論文 86-23-5-18



# 단일오류 정정 및 Erasure 발생을 위한 R-S 복호기 설계

# (R-S Decoder Design for Single Error Correction and Erasure Generation)

金 容 哲*, 宋 東 一*, 金 英 雄*, 李 根 泳**

(Yong Serk Kim, Dong Il Song, Young Woong Kim and Kuen Young Lee)

**要 約**

R-S 부호는 전송상에서의 산발에러(random error) 및 연집에러(burst error)를 정정하는데 매우 효과적이나 B/M 알고리즘을 이용할 경우 하드웨어가 커진다. 또한 전체의 에러정정 시스템을 두개의 R-S 복호기와 De-interleave로 구성할 경우, Erasure 플래그 비트를 사용하므로 De-interleave 부분의 입출력라인이 9비트가 되어 이부분의 하드웨어가 늘어나서 시스템이 커지게 된다.

본 논문의 R-S 복호기는 에러정정부분과 Erasure 발생 부분으로 구성이 되는데 에러정정 부분은 직접 복호법, Erasure 발생부분은 Erasure 플래그 비트를 사용하지 않는 방법을 이용 하였다. 이 두가지 방법을 적용한 R-S 복호기를 전체의 에러정정 시스템에 사용할 경우 매우 하드웨어가 간단화 될 수 있음을 나타낸다.

**Abstract**

**Reed-Solomon (R-S) code is very effective to correct both random and burst errors over a noise communication channel. However, the required hardware is very complex if the B/M algorithm was employed. Moreover, when the error correction system consists of two R-S decoder and de-interleave, the I/O data bus lines becomes 9 bits because of an erasure flag bit. Thus, it increases the complexity of hardware.**

**This paper describes the R-S decoder which consists of a error correction section that uses a direct decoding algorithm and erasure generation section and a erasure generation section which does not use the erasure flag bit. It can be shown that the proposed R-S decoder is very effective in reducing the size of required hardware for error correction.**

## I. 序 論

R-S부호 (Reed-Solomon Code)는 BCH(Bose Chaudburi Hocquenghem)부호의 일종으로서 다중 오류를 정정하는 가장 강력한 부호로 알려져 있다.[1]

그러나 이 R-S부호의 복호 과정에는 복잡한 에러 위치 다항식을 구해야 하는 단점이 있다.[2] 본 논문에서는 이러한 과정을 생략한 직접 복호법을 적용하고 수식을 변경하므로써 구조가 간편해진 복호기를 설계하고, 또한 간단한 곱셈만을 이용하므로서 하드웨어 크기의 최소화를 시도하였다. 설계 실험은 단일 에러(single error) 정정과 복수에러(multiple error)에서의 Erasure 발생능력이 있는 단축형 R-S부호(Shortened reed-solomon code) (14, 12, 3)에서 수행되었다.

먼저 R-S부호의 기본 특성을 고찰하면 다음과 같다.

*正會員, 三星電子(株) 綜合研究所
(R & D Center, Sam Sung Electronics Co.)
**正會員, 成均館大學校 電子工學科
(Dept. of Elec. Eng., Sung Kyun Kwan Univ.)

接受日字 : 1985年 11月 2日

R-S부호는 비이원화 BCH부호(non binary BCH code)이므로 부호의 구성원소는 심벌(symbol)이 된다. 일반적으로 GF($2^m$) 상의 원소로 구성되는 t개 에러 정정 R-S 부호는 각원소를 심벌로 표시하면 각 심벌은 m비트로 이루어지므로 부호의 길이는 $n=m(2^m-1)$이고 데이터 길이는 $k=m(2^m-1-2t)$인 2원(n, k) 선형 부호가 성립될 수 있다. 이 부호는 t심벌이하의 에러 형태를 정정할 수 있는 능력을 갖게 된다.

또한 GF($2^m$)내에서 R-S 부호는 다음과 같은 매개변수가 성립한다.[3]

패 리 티 수 : $n-k=2t$

최 소 거 리 : $d(min)=2t+1$

에러정정수 : t

의 Erasure발생능력이 있는 단축형 R-S 부호sbortened reed-solomon code) (14, 12, 3)에서 수행되었다.

이원체 GF(2) 상의 m차원시 다항식(primitive polynomial) p(x)의 근을 $\alpha$라고 하면 그 확대체 GF($2^m$) 상의 원소로 표시되는 R-S 부호가 형성된다.[4] 이때 t개의 에러를 정정할 수 있는 R-S부호의 생성다항식 g(x)는

$$g(x)=\prod_{i=0}^{2t-1}(x+\alpha^i) \qquad (1)$$

이 되며 이 g(x)에 의해 생성된 부호는 GF($2^m$) 상의 $(m(2^m-1),\ m(2^m-1-2t))$의 블럭부호가 된다.

주어진k개 심벌의 데이터 워드를

$$d(x)=d_{k-1}x^{k-1}+d_{k-2}x^{k-2}+\cdots\cdots+d_1x+d_0 \qquad (2)$$

라하고, 코드워드를

$$\begin{aligned}c(x)&=c_{n-1}x^{n-1}+c_{n-2}x^{n-2}+\cdots\cdots+c_1x+c_0\\&=d_{k-1}x^{n-1}+d_{k-2}x^{n-2}+\cdots+d_0x^{n-k}+p_{r-1}x^{n-k-1}+\\&\qquad p_{r-2}x^{n-k-2}+\cdots p_0\end{aligned}$$

단, $r=n-k$ (3)

라 하면, 코드워드의 k개 심벌 $c_{n-1}, c_{n-2}, \cdots c_{n-k+1}, c_{n-k}$는 데이타 워드 $d_{k-1}, d_{k-2}, \cdots, d_1, d_0$와 같으며 코드워드의 나머지 $c_{n-k-1}, c_{n-k-2}, \cdots, c_1, c_0$는 패리티로써 n-k차의 생성다항식 g(x)에 의해서 구해지며, 그 값은 $x^{n-k}\,d(x)/g(x)$의 나머지와 같다.

따라서 $x^{n-k}d(x)=g(x)q(x)+p(x)$으로 나타낼 수 있다. (4)

결과적으로

$$\begin{aligned}c(x)&=c_{n-1}x^{n-1}+\cdots+c_1x+c_0=\sum_{i=n-k}^{n-1}c_ix^i+\sum_{i=0}^{n-k-1}c_ix^i\\&=x^{n-k}d(x)+p(x)\end{aligned}$$

에서 식(4)에 의하여

$$c(x)=x^{n-k}d(x)+p(x)=g(x)q(x)$$

따라서 생성다항식g(x)는 코드워드c(x)의인수가 됨을 알 수 있다. 회로구성은 일반적으로 RS 부호의 패리티 생성 회로는 $x^{n-k}$을 곱하는 회로와 g(x)로 나누어 나머지를 구하는 회로를 사용한다. 식(1)을 이용하면 t=1인 경우의 GF($2^8$)에서의 g(x)는 $g(x)=x^2+\alpha^{25}x+\alpha$이 되며, 간단하게 RS(14, 12) 부호기를 구성할 수 있다.

RS(14, 12) 부호는 최소거리를 3으로 갖는 단일에러 정정부호로 이에 대한 특징이 표 1에 나타낸다.

**표 1. RS(14, 12) 부호의 특징**
**Table 1. Characteristics of RS(14, 12) code.**

| Items | Contents |
|---|---|
| Error Correction Code | RS(14, 12, d=3) OVER GF($2^8$) |
| Primitive Polynomial | $p(x)=x^8+x^4+x^3+x^2+1$ |
| Generator Polynomial | $g(x)=x^2+\alpha^{25}x+\alpha$ |

본문에서는 직접복호법에 의해 에러값, 에러판별조건을 유도하여 이것을 에러정정부에 적용하여 어떻게 하드웨어가 간단화 될 수 있는가를 보이고, erasure 발생시 erasure플래그 비트를 사용하지 않고 16 진수 "FF"를 erasure로 하는 새로운 방법을 설명 하고자 한다.

## Ⅱ. R-S 부호의 복호

### 1. 일반적 복호 과정

R-S 복호기의 복호과정은 에러가 섞여있는 수신 데이타워드(received dataword)를 받아서 a. 신드롬(syndrome)의 계산 b. 에러위치 다항식의 계산 c. 에러 위치 다항식으로부터 에러위치계산 d. 에러값의 계산의 4단계를 거치게 된다. 여기서 2단계는 B/M (Berlekamp/Massey) 알고리즘을, 3단계는 CHIEN SEARCH 방법을 이용하게 된다. 에러 정정은 4단계에서 얻어진 에러값을 에러 위치에서 버퍼에 저장되어 있던 에러값에 더함으로써 (배타연산) 에러를 정정할 수 있도록 한다.

여러단계중 2단계는 매우 복잡하여 많은 하드웨어를 차지한다. CHIEN은 이 단계를 생략한 직접복호법을 이원화 부호에 대해서 제안했다.[5] 이방법은 신드롬 레지스터를 쉬프트 해서 그 상태를 직접 살펴봄으로써 에러의 위치를 구한다.

비이원화 부호에 대한 이방법의 일반화는 FORNEY[6]에 의해 시사되었지만 그 명확한 형태는 나와있지 않다. 본 논문에서는 에러 위치다항식 구성단계를 생략한 직접복호법을 R-S 부호에 적용하여 단일오류 정정복호기 설계를 간단화 시킨다.

2. 직접 복호법에 의한 복호 과정

R-S 복호에서의 직접 복호법(direct decoding method)은 신드롬(syndrome)으로부터 에러위치 다항식(error locator polynomial)을 구성하는 단계가 생략된 것을 말한다. 단일 에러정정 복호기의 경우 직접복호법이 통상 이용될 수 있으나, 문제는 유한필드 내에서 서로다른 두수의 곱셈 혹은 나눗셈이 사용되는 것이었다.

따라서 본 논문의 핵심은 복잡한 연산을 하지않고 복호를 할 수 있는 간단한 복호기를 구성하는데 있다.

여기서는 GF($2^4$) 내에서의 RS(14, 12)부호를 이용하여 에러위치, 에러값, 에러판별조건을 수식을 변경시켜 구해보기로 한다.

그러면 수식을 변경하므로써 최소화된 복호기를 구성하자.

$\alpha$를 GF($2^4$)상의 원시근이라 하면, RS(14, 12) 부호의 패리티 검사행렬은 식(5)으로 주어질 수 있다.[7] 부호벡터 $\bar{c}=(c_0, c_1, c_2, \cdots, c_{13})$, $c_i \in GF(2^4)$은 $H^T \cdot C = 0$을 만족한다.

$$H=\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ \alpha^{13} & \alpha^{12} & \alpha^{11} & \alpha^{10} & \alpha^{9} & \alpha^{8} & \alpha^{7} & \alpha^{6} & \alpha^{5} & \alpha^{4} & \alpha^{3} & \alpha^{2} & \alpha & 1 \end{bmatrix} \quad (5)$$

RS(14, 12) 부호의 에러위치 판별식을 유도하자.

신드롬을 $s_i$라 하면

$$s_i = r(\alpha^i) \quad (6)$$
$$= e(\alpha^i) \quad (0 \le i \le 2t-1)$$

에러값을 $\bar{e}=(e_0, e_1, e_2, e_3 \cdots\cdots e_{13})$, $e_i \in GF(2^4)$로 하면

$e(x) = r(x) - c(x)$가 된다.

에러 위치를 $\alpha^j$, 에러 값을 $e_j$라 하면

$$s_i = \sum_{j=0}^{13} e_j (\alpha j)^i, \quad 0 \le i \le 2t-1 \quad (7)$$

$u(\le t)$개의 에러가 k번째 코드워드에서 발생했다면, 이들 에러의 위치를 $x_j = \alpha^{k_j}$, 에러값 $y_j = e_{k_j}$ $(j=1, 2, \cdots, u)$로 하면 $s_i$는 식(8)로 표현된다.

$$s_i = \sum_{j=0}^{u-1} y_j x_j^i \quad (0 \le i \le 2t-1) \quad (8)$$

복호는 신드롬으로부터 미지수 쌍 $(x_j, y_i)$를 구하는 것이다.

식(7)으로부터

$$s_0 = \sum_{j=0}^{13} e_j \quad (9)$$

$$s_1 = \sum_{j=0}^{13} e_j \cdot \alpha j \text{ 을 얻을 수 있다.} \quad (10)$$

단일 에러 위치 다항식 $\sigma(x)$를 $\sigma(x) = 1 + \alpha^i x = 1 + \sigma_1 x$라 놓자.

$\sigma(x)$의 일차항 계수 $\sigma_1$를 j만큼 변형시킨 값을 $\sigma_1^{(j)}$라 하면 $\sigma_1^{(j)} = \alpha^{-(13-j)} \sigma_1$로 할 수 있으며, CHIEN SEARCH 방법에 의해 에러위치는 $\sum_{i=0}^{1} \sigma_i^{(j)} = 0$ 일때이다. (11)

(j= 0 ~j= 13까지의 레지스터의 쉬프트수)

한편, (6), (9)로 부터

$s_0 = e_0 + e_1 + e_2 + \cdots\cdots + e_{13}$

$= r_0 + r_1 + r_2 + \cdots\cdots + r_{13}$이므로 (12)

수신부호(received codeword) $\bar{r} = (r_0, r_1, \cdots, r_{13})$의 14개 심벌중 1개가 에러 이라면 나머지 13 심벌은 ZERO가 되므로 에러 값 $y_j = e_j = s_0$가 된다. (13)

그러면 에러위치 $x_j$를 정해주는 에러 위치 판별식을 유도하자.

식(9), (10)으로부터

$\alpha^j = s_1 / s_0 = \sigma_1$ (14)

본래 B/M 알고리즘을 사용한다면, 신드롬 $s_0, s_1$를 구한후 몇개의 단계를 거친후 결과 값으로부터 $\sigma(x)$의 계수를 구하지만[8] 직접복호법에서는 식(14)에서 처럼 직접 $\sigma_1$이 구해진다.

그러면 식(11)을 이용하여,

$1 + \sigma_1^{(j)} = 0$ (j= 0 ~j= 13)

$1 + \alpha^{-(13-j)} s_1 / s_0 = 0$

$1 + \alpha^{-13} s_1 / s_0 \alpha^j = 0$

$s_0 \alpha^{13} + s_1 \alpha^j = 0$ (단j는 신드롬 레지스터의 쉬프트수) (15)

따라서 신드롬 레지스터를 j= 0 ~j= 13까지 쉬프트 시키면서 식(15)가 성립하면 에러가 발생한 것으로 판별하여 이 순간에 에러 심벌에 에러 값을 더함으로 에러를 정정하게 된다.

또 하나의 방법은, 식(10)을 변형하므로써 구할 수 있다.

RS(14, 12) 부호의 경우 $x_j = \alpha^{13-j}$라 놓을 수 있으므로

$s_1 = \sum_{j=0}^{13} e_j \alpha^{13-j}$이 된다. (10)'

식(9), (10)'에서

$s_1 / s_0 = \alpha^{13-j}$를 얻는다. (16)

따라서 식(15)와 식(16)은 동일하다.

지금까지의 수식을 정리하면,

| |
|---|
| 에러 위치 : $x_j = \alpha^{13-j}$ |
| 에러 값 : $y_j = s_0$ |
| 에러 판별조건 : $s_0 \alpha^{13} + s_1 \alpha^j = 0$ |

의 결과를 얻는다. (단, 본논문에서 사용되는 덧셈은 배타 연산이고, 곱셈은 GF($2^4$) 내에서의 연산임) 정리된 수식을 이용하여 j= 0, ~j= 13까지 값을 변화시키면서 다음의 도표 2를 얻는다. 지금까지의 복호기 설계이론을 이용하여 에러 정정 section의 복호기를 구성하면 그림 1과 같다.

표 2. RS(14, 12) 부호의 에러 정정section의 복호 과정

**Table 2.** Decoding procedure of RS(14, 12) code in error correction section.

| 쉬프트 수 | 에러 위치 | 검출 심벌 | 에러값 | 에러 판별 조건 |
|---|---|---|---|---|
| j=0 | $x_0=\alpha^{13}$ | $r_{13}$ | $s_0$ | $s_0\alpha^{13}+s_1\alpha^0=0$ |
| j=1 | $x_1=\alpha^{12}$ | $r_{12}$ | $s_0$ | $s_0\alpha^{13}+s_1\alpha^1=0$ |
| j=2 | $x_2=\alpha^{11}$ | $r_{11}$ | $s_0$ | $s_0\alpha^{13}+s_1\alpha^2=0$ |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |
| j=12 | $x_{12}=\alpha^1$ | $r_1$ | $s_0$ | $s_0\alpha^{13}+s_1\alpha^{12}=0$ |
| j=13 | $x_{13}=\alpha^0$ | $r_0$ | $s_0$ | $s_0\alpha^{13}+s_1\alpha^{13}=0$ |

## Ⅲ. R-S 복호기(decoder)의 시스템 구성

R-S복호기는 크게 에러 정정section과 ERASURE 발생 section으로 나눌 수 있다.

1. 에러 정정 section

이는 데이터를 받는 첫관문으로 수신부호를 $\bar{r}=(r_{13}, r_{12}, \cdots, r_1, r_0)$라 하면 첫 8 비트 레지스터를 거치면서 $s_0=r_{13}+r_{12}+\cdots+r_1+r_0$, $s_1=\alpha^{13}r_{13}+\alpha^{12}r_{12}+\cdots+\alpha r_1+r_0$의 수식에 의해 $s_0$, $s_1$를 구하고 $s_0\alpha^{13}+s_1\alpha^j$의 결과치가 ZERO가 되는가를 살펴서 에러 발생을 감시한다.

에러 정정 section에서는 한코드워드 $\bar{r}$중 한개의 에러 만이 발생했을 때 정정이 가능하다. 두개 이상의 에러가 발생했을 경우는 정정을 못하고 ERASURE 발생 section으로 보내게 된다.

그림 1에 에러 정정 section의 복호기를 보여주고 있는데 매우 간단한 $\alpha$, $\alpha^{13}$의 곱셈만으로 정정이 이루어지므로 하드웨어 크기를 상당히 줄일 수 있다. 본래 복호기는 임의의 두수에 의한 곱셈회로, 나눗셈회로가 반드시 쓰여서 많은 하드웨어를 차지했었다. 그림 4(a), 4(b)에 $\alpha$, $\alpha^{13}$ 연산회로를, 그림 2에 R-S 복호기 전체 시스템을 나타낸다.

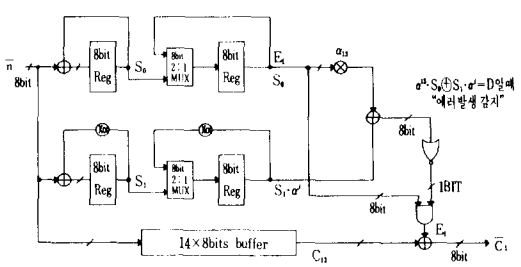


**그림 1.** RS(14, 12)부호의 에러 정정section 복호기
**Fig. 1.** Error correction section decoder of RS (14, 12) code.

2. ERASURE 발생 section

에러가 2개 이상 발생하여 정정을 못하고 ERASURE 발생 section으로 데이터가 넘어 왔을때 한코드워드를 ERASURE로 만든다.

이 section에서는 일반적으로 9비트로 한 데이터 심벌을 나타내고 EARSURE는 MSB한 비트를 ERASURE 플래그 비트로 하여 ERASURE는 그 비트를 세트시키는 방법을 사용하여 왔으나, 본 논문 에서는 16진수값 "FF"로 만들었다. 이 방법을 음성전달 시스템에 적용한다면 A/D변환시"FF"값은 거의 없으므로 "FF"를 ERASURE로 사용할 수 있어 ERASE 및 에러를 정정할 수 있는 R-S 복호기와 함께 이용한다면 전체 에러정정 시스템의 하드웨어 크기를 줄일 수 있다. 그림 3에 회로를 나타낸다. 식(6)에서 $s_0=r(\alpha^0)$, $s_1=r(\alpha)$가 되는데 에러 정정 section을 거친 코드워드를 받아서 $s_0$, $s_1$를 연산하므로써

(i) $s_0=s_1=0$

에러가 없거나 에러 정정section에서 정정을 한 경우

(ii) $s_0=s_1\neq 0$

에러가 2개이상 발생한 경우

임을 판단하여 (ii)의 경우 그 한 코드워드를 ERASURE로 만든다.

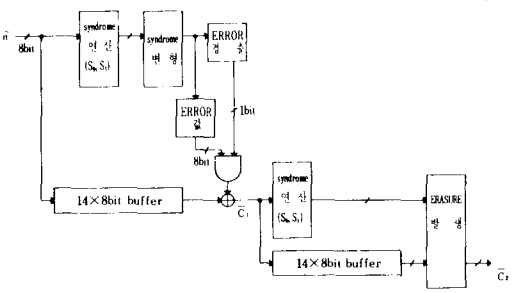


**그림 2.** RS(14, 12)의 복호기 시스템
**Fig. 2.** Decoder system of RS(14, 12) code.

## Ⅳ. 복호기 시스템에 대한 실험 및 결과

R-S 복호기의 하드웨어는 크게 에러 정정section과 ERASURE발생 section의 두부분으로 구성되므로 이에 대한 테스트도 별도로 진행한 후 두 section를 결합해서 하였다. 테스트 결과 데이터는 전체 복호기 시스템만을 첨부하였다. 실험을 위해서 RS(14, 12) 부호는 두개의 패리티를 가지게 되는데, 데이터워드($c_{13}, c_{12}, \cdots c_3, c_2$)의 12개 데이터 심벌을 SPC-3000(삼성 퍼스널 컴퓨터)에 입력시켜 이에대한 패리티($c_1, c_0$)를 구하므로써 코드워드 $\bar{c}=(c_{13}, c_{12}, \cdots, c_1, c_0)$를 얻었다. 코드워드는 모두 5가지를 만들었는데, 60개의 데이터 심벌

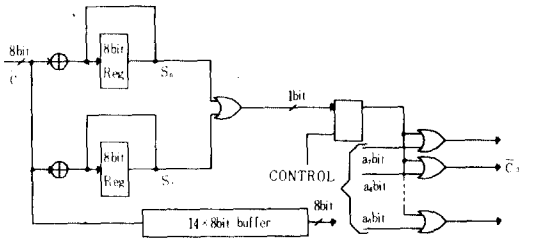


**그림 3**. RS(14, 12) 부호의 ERASURE발생 section복호기

**Fig. 3**. Erasure generation section decoder of RS (14, 12) code.

을 각각 12개씩 이용하여 각각 2개씩의 패리티를 발생시켰는데 표3에 나타낸다. 이 표를 사용하여 테스트 코드워드 $\bar{r}$을 만들 수 있다.

테스트 코드워드 $\bar{r}=(r_{13}, r_{12}, \cdots, r_1, r_0)$라 하면 이것들은 표3에서의 코드워드를 에러 형태의 여러가지로 만들어 표4를 완성하였다. 이 표는 에러가 없는 경우에서 3개에러까지 여러 에러가능성을 고려하였는데, 이 10개 테스트 코드워드를 순차적으로 SPC-3000에 입력시킨 후 이것들을 다시 복호기 시스템에 입력시켜 이 시스템이 제대로 동작이 되는가를 살펴보았다. 그림2에 전체적인 복호기 시스템이 나타나 있는데 시스템 구성상 데이타의 이동은 2개의 버퍼 단을 거치게 되므로 복호기 시스템에 입력된 데이타 출력은 입력후 2개 버퍼만큼의 지연이 생기게 되었다. 예를 들어 첫번째 블럭에 입력되는 테스트 코드워드는 $\bar{r}=$ (06, EA, 14, ···, 14, ···, D8, 9E)가 2개 블럭을 지난 세번째 블럭에서 출력이 되어 DB라는 에러 심벌이 CA라는 올바른 데이터 심벌로 정정이 됨을 알 수 있다. 표5에 복호기 시스템의 출력변화를 나타내고 있는데 에러가 2개 이상이면 ERASURE를 만들어냄을 알 수 있다. 따라서 복호기 시스템의 테스트 결과는 매우 성공적이었다.

**표 3**. RS(14, 12) 부호의 코드워드

**Table 3**. Codeword of RS(14, 12) code.

| | dataword $\bar{d}$ | | | | | | | | | | | | 패리티 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $c_{13}$ | $c_{12}$ | $c_{11}$ | $c_{10}$ | $c_9$ | $c_8$ | $c_7$ | $c_6$ | $c_5$ | $c_4$ | $c_3$ | $c_2$ | $c_1$ | $c_0$ |
| 1 | 06 | EA | 14 | 5F | BE | CA | 01 | F0 | 1A | 26 | 02 | 5A | D8 | 9E |
| 2 | D3 | 25 | 1C | 12 | FB | 2C | 1B | AD | 19 | D1 | 31 | FC | 46 | DA |
| 3 | 45 | F2 | B2 | 90 | B3 | D1 | A4 | AA | 15 | 6D | 1C | 70 | 38 | D5 |
| 4 | 1C | 3D | F4 | 2C | FA | 6C | 16 | 01 | D1 | E4 | 55 | F1 | D2 | 3B |
| 5 | D2 | 3C | 1E | EA | 94 | 2C | 02 | A0 | BA | 1A | ED | 1A | 2E | 75 |

• $c_0$—$c_{13}$ : 16진수 값

**표 5**. SPC-3000에 의한 복호기 시스템의 출력변화

**Table 5**. Output variation of of decoder system by SPC-3000.

| 입력블럭 | 출력블럭 | 입력형태 | 출력형태 | 비 고 |
|---|---|---|---|---|
| 1 | 3 | 단일에러 | 에러 정정 | DB→CA 정정 |
| 2 | 4 | 에 러 무 | 그대로 출력 | |
| 3 | 5 | 2개에러 | ERASURE | 한코드워드가 FF |
| 4 | 6 | 단일에러 | 에러 정정 | 13 →E4 정정 |
| 5 | 7 | 3개에러 | ERASURE | 한코드워드가 FF |

## V. 結 論

부호 이론을 이용하여 하드웨어의 크기를 최소화 시킨 R-S복호기를 구성하였다. GALOIS FIELD내에서 나눗셈은 어려워서 일반적으로 ROM을 이용하는 것이 보통인데, 구성된 시스템은 곱셈만을 이용하였으며 B/M 알고리즘을 사용하지 않는 직접복호법을 사용하였다. 그런데 본 논문에서 적용된 직접 복호법은 단일에러 정정에는 매우 효과적이나 그 이상의 에러정정에서는 새로운 직접 복호기법이 요구되며 이에 대한 연

**표 4**. RS(14, 12) 부호의 테스트 코드워드 $\bar{r}$

**Table 4**. Test codeword of RS(14, 12) code.

| 에러 형태 | | 수신코드워드 $\bar{r}$ | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | $r_{13}$ | $r_{12}$ | $r_{11}$ | $r_{10}$ | $r_9$ | $r_8$ | $r_7$ | $r_6$ | $r_5$ | $r_4$ | $r_3$ | $r_2$ | $r_1$ | $r_0$ |
| 단일 에러 | 1 | 06 | EA | 14 | 5F | BE | *DB | 01 | F0 | 1A | 26 | 02 | 5A | D8 | 9E |
| 에 러 무 | 2 | D3 | 25 | 1C | 12 | FB | 2C | 1B | AD | 19 | D1 | 31 | FC | 46 | DA |
| 2개 에러 | 3 | 45 | *1C | B2 | 90 | B3 | D1 | A4 | AA | 15 | 6D | 1C | 70 | *BE | D5 |
| 단일 에러 | 4 | 1C | 3D | F4 | 2C | FA | 6C | 16 | 01 | D1 | *13 | 55 | F1 | D2 | 3B |
| 3개 에러 | 5 | *A0 | 30 | 1E | EA | 94 | 2C | 02 | A0 | BA | 1A | ED | *AC | 2E | *0C |

• $c_0$—$c_{13}$ : 16진수 값 *표시 심벌은 에러임.

구가 필요하다. 또한, 에러정정section에서 정정에 실패하였을 때는 ERASURE를 만드는데 ERASURE 플래그비트를 사용하지 않았다.

본 시스템에서는 16진수FF를 ERASURE로 간주하였으나 16진수 00도 가능하리라 본다.

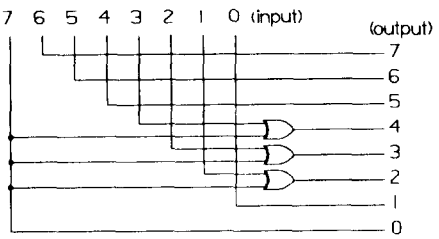


**그림 4 (a)**. $x\alpha$ 연산회로
**Fig. 4 (a)**. Circuit with multiplies by $\alpha$.

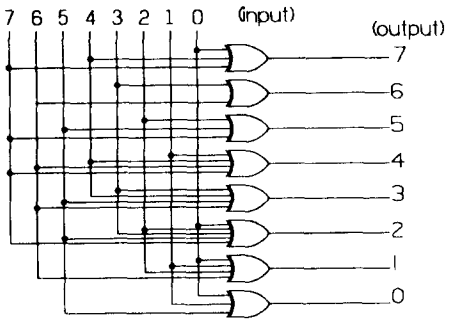


**그림4 (b)**. $x\alpha^{13}$연산 회로(0-7 : LSB-MSB를 나타냄)
**Fig. 4 (b)**. Circuit which multiplies by $\alpha^{13}$.

본 논문에서 구성된 복호기 시스템중 ERASURE 발생 section은 음성전달(예를 들면 디지탈 오디오) 시 발생하는 에러를 정정하는데 함께 이용하면 효과적이다. 특히 ERASURE를 16진수 FF로 간주하는 방법은 매우 특이한 방법이라 생각된다.

### 별첨. TEST DATA SHEET 1.

Mode Select Completed····· A port-Data Out B p-Data In
C port:C0-CLK C1-OR Out C2-DATA E C3-PARITY C7-Reset

| BLOC NO. | CLK NO. | DAT-IN | DAT-OUT |
|---|---|---|---|
| 1 | 1 | 06 | 30 |
| 1 | 2 | EA | FF |
| 1 | 3 | 14 | FF |
| 1 | 4 | 5F | FF |
| 1 | 5 | BE | FF |
| 1 | 6 | DB | FF |
| 1 | 7 | 01 | FF |
| 1 | 8 | F0 | FF |
| 1 | 9 | 1A | FF |
| 1 | 10 | 26 | FF |
| 1 | 11 | 02 | FF |
| 1 | 12 | 5A | FF |
| 1 | 13 | D8 | FF |
| 1 | 14 | 9E | FF |
| 1 | 15 | 9E | FF |
| 1 | 16 | 9E | FF |
| 2 | 1 | D3 | FF |
| 2 | 2 | 25 | FF |
| 2 | 3 | 1C | FF |
| 2 | 4 | 12 | FF |
| 2 | 5 | FB | FF |
| 2 | 6 | 2C | FF |
| 2 | 7 | 1B | FF |
| 2 | 8 | AD | FF |
| 2 | 9 | 19 | FF |
| 2 | 10 | D1 | FF |
| 2 | 11 | 31 | FF |
| 2 | 12 | FC | FF |
| 2 | 13 | 46 | FF |
| 2 | 14 | DA | FF |
| 2 | 15 | DA | 06 |
| 2 | 16 | DA | 06 |
| 3 | 1 | 45 | 06 |
| 3 | 2 | 1 | EA |
| 3 | 3 | B2 | 14 |
| 3 | 4 | 90 | 5F |
| 3 | 5 | B3 | BE |
| 3 | 6 | D1 | CA |
| 3 | 7 | A4 | 01 |
| 3 | 8 | AA | F0 |
| 3 | 9 | 15 | 1A |
| 3 | 10 | 6D | 26 |
| 3 | 11 | 1C | 02 |
| 3 | 12 | 70 | 5A |
| 3 | 13 | BE | 5A |
| 3 | 14 | D5 | 5A |
| 3 | 15 | D5 | D3 |
| 3 | 16 | D5 | D3 |
| 4 | 1 | 1C | D3 |
| 4 | 2 | 3D | 25 |
| 4 | 3 | F4 | 1C |
| 4 | 4 | 2C | 12 |
| 4 | 5 | FA | FB |
| 4 | 6 | 6C | 2C |
| 4 | 7 | 16 | 1B |
| 4 | 8 | 01 | AD |
| 4 | 9 | D1 | 19 |
| 4 | 10 | 13 | D1 |
| 4 | 11 | 55 | 31 |
| 4 | 12 | F1 | FC |
| 4 | 13 | D2 | FC |
| 4 | 14 | 3B | FC |
| 4 | 15 | 3B | FF |
| 4 | 16 | 3B | FF |

### 별첨. TEST DATA SHEET 2

본 RS(14, 12) 복호기는 DE-INTERLEAVE와 ERASURE 및 에러를 정정할 수 있는 R-S 복호기와 함께 이용할 때 의미를 가진다.[9]

에러정정 능력을 높이는 방법으로 2개의 RS부호를 사용하는 것 외에 이원화 부호와 비이원화 부호를 결합하여 사용하는 CONCATENATED-CODING 방법이[10] 있는데 본 논문의 에러정정 section을 외부호(OUTER CODE)로써 이용한다면 매우 효과적인 에러정정 시스템이 될 수 있으리라 생각되므로 이에 대한 많은 연구가 요망된다.

## 參 考 文 獻


[1] Kuang Yung Liu, "Architecture for VLSI Design of Reed-Solomon Decoders". *IEEE Trans. on computer,* vol. C-33, no. 2, pp. 178, Feb. 1984.

[2] W. Wesley Peterson, *E.J. Weldon, Jr.,* Error-Correcting Codes, pp. 288-298, 1972.

[3] 이만영, 안교한 "REED-SOLOMON 부호의 부호기 및 부호기 구성," 대한전자공학회 84년도 하계학술 논문집, vol. 7, no. 1, pp. 176-179, 1984.

[4] 이만영, 부호이론, 희중당, pp. 191-192.

[5] R.T. Chien, "Cyclic Decoding Procedures for BCH Codes', *IEEE Trans. Information* Theory, vol. IT-10, pp. 357-363, Oct. 1964.

[6] Forney, G.D., *On Decoding BCH Codes.* IEEE Trans. Information Theory, IT-11, pp. 549-557, Oct. 1965.

[7] W. Wesley Peterson, *E. J. Weldon,* Jr., Error-Correcting Codes, pp. 269-271, 1972.

[8] G.A. Kumar, R.O. Claus, N. Asimopoulos, F.G. Gray, *Hardware/Software Tradeoffs for the Decoding of Reed-Solomon Codes.* IEEE pp. 236-240, 1983.

[9] Philips, Electronic Components and Applications, vol. 4, no. 3, pp. 131-141, May, 1982.

[10] W. Wesley Peterson, *E.J. Weldon,* Jr. Error-Coorrecting Codes, pp. 139-141, 1972.