論 文

CRC 오류검출부호의 성능 분석

正會員 廉 興 烈* 正會員 權 周 漢** 準會員 梁 承 杜** 正會員 李 晩 榮***

Performance Analysis of CRC Error Detecting Codes

Heung Youl YOUM*, Joo Han KWON**, Seung Doo YANG**,
Man Young RHEE* *Regular Members*

要 約 본 논문에서는 단축 Hamming 부호의 일종이며 오류검출용 검사비트 수가 16인 CRC-CCITT 부호와 원시다항식 CRC 부호에 대한 성능 분석을 위하여 필수적으로 요구되는 중분포 (weight distribution)를 구하는 기법과 오류검출 성능을 분석하는 기법을 제안하였고, 두 CRC (cyclic redundant code) 부호를 CCITT에서 광대역 ISDN의 가입자망 인터페이스의 전송방식으로 권고된 ATM (asynchronous transfer mode) 전송방식의 오류검출용 부호로 적용하여 현재 고려되고 있는 cell 크기에 대한 중분포 및 미검출오류확률(undetected error probability)을 구한후, 두 오류검출부호의 성능을 비교/분석하였다.

분석 결과, 현재 고려되는 셀 크기에 대해 CRC-CCITT 부호의 성능이 원시다항식 CRC 부호의 성능보다 더 우수함이 입증되었다. 이를 위한 모든 계산을 IBM PC/AT를 이용하여 수행하였다. 한편 본 논문에서 제안한 단축 Hamming 부호의 성능 분석 기법은 지금까지 디지틀 통신시스템에 적용되고 있고 또는 적용예정인 CRC 오류검출 부호의 성능 분석에 이용될 수 있다.

ABSTRACT In this paper, the CRC-CCITT code and primitive polynomial CRC code are selected for analysing error detecting performance. However, general formulas for obtaining the weight distribution of these two CRC codes are not so far derived. So, a new method for calculating the weight distribution of the shortened cyclic Hamming code is presented and an undetected error probability of these two codes is obtained when used in cell of ATM for broadband ISDN user-network interface. Consequently, we show that CRC code is performing better than primitive polynimial CRC code for error detection of ATM Cell and shortening a code too much does affect its error detection performance. All the computer simulation is performed by IBM PC/AT.

*韓國電子通信研究所
Transmission Systems Section Electronics and Telecommunications Research Institute
**三星電子
Samsung Electronics.
***漢陽大學校 電子通信工學科
Dept. of Electronic Communication Engineering Han Yang University
論文番號：89-57(接受1989. 6. 9)

I. 서 론

지난 30여년 동안 전송로에서 발생한 오류를

제어하기 위한 부호이론에 대한 연구는 오류 검출/정정 능력이 우수한 오류 정정/검출 부호의 개발, 보다 효율적이고 간단한 부호화 및 복호 알고리즘 개발 등에 대하여 중점적으로 수행되어 왔다.[1]

신뢰성있는 정보 전송을 위하여 통신시스템에서 널리 이용되고 있는 오류제어기법은 FEC (forward error correction) 기법, ARQ (automatic repeated-request) 기법, 그리고 두방식을 결합한 복합오류제어 (hybrid error control) 기법 등이 이용되어 수행되었다.

FEC 기법에서는 기본적으로 부호화율 (k/n)이 비교적 작은 오류정정부호를 이용하며, 오류 발생 검출, 오류 위치 및 오류값 추정, 그리고 오류정정으로 구성된 세 단계를 거쳐 수행되며, 재전송을 요구할 수 없는 실시간 데이타 전송시스템에서의 정보의 신뢰성을 향상시키기 위해 도입/적용되고 있다. ARQ 기법은 일반적으로 부호화율이 비교적 큰 오류 검출부호를 이용하며, 수신기에서 전송된 데이타 블럭에 오류 발생 유무만을 검출한후 오류가 발생되었다고 판단되면 송신단에 해당 데이타 블럭의 재전송을 요구함으로서 해당 데이타 블록의 오류를 제어한다.

ARQ 기법은 시스템의 복잡도 측면에서 FEC 기법보다 비교적 간단하게 구현될 수 있으므로 통신채널의 오류 발생 확률이 높지 않은 패킷 교환 데이타망, 컴퓨터 통신망, 그리고 위성을 통한 데이타 통신망 등에 널리 적용되어 왔다.[3] ARQ 기법에서 널리 이용되고 있는 대표적인 오류검출부호로는 선형 부호(linear cyclic code)의 일종인 CRC 부호을 들수 있다. 따라서 CRC 오류검출부호의 성능은 ARQ 기법을 이용한 통신시스템의 성능에 직결되는 매우 중요한 변수가 된다. 그리고 복합 오류제어 기법은 FEC 기법과 ARQ 기법을 혼합한 기법으로 주로 위성을 통한 데이타 전송시스템에 있어서 성능 향상 및 전송 효율의 향상을 위하여 최근 연구/도입되고 있다.

본 논문에서는 단축 Hamming 부호의 일종이며 오류검출용 검사 비트수가 16인 CRC-CCITT 부호와 원시다항식 CRC 부호를 도입하여, 이들 단축 Hamming 부호에 대한 일반적인 성능 분석 기법을 제안하였고, 두 CRC 부호를 CCITT 광대역 ISDN 가입자망 인터페이스의 전송방식으로 권고된 ATM의 cell에 도입/적용하여 현재 고려되고 있는 각각의 cell 크기에 대한 중분포 (weight distribution) 및 미검출오류확률 (undetected error probability)을 구하고, 두 부호에 대한 성능을 비교/분석하였다.

Ⅱ. 오류검출부호

Ⅱ.1 오류검출부호의 종류 및 오류검출 알고리즘

현재까지 디지틀 통신시스템에 널리 이용되고 있는 CRC 부호는 CRC-12, CRC-16, CRC-CCITT, 그리고 CRC-32 등이며, 이 부호들의 생성다항식 (generator polynomial)은 다음과 같다.[3][5]

*CRC-12의 생성다항식: $x^{12}+x^{11}+x^3+x^2+1$

*CRC-16의 생성다항식: $x^{16}+x^{15}+x^2+1$

*CRC-CCITT의 생성다항식: $x^{16}+x^{12}+x^5+1$

*CRC-32 의 생성다항식:

$$x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^8+x^7+x^5+x^4+x^2+1$$

CRC-12 부호는 6-비트 캐릭터에 대한 오류검출부호로, CRC-16 부호는 BSC (binary synchronous communications) 프로토콜에서의 8-비트로 구성된 EBCDIC (extended binary coded decimal interchange code) 케릭터용 오류검출부호로, CRC-CCITT 부호는 SDLC (synchronous data link control) 프로토콜에서의 데이타 필드에서 발생한 오류를 제어하기 위하여 이용되고 있다. 특히 CRC-CCITT 부호는 미국 IBM 사의 8-inch 프러피 디스크의 데이타 보호를 위하여 적용되고 있으며, 많은 여타 사의 FDC (floppy

disk controller)에 적용되고 있는 대표적인 오류 검출부호이다. 그리고 CRC-32 부호는 LAN (local area network)의 일종인 Ethernet의 데이터 링크 레이어의 데이타 필드에 대한 오류검출을 위해 채택/사용되고 있다.[11]

CRC 오류검출부호를 이용한 오류 검출 과정은 기본적으로 다음과 같다. 전송될 k비트의 정보 블럭은 다음과 같은 k-1 차의 정보다항식, $d(x)=d_0+d_1x+d_2x^2+\cdots+d_{k-1}x^{k-1}$으로 표현될 수 있다. 일반적으로 CRC 오류검출부호의 생성다항식은 차수가 n-k 차인 g(x)로 표현되므로, 오류 검출을 위한 검사비트 (redundancy bit)를 다항식 형태로 표현한 검사 다항식 (redundant check polynomial), r(x)는 다음과 같은 식을 이용하여 구해진다.[1][6][7]

$$x^{n-k}d(x)=q(x)g(x)+r(x) \qquad (1)$$

여기서, n-k: 생성다항식의 차수

q(x): $x^{n-k}d(x)$를 g(x)로 나눈 몫다항식

r(x): 검사다항식

따라서 오류검출을 위한 CRC 검사 비트는 (1)식의 r(x)에서 구해진다. 송신기는 k비트의 정보 다항식에 n-k 비트의 여분의 검사 다항식을 부가하여 실제 통신로로 전송될 부호어를 생성한다. 이는 (2)식과 같이 표현된다.

$$c(x)=x^kd(x)+r(x) \qquad (2)$$

(1)과 (2)식에서 알 수 있듯이 부호 다항식, c(x)는 g(x)로 나누어 떨어진다.

한편, 전송로에서 발생한 오류다항식 (error polynomial)를 e(x)라 하면, 전송로를 통해 수신된 수신다항식 (received polynomial), v(x)는 (3)식과 같이 표현될 수 있다.

$$v(x)=c(x)+e(x) \qquad (3)$$

(1),(2),(3) 식에서 알 수 있듯이 전송채널에서 오류가 발생하지 않았을 경우, 즉 e(x)=0인 경우, v(x)는 g(x)로 나누어 떨어지므로 나머지 다항식이 "0"이 된다. 그러나 전송채널에서 오류가 발생한 경우, 즉 e(x)=0인 경우, v(x)는 g(x)로 나누어 떨어지지 않음으로 나머지 다항식이 "0"이 되지 않는다. 위와같은 특성을 이용하여 수신기는 수신 다항식에 오류 발생 유무를 검출하기 위하여 v(x)를 g(x)로 나눈후 나머지가 "0"인 경우 해당 정보 블럭에 오류가 발생하지 않은 것으로, 나머지가 "0"이 아니면 해당 정보 블럭에 오류가 발생한 것으로 판단하여 송신기로 해당 정보 블럭을 재전송하도록 요구하므로서 해당 정보 블럭에 발생한 오류를 교정하고 있다. 그러나 오류다항식이 부호다항식, 즉 부호어와 같은 형태로 발생한 경우, 오류가 발생했음에도 불구하고 수신기는 오류를 검출하지 못할 것이다. 이와 같은 사건을 미오류검출사건이라 지칭하며 이와 같은 사건이 발생할 확률이 오류검출부호의 성능을 나타내는 미검출오류확률이다. 따라서 오류검출 부호의 성능을 분석하기 위해서는 g(x)에 의해 생성 가능한 부호어를 구하는 것이 먼저 이루어져야 한다.

CRC-CCITT 오류검출부호를 이용한 검사 비트는 〈그림 1〉과 같은 쉬프트 레지스터와 exclusive-OR gate로 구성된 회로에 의해 생성된다. 수신단의 오류 검출기는 〈그림 1〉과 비슷한

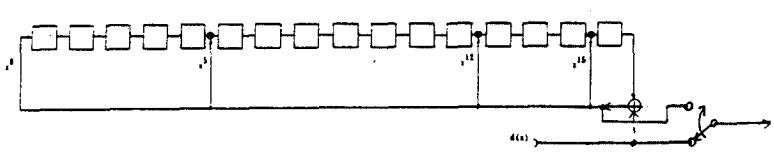


그림 1. CRC 비트 생성기
generator circuit for CRC bit

회로로 구현될 수 있다.

Ⅲ. ATM 전송방식

ISDN (Integrated Service Digital Network)은 음성과 비음성 서비스를 하나의 통합된 망을 통해서 제공하는 것을 바탕으로 하여 세계각국에서 활발히 구축되고 있다. ISDN은 기본 액세스(basic access)인 2B+D (B: 64Kb/s, D: 16 Kb/s)서비스와 1차 액세스(primary access)인 23B+D (또는 30B+D) 서비스 제공을 바탕으로 한 협대역 ISDN과 H1(1.544Mb/s) 급 이상의 고속 데이타, 고속 비디오 서비스 등의 광대역 서비스 제공을 바탕으로 하는 광대역 ISDN으로 분류될 수 있다. 현재 CCITT에서 권고되고 있는 광대역 ISDN의 가입자망 인터페이스 (UNI: user network interface)의 기준 모델은 〈그림 2〉와 같다. [4]

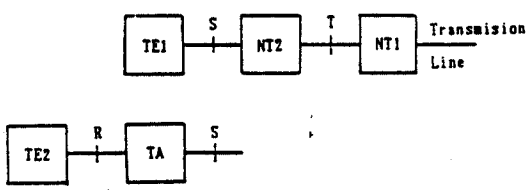


그림 2. 광대역 ISDN의 UNI의 기준 모델
reference model for UNI of broadband ISDN

현재 CCITT에서 논의/고려되고 있는 광대역 ISDN의 UNI에 적용되는 전송방식은 기본적으로 STM (synchronous transfer mode) 전송 방식과 ATM (asynchronous transfer mode) 전송 방식이 고려되고 있으나, 이중 ATM 전송방식이 권고될 예정이다. STM 전송방식은 프레임을 구성한후 특정채널의 데이타를 특정 타임슬롯에 할당하여 전송하는 전송방식이다. 이 방식을 이용하면 특정 채널이 서비스를 포함하지 않더라도 해당 타임슬롯을 다른 채널이 이용할 수 없는 단점이 있으나, 회로 구현이 간단하여 지금까지의 많은 디지탈 전송시스템에 널리 이용되고 있는 전송 방식이다. ATM 전송방식은 〈그림 3〉과 같이 프레임을 구성한후 시분할 기법을 이용하여 가입자 정보를 패켓단위로 구분하여 전송하는 패켓 지향적인 전송방식으로서, 가입자의 정보는 cell 단위의 패캣에 삽입되어 전송된다.[4]

cell은 〈그림 4〉와 같이 정보부와 헤더부로 구성되어 있으며, 정보부에는 가입자의 데이타를, 헤더부에는 정보부에 대한 서비스 종류, 라우팅, 오류정정/검출 부호 비트 등의 정보를 포함한다. 따라서 이 전송방식을 이용하면 가입자의 여러 다양한 서비스에 융통성있게 대처할 수 있으나 시스팀 실현의 복잡도가 STM 전송방식보다 큰 단점이 있다.

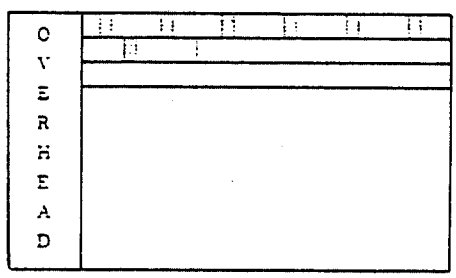


그림 3. 고려되고 있는 ATM 전송방식에서의 프레임 구성
frame format for ATM

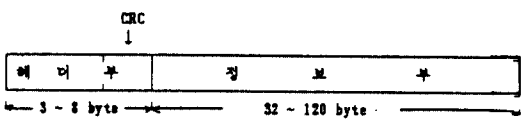


그림 4. ATM 전송방식에서의 cell 구조
cell structure for ATM

현재 (89.2) CCITT에서는 cell의 헤더부 크기를 3-8바이트, 정보부의 크기를 32-120 바이트로 잠정 권고하고 있다.[4]본 논문에서는 두 종류의 CRC 오류검출부호를 헤더부에 도입하여 정보부 및 헤더부에서 발생한 오류를 검출하는 방안을 제안하였다. 이를 기반으로 하여, 일반적인 단축

Hamming 오류검출부호의 성능 분석의 일반적인 이론을 분석하고, 두 오류검출부호에 적용될 수 있는 성능 분석 알고리즘을 제시하고 두 CRC 검출부호의 성능을 분석한후 이를 비교/분석한다.

Ⅳ. 미검출오류확률

Ⅳ-1. MacWilliams 항등식과 미검출오류확률

MacWilliams 항등식은 (n,k) 선형 부호, C의 중 분포 (weight distribution)와 C의 쌍대부호 (dual code), C'의 중분포 간의 관계식을 나타내며, 이는 (3)식과 같이 표현된다.[1][2][6][7]

$$A(x)=2^{-(n-k)}(1+x)^n B(1-x/1+x) \qquad (3)$$

여기서 A(x): C 의 중분포 다항식
B(x): C'의 중분포 다항식

(3)식은 C의 정보장(k)과 C'의 정보장(n-k)의 관계가 k〉〉n-k을 만족할 경우, C'의 중분포를 구한후 C의 중분포를 쉽게 구할 수 있음을 의미한다.

이원대칭채널 (BSC: binary symetric channel) 을 통해 (n,k) 선형 부호가 전송되었을 경우, 채널상에서 발생한 오류 다항식이 (n,k) 선형 부호의 부호어와 일치하면, 수신다항식은 또 다른 (n,k) 선형부호의 부호어가 되어, 수신기는 송신 벡타에 오류가 발생했음에도 불구하고 이를 검출할 수 없게 된다. 이와 같은 사건은 오류 다항식이 (n,k) 선형 부호의 부호어중 하나와 같은 형태로 일어날 경우 발생하며, 이와같은 오류형태가 발생할 확률이 미검출오류확률이라 정의된다. 따라서 (n,k) 선형 부호의 미검출오류확률은 (4)식과 같이 표현될 수 있다.

$$P_{ud}(e)=\sum_{i=1}^{n} A_i e^i (1-e)^{n-i}$$
$$= \sum_{i=d_{min}}^{n} A_i e^i (1-e)^{n-i} \qquad (4)$$

여기서 e는 BSC에서의 천이확률 (transition probability), dmin은 선형부호 C의 최소거리 (minimum distance)이다. 그리고 A_i는 (5)식과 같은 중분포다항식의 중분포 계수를 의미한다.

$$A(x)=A_0+A_1x^1+A_2x^2+\cdots+A_nx^n$$
$$=\sum_{i=0}^{n} A_i x^i \qquad (5)$$

(4)식은 (5)식을 이용하여 (6)식과 같이 변경될 수 있다.

$$P_{ud}(e)=(1-e)^n(A(e/1-e)-1) \qquad (6)$$

(6)식을 (3)식에 대입하면, 미검출오류확률은 쌍대부호의 중분포로 (7)식과 같이 표현될 수 있다.

$$P_{ud}(e)=2^{-(n-k)}B(1-2e)-(1-e)^n$$
$$=2^{-(n-k)}\sum_{i=0}^{n} B_i(1-2e)^i-(1-e)^n \qquad (7)$$

여기서, $\sum_{i=0}^{n} B_i=2^{n-k}$

(n,k) 선형부호, 또는 쌍대부호의 중분포를 알 수 있을 경우, (n,k) 선형부호의 미검출오류확률은 (4)식 또는 (7)식을 이용하여 구해진다. 특히 n-k 〈〈k인 경우, 쌍대부호의 중분포는 원래 부호의 중분포보다 쉽게 구해지므로 (7)식을 이용하여 이 부호의 미검출오류확률을 구할 수 있다. 따라서 C와 C'의 부호의 특성이 명확히 알려져 있고 C'의 중분포 B_i가 비교적 쉽게 구해질 수 있을 경우, 미검출오류확률은 (7)식을 이용하여 구하는 것이 유리하다. 그러나 C와 C'관계가 명확하지 않고 C'의 중분포가 쉽게 구할 수 없을 경우, 미검출오류확률은 binomial approximation 방법 또는 컴퓨터 시뮬레이션 기법[9]등을 이용하여 C의 중분포 A_i를 추정하여, 이를 이용하여 미검출오류확률을 구해야 한다. 이후부터 C와 C'의 관계가 명확한 Hamming 부호를 단축한 단축 Hamming 부호의 특성을 이용하여, 이 부호의 미검출오류확률을

구하는 새로운 기법을 제안한다.

Ⅳ-2. 단축 Hamming 부호의 특성

선형부호의 미검출오류확률이 e가 증가함에 따라 단조증가한 경우, 이 부호는 proper 부호라 정의된다. (n,k) 선형부호가 미검출오류확률의 상한치 (upper bound)를 만족하는 선형부호의 종류는 다음과 같은 특성에서 알수 있다. 특히 proper 부호의 미검출오류확률은 $2^{-(n-k)}$ 상한치이내임이 증명되었다.[5] 여기서는 proper 부호와 관련된 특성을 도출한다.

[특성 1] 선형부호는 반드시 proper 부호가 아니다.
[특성 2] 순회부호는 반드시 proper 부호가 아니다.
[특성 3] Single parity-check 부호는 proper 부호이다.
[특성 4] 최대장계열부호는 proper 부호이다.
[특성 5] 이중오류정정 BCH 부호는 proper 부호이다.
[특성 6] Hamming 부호와 Golay (23,12) 부호 등의 완전 (perfect) 부호는 proper 부호이다.
[특성 7] 완전부호의 쌍대부호는 proper 부호이다.
[특성 8] proper 부호의 쌍대부호는 반드시 proper 부호가 아니다.
[특성 9] 단축 Hamming 부호는 항상 proper 부호가 아니다.
[특성 10] 확대 Hamming 부호는 proper 부호이다.

여기서 단축 Hamming 부호의 미검출오류확률과 밀접한 관련이 있는 [특성 9]를 다음과 같이 증명한다.

[특성 9] 단축 Hamming 부호는 언제나 proper 부호가 아니다.
[증명]
다음과 같은 m차 원시다항식을 생성다항식으로 갖는 (n,k) 순회 Hamming 부호를 생각하여 보자.

$$g(x)=x^{10}+X^3+1$$

위의 생성다항식으로 부터 (1023,1013) Hamming 부호의 비조직형 생성행열은 다음과 같이 구성될 수 있다.

$$G=\begin{bmatrix} 1\,0\,0\,1\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\cdots 0 \\ 0\,1\,0\,0\,1\,0\,0\,0\,0\,0\,0\,1\,0\,0\cdots 0 \\ 0\,0\,1\,0\,0\,1\,0\,0\,0\,0\,0\,0\,1\,0\cdots 0 \\ \vdots \\ \end{bmatrix}$$

위의 생성다항식으로 얻어진(1023,1013) Hamming 부호를 1012 비트 단축시킨 (11,1) 단축 Hamming 부호의 생성다항식은 다음과 같다.

$$G=[1\,0\,0\,1\,0\,0\,0\,0\,0\,0\,1]$$

따라서 이 생성다항식을 이용한 (11,1) 단축 Hamming 부호의 미검출오류확률은 다음과 같다.

$$P_{ud}=e^3(1-e)^8$$

미검출확률의 최대값은 e=3/11 일때 임을 알 수 있다. 따라서 이론적인 미검출오류확률의 상한치와 (11,1) 단축 Hamming 부호의 미검출확률의 최대값을 비교하면 다음과 같다.

$$P_{ud}(3/11)=1.59\times 10^{-3} \rangle 2^{-(11-1)}=9.77\times 10^{-4}$$

그러므로 단축 Hamming 부호는 반드시 proper 부호가 아니다.
〈Q.E.D〉

V. 단축 Hamming 부호의 성능 분석

본 절에서는 검사비트의 길이가 16인 CRC 오류검출부호를 도입한후, 이에 대한 미검출오류확률을 구하기 위하여 단축 Hamming 부호의 일반적인 중 분포를 구하는 방법과 단축 순회 Hamming 부호와 쌍대 부호와의 대수학적 관계를 이용하여 이에 대한 미검출오류확률을 구하는 기법을 제시한다. 이를 위해 도입된 오류검출부호는 CRC-CCITT 오류 검출부호와 16차 원시다항식으로 생성된 오류검출부호이다.

용어의 혼동을 피하기 위하여 Hamming 부호의 여러 version과 각각 부호의 중분포에 대한 약어를 다음과 같이 정의한다.

C_2m_{-1}: $(2^m-1, 2^m-m-1)$ 순회 Hamming 부호
$C_2m_{-1,e}$: $(2^m-1, 2^m-m-2)$ 소거 Hamming 부호
C_n: 순회 Hamming 부호를 단축시킨 (n,n-m) 단축 Hamming 부호
C'_n: C_n의 쌍대부호
$C_{n,e}$: 소거 Hamming 부호를 단축시킨 (n,n-m) 단축 소거 Hamming 부호
$C'_{n,e}$: $C_{n,e}$의 쌍대부호
$B_{n,l}$: C'_n의 중분포
$B_{n,l,n}$: $C'_{n,e}$의 중분포

V.1 CRC-CCITT 오류검출부호의 성능

CRC-CCITT 오류검출부호의 생성다항식은 다음과 같이 생성된다.

$$g(x)=(1+x)p(x)=(1+x)(x^{15}+x^{14}+x^{13}+x^{12}+x^4+x^3+x^2+x+1)$$
$$=x^{16}+x^{12}+x^5+1$$

여기서 p(x)는 15차 원시다항식이다.

CRC-CCITT 오류검출부호의 생성다항식은 15차의 원시다항식에 1+x를 곱하여 형성된다. 따라서 CRC-CCITT 오류검출부호는 15차의 원시다항식을 생성다항식으로 갖는 기존의 $(2^m-1, 2^m-m-1)$ Hamming 부호의 부호어중 짝수 중 (even weight)을 갖는 부호어로 구성되어 있는 부호로서, 이는 소거 Hamming 부호 (expurgated Hamming code)라 정의되고 있다.[6][7][8] 한편 ATM cell의 부호장 (code length)이 최소 33에서 최고 128 바이트이므로, 소거 Hamming 부호를 단축하여 cell의 부호장에 맞는 단축 소거 Hamming 부호를 구성한후, 이 단축 소거 Hamming 부호의 쌍대부호의 중분포를 구한후 이 부호들의 대수학적 특징을 이용하여 오류검출부호의 미검출오류확률을 구할 수 있다.

단축 소거 Hamming 부호로 부터 직접 쌍대부호의 중분포를 구하는 것은 쉽지 않으므로 본 논문에서는 원시다항식 p(x)로 부터 생성된 순회 Hamming 부호를 단축시킨 단축 Hamming 부호 (C_n)을 이용하여 C'_n으로 부터 $C'_{n,e}$과의 중분포 관계식을 이용하여 소거 단축 Hamming 부호의 중분포를 구하는 간접적인 방식을 이용하였다.

GF(2^m)상의 m차 원시다항식은 일반적으로 (8)식과 같이 표기된다.

$$p(x)=\sum_{j=0}^{m} p_j x^j \qquad (8)$$

(8)식의 원시다항식을 이용하여 생성된 $(2^m-1, 2^m-m-1)$ Hamming 부호의 쌍대부호는 "0"벡타를 제외한 모든 부호어의 중이 2^{m-1}인 simplex 부호로서, 부호의 제원은 다음과 같다.[6][7][8]

부호장 (code length), $n=2^m-1$
정보장 (information length), $k=m$
최소거리 (minimum distance), $d_{min}=2^{m-1}$

한편 simplex 부호는 p(x)의 계수 p_j가 feedback 탭의 계수인 m 단 시프트 레지스터를 이용하여 생성된다. 그리고 연속된 계열의 주기가 2^m-1이므로 simplex부호는 최대장계열부호 (maximal length sequence code, 이하 C_{max}라 표기)라 정의되기도 한다. CRC-CCITT 오류검출부호는 m이 15이므로 이와 대응되는 최대장계열부호, C_{max}는 〈그림 5〉와 같은 회로를 이용하여 형성될 수 있

다.

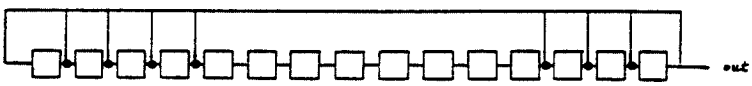


그림 5. C_{max}를 구하기 위한 회로

C_{max}는 (8)식을 이용하여 다음과 같이 표현될 수 있다.

$a_0=1$

$a_i=0,\ 1\le i\le m-1$

$$a_i=\sum_{j=0}^{m-1} p_j a_{i+j-m},\ m\le i< 2^m-1 \qquad (9)$$

$a_0=1$, $a_i=0$, $1\le i\le m-1$로 설정은 m 단의 시프트 레지스터의 초기값을 $GF(2^m)$ 상의 체원소 (field element) 중의 하나의 원소에 대응한 것이며, 여기서는 α에 대응한 벡터값을 회로의 초기값으로 설정하였다. (9)식을 이용하여 순차적으로 $a_0, a_1, a_2, \cdots, a_m, \cdots, a_{2^m-1}$을 생성하여, 이 계열을 2^m-1 비트씩 나눈 2^m-1 개의 계열은 각각 C'_{2^m-1}의 한 부호어가 된다. 그리고 이 계열을 n bit 크기로 나눈 2^m-1개의 벡터가 영벡타를 제외한 C'_n의 모든 부호어이므로 이 계열을 이용하여 C'_n의 중, $B_{n,i}$ 을 계산할 수 있다. 따라서 C'_n의 중분포 $B_{n,i}$와 C_n의 중분포 $B_{n,i,e}$의 관계식을 이용하여 $B_{n,i,e}$를 구할 수 있다. 따라서 CRC-CCITT 오류검출부호의 미검출오류확률은 $B_{n,i,e}$를 이용하여 계산된다. 그러나 이 방법은 최대장계열을 한 비트씩 쉬프트한후 이에 대한 중을 계산해야 하므로 m이 증가함에 따라 엄청난 반복계산이 요구되고 따라서 계산시간 또한 오래 걸리는 단점이 있다. CRC-CCITT 오류검출부호의 경우, m이 15이므로 ($2^{15}-1=32767$) 개의 최대장계열비트를 32767번 반복하여 중분포를 구할 수 있다. 이러한 문제점을 극복하기 위하여 Trace 개념을 사용하여 a_i를 비트 단위가 아닌 여러 비트 단위로 계산하는 기법의 이론적 배경을 다음부터 제시한다..

$GF(2^m)$ 상의 임의의 유한체 원소를 β라 할 경우, β의 Trace 값을 다음과 같이 정의된다,.

$$Tr(\beta)=\sum_{j=0}^{m-1} \beta^{2^j}\ Tr(\beta)\in GF(2) \qquad (10)$$

그리고 p(x)의 근을 α라 정의할 때 b_i를 다음과 같이 정의한다.

$b_i=Tr(\alpha^i),\ 0\le i\le 2^m-1$

임의의 h, $0\le h<m$에 대해 α^{2^h}가 p(x)의 근이 되므로, 이 특성을 (8)식에 대입하고 Trace의 선형성을 이용하여 다음과 같은 식이 유도될 수 있다.

$$p(\alpha^{2^h})=\sum_{j=0}^{m} p_j(\alpha^{2^h})^j$$

$$=\sum_{j=0}^{m-1} p_j(\alpha^{2^h})^j+p_m(\alpha^{2^h})^m \qquad (11)$$

(11) 식은 다음과 같이 변형될 수 있다.

$$(\alpha^{2^h})^m=\sum_{j=0}^{m-1} p_j(\alpha^{2^h})^j$$

윗식의 양변에 α^i를 곱한후 Trace를 취하면 다음과 같은 식이 유도된다.

$$Tr(\alpha^{i+m2^h})=Tr(\alpha^i\sum_{j=0}^{m-1} p_j(\alpha^{j2^h}))$$

$$=Tr(\sum_{j=0}^{m-1} p_j(\alpha^{i+j2^h}))$$

$$=\sum_{j=0}^{m-1} p_j\ Tr(\alpha^{i+j2^h}))$$

$b_i=Tr(\alpha^i)$이므로 위의 식은 다음과 같이 변형될 수 있다.

$$b_{i+m2^h}=\sum_{j=0}^{m-1} p_j b_{i+j2^h},\ 0\le i\le 2^m-2 \qquad (12)$$

h=0인 경우, (12) 식은 다음과 같이 된다.

$$b_{i+m}=\sum_{j=0}^{m-1} g_j b_{i+j},\ 0\leq i\leq 2^m-2$$

위 식에서 i+m를 k로 치환하면,

$$b_k=\sum_{j=0}^{m-1} g_j b_{i+j-m},\ 0\leq i\leq 2^m-2$$

로 되어 (9)식과 같은 형태로 변경될 수 있다. 따라서 (9)식, (12)식으로 부터 특정 정수 u에 대해 (13)식이 성립한다.

$$b_{i+u}=a_i \tag{13}$$

한편 $h\neq 0$인 일반적인 경우를 유도하기 위하여 (12)식에서 $i+m2^h=k$로 치환하면 다음과 같이 변형된다.

$$b_k=\sum_{j=0}^{m-1} g_j b_{k+(j-m)2^h} \tag{14}$$

(14)식은 (9)식에 a를 b로, p를 g로 치환하고, $h=0$인 경우 (9)식과 (14)식은 완전히 일치하므로 $h\neq 0$인 일반적인 경우에 대한 최대장계열을 (9)식 대신(14)식의 형태로 표현하면 (15)식과 같다.

$$a_i=\sum_{j=0}^{m-1} g_j a_{i+(j-m)2^h} \tag{15}$$

따라서 (15)식을 이용하면, 최대장계열은 비트 단위가 아닌 2^h 비트 단위로 구할 수 있다. 2^h개의 $\bar{a}_i$를 하나의 벡터로 구성하기 위해 벡터 $\bar{a}_i$를 다음과 같이 정의하자.

$$\bar{a}_i=(a_{i2^h}, a_{i2^h+1}, \cdots, a_{i2^h+2^h-1}) \tag{16}$$

(16)식을 (15)식에 대입하여 전개하면 (17)식이 유도된다.

$$\begin{aligned}\bar{a}_i &= \Big(\sum_{j=0}^{m-1} g_j a_{(i+j-m)2^h},\ \sum_{j=0}^{m-1} g_j a_{(i+j-m)2^h+1},\ \cdots,\\ &\quad \sum_{j=0}^{m-1} g_j a_{(i+j-m)2^h+2^h-1}\Big)\\ &= \sum_{j=0}^{m-1} g_j \left(a_{(i+j-m)2^h},\ a_{(i+j-m)2^h+1},\ \cdots,\right.\\ &\quad \left. a_{(i+j-m)2^h+2^h-1}\right)\\ &= \sum_{j=0}^{m-1} g_j \bar{a}_{i+j-m}\end{aligned} \tag{17}$$

여기서 2^h는 구체적으로 계산시 처리해야할 비트 수이다. 따라서 효율적으로 계산을 수행하기 위하여 컴퓨터에서의 한 워드 크기에 해당하는 h를 선택하는 것이 계산속도를 빠르게 할 수 있다. (17) 식을 이용하여 (16) 식의 2^h비트로 구성된 최대장계열 벡타를 순차적으로 구하기 위하여 (9) 식에 의해 $a_0, a_1, a_2, \cdots, a_{m2^h-1}$을 먼저 구해야 한다. 위의 계열을 2^h비트 단위로 나누어 $\bar{a}_0, \bar{a}_1, \cdots, \bar{a}_{m-1}$의 최대장 벡터를 구성한다. 그러면 나머지 최대장계열 벡터들, $\bar{a}_m, \bar{a}_{m+1}, \bar{a}_{m+2}, \cdots, \bar{a}_{2^m-h-1}$, 은 (17)식을 이용하여 m개의 벡터들 끼리의 벡터 단위 Exclusive OR 연산을 통해 차례대로 구해질 수 있다. 이렇게 구해진 최대장 계열 벡타들로 부터 C'_{2^m-1}의 부호어는 구해질 수 있다. 그리고 C'_n의 중분포를 구하기 위하여, 위에서 구한 최대장계열을 n 비트 단위로 나눈후, 각각의 벡터가 C'_n의 부호어가 되므로 부호장이 n인 2^m-1 개의 부호어는 (18)식과 같이 형성된다.

$$\bar{v}_i=(a_i, a_{i+1}, a_{i+2}, \cdots, a_{i+n-1}),\ 0\leq i\leq 2^m-2 \tag{18}$$

(18)식으로 부터 C'_n의 부호어의 중은 (19)식을 이용하여 구할 수 있다.

$$w(\bar{v}_{i+1})=w(\bar{v}_i)+a_{i+n}-a_i \tag{19}$$

여기서 w는 벡터의 중을 나타내며, a_{i+n}에서의 i+n은 modulo 2^m-1 연산이 적용된다. (19)식을 이용하여 C'_n의 중분포$(B_{n,i})$를 구할 수 있다. 한편, 단축 소거 Hamming 부호의 쌍대부호 $C'_{n,e}$의 중분포 $B_{n,i,e}$와 $B_{n,i}$의 관계는(20)식과 같다.[12]

$$B_{n,i,e}=B_{n,i}+B_{n,n-i} \tag{20}$$

(20)식에서 구한 중분포를 (7)식에 대입하면 BSC에서의 채널천이확률, e에 따른 단축 소거 Ham-

ming 부호의 미검출오류확률을 구할 수 있다.

V-2. 16차 원시 다항식 CRC 오류검출부호의 성능

CRC-CCITT 오류검출부호의 생성 다항식은 16 차의 원시다항식이 아니므로 일단 p(x)로 Hamming 부호, C_{2^m-1}를 단축시켜서 원하는 부호장 및 정보장을 갖는 단축 Hamming 부호, C_n의 쌍대부호의 중분포를 구한후, 이를 (20)식에 대입하여 단축 소거 Hamming 부호의 쌍대부호의 중분포, $B_{n,l,e}$를 구했다. 그러나 오류검출부호의 생성다항식이 원시다항식일 경우, 단축 소거 Hamming 부호가 아닌 원래의 Hamming 부호, C_{2^m-1} 를 구성할 수 있다. 본 논문에서는 cell의 오류검출부호의 생성다항식을 16차 원시다항식 중 하나를 채택하여 V.1 절과 비슷한 과정을 반복하여 (n,n-m) 단축 Hamming 부호의 쌍대부호의 중분포와 미 검출오류확률을 계산하였다. 16차 원시다항식은 (21)식과 같다.[1][6]

$$g(x)=x^{16}+x^{12}+x^3+x+1 \qquad (21)$$

VI. 성능 분석 절차 및 분석 결과 검토

CRC-CCITT 부호와 원시다항식 CRC 부호의 쌍대부호의 중분포 및 미검출오류확률을 각각 구하기 위해서 다음과 같은 절차에 의해 성능분석을 수행하였다. 이를 위해 이용된 컴퓨터는 IBM PC / AT 이고, 프로그램 언어는 Basic 언어를 이용하였다. 그리고 프로그램 길이는 400 라인 정도였다.

VI.1 CRC-CCITT 오류검출부호의 성능 분석 절차

단계 1) (9)식에 의해 a_0-a_{239}까지 최대장계열을 생성.

단계 2) a_0-a_{239}를 2^4 (=16) 비트 단위로 분할하여 벡터 $\bar{a}_0$-a_{14} 형성

단계 3) (17)식을 이용하여, 벡터 $\bar{a}_{15}$-$\bar{a}_{2047}$ 형성.

단계 4) 최대장계열 a_0-a_{32767} 배열.

단계 5) C'_n의 부호장이 n이므로 a_0-a_{n-1}까지의 중을 구한후, 이를 벡터 $\bar{v}_0$의 중으로 선택.

단계 6) (19)식을 이용하여 32767 개의 벡터 $\bar{v}_i$의 해밍중 (Hamming weight) 계산.

단계 7) (20)식을 이용하여 $B_{n,l,e}$를 계산.

단계 8) (7)식을 이용하여 CRC-CCITT의 미검출 오류확률 구함.

VI.2 원시다항식 CRC 오류검출부호의 성능 분석 절차

단계 1) (9)식에 의해 a_0-a_{255}까지 최대장계열 생성.

단계 2) a_0-a_{255}를 2^4 (=16) 비트 단위로 분할하여 벡터 $\bar{a}_0$-$\bar{a}_{15}$ 형성.

단계 3) (17)식을 이용하여, 벡터 $\bar{a}_{16}$-$\bar{a}_{4095}$ 형성.

단계 4) 최대장계열 a_0-a_{65535} 배열.

단계 5) C'_n의 부호장이 n 이므로 a_0-a_{n-1}까지의 해밍중을 구한후, 이를 벡터 $\bar{v}_0$의 중으로 선택.

단계 6) (19)식을 이용하여 65535 개의 벡터 $\bar{v}_i$의 중 계산.

단계 7) (7)식을 이용하여 원시다항식 CRC 부호의 미검출오류확률 계산.

CCITT에서는 ATM cell의 정보부의 크기를 32-120 바이트로, 헤더부의 크기를 3-8 바이트로 잠정 권고하고 있음을 고려하여, (n,n-m) 단축 소거 Hamming 부호의 부호장 n을 ATM cell의 최대 크기와 최소 크기내에서 변환시켜 가면서 각 길이 n에 대해 두개의 오류검출부호에 대한 중 분포들을 구하고 이를 기초로하여 각 오류검출부호에 대한 미검출오류확률을 계산하였다. VI.1과VI.2절의 과정을 이용하여 얻어진 CRC-CCITT 오류검출부호와 원시다항식 CRC 오류검출부호의 중분포는 〈부록〉의 〈표1〉 〈표2〉와 같으며, 단축 정도에 따른 미검출오류확률 및 $2^{-(n-k)}$ 상한치와의

관계 〈그림 6〉 〈그림 7〉과 같으며, 길이가 126, 94, 62, 32일 경우의 두 오류검출부호의 성능 비교는 각각 〈그림 8〉, 〈그림 9〉, 〈그림 10〉, 〈그림 11〉과 같다. 위의 그림에서 알 수 있듯이, CRC-CCITT 의 미검출오류확률은 부호장이 32-120 바이트 범위내에서 원시다항식 CRC 부호보다 미검출오류확률 측면에서 우수하고, 부호장이 32 바이트 이하일 경우 원시다항식 CRC 부호가 성능이 CRC-CCITT 오류검출부호보다 우수하다. 그리고 〈그림 6〉에서 알 수 있듯이 CRC-CCITT 부호의 미검출오류확률은 부호장이 8바이트 일때 $2^{-(n-k)}$ 상한치를 초과하고 있다. 또한 두 오류검출부호 공히 부호장이 6 바이트 이하일 경우 $2^{-(n-k)}$ 상한치를 초과함을 알 수 있었다.

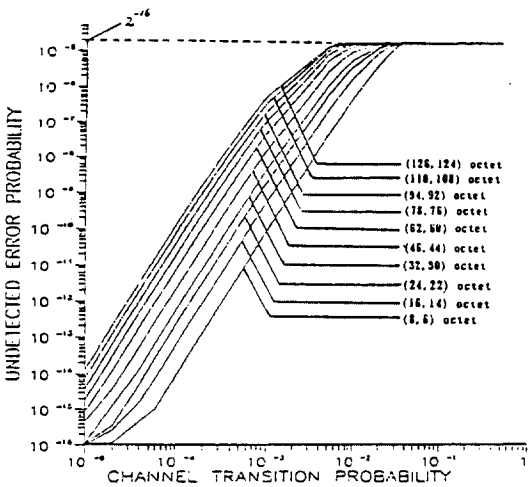


그림 6. CRC-CCITT 오류검출부호의 미검출오류확률
undetected error probability of CRC-CCITT

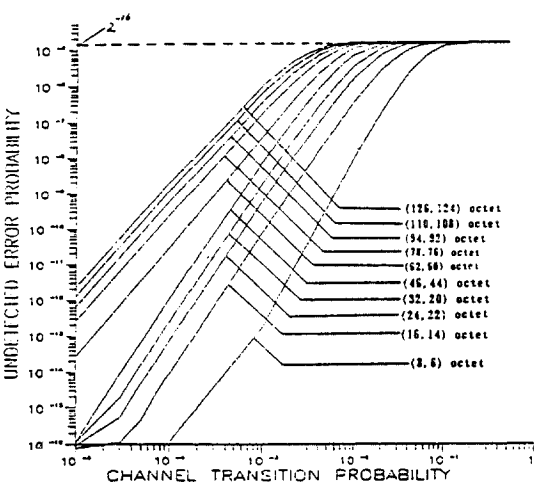


그림 7. 원시다항식 CRC 오류검출부호의 미검출오류확률
undetected error probaility of primitive CRC

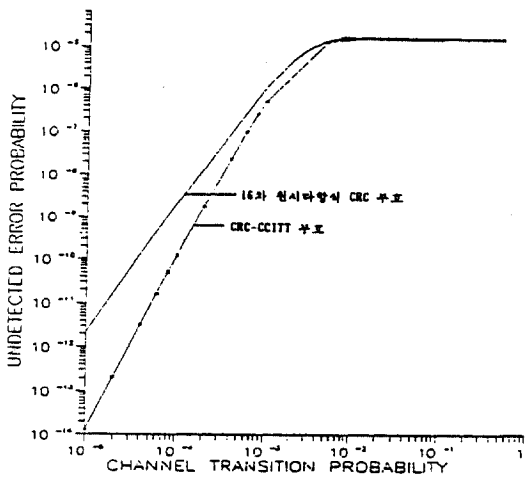


그림 8. 부호장이 126 일때의 두 부호의 성능 비교
comparison of the undetected error probability, n=126

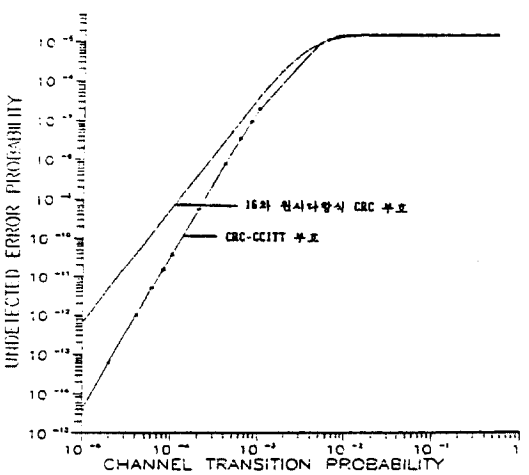


그림 9. 부호장이 94 일때의 두 부호의 성능 비교
comparison of the undetected error probability, n=94

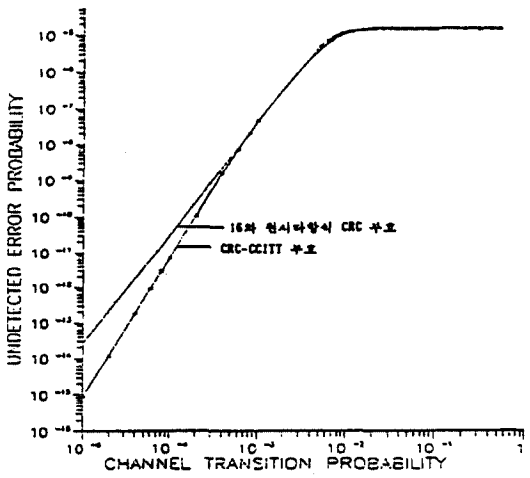


그림 10. 부호장이 62 일때의 두 부호의 성능 비교
comparison of the undetected error probability, n=62

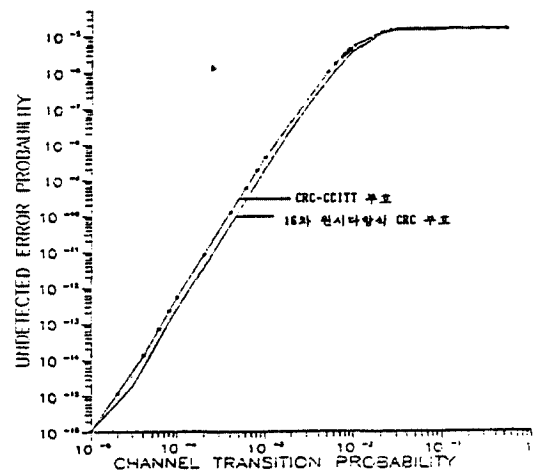


그림 11. 부호장이 32 일때의 두 부호의 성능 비교
comparison of the undetected error probability, n=32

Ⅶ. 결 론

CCITT에서는 광대역 ISDN의 UNI의 전송방식으로 ATM 전송방식을 권고하였으며, cell에 발생하는 오류제어 기법에 대한 연구가 활발히 논의/토의되고 있다.[4]본 논문에서는 cell에서 발생하는 오류를 검출하기 위하여 CRC-CCITT 부호와 GF(2) 상의 16차의 원시다항식 CRC 오류검출부호를 도입하였다. 그리고 Hamming 부호와 단축 Hamming 오류검출부호의 성능을 분석하기 위하여 미검출오류확률에 대한 일반적 이론을 분석/제시하였고, (n,k) 선형 Hamming 오류검출부호와 단축 소거 Hamming 오류검출부호의 중분포를 구하기 위한 새로운 기법을 제시한후, 이 결과를 이용하여 CCITT-CRC와 원시다항식 CRC 오류검출부호를 ATM cell에 적용하였다고 가정하고 이에 대한 미검출오류확률을 구했다. 특히 cell 크기가 현재 CCITT에서 확실히 권고되지 않았음을 고려하여 단축 정도를 변화하면서 두 CRC 오류검출부호의 성능을 분석/비교하였다.

분석/비교 결과, CRC-CCITT 오류검출부호는 원시다항식 CRC 오류검출부호에 비교하면 단축정도가 크지않을 경우 미검출오류확률 측면에서 성능이 더 우수하나, 단축정도가 매우 클 경우 원시다항식 CRC 오류검출부호가 성능이 더 우수함을 알 수 있었다. 그리고 두 오류검출부호 공히 단축 정도가 클경우 $2^{-(n-k)}$ 상한을 만족하지 않았다.

參 考 文 獻

1. 이만영, 부호 이론, 희중당, 1984.
2. S.K.L.Y. Cheong, E.R. Barnes, and D.U. Friedman, " Some properties of Undetected Error Probability of Linear Code," IEEE Trans. on Inform. Theory, vol.IT-25, pp.110-112, Jan. 1979.
3. W.Stallings, Data and Computer Communications, MacMillan Publishing Co, New York, 1985.
4. CCITT, "Part C of the Seoul Meeting", 25.Jan.-5.Feb. 1988.

5. S.K. L.Y. Cheong and M.E. Hellman, "Concerning a Bound on Undetedcted Error Probability," IEEE Trans. on Inform. Theory, vol.IT-22, pp.235-237, Mar. 1976.
6. F.J. MacWilliams and N.J.A. Sloane, Theory of Error--Correcting Codes, North-Holland, Amsterdam. The Netherlands, 1977.
7. S.Lin and D.J. Costello, Error Control Coding: Fundamentals and Applications, Prentice-Hall, Englwood Cliffs, N.J, 1983.
8. E.R. Berlekamp, Algebraic Coding Theory, McGraw-Hill Book Co., New York, 1968.
9. J.K. Wolf, and A.H. Levesque, "On the Probability of Undetected Error for Linear Block Codes", IEEE Trans. on Commun. Theory, vol.COM-30, pp.317-324, Feb. 1982.
10. S.C. chang and Jack K. Wolf, "A Simple Derivation of the MacWilliams' Identity for Linear Codes," IEEE Trans. on Inform. Theory, vol.IT-26, pp.476-477, July. 1980.
11. T.Fujiwara, T.Kasami and S. Lin, "Error Detecting Capabilities of the Shortened Hamming Code Used for the Ethernet." 일본 전자통신학회 논문지, vol.J69-A, No.6, June. 1986.
12. T. Fujiwara, T.Kasami, A. Kitai, and S. Lin, "On the Undetected Error Probability for Shortened Hamming Codes", IEEE Trans, on Commun., vol.COM-33, No. 6, June. 1985.
13. T.Kasami, T. Klove and S.Lin, "Linear Block Codes for Error Detection" IEEE Trans. on Inform. Theory, vol IT-29, pp.131-136, Jan. 1983.

부록 CRC-CCITT 와 원시다항식 CRC 오류검출부호의 중 분포

표 1. CRC-CCITT 오류검출부호의 쌍대부호의 중분포
weight distribution of CRC-CCITT's dual code

(126, 124) octet				(110, 108) octet				(94, 92) octet			
i	$B_{a.l.e}$	i	$B_{a.l.e}$	i	$B_{a.l.e}$	i	$B_{a.l.e}$	i	$B_{a.l.e}$	i	$B_{a.l.e}$
0	1	504	1939	0	1	440	1185	0	1	376	1653
452	2	505	1903	394	1	450	1312	331	2	387	1332
453	4	510	1822	395	2	451	1166	332	5	388	1257
454	6	511	1684	396	16	452	1147	333	6	389	1237
455	17	512	1458	397	47	453	1044	334	12	390	1193
456	20	513	1276	398	90	454	1018	335	19	391	1036
457	34	514	1297	399	145	455	975	336	47	392	1006
458	65	516	1188	400	125	456	922	337	54	393	951
459	55	516	1025	401	82	457	802	338	67	394	858
460	83	517	895	402	47	458	799	339	85	395	825
461	77	518	807	403	56	459	696	340	104	396	679
462	98	519	689	404	54	460	554	341	103	397	654
463	105	520	631	405	72	461	528	342	75	398	548
464	93	521	718	406	103	462	507	343	94	399	496
465	66	522	624	407	146	463	465	344	99	400	402
466	54	523	567	408	144	464	519	345	116	401	322
467	90	524	552	409	164	465	458	346	104	402	239
468	78	525	576	410	196	466	464	347	173	403	164
469	101	526	496	411	229	467	446	348	158	404	168
470	152	527	518	412	253	468	533	349	164	405	173
471	201	528	539	413	446	469	228	350	239	406	104
472	258	529	563	414	464	470	196	351	322	407	116
473	297	530	495	415	458	471	164	352	402	408	59
474	302	531	382	416	519	472	144	353	495	409	34
475	258	532	381	417	483	473	146	354	548	410	15
476	301	533	258	418	587	474	103	355	651	411	103
477	302	534	302	419	528	475	72	356	679	412	104
478	495	535	237	420	554	476	54	357	825	413	85
479	563	536	258	421	638	477	56	358	859	414	67
480	535	537	201	422	739	478	47	359	951	415	54
481	510	538	152	423	802	479	62	360	1006	416	47
482	496	539	104	424	922	480	125	361	1036	417	19
483	576	540	78	425	975	481	145	362	1193	418	12
484	552	541	90	426	1018	482	90	363	1237	419	6
485	567	542	54	427	1044	483	47	364	1257	420	5
486	624	543	66	428	1147	484	16	365	1332	421	2
487	718	544	93	429	1166	485	2	366	1459	252	1
488	631	545	105	430	1312	486	1	367	1421		
489	689	546	98	431	1185	888	1	368	1489		
490	807	547	77	432	1413			369	1665		
491	895	548	83	433	1655			370	1636		
492	1025	549	55	434	1674			371	1764		
493	1188	550	65	435	1911			372	2071		
494	1297	551	94	436	1830			373	1922		
495	1276	552	28	437	1954			374	1967		
496	1458	553	17	438	1981			375	1947		
497	1684	554	6	439	2097			376	1856		
498	1822	555	4	440	2168			377	1947		
499	1903	556	2	441	2097			378	1967		
500	1939	1008	1	442	1921			379	1922		
501	1912			443	1954			380	2071		
502	1810			444	1830			381	1764		
503	1575			445	1911			382	1635		
504	2021			446	1671			383	1645		
505	1575			447	1655			384	1443		
506	1810			448	1413			385	1421		
507	1912										

(78, 76) octet				(62, 60) octet				(46, 44) octet				(32, 30) octet			
i	$B_{a.l.e}$	i	$B_{a.l.e}$	i	$B_{a.l.e}$	i	$B_{a.l.e}$	i	$B_{a.l.e}$	i	$B_{a.l.e}$	i	$B_{a.l.e}$	i	$B_{a.l.e}$
0	1	325	1179	0	1	260	553	0	1	208	110	0	1	154	40
270	2	326	1224	213	5	269	256	155	5	209	75	99	12	155	18
271	5	327	1106	214	3	270	202	154	13	210	55	100	10	156	16
272	6	328	1098	215	20	271	176	155	27	211	66	101	10	157	12
273	15	329	1000	216	28	272	180	156	52	212	52	102	40	256	1
274	21	330	881	217	64	273	170	157	66	213	27	103	48		
275	25	331	747	218	77	274	180	158	55	214	13	104	56		
276	39	332	652	219	98	275	147	159	75	215	5	105	48		
277	34	333	624	220	134	276	134	160	110	388	1	106	85		
278	51	334	461	221	143	277	98	161	99			107	102		
279	71	335	329	222	186	278	77	162	155			108	92		
280	97	336	163	223	178	279	64	163	216			109	135		
281	135	337	171	224	180	280	28	164	250			110	217		
282	153	338	126	225	175	281	20	165	329			111	245		
283	163	339	93	226	222	282	3	166	381			112	352		
284	150	340	150	227	256	283	5	167	458			113	482		
285	93	341	163	228	353	486	1	168	649			114	663		
286	126	342	153	229	446			169	875			115	867		
287	171	343	135	230	655			170	949			116	1020		
288	165	344	97	231	763			171	1338			117	1348		
289	209	345	71	232	988			172	1430			118	1595		
290	461	346	51	233	1273			173	1660			119	1966		
291	524	347	34	234	1325			174	1964			120	2345		
292	652	348	39	235	1325			175	1896			121	2561		
293	747	349	25	236	1432			176	2059			122	2710		
294	881	350	21	237	1562			177	2214			123	2705		
295	1000	351	15	238	1664			178	2296			124	2630		
296	1038	352	6	239	1808			179	2348			125	2708		
297	1106	353	5	240	1999			180	2254			126	2930		
298	1224	354	2	241	2118			181	2384			127	3143		
299	1179	624	1	242	2042			182	2507			128	3252		
300	1319			243	1983			183	2364			129	3143		
301	1481			244	2064			184	2238			130	2936		
302	1457			245	1972			185	2364			131	2705		
303	1607			246	2044			186	2507			132	2634		
304	1683			247	2028			187	2384			133	2705		
305	1755			248	2024			188	2294			134	2716		
306	1715			249	2068			189	2340			135	2564		
307	1867			250	2044			190	2236			136	2545		
308	1900			251	1972			191	2214			137	1960		
309	1583			252	2064			192	2089			138	1595		
310	2037			253	1983			193	1896			139	1341		
311	2170			254	2042			194	1964			140	1024		
312	2234			255	2118			195	1660			141	867		
313	2170			256	1939			196	1498			142	663		
314	2037			257	1868			197	1338			143	482		
315	1983			258	1654			198	949			144	352		
316	1900			259	1552			199	875			145	255		
317	1867			260	1433			200	649			146	217		
318	1715			261	1325			201	499			147	155		
319	1759			262	1325			202	381			148	95		
320	1683			263	1273			203	323			149	102		
321	1607			264	988			204	253			150	45		
322	1457			265	763			205	214			151	48		
323	1341			266	655			206	155			152	50		
324	1319			267	446			207	99			153	48		

표 2. 원시다항식 CRC 오류검출부호의 쌍대부호의 중분포
weight distribution of primitive CRC's dual code

(126,124) octet

i	B_{n-i}	i	B_{n-i}
0	1	512	1613
158	5	513	1546
459	10	514	1271
460	12	515	1177
461	26	516	1185
462	28	517	1188
463	8	518	1176
464	13	519	1088
465	37	520	1011
466	71	521	893
467	99	522	821
468	115	523	783
469	154	524	695
470	156	525	663
471	144	526	532
472	213	527	505
473	322	528	408
474	409	529	403
475	414	530	416
476	540	531	354
477	602	532	381
478	730	533	307
479	662	534	291
480	673	535	253
481	682	536	281
482	668	537	223
483	664	538	181
484	685	539	145
485	758	540	122
486	802	541	71
487	839	542	49
488	764	543	68
489	786	544	53
490	741	545	57
491	871	546	52
492	916	547	63
493	922	548	42
494	1028	549	46
495	1251	550	62
496	1496	551	27
497	1594	552	21
498	1759	553	19
499	1949	554	10
500	1936	555	6
501	1967	556	18
502	1945	557	14
503	1893	558	11
504	1830	559	8
505	1691	560	5
506	1547	561	7
507	1424	562	2
508	1435	1008	1
509	1546		
510	1504		
511	1528		

(110,108) octet

i	B_{n-i}	i	B_{n-i}
0	1	456	944
402	8	457	866
403	24	458	871
404	51	459	817
405	84	460	660
406	129	461	517
407	181	462	488
408	208	463	422
409	208	464	352
410	262	465	288
411	343	466	254
412	361	467	231
413	396	468	219
414	446	469	208
415	628	470	281
416	704	471	234
417	750	472	192
418	848	473	155
419	817	474	128
420	855	475	81
421	818	476	78
422	852	477	94
423	818	478	58
424	786	479	57
425	901	480	29
426	866	481	35
427	923	482	28
428	923	483	45
429	984	484	67
430	1038	485	62
431	1230	486	31
432	1458	487	26
433	1643	488	30
434	1727	489	23
435	1797	490	21
436	1882	491	12
437	1888	492	11
438	1967	493	3
439	1938	880	1
440	1863		
441	1907		
442	1781		
443	1698		
444	1597		
445	1602		
446	1534		
447	1745		
448	1792		
449	1686		
450	1555		
451	1508		
452	1372		
453	1142		
454	972		
455	871		

(94,92) octet

i	B_{n-i}	i	B_{n-i}
0	1	391	948
337	7	392	756
338	2	393	731
339	10	394	636
340	21	395	587
341	44	396	572
342	47	397	444
343	65	398	389
344	105	399	338
345	93	400	389
346	173	401	343
347	267	402	313
348	334	403	288
349	314	404	271
350	344	405	239
351	395	406	159
352	490	407	129
353	510	408	126
354	577	409	148
355	678	410	168
356	782	411	122
357	865	412	88
358	908	413	78
359	1038	414	56
360	1056	415	41
361	1172	416	31
362	1229	417	53
363	1141	418	44
364	1138	419	24
365	1138	420	5
366	1260	421	8
367	1268	422	3
368	1255	423	2
369	1450	352	1
370	1755		
371	2000		
372	1845		
373	1867		
374	1978		
375	2080		
376	2005		
377	1940		
378	1948		
379	1981		
380	1880		
381	1916		
382	1879		
383	1747		
384	1664		
385	1624		
386	1547		
387	1522		
388	1358		
389	1233		
390	1065		

(78,76) octet

i	B_{n-i}	i	B_{n-i}
0	1	338	734
276	5	339	603
277	11	332	456
278	36	333	378
279	50	334	349
280	62	335	301
281	94	336	264
282	129	337	264
283	117	338	253
284	141	339	264
285	155	340	298
286	185	341	240
287	267	342	147
288	376	343	87
289	457	344	90
290	474	345	122
291	554	346	116
292	578	347	75
293	636	348	50
294	706	349	35
295	674	350	24
296	1002	351	17
297	1164	352	10
298	1210	353	9
299	1215	354	1
300	1280	624	1
301	1498		
302	1518		
303	1575		
304	1614		
305	1662		
306	1873		
307	2010		
308	2111		
309	2177		
310	2265		
311	2231		
312	2152		
313	2145		
314	2005		
315	1795		
316	1647		
317	1508		
318	1940		
319	1874		
320	1705		
321	1544		
322	1337		
323	1166		
324	1164		
325	1274		
326	1250		
327	1048		
328	937		
329	871		

(52,50) octet

i	B_{n-i}	i	B_{n-i}
0	1	264	829
216	2	265	660
211	3	266	572
212	10	267	519
213	11	268	418
214	23	269	332
215	30	270	303
216	35	271	343
217	28	272	255
218	35	273	198
219	45	274	193
220	56	275	174
221	51	276	167
222	87	277	129
223	123	278	98
224	181	279	51
225	223	280	41
226	268	281	32
227	311	282	22
228	448	283	11
229	645	284	25
230	578	285	20
231	659	286	6
232	875	496	1
233	1032		
234	1294		
235	1441		
236	1530		
237	1531		
238	1634		
239	1728		
240	1886		
241	2208		
242	2137		
243	2146		
244	2145		
245	2234		
246	2224		
247	2340		
248	2218		
249	2310		
250	2221		
251	2327		
252	2375		
253	2264		
254	1934		
255	1788		
256	1595		
257	1435		
258	1264		
259	1214		
260	1210		
261	1235		
262	1337		
263	1185		

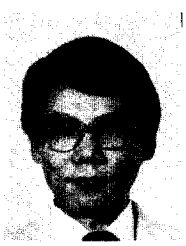
(46,44) octet

i	B_{n-i}	i	B_{n-i}
0	1	204	357
150	9	205	260
151	12	206	247
152	27	207	146
153	12	208	86
154	15	209	81
155	10	210	72
156	11	211	51
157	11	212	35
158	18	213	22
159	51	214	16
160	60	215	17
161	98	216	16
162	186	217	5
163	234	218	4
164	291	219	10
165	401	220	8
166	439	221	3
167	596	568	1
168	762		
169	948		
170	1069		
171	1187		
172	1570		
173	1561		
174	1673		
175	1642		
176	1884		
177	2010		
178	2204		
179	2588		
180	2824		
181	3088		
182	3058		
183	2791		
184	2641		
185	2525		
186	2477		
187	2371		
188	2344		
189	2169		
190	2021		
191	2083		
192	1895		
193	1687		
194	1460		
195	1291		
196	1184		
197	1191		
198	999		
199	782		
200	649		
201	674		
202	579		
203	404		

(32,30) octet

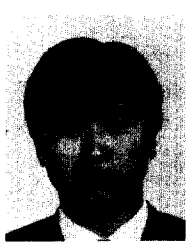
i	B_{n-i}
0	1
101	1
102	5
103	14
104	29
105	37
106	45
107	88
108	131
109	214
110	338
111	432
112	536
113	591
114	611
115	778
116	976
117	1185
118	1413
119	1648
120	1926
121	2297
122	2687
123	2864
124	3055
125	3314
126	3391
127	3229
128	3064
129	3131
130	3206
131	3035
132	2857
133	2482
134	2258
135	2239
136	1985
137	1673
138	1452
139	1270
140	1068
141	893
142	757
143	645
144	515
145	414
146	263
147	227
148	145
149	86
150	62
151	52
152	59
153	37
154	16

i	B_{n-i}
155	10
156	4
157	1
256	1

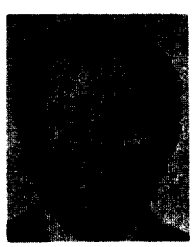


廉 興 烈(Heung Youl YOUM) 正會員
1959年 2 月10日生
1981年：漢陽大學校 電子工學科 卒業
1983年：漢陽大學校 大學院 電子工學科 工學碩士
1983年～現在：漢陽大學校 大學院 電子工學科 博士課程中
1982年12月～現在：韓國電子通信研究所 전송시스템研究室 先任研究員

權 周 漢(Joo Han KWON) 正會員
1964年10月 1 日生
1987年 2 月：漢陽大學校 電子通信科 卒業
1989年 2 月：漢陽大學校 大學院 電子通信科 卒業, 에러정정부호 專攻
現在： 三星電子綜合研究所 研究 2 担當 勤務



梁 承 杜(Seung Doo YANG) 準會員
1965年 1 月13日生
1983年 3 月～1987年 2 月：漢陽大學校 工科大學 電子通信工學科(學士)
1987年 3 月～1989年 2 月：漢陽大學校大學院 電子通信工學科(碩士)
1989年～現在：三星電子(株) 미니컴퓨터 開發室 研究員



李 晩 榮(Man Young RHEE) 正會員
1924年11月30日生
서울大學校電氣工學科 卒業
美國Colorado 大學院卒業(工學博士 1958年)
美國Boeing 會社研究員
美國Virginia 工大教授
美國California 工大 JPL NASA研究員
國防科學研究所副所長
韓國電子通信(株)代表理事社長
漢陽大學校工科大學電子通信工學科 教授