / 오류의 개수	곱 셈 기	XOR 수	역원계산기
$t=1$	2	4	2
$t=2$	16	32	4
$t=3$	150	132	5
$t=4$	1058	722	6
$t=5$	11346	4028	7
합 계	12572	4918	24

〈표 2〉 5중(31,21)RS부호의 복잡도 비교

비교항목 \ 종 류	PGZ 복호 알고리즘	효율적인 복호알고리즘
곱셈기수	12572	61
X-OR 수	4918	216
역원 계산기수	24	1
2×1 MUX수	0	6
인버터수	0	3
비교기수	0	2

사. 예 제

(31, 21) RS부호에서 부호다항식 $C(X)=0$를 전송했을 때 수신 다항식이 $r(x)=\alpha^{4}x^{20}+\alpha^{19}x^{15}+\alpha^{5}x^{10}+\alpha^{10}x^{5}+\alpha^{11}x$ 라면 이를 복호하는 과정은 다음과 같다.

1) 오증 계산

식 (7)을 사용하며 오증을 구하면 다음과 같다.

$r(x)=\alpha^{4}x^{20}+\alpha^{19}x^{15}+\alpha^{5}x^{10}+\alpha^{10}x^{5}+\alpha^{11}x$

S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9	S_{10}
α^{21}	α^{28}	α^{7}	α^{28}	α^{4}	α^{16}	α^{23}	α^{6}	α^{8}	1

2) 오류 위치 다항식 계산

식 (9), (10)에 의해서

$A(x)=x^{10}$

$S(x)=\alpha^{21}x9+\alpha^{28}x^{8}+\alpha^{7}x^{7}+\alpha^{28}x^{6}+\alpha^{4}x^{5}+\alpha^{16}x^{4}+\alpha^{23}x^{3}+\alpha^{6}x^{2}+\alpha^{8}x+1$

이다.

식 (13), (14), (15), (16)을 이용해서$R_i(x)$, $Q_i(x)$, $\lambda_i(x)$, $\mu_i(x)$를 구하는데$R_i(x)$ or $Q_i(x)$ 〈5까지 다음을 반복한다.

(단계 1)

l_0	1
s_0	1
$R_1(x)$	$\alpha^{28}x^{6}+\alpha^{7}x^{5}+\alpha^{28}x^{7}+\alpha^{4}x^{6}+\alpha^{16}x^{5}+\alpha^{23}x^{4}+\alpha^{6}x^{3}+\alpha^{8}x^{2}+x$
$Q_1(x)$	$Q_0(x)$
$\lambda_1(x)$	x
$m_1(x)$	$m_0(x)$

(단계 2)

l_1	1
s_1	1
$R_2(x)$	$\alpha^{23}x^{4}+\alpha^{17}x^{7}+\alpha^{3}x^{5}+\alpha^{11}x^{3}+x^{2}+\alpha^{14}x+\alpha^{28}$
$Q_2(x)$	$Q_1(x)$
$\lambda_2(x)$	$\alpha^{21}x+\alpha^{28}$
$m_2(x)$	$m_1(x)$

(단계 3)

l_2	-1
s_2	0
$R_3(x)$	$\alpha^{21}x^{8}+\alpha^{30}x^{7}+\alpha^{30}x^{6}+\alpha^{27}x^{5}+\alpha^{23}x^{4}+\alpha^{11}x^{3}+\alpha^{25}x^{2}++\alpha x+\alpha^{23}$
$Q_3(x)$	$R_2(x)$
$\lambda_3(x)$	$\alpha^{11}x^{2}+\alpha^{18}x+\alpha$
$m_3(x)$	$\lambda_2(x)$

(단계 4)

l_3	0
s_3	1
$R_4(x)$	$x^7+\alpha^{22}x^6+\alpha^{21}x^5+\alpha^{15}x^4+\alpha^6x^3+\alpha^{27}x^2+\alpha^{12}x+\alpha^{13}$
$Q_4(x)$	$Q_3(x)$
$\lambda_4(x)$	$\alpha^3x^2+\alpha^{28}x+x$
$m_4(x)$	$m_3(x)$

(단계 5)

l_4	-1
s_4	0
$R_5(x)$	$\alpha^{12}x^7+\alpha^{13}x^6+\alpha^{13}x^5+\alpha^{29}x^4+x^3+\alpha^{10}x^2+\alpha^{21}x+\alpha^{28}$
$Q_5(x)$	$R_4(x)$
$\lambda_5(x)$	$\alpha^{26}x^3+\alpha^{20}x^2+\alpha^{14}x+\alpha^{28}$
$m_5(x)$	$\lambda_4(x)$

(단계 6)

l_5	0
s_5	1
$R_6(x)$	$\alpha^{17}x^6+\alpha^{21}x^5+\alpha x^4+\alpha x^3+\alpha^{13}x^2+\alpha^{19}x+\alpha^{23}$
$Q_6(x)$	$Q_5(x)$
$\lambda_6(x)$	$\alpha^{26}x^3+\alpha^{17}x^2+\alpha^{11}x+\alpha^{23}$
$m_6(x)$	$m_5(x)$

(단계 7)

l_6	-1
s_6	0
$R_7(x)$	$\alpha^{10}x^6+\alpha^7x^5+\alpha^{26}x^4+\alpha^{12}x^2+\alpha^{29}x+\alpha^{20}$
$Q_7(x)$	$R_6(x)$
$\lambda_7(x)$	$\alpha^{26}x^4+\alpha^{17}x^3+\alpha^{28}x^2+\alpha^{15}x+\alpha^{20}$
$m_7(x)$	$\lambda_6(x)$

(단계 8)

l_7	0
s_7	1
$R_8(x)$	$\alpha^{13}x^5+\alpha^{18}x^4+\alpha^{11}x^3+\alpha^{29}x^2+\alpha^{20}x+\alpha^{23}$
$Q_8(x)$	$Q_7(x)$
$\lambda_8(x)$	$\alpha^2x^4+\alpha^{16}x^3+\alpha^{16}x^2+\alpha^8x+\alpha^{23}$
$m_8(x)$	$m_7(x)$

(단계 9)

l_8	-1
s_8	0
$R_9(x)$	$\alpha^{10}x^5+\alpha^{24}x^4+\alpha^{21}x^3+\alpha^{13}x^2+\alpha^4x+\alpha^5$
$Q_9(x)$	$R_8(x)$
$\lambda_9(x)$	$\alpha^0x^5+\alpha^{23}x^4+\alpha x^3+\alpha^8x^2+\alpha^{20}+\alpha^5$
$m_9(x)$	$\lambda_8(x)$

(단계 10)

l_9	0
s_9	1
$R_{10}(x)$	$\alpha^{13}x^4+\alpha^4x^3+\alpha^{28}x^2+x+\alpha^{11}$
$Q_{10}(x)$	$Q_9(x)$
$\lambda_{10}(x)$	$\alpha^{22}x^5+\alpha^{27}x^4+\alpha^6x^3+\alpha^{23}x+\alpha^{11}x+\alpha^{11}$
$m_{10}(x)$	$m_9(x)$

$R_{10}(x)$의 차수가 4차이므로 $\lambda_{10}(x)$로부터 오류 위치 다항식 $\sigma(x)$을 구한다.

3) 오류위치 다항식

식 (3)을 이용하면 $\lambda_{10}(x)$로부터 $\Sigma(x)$를 구할 수 있다.

$$\lambda_{10}(x)=\alpha^{22}x^5+\alpha^{27}x^4+\alpha^6x^3+\alpha^{23}x+\alpha^{11}x+\alpha^{11}$$
$$\sigma(x)=x^5+\alpha^5x^4+\alpha^{15}x^3+\alpha x^2+\alpha^{20}x+\alpha^{20}$$

4) 오류위치 다항식으로부터 변환 오류 위치 다항식 계산

식 (18)을 이용하면

E_{11}	E_{12}	E_{13}	E_{14}	E_{15}	E_{16}	E_{17}	E_{18}	E_{19}	E_{20}	E_{21}
α^{22}	α^{26}	α^{3}	α^{29}	α^{27}	α^{28}	1	α^{10}	α^{10}	α^{20}	α^{21}
E_{22}	E_{23}	E_{24}	E_{25}	E_{26}	E_{27}	E_{28}	E_{29}	E_{30}	E_{31}	$=E_0$
α^{24}	α^{27}	α^{24}	α^{12}	α^{24}	α^{30}	α^{15}	α^{16}	α^{20}		α^{5}

5) 오류값 *ei*

식 (19)를 이용하면

e_0	e_1	e_2	e_3	e_4	e_5	e_6	e_7	e_8	e_9	e_{10}
0	α^{11}	0	0	0	α^{10}	0	0	0	0	α^{5}
e_{11}	e_{12}	e_{13}	e_{14}		e_{15}	e_{16}	e_{17}	e_{18}	e_{19}	e_{20}
0	0	0	0		α^{19}	0	0	0	0	α^{4}
e_{21}	e_{22}	e_{23}	e_{24}		e_{25}	e_{26}	e_{27}	e_{28}	e_{29}	e_{30}
0	0	0	0		0	0	0	0	0	0

제 4 장 C 및 VHDL 시뮬레이션

본 장에서는 오류 정정 능력이 5인 (31, 21) RS부호의 부호기 및 복호기를 C 언어로 시뮬레이션한 결과를 보인다. 시뮬레이션을 위한 복호알고리즘에서 제시된 오류정정능력이 5인 (31, 21) RS부호의 부호화 및 복호알고리즘을 이용하였다. (31, 21) RS부호의 부호화 및 복호 루틴은 부호기 시뮬레이션 루틴과 복호기 시뮬레이션 루틴으로 구성된다. 부호기 프로그램은 enc.c이며 복호기 프로그램은 main 루틴에서 12개의 외부 함수를 호출하여 실현하였다.

main 루틴에서 사용한 외부함수는 다음과 같다.

```
extern int    gf[31][5];
extern void   syndrome(int S[10][5]);
extern void   init_DATA(int DataRam[31][5]);
extern void   mult(int *, int *, int *);
extern int    SearchDegree(int a[10][5]);
extern int    SearchHighOrder(int x[10][5]);
extern void   xor(int *, int *, int *);
extern void   move(int *, int *);
extern void   zero(int *);
extern void   inv(int *, int *);
extern int    or(int S[10][5]);
extern void gf_init();
```

(31, 21) RS부호의 시뮬레이션을 위한 외부 함수의 기능은 다음과 같다.

gf[31][5]: $p(a) = 1 + a^2 + a^5 = 0$에서 Galois Field $GF(2_5)$값이 저장된 배열이다.

syndrome(int S[10][5]): 오증요소를 구하는 함수이다.

init_DATA(int DataRam[31][5]): 수신 데이타를 보관하는 함수이다.

mult(int *, int *, int *): 원소 B와 원소 C를 곱하며 A에 저장하는 함수이다.

SearchDegree(int a[10][5]): 최고차를 구하는 함수이다.

SearchHighOrder(int x[10][5]): 최고차항을 구하는 함수이다.

xor(int *, int *, int *): B xor C하며 A에 저장하는 함수이다.

move(int *, int *): B의 내용을 A에 옮기는 함수이다.

zero(int *): 모든 요소를 "0"으로 만드는 함수이다.

inv(int *, int *): B의 역원을 A에 저장하는 함수이다.

or(int S[10][5]): S의 내용이 모두 "0"이면 "0"을 출력하고, 하나라도 "1"이

면 "1"을 출력하는 함수이다.

gf_init() : $GF(2^5)$를 생성하는 함수이다.

프로그램 시뮬레이션 결과 복호 알고리즘이 정상적으로 동작함을 확인하였다. 이 결과는 VHDL로 부호화한 후 이를 논리레벨 시뮬레이션하였다. 본 VHDL 실현은 RS부호의 복호기를 FPGA로 실현할 때 적극적으로 활용될 수 있다. VHDL은 다음을 가정하고 실현하였다.

$C(x)$ = "0"라 가정하고, 수신다항식 $r(x) = a^{15}x^4 + a^{12}x^3 + a^2x^2 + a^{19}x + a^{23}$은 그림 7과 같다.

이를 근거로 계산된 오증 다항식은 $S(x) = \alpha^6x^9 + \alpha^7x^8 + \alpha^{25}x^7 + \alpha^{29}x^6 + \alpha^2x^5 + \alpha^2x^4 + \alpha^{12}x^3 + \alpha^2x^2 + \alpha^{10}x + \alpha^6$ 이고, 이는 그림 8과 같다. 오증 다항식을 근거로 계산 $\lambda(x) = \alpha^{13}x^5 + \alpha^{28}x^4 + \alpha^3x^3 + \alpha^5x^2 + \alpha^3x + \alpha^{23}$ 된 $\lambda(x)$와 오류위치 다항식 값은 $\sigma(x) = x^5 + \alpha^{15}x^4 + \alpha^{21}x^3 + \alpha^{23}x^2 + \alpha^{21}x + \alpha^{10}$ 이고, 이는 그림 9과 같이 시뮬레이션 되었다.

$\sigma(x)$와 $S(x)$로 계산한 변환된 오류 값은 $e(x) = a^{15}x^4 + a^{12}x^3 + a^2x^2 + a^{19}x + a^{23}$ 이고, 이는 그림 10와 같이 시뮬레이션 되었다. 그림 11은 $C(x) = r(x) + e(x)$ 하여 오류가 정정됨을 확인하였다.

여기서, mcount 는 클럭의 수, mreset은 전체 reset clk, cx는 복호된 부호어, wrx는 수신된 데이터, wb는 error 값, ws는 오증요소, wsgma는 σ값, wrmda는 λ값이다.

모든 부호어가 "0"일 때 오류는 그림 7의 뒷부분에 5개 발생했다고 가정하고 $C(x)$에 오류가 복구된 것을 확인하였다. 따라서 제시된 복호알고리즘의 정당성을 확인할 수 있었다.

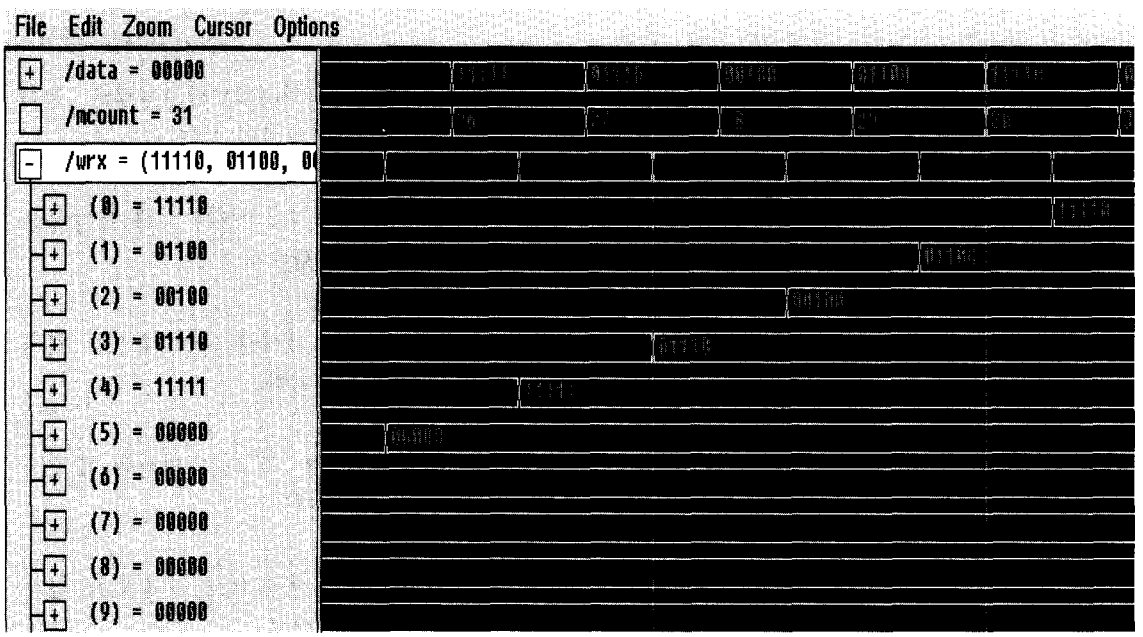


그림 7 수신된 데이터

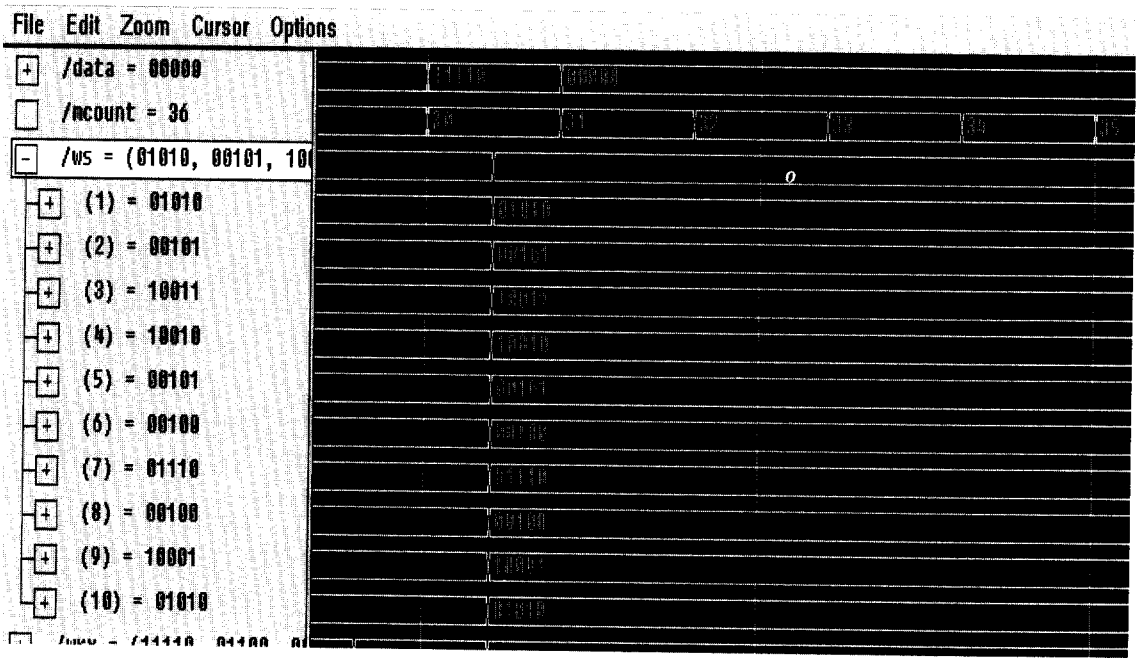


그림 8 오증요소

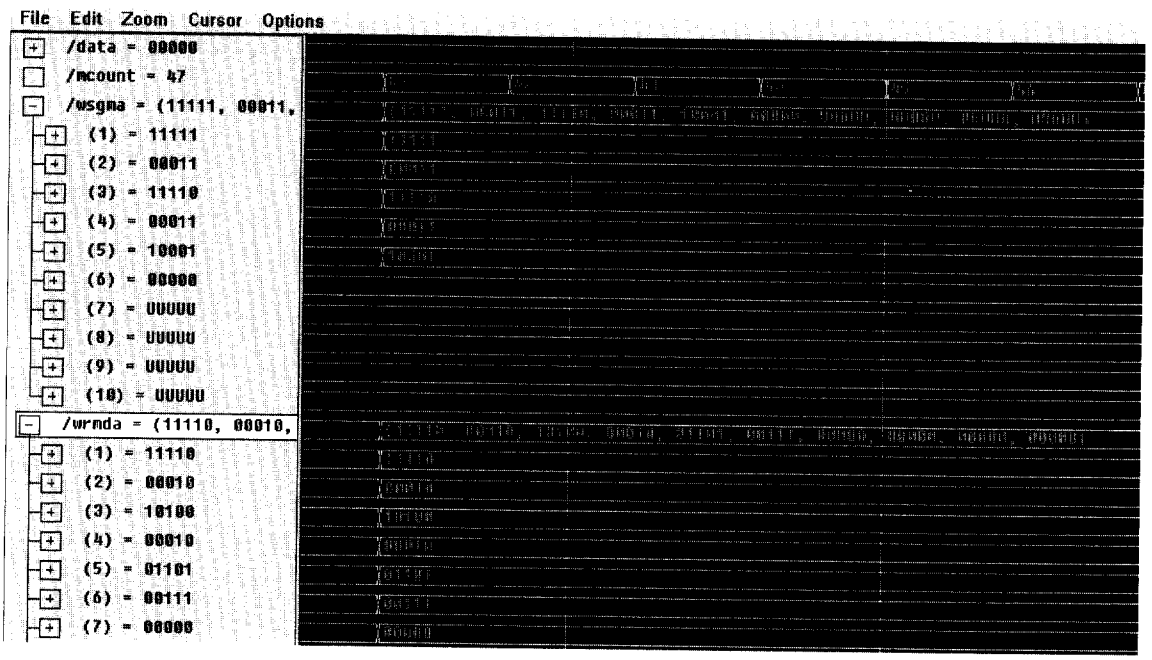


그림9 σ, 및 λ값

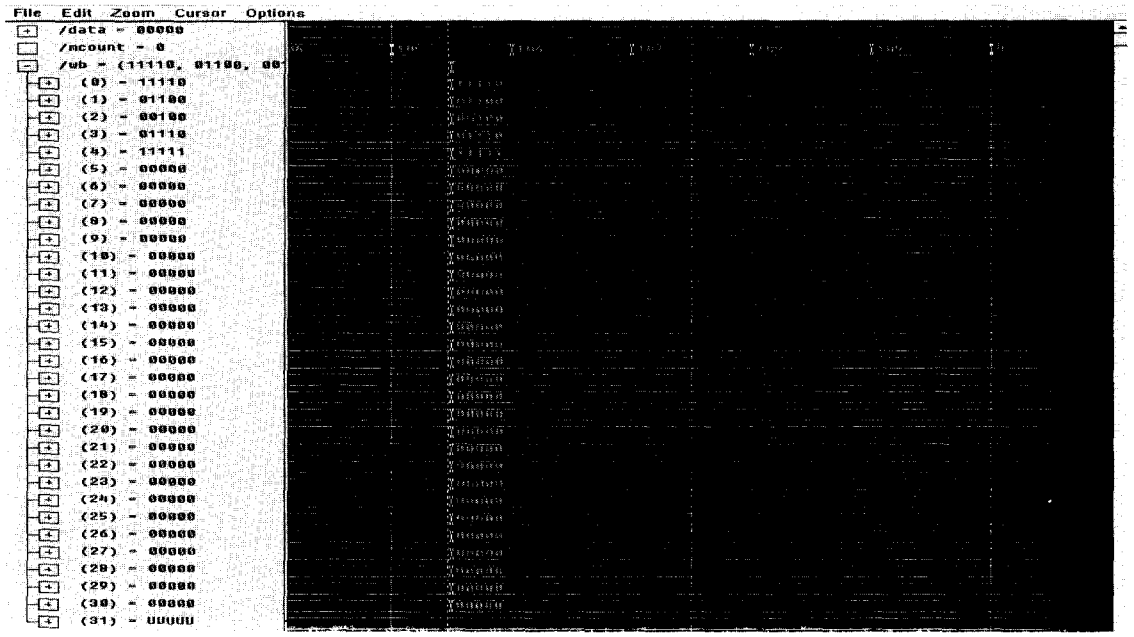


그림10 오류값

그림11 부호어

제 5 장 결 론

본 논문에서는 오류정정능력이 매우 큰 RS 부호의 부호화 알고리즘과 복호알고리즘을 제안하였으며, 이 알고리즘을 이용하여 오류정정능력이 5인 (31, 21) RS 부호의 부호기 및 복호기를 설계하였다. 그 결과 PGZ복호알고리즘을 이용한 복호기보다 곱셈기는 1248개, X-OR는 4702개, 역원계산기는 23개 줄어든 반면, 2×1 MUX 6개, 인버터 3개, 비교기 2개 필요함을 확인하였고 제안된 복호알고리즘을 이용한 복호기에는 역원기가 하나 필요하고 셀이 다 매우 규칙적이며 단순한 특성이 있음을 알 수 있었다. 제안된 복호기는 C 언어를 이용하여 시뮬레이션하였고, 그 타당성을 VHDL 언어의 시뮬레이션으로 검증하였다. 앞으로 이의 ASIC 또는 FPGA실현에 관한 연구를 실행할 예정이다.

참고문헌

[1] 이만영, "BCH 부호와 Reed-Solomon 부호," 민음사, 1990.

[2] Man Young Rhee, "Error Correcting Coding Theory", McGRAW-HILL, 1989

[3] E. R. Berlekamp, "Bit-serial Reed-Solomon encoders," IEEE Trans. Inform. Theory, vol. IT-28, no. 6, Nov. 1982.

[4] T. K. Truong, L. J. Deutsch, I. S. Reed, J. S. Hsu, K. Wang, and C. S. Yeh, "The VLSI design of a Reed-Solomon encoder using Trans. Comput., vol C-33, pp. 906-911, Oct. 1984.

[5] R. P. Brent and H. T. Kung, "Systolic VLSI arrays for polynomial GCD computations," Dep. Comput. Sci., Carnegie-Mellon Univ., Pittsburgh, PA, Rep., 1982.

[6] I. S. Reed, R. A. Scholtz, T. K. Truong, and L. R. Welch, "The fast decoding of Reed-Solomon codes using Fermat theoretic transforms and continued fractions," IEEE Trans. Inform. Theory, vol. IT-24, pp. 100-106, 1978.

[7] Howard M. Shao, T. K. Truong, Leslie J. Deutsch, Joseph H. Yuen, Irving S. Reed, "A VLSI Design of a Pipeline Reed-Solomon Decoder", IEEE Tran. Comput. vol. C-34, pp.393-402, May, 1965.

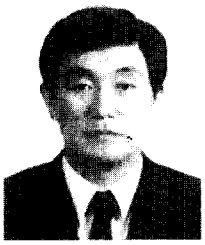
□ 著者紹介

강 경 식

1983년 2월 정주대학교 전자공학과(학사)
1989년 2월 한양대학교 대학원 전자통신공학과(석사)
1994년 3월 현재 정주대학교 대학원 전자공학과 박사과정
1983년 3월 - 현재 주성대학 공학1학부 조교수

※ 주관심분야 : 부호이론, 암호이론, 전자 상거래 등임

박 진 수



1975년 2월 한양대학교 전자공학과(학사)
1977년 2월 한양대학교 대학원 전자통신공학과(석사)
1985년 3월 한양대학교 대학원 전자통신공학과(박사)
1987년 2월 - 1988년 2월 Univ. Colorado at Colorado Spring (Post Doc.)
1978년 2월 - 현재 청주대학교 전자공학과 교수
1998년 1월 - 현재 한국통신학회 충북지부 지부장

※ 주관심분야 : 디지털 이동통신, 부호이론, Spread Spectrum 통신, 무선LAN 등임.