한국통신학회 1985년도 춘계학술발표회 논문집 (1985. 4. 27)

새로운 패러티 검사합 구성법에 의한 콘볼루션날 디코더 설계

85301

*박 만 효 강 창 언
연 세 대 학 교 공 과 대 학 전 자 공 학 과

Convolutional Decoders Using A New Parity Check Sums Method

Man Hyo PARK Chang Eon KANG
Dept. of Electronics Engineering, Yonsei Univ.

ABSTRACT

Using the (12,3) uniform convolutional codes, an effective codec system has been designed and constructed. Since uniform codes have the equidistance between code words, an encoding system can be constructed easily. A decoder has been designed by majority logic method.

A new method is proposed-giving information about parity check sums. Using ROM instead of majority logic gate, the decoding propagation delay time is decreased from 640nsec to 450nsec. Experimental results show that three or less random errors can be corrected.

1. 서 론

위성 통신이나 우주 통신과 같은 장거리 통신이나, 통신로상에 에러가 많이 발생하는 경우에는 에러 정정코드를 이용하여 통신로상의 에러를 처리하게 되면, 보다 효율적인 통신 시스템을 구성할 수 있다.

1948년 SHANNON에 의해 정보이론이 처음 정립된 후, 1955년 ELIAS가 콘볼루션날 코드를 이용한 에러 정정 방법을 소개했으며, 1963년 MASSEY가 다수결 논리 방법을 콘볼루션날 코드에 적용시킬 수 있음을 보였고, 1966년 MASSEY에 의해 유니폼 코드가 제안되었다.

유니폼 코드는 콘볼루션날 코드의 한 부류로서 랜덤에러 정정 능력이 우수하지만, 디코딩시 다수결 논리 GATE의 입력으로 들어가는 패러티 검사합을 시행착오적으로 찾아야 된다는 번거로움 때문에 잘 연구되어지지 않았다.

이에 본 논문에서는 패러티 검사합을 쉽게 찾을 수 있는 방법을 제시했으며, 이 방법에 의해서 구한것을 실제 실험에 적용하여 틀림없음을 확인할 수 있었다. 실제 실험에서는 다수결 논리 GATE 대신 ROM을 사용하여 디코딩 전파지연 시간을 줄였다.

2. 이론적 해석

(1) 콘볼루션날 코드의 인코딩

일반적인(n,k) 콘볼루션날 코드의 인코더에서 사용하는 부호가 2진수라면, k개의 정보비트의 한 블럭이 인코더의 입력으로 인가되면, 구속장이 N이고, (N-1)단의 기억소자로 구성된 인코더는, 그 때의 k개 정보 비트 블럭 뿐만아니라 그 이전의 (N-1) 정보 블럭들에 걸쳐서 선정된 각 기억 소자들의 출력을 MODULO-2 가산기에 입력시켜 선형 결합시킴으로써 출력이 k비트보다 많은 n비트인 코드 SEQUENCE를 만든다.

코드 SEQUENCE가 만들어지는 과정을 상세히 서술하면 다음과 같다.

(n,k) 콘볼루션날 코드에 대한 N디지트의 k(n-K) 벡터의 집합인 SUB-GENERATOR는

$$g(i,j) = (g_0(i,j)\ g_1(i,j)\ \cdots\ g_{N-1}(i,j)) \quad (1)$$

로 주어지며, 여기서 i=1,2,···,K 이고 j=1,2,···,n-k이다.

식(1)로부터 반무한대의 행렬인 콘볼루션날 코드의 GENERATOR MATRIX를 구할수 있는데 그 구조는 다음과 같다.

$$G_\infty = \begin{bmatrix} G_0 G_1 G_2 \cdots G_{N-1} \\ G_0 G_1 \cdots G_{N-2} G_{N-1} \\ G_0 \cdots G_{N-3} G_{N-2} G_{N-1} \\ \ddots \\ G_0 \; G_1 \; G_2 \cdots \\ \downarrow \\ \infty \end{bmatrix} \quad (2)$$

위의 식에서 빈칸은 모두 0을 나타낸다.

또, $G_0 = [I_K P_0]$ $G_x = [0 P_x]$ (3)

이며 I_K 는 k x k IDENTITY MATRIX

0 는 k x k ZERO MATRIX

P_ℓ 은 k x (n-k) MATRIX 인데,

$0 \le \ell \le N-1$ 이며, 그 형태는 다음과 같다.

$$P_\ell = \begin{bmatrix} g_\ell(1,1) g_\ell(1,2) g_\ell(1,3) \cdots g_\ell(1,n-K) \\ g_\ell(2,1) g_\ell(2,2) g_\ell(2,3) \cdots g_\ell(2,n-K) \\ \vdots \\ g_\ell(K,1) g_\ell(K,2) g_\ell(K,3) \cdots g_\ell(K,n-K) \end{bmatrix} \quad (4)$$

여기서 m을 인코딩되는 반무한대의 정보 SEQUENCE라 하면, 이 정보 SEQUENCE는 다음과 같이 순서로된 정보 블럭으로 나타낼 수 있다.

$$m = \{ m_0(1) m_0(2) \cdots m_0(K) m_1(1) m_1(2) \cdots m_1(K) m_2(1) m_2(2) \cdots m_2(K) \cdots \} \quad (5)$$

식 (5)에서 어떤 x번째 블럭 m_x'이 k개의 정보 디지트들로 구성되어 있다면

$$m_x' = m_x(1) m_x(2) \cdots m_x(K) \quad (6)$$

임을 나타낸다.

또, 어떤 양의 정수 ℓ 에 대해 지연인자 D^ℓ 을 정의하면, 코드 SEQUENCE c는 반무한대의 정보 SEQUENCE와 GENERATOR MATRIX의 곱으로 얻어진다.

즉, $c = m G_\infty$

$$= \sum_{x=0}^{\infty} m_x(1) D^x g_\infty(1) + \sum_{x=0}^{\infty} m_x(2) D^x g_\infty(2) + \cdots + \sum_{x=0}^{\infty} m_x(k) D^x g_\infty(K) \quad (7)$$

식(7)은 k개의 정보 디지트가 n개의 코드 SEQUENCE로 바뀌었음을 나타낸다. 즉,

$$c_x' = c_x(1) c_x(2) \cdots c_x(n) \quad (8)$$

이다.

식 (2)(3)(4)에서, x번째의 코드 블럭에 있는 n개 디지트들을 다음식으로 쓸수 있다.

$$c_x(i) = m_x(i) \quad (i = 1, 2, \cdots, K) \quad (9)$$

$$c_x(K+j) = \sum_{y=0}^{N-1} m_{x-y}(1) g_y(1,j) + \sum_{y=0}^{N-1} m_{x-y}(2) g_y(2,j) + \cdots + \sum_{y=0}^{N-1} m_{x-y}(K) g_y(K,j)$$

$(j=1,2,\cdots,n-K)$ (10)

식 (9) 에서 알 수 있듯이, x 번째 순간에 인코더 안으로 들어오는 k개의 정보 디지트들은 x번째 코드 블럭의 처음 k 개의 디지트와 동일하다. 그리고 식 (10)은 x번째 코드 블럭의 나머지 (n-k) 개 디지트를 나타내는데 이것은 페러티 검사 디지트들로써, 현재의 정보 블럭뿐만 아니라 이전 $(N-1)$개 정보 블럭에 의해서도 표현되어 짐을 알 수 있다. 식 (9)와 (10)에서, 각 입력 정보 디지트단에 자리이동단이 (N-1)개 있으므로 결국 k(N-1)단의 자리이동 레지스터로 인코딩 회로를 구성할 수 있다.

한편, (n,k) 콘볼루션날 코드는 선형 코드이므로 페러티 검사 MATRIX로도 나타낼 수 있다.

$$H_\infty = \begin{bmatrix} P_0^T I_K \\ P_1^T 0 \; P_0^T I_K \\ \vdots \\ P_{N-1}^T 0 \; P_{N-2}^T 0 \; P_{N-3}^T 0 \cdots P_0^T I_K \\ P_{N-1}^T 0 \; P_{N-2}^T 0 \cdots P_1^T 0 \; P_0^T I_K \\ \downarrow \\ \infty \end{bmatrix} \quad (11)$$

여기서 P_x^T 는 P_x 의 전치 MATRIX를, 빈칸은 모두 0을 나타낸다. 식 (2)와 (11)로 부터

$$G_\infty H_\infty^T = 0 \quad (12)$$

됨을 쉽게 알 수 있으므로

$$c H_\infty^T = 0 \quad (13)$$

을 만족하는 c만이 코드 SEQUENCE가 된다.

그런데, 실제의 인코더는 유한개의 디지트만을 저장할 수 있으므로, 반 무한대의 SEQUENCE는 적당히 잘라 줄여서 사용한다. 즉, 식 (2)는

$$G = \begin{bmatrix} G_0 G_1 G_2 \cdots G_{N-1} \\ G_0 G_1 \cdots G_{N-2} \\ G_0 \cdots G_{N-3} \\ \ddots \\ G_0 \end{bmatrix} \quad (14)$$

로, 또 식 (11)은 다음과 같이 된다.

$$H = \begin{bmatrix} P_0^T I_K \\ P_1^T 0 \; P_0^T I_{n-K} \\ \vdots \\ P_{N-1}^T 0 \; P_{N-2}^T 0 \; P_{N-3}^T 0 \cdots P_0^T I_{n-K} \end{bmatrix} \quad (15)$$

식 (14)와 (15)에서

$$G H^T = 0 \quad (16)$$

됨을 알 수 있다. 그러므로 코드 SEQUENCE인 nN 디지트의 임의의 벡터 v는

$$v \cdot H^T = 0 \quad (17)$$

을 만족해야 한다. 여기서 $d(u,v)$를 0번째 블럭

안에 있는 서로 다른 두 벡터 $\mathbf{U}$와 $\mathbf{V}$의 Hamming 거리라 하면, 최소거리 d_{min} 은 가장 작은 $d(\mathbf{U},\mathbf{V})$와 같다. 또 $\mathbf{Z}=\mathbf{U}+\mathbf{V}$라 놓으면

$$d(\mathbf{U},\mathbf{V})=W(\mathbf{Z}) \qquad (18)$$

가 되는데, $W(\mathbf{Z})$는 $\mathbf{Z}$의 weight이다.

일반적으로 콘볼루션날 코드는, nN개의 연속적인 디지트 안에서 발생하는 $[(d_{min}-1)/2]$ 이하의 에러들을 정정할 수 있는데, $[x]$ 는 x를 넘지 않는 최대의 정수를 의미한다.

(2) 신 드 롬 계 산

인코더에서 나온 코드 SEQUENCE는 통신로 상에서 잡음의 영향을 받게 된다. 이 잡음의 SEQUENCE를 $\mathbf{e}$ 라고 하면, 수신된 SEQUENCE는 다음과 같이 된다.

$$\mathbf{r}=\mathbf{c}+\mathbf{e} \qquad (19)$$

이 때 수신된 SEQUENCE에 대한 신드롬(Syndrome)을 다음과 같이 정의한다.

$$\mathbf{S}=\mathbf{r}\mathbf{H}_\infty^T \qquad (20)$$

그런데 식 (12),(19)를 (20)에 대입하면

$$\begin{aligned}\mathbf{S}&=\mathbf{r}\mathbf{H}_\infty^T\\&=(\mathbf{c}+\mathbf{e})\mathbf{H}_\infty^T\\&=\mathbf{e}\mathbf{H}_\infty^T\end{aligned} \qquad (21)$$

가 된다. 이것은 신드롬 디지트는 에러 디지트의 형태로도 나타낼 수 있음을 보이는 것으로, r대신에 e를 대입하면 x 번째 블럭의 (n-k) 신드롬 디지트는 j=1,2,···,n-k인 경우에 다음과 같이 된다.

$$\begin{aligned}S_x(j)=&e_x(k+j)+\sum_{i=1}^{k}e_x(i)q_{0n}(i,j)+\sum_{i=1}^{k}e_{x-1}(i)q_{1n}(i,j)\\&+\cdots+\sum_{i=1}^{k}e_{x-(N-1)}(i)q_{N-1}(i,j)\end{aligned} \qquad (22)$$

여기서 $t<0$이면 $e_t=0$ 이다.

식 (22)는 수신된 SEQUENCE의 신드롬은 에러 SEQUENCE에 관한 정보를 갖고 있으므로 디코딩에 사용할 수 있음을 알 수 있다. 그러므로 식 (22)와 (7)에서 인코더는 수신된 SEQUENCE의 신드롬 계산에 그대로 사용할 수 있음을 알 수 있다.

(3) 디 코 딩

디코딩은 다수결 논리 gate를 사용해서 수행하는데 그러기 위해서는 반드시 페러티 검사 합이 구성되어야 한다.

신드롬 디지트의 합을 다음과 같이 쓸 수 있다고 하자.

$$A=\sum_{j=1}^{nK}a_jS_0(j)+\sum_{j=1}^{nK}a_jS_1(j)+\cdots+\sum_{j=1}^{nK}a_jS_{N-1}(j) \qquad (23)$$

여기서 a_j 는 0이나 1이다.

A 는 에러 디지트의 합이므로

$$A=\sum_{l=1}^{n}b_le_0(l)+\sum_{l=1}^{n}b_le_1(l)+\cdots+\sum_{l=1}^{n}b_le_{N-1}(l) \qquad (24)$$

이며, b_l 은 0 이나 1 이다.

식 (24)를 페러티 검사 합이라고 한다.

종래에는 이 페러티 검사 합을 시행 착오적인 방법으로 찾았으나, 여기서는 $e_0(1)$에 직교한 합을 다음의 단계를 거쳐서 찾을 수 있는 새로운 방법을 제시하였다.

[1단계] 식 (22)처럼 신드롬 디지트를 에러 디지트의 합으로 표현한 것을, 각 블럭에 있는 각각의 에러 디지트 $e_l(i)$ (l =0,1,2,···,N-1 이고 i=1,2,3,···,n)의 순서대로 행으로 나열하여 표현한다.

[2단계] 각각 의 에러 디지트가 속해있는 신드롬을 찾아 그기에 해당되는 에러 디지트의 행에, 관계되는 신드롬 디지트를 모두 나열한다.

[3단계] 각 $e_l(i)$에서 2개 이상의 신드롬 항이 있는 것에 대하여,위·아래 행에서 공통으로 표현된 것이 있으면 그것을 찾아서 묶어준다. (즉, 더해준다.) 단, $e_0(1)$ 안에 묶을 수 있는 같은 것이 있으면 묶지 못한다.

[4단계] 어떤 행에서 묶을 수 있는 신드롬 항들이 $e_0(1)$안에 있는 것과 같은 것이 없으면 다른행과 관계없이, 그 행안에서 단독으로라도 그것들을 묶을 수 있다.

[5단계] 맨 먼저묶고 남은 것이나 혹은 원래 $e_l(i)$에 관계되는 신드롬 디지트가 짝수개이면 $e_0(1)$에 없는 것은 없앤다.

[6단계] 5단계에서 없앤것을 위·아래 행에서 찾아서 없앤다.

[7단계] 위와 같은 단계들을 행한 다음, 또 짝수개가 남은행을 찾아 5,6단계를 반복한다.

[8단계] 최종적으로 각 $e_l(i)$행에서 신드롬 항들을 더이상 묶을 수 없으면 수행을 멈춘다.

위의 단계들을 수행하면서 신드롬를 묶을 수 있는 것들의 합으로 나타내면 그것이 바로 페러티검사합이다.

다수결 논리 gate의 출력은, 입력의 과반수 이상이 1이면 1이고 그렇지 않으면 0이다.

본 논문에서는 이 다수결 논리 규칙에 의거하여 다수결 논리 gate대신 ROM을 사용함으로써 회로를 간단히 줄일수 있을뿐만 아니라, 디코딩 전파지연시간도 단축시킬 수 있는 새로운 방법을 채택했다.

3. (12, 3) 유니폼 코드의 CODEC설계

(1) (12, 3) 유니폼 코드

어떤 소수의 거듭 제곱수 q에 대해서 첫번째 정보 디지트가 0인 경우를 제외하고 모든 코드어들의 최소거리가 똑같은 q진수 코드를 유니폼 코드라고 한다. 이 유니폼 코드의 기본 형태는 다음과 같이 주어진다.

(1) 코드 디지트수 n = 2^q (2) 정보디지트수 K=1

(3) 코드율 R = $1/2^q$ (4)구속장 N=q+1

(5) Sub-generator $g(1,j)=(1, j_0, j_1, \cdots, j_{q-1})$

여기서 j=1, 2 $\cdots$, 2^q-1이고,

$(j_0, j_1, j_2, \cdots, j_{q-1})$은 q의 2진 표현이다.

6) Sub-generator수 = 2^{q-1}

7) 최소거리 $d_{min} = (q+2)\ 2^{q-1}$

8) 패러티 검사합수 $J = (q+2)\ 2^{q-1} - 1$

9) 랜덤에러 정정갯수 $t \leq [(q+2)2^{q-2} - \frac{1}{2}]$

본 논문에서는 q=2인 2진수의 경우를 살펴본다.

그러면 위의 식들에 의해 N=3, R= 1/4인 (4,1) 코드가 되며 $d_{min}=8$, J=7, $t \leq 3$이 된다.

또 3개의 Sub-generator는

i) $g(1,1)=(g_0(1,1)\ g_1(1,1)\ g_2(1,1)) = (1\ 1\ 0)$ (25)

ii) $g(1,2)=(g_0(1,2)\ g_1(1,2)\ g_2(1,2)) = (1\ 0\ 1)$ (26)

iii) $g(1,3)=(g_0(1,3)\ g_1(1,3)\ g_2(1,3)) = (1\ 1\ 1)$ (27)

가 된다.

(2) 인코더

N=3인 경우의 generator matrix는

$$G = \begin{bmatrix} 1111 & 0101 & 0011 \\ & 1111 & 0101 \\ & & 1111 \end{bmatrix} \quad (28)$$

와 같이 된다.

그러므로 (12, 3) 유니폼 코드의 인코더는 그림1과 같다.

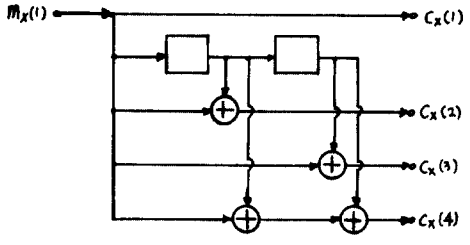

그림1. (12, 3) 유니폼 코드의 인코더

Fig.1. An encoder for the (12, 3) uniform codes

(3) 디코더

디코더를 구성하기 위해서 식(22)를 전개한 뒤 $e_0(1)$에 직교되는 패러티 검사합을 본 논문에서 제시한 방법으로 찾으면 다음과 같은 2가지 경우를 구성할수 있다.

경우 1)

$A_1 = S_0(1) = e_0(1) + e_0(2)$

$A_2 = S_0(2) = e_0(1) + e_0(3)$

$A_3 = S_0(3) = e_0(1) + e_0(4)$

$A_4 = S_1(3) = e_0(1) + e_1(1) + e_1(4)$

$A_5 = S_1(1) + S_1(2) = e_0(1) + e_1(2) + e_1(3)$

$A_6 = S_2(1) + S_2(3) = e_0(1) + e_2(3) + e_2(4)$

$A_7 = S_2(2) = e_0(1) + e_2(1) + e_2(3)$

경우 2)

$A_1 = S_0(1) = e_0(1) + e_0(2)$

$A_2 = S_0(2) = e_0(1) + e_0(3)$

$A_3 = S_0(3) = e_0(1) + e_0(4)$

$A_4 = S_1(1) = e_0(1) + e_1(1) + e_1(2)$

$A_5 = S_1(2) + S_1(3) = e_0(1) + e_1(3) + e_1(4)$

$A_6 = S_2(1) + S_2(3) = e_0(1) + e_2(2) + e_2(4)$

$A_7 = S_2(2) = e_0(1) + e_2(1) + e_2(3)$

위의 각 경우에 대해, $e_0(1)$은 A_1에서 A_7까지를 이용하여 다수결 논리 방법으로 추정할 수 있다.

(12, 3) 유니폼 코드의 디코더는 그림 2 와 같다.

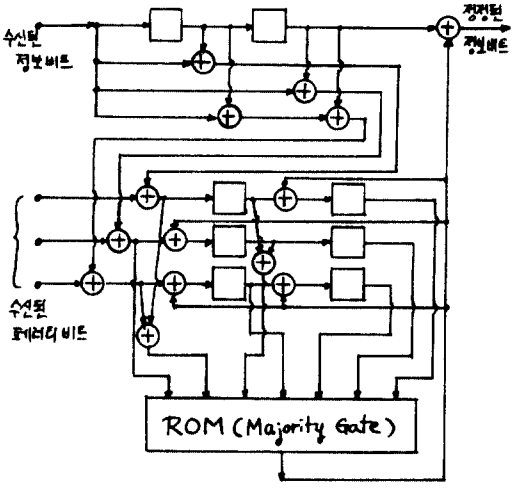

그림 2. (12, 3) 유니폼 코드의 디코더

Fig.2. A decoder for the (12, 3) uniform codes

4. 실험 및 결과 고찰

실험은 HP5035T Logic Lab과 Apple II 및 interface card를 이용하여 수행했으며, 다수결 논리 gate 대신 ROM 2716을 사용함으로써 디코딩 전파지연 시간을 640 nsec에서 450 nsec로 줄일수가 있었다. 실험결과 이 시스템을 사용하면 12 비트에 걸쳐서 있는 3개이하의 랜덤에러를 완벽히 정정할 수 있음을 알수 있었다. 또 (12, 3)일때의 performance를 Pc(E), (32, 4)일때의 P1(E), (4,1)일때의 P2(E) 및 코딩을 안 했을때의 $p_{nc}(E)$를 구한 결과를 그래프로 그리면 그림3과 같다. 퍼포먼스 분석결과 통신로상의 에러가 10^{-4}이하일때 이 시스템은 효율적임을 알 수 있었다.

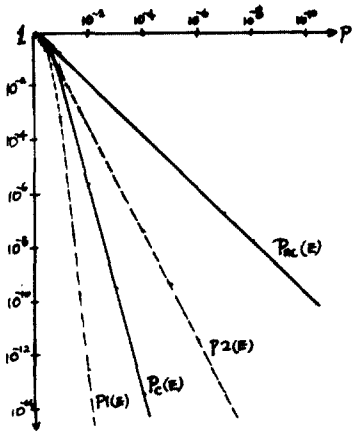

그림 3. 퍼포먼스 그래프

Fig.3. Performance graph

5. 결론

패러티 검사합을 찾는 방법을 새롭게 제시하여 그 방법에 따라 구한 합을 실제 실험을 통해서 확인한 결과, 예상했던 결과를 얻을 수 있었으며, 디코딩시 다수결 논리 gate 대신 ROM을 사용하는 방법을 처음시도했다. 다수결 논리 방법을 이용한 디코딩법은 정확히 한 구속장 만큼의 지연이 있으므로 신속성이 요구되는 시스템에서의 에러 조정에 효과가 있으며, 하드웨어가 간단하고, 경제적이므로 실용적인 측면에서 응용이 가능하리라고 생각되며, 앞으로 우리 나라에서도 펼쳐질 위성통신 방법에 콘볼루션날 코드를 이용한 에러 정정 방법이 절대적으로 필요하리라고 예상된다.

참고문헌

1. J.L. Massery, "Threshold Decoding," Cambridge, Mass.:M.I.T. Press,1963.
2. W.W.Peterson and E.J.Weldon, Jr., "Error-Correcting Codes," 2nd ed. Cambridge, Mass.:M.I.T.Press,1972.
3. R.E.Blahut,"Theory and Practice of Error Control Codes," N.Y.,Addison-Wesley,1983.
4. C.E.Shannon, "A mathematical theory of communication", Bell System Tech. J., Vol.27, PP.379-423, PP.623-656, July, 1948.
5. J.L.Massery, "Uniform Codes", Trans. Information Th.,Vol.IT-12,PP.132-134, April, 1966.
6. P.Elias, "Coding for Noisy channels", IRE Conv.Rec., pt.4, PP.37-46,1955.
7. L.D.Rudolph, "Generallized threshold decoding of convolutional codes", IEEE Trans. Information Th., Vol.IT-16, PP.739-743, Nov., 1970.