지상파 DMB용 Outer 인코더/디코더의 설계 및 구현

원 지 연[†] · 이 재 흥[††] · 김 건[†††]

요 약

본 논문에서는 차세대 디지털 방송규격인 지상파 DMB용 Outer 인코더/디코더를 설계 하고 ALTERA의 FPGA를 이용하여 구현하고 검증하였다. 인코더 부분에서는 입력되는 MPEG-2 TS 패킷(188바이트)으로부터 비트 시리얼 알고리즘을 이용한 RS(Reed-Solomon) 인코더를 이용해 패리티바이트(16바이트)를 생성하고 군집에러를 효과적으로 수정하기위해 콘볼루션 인터리버를 구현해 데이터를 분산 출력 시켰다. 디코더 부분에서는 인코더에서 송신된 데이터에서 DMB에 적합한 동기바이트 검출하는 알고리즘을 제시하였으며, RS디코더는 수정된 유클리드 알고리즘을 적용하여 회로구성을 간략화 하였다. 본 시스템은 하나의 패킷에서 최대 8바이트의 에러를 수정할 수 있고, C언어를 이용하여 알고리즘을 검증하고 VHDL로 작성하였으며, FPGA 칩 상에서 회로를 검증하였다.

The Design and Implementation of Outer Encoder/Decoder for Terrestrial DMB

Ji-Yeon Won[†] · Jae-Heung Lee[††] · Geon Kim[†††]

ABSTRACT

In this paper, we designed the outer encoder/decoder for the terrestrial DMB that is an advanced digital broadcasting standard, implemented, and verified by using ALTERA FPGA. In the encoder part, it was created the parity bytes (16 bytes) from the input packet (188byte) of MPEG-2 TS and the encoded data was distributed output by the convolutional interleaver for preventing burst errors. In the decoder part, it was proposed the algorithm that detects synchronous character suitable to DMB in transmitted data from the encoder. The circuit complexity in RS decoder was reduced by applying a modified Euclid's algorithm. This system has a capability to correct error of the maximum 8 bytes in a packet. After the outer encoder/decoder algorithm was verified by using C language, described in VHDL and implemented in the ALTERA FPGA chips.

키워드 : DMB, 인터리버(Interleaver), 디인터리버(Deinterleaver), 동기검출기(Sync Detector), RS 인코더/디코더(RS Encoder/Decoder)

1. 서 론

DMB(Digital Multimedia Broadcasting) 기술은 방송국의 증가로 인한 주파수 자원의 고갈과 아날로그 FM 방송의 혼선 및 이동시 음질저하 문제의 해결책으로 처음 생겨나게 되었다. DMB란 기존의 라디오방송에서 음성뿐만 아니라 문자·CD 수준의 음질·양방향성·다양한 데이터 서비스등 멀티미디어 데이터를 서비스하는 차세대 라디오 방송을 말하는 것으로, DMB는 이동수신이 뛰어나기 때문에 소형TV·PDA등 휴대용 단말기를 통한 전달이 용이 하다. 지상파 DMB는 TV채널 1개를 통해 3개의 동영상과 10개 정도의 오디오·데이터 방송을 구현 할 수 있다. 국내에서 DMB 방송의 명칭은 당초 DAB(Digital Audio Broadcasting)라 부르기도 했으나 오디오 이외에 비디오나 문자 데이터등을 포함한다는 ITU의 규정에 의거 DMB로 개칭해서 사용하게 되었다. 국내 지상파 DMB는 1997년 지상파 디지탈 방송 추진협의회를 시작으로 도입되었으며 2001년 디지털 라디오 방송추진 위원회를 거치면서 유럽형 Eureka-147을 잠정 표준으로 결정하였으며 2003년 후반기에 시험방송을 시작할 예정에 있다[8].

기존의 실시되고 있는 디지털방송(DTV) 방식의 경우 고정형 수신기를 장착하지만 지상파 DMB의 경우 이동형단말기 수신을 목적으로 하기 때문에 채널 상에서 발생 할 수 있는 오류의 양이 더욱 클 것으로 생각된다.

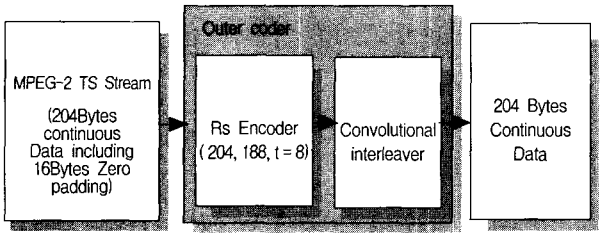
본 논문에서는 지상파 DMB의 채널 상에서 발생하는 에러를 수정 할 수 있도록 Outer 인코더/디코더를 설계함으로써 지상파 DMB 서비스의 질을 보다 향상 될 수 있도록 하는데 그 목적을 두었다.

† 준 회 원 : 한밭대학교 정보통신전문대학원 컴퓨터공학과
†† 정 회 원 : 한밭대학교 정보통신·컴퓨터공학부 교수
††† 정 회 원 : 한국전자통신연구원 방송시스템연구부 연구원
논문접수 : 2003년 10월 17일, 심사완료 : 2004년 1월 7일

RS(204, 188, t = 8) 인코더 / 디코더 규격에 군집에러를 분산 시킬 수 있도록 콘볼루션 인터리버 / 디인터리버를 추가하였으며, 디코더부에 DMB방송 서비스에 적합한 동기신호 검출기를 포함하고 있다. 그리고 Altera사의 Stratix 디바이스를 이용하여 구현하였으며 PCI를 이용한 데이터 송수신으로 로직을 검증하였다.

2. DMB용 Outer 인코더

DMB용 Outer 인코더는 전송할 데이터를 인코딩하는 부분으로 RS 인코더와 콘볼루션 인터리버로 구성된다.



(그림 1) DMB용 Outer 인코더

2.1 RS(Reed-Solomon) 인코더

RS 인코더는 q개의 원소를 갖는 $GF(q)$상에서 $q=2^m$인 경우만을 고려한다. GF상의 각 원소는 m 비트로 이루어지며, (n, k) RS부호의 오류정정 능력이 t 라면, 각 파라미터는 다음과 같이 정의된다[1, 7].

부호장 : $n = 2^m - 1$(심볼) = $m(2^m-1)$ (비트)
정보장 : $k = n-2t$(심볼) = $m(n-2t)$ (비트)
검사장 : $n - k = 2t$(심볼) = $2tm$ (비트)
최소거리 : $d = n - k + 1$

부호장은 정보장과 검사장으로 구성되며, 검사장은 RS(n, k, t) 코드 생성다항식과 정보장과의 곱으로 구할 수 있다. 그러나 두 수의 곱으로 만들어진 검사장의 값은 너무 커지게 된다. 때문에 주어진 필드 생성다항식으로 생성된 GF상의 값을 이용하여 나눗셈 연산을 하면 $2^t m$ 비트 크기의 검사장을 구할 수 있다. 생성 다항식 $g(x)$는 일반적으로 식 (2.1)과 같이 표현한다.

$$g(x) = \prod_{i=0}^{2t-1}(x+\alpha^i) \tag{2.1}$$

α는 $GF(2^m)$ (Galois Field)상의 원시원(Primitive element)이다. (n, k, t) RS 부호의 정보다항식을 $d(x) = d_0 + d_1 x + \cdots + d_{k-1}x^{k-1}$이라하고 검사다항식을 $p(x) = p_0 + p_1 x + \cdots + p_{n-k-1}x^{n-k-1}$이라하면 부호화 된 RS 부호의 다항식은 정보 다항식과 검사 다항식의 합으로 표현한다.

$$c(x) = d(x) + p(x)$$
$$c_i \in GF(2^m) \tag{2.2}$$

검사다항식 $p(x)$는 $d(x)$와 $g(x)$의 곱이며, 순회 부호의 일반적인 부호화 방법은 심볼 당 주어진 필드생성다항식 f(x)의 나눗셈으로 간단히 할 수 있다.

$$p(x) = f(x) \times q(x) + r(x) = r(x) \tag{2.3}$$

위식을 식 (2.2)에 대입하면

$$c(x) = d(x) + r(x) \tag{2.4}$$

가 된다.

따라서 정보 다항식 $d(x)$에 생성 다항식 $g(x)$를 곱하여 검사 다항식 $p(x)$를 구하고 $p(x)$의 나머지 $r(x)$를 구하여 $d(x)$와 합하면 부호다항식 $c(x)$를 구하는 RS 부호화가 이루어진다.

본 논문에서는 DVB-T표준[EN 300 744]을 따르며RS(255, 239, t = 8)으로부터 유도된 간략화 된 RS(204, 188, t = 8)를 구현하였다.

코드생성다항식은

$$g(x) = (x+\lambda^0)(x+\lambda^1)(x+\lambda^2)\cdots(x+\lambda^{15}),\ ^{\lambda=02} \tag{2.5}$$

필드 생성 다항식은

$$p(x) = x^8 + x^4 + x^3 + x^2 + 1 \tag{2.6}$$

일 때,

생성 다항식은 아래와 같이 구할 수 있다.

$$\begin{aligned} G_{16}(x) &= (x-\alpha^0)(x-\alpha^1)(x-\alpha^2)\cdots\cdots(x-\alpha^{15}) \\ &= \prod_{i=0}^{2t-1}(x+\alpha^i) = g_0 + g_1 x^1 + g_2 x^2 + \cdots + g_{15}x^{15} + x^{16} \\ &= \alpha^{120} + \alpha^{225}x^1 + \alpha^{194}x^2 + \alpha^{182}x^3 + \alpha^{169}x^4 + \alpha^{147}x^5 \\ &+ \alpha^{191}x^6 + \alpha^{91}x^7 + \alpha^{3}x^8 + \alpha^{76}x^9 + \alpha^{161}x^{10} + \alpha^{102}x^{11} \\ &+ \alpha^{109}x^{12} + \alpha^{107}x^{13} + \alpha^{104}x^{14} + \alpha^{120}x^{15} + x^{16} \end{aligned}$$

생성다항식 GP의 계수 g_n과 정보다항식 계수 m_k를 서로 곱하여 정리를 하면 다음과 같다.

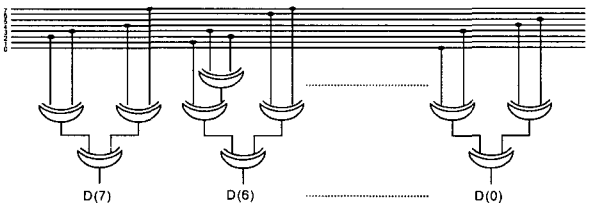
$$m_k = a_7\alpha^7 + a_6\alpha^6 + a_5\alpha^5 + a_4\alpha^4 + a_3\alpha^3 + a_2\alpha^2 + a_1\alpha^1 + a_0\alpha^0$$
$$\begin{aligned} m_k \times \alpha^{120} = &(a_7\alpha^7 + a_6\alpha^6 + a_5\alpha^5 + a_4\alpha^4 + a_3\alpha^3 + a_2\alpha^2 \\ &+ a_1\alpha^1 + a_0\alpha^0) \times (\alpha^5 + \alpha^4 + \alpha^3 + \alpha^2 + 1) \end{aligned}$$

같은 방법으로 정보다항식을 $g_0, g_1, g_2, g_3, \cdots, g_{15}$와의 곱으로 표현하면 <표 1>과 같다.

<표 1>을 각각의 입력에 대해 회로로 표현하게 되면 아래 (그림 2)와 같이 구성 된다. (그림 2)는 위의 표에서 α^{120} 일 때의 회로도를 나타낸 것이다.

〈표 1〉 GF(256)에 대한 근의 곱셈 결과

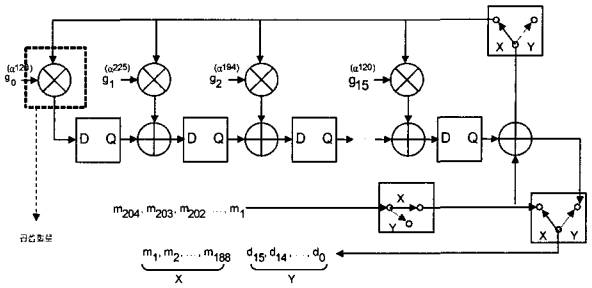
차수 / G(x) 계수	x^7	x^6	x^5	x^4	x^3	x^2	x^1	x^0
α^{120}	156 10011100	206 11001110	103 01100111	51 00110011	133 10000101	222 1101110	115 1110011	57 111001
α^{225}	228 11100100	242 11110010	249 11111001	252 11111100	26 11010	233 11101001	144 10010000	200 11001000
α^{194}	12 1100	6 0110	131 10000011	193 11000001	236 11101100	122 1111010	49 110001	24 11000
α^{182}	102 1100110	179 10110011	89 1011001	172 10101100	48 110000	254 11111110	153 10011001	204 11001100
α^{169}	247 11110111	251 11111011	253 11111101	126 1111110	72 1001000	83 101111	222 11011110	239 11101111
α^{147}	84 1010100	42 101010	149 10010101	202 11001010	177 11001010	12 1100	82 1010010	169 10101001
α^{191}	98 1100010	49 110001	24 11000	12 1100	100 1100100	208 11010000	138 10001010	197 11000101
α^{91}	117 1110101	186 10111010	93 1011101	46 101110	98 1100010	68 1000100	215 11010111	235 11101011
α^{3}	16 10000	136 10001000	196 11000100	226 11100010	97 1100001	160 10100000	64 1000000	32 100000
α^{76}	248 11111000	124 1111100	190 10111110	223 11011111	151 10010111	51 110011	225 11100001	240 11110000
α^{161}	155 10011011	205 11001101	230 11100110	243 11110011	226 11100010	234 11101010	110 1101110	55 110111
α^{102}	194 11000010	97 1100001	176 10110000	216 11011000	174 10101110	149 10010101	8 1000	132 10000100
α^{109}	141 10000011	198 10111100	227 11100011	241 11110001	245 11110101	119 1110111	54 110110	27 11011
α^{107}	54 00101100	27 00010001	141 10001101	198 11000110	213 11010101	220 11011100	216 11011000	108 1101100
α^{104}	176 1011000	216 11011000	108 1101100	54 110110	171 10101011	229 11100101	194 11000010	97 1100001
α^{120}	156 10011100	206 11001110	103 01100111	51 00110011	133 10000101	222 1101110	115 1110011	57 111001



(그림 2) 곱셈회로의 세부 회로도

위와 같은 방식으로 <표 1>의 값들을 나타내고 이를 아래 (그림 3)과 같은 형식으로 나눗셈회로를 구현하여 입력되는 데이터로부터 연속적으로 인코딩하여 결과를 출력 할 수 있도록 구현하였다.

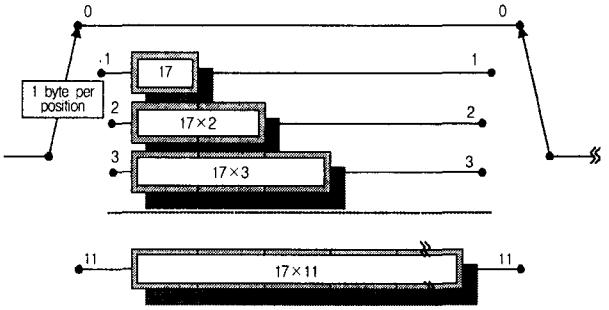
(그림 3)에서처럼 입력된 데이터는 204바이트이고 이중 188바이트는 출력 포트와 연산 블록으로 연결되게 되고 189번째 데이터부터 204번째 데이터는 연산 블록 쪽의 출력 값이 출력 포트로 출력되게 된다. 최종적으로 출력포트로 출력되는 값은 정보에 해당되는 188바이트((그림 3)의 'X')의 정보 데이터와 16바이트((그림 3)의 'Y')의 패리티바이트를 가져가게 된다.



(그림 3) RS 인코더 블록도

2.2 콘볼루션 인터리버(Convolutional Interleaver)

데이터는 전송되는 동안에 여러 가지 장애에 의한 데이터 에러가 발생할 수 있다. 콘볼루션 인터리버는 데이터를 분산, 처리하여 수신 데이터의 복원 율을 향상시키므로 특정 부분에 연속적으로 에러가 생기는 군집 오류(Burst Error)에 강한 면모를 보인다.



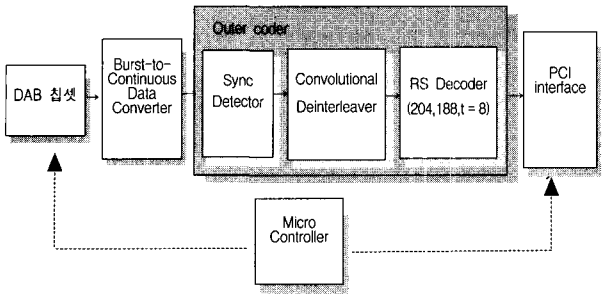
(그림 4) 콘볼루션 인터리버의 구조

(그림 4)와 같은 방식으로 RS부호로 인코딩 된 204바이트 패킷을 인터리빙 한다. 각각의 브랜치는 총 12개이며, 각 브랜치는 17바이트×N(N = 0, 1, 2, ···, 11) 단위로 구현된다. 일반적으로 콘볼루션 인터리버는 FIFO 쉬프트 레지스터로 구현하지만 본 연구에서는 메모리(1135×8비트)를 사용하여 구현하였다.

처음 입력 데이터인 동기 바이트(0×47)는 '0' 브랜치를 통해 지연 없이 출력되고 2번째 데이터는 '1' 브랜치를 통해 17번의 지연 후 출력되며 동일한 방식으로 마지막 '11' 브랜치는 187번의 지연 후에 데이터가 출력 된다. 이러한 방식으로 데이터를 처리하게 되면 동기 바이트는 항상 '0' 브랜치를 통해 출력되기 때문에 중간에 잘못된 클록이나 데이터의 오류를 최소화 할 수 있다.

3. DMB용 Outer 디코더

DMB용 Outer 디코더는 동기 검출, 콘볼루션 디인터리버와 RS 디코더로 구성된다.



(그림 5) Outer 디코더 전체블럭도

3.1 동기 검출

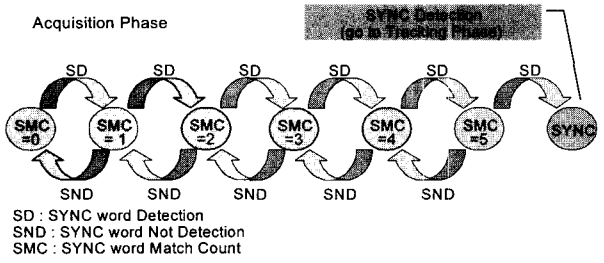
동기 검출 부는 입력되는 MPEG-2 TS에서 204바이트 마다 반복되는 동기바이트를 획득해서 뒷 블록의 연산에서 사용할 프레임 동기신호를 생성하여 연산을 할 수 있도록 한다.

동기 검출 부는 하나의 SMC(Sync word Match Count)를 이용하여 204바이트 단위로 동기 신호가 있는지를 조사하여 동기 획득여부를 판단할 수 있도록 하는 알고리즘을 제시하였다. 제시한 알고리즘은 중간에 동기신호를 연속하여 3번 잃을 때까지는 동기신호가 유지되도록 하였다. 그렇게 때문에 단순한 노이즈로 인해 동기를 한두개 잃었을 때에도 동기신호가 유지될 수 있게 하였다.

동기 검출 부의 구조는 크게 두 개의 부분으로 나누었다. 동기를 찾는 획득부분(Acquisition Phase)과 동기를 유지하는 추적부분(Tracking Phase)이다.

3.1.1 획득 부분

(그림 6)은 동기 획득부분으로 입력되는 데이터(바이트)를 이용해서 동기바이트를 찾는 부분을 나타낸다.



(그림 6) 동기 검출

동기 바이트란 인코더에서 보낸 데이터에서 204바이트의 처음에 동기 바이트인 '0×47' 값을 말한다. (그림 6)에서 SD는 204바이트마다 데이터를 체크해서 동기 바이트인지 판별해서 SMC를 증가시킨다. 이렇게 증가한 SMC는 6개까지 증가하면 동기검출확인 상태로 전환되며 동기 유지부로 이동한다.

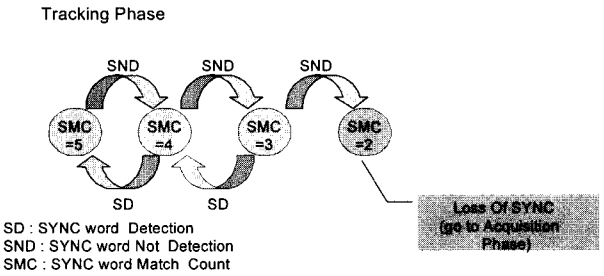
3.1.2 동기 유지

동기 유지 부는 동기 획득부에서 동기를 찾은 것을 계속 유지하고 있는지 판별하는 부분이다.

(그림 7)에서 동기를 연속 3번 잃으면 동기 획득부로 되돌아간다.

즉, 입력되는 동기 바이트값에 따라 SMC값을 증가와 감소시키면서 SMC값이 3보다 작게 되면 동기를 잃어버리게 된다.

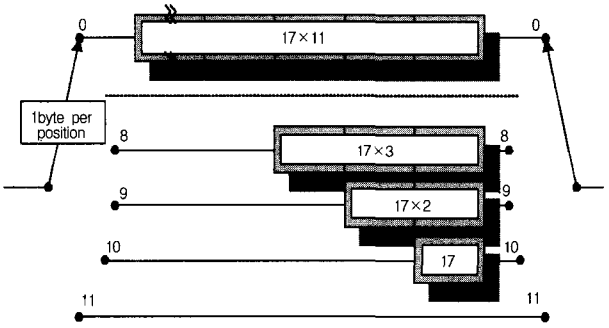
동기 획득부로 이동하게 되면 SMC가 0으로 초기화 되어 다시 동기를 찾게 된다.



(그림 7) 동기 유지 부

3.2 콘볼루션 디인터리버

콘볼루션 디인터리버는 (그림 8)과 같이 인터리버의 역구조 방식으로 되어있어 인터리빙된 입력데이터는 디인터리버를 거쳐서 정상적인 순서의 데이터가 출력된다.



(그림 8) 콘볼루션 디인터리버의 구조

브랜치는 12개로 인터리버와 같지만 각 브랜치별로 지연되는 길이는 인터리버와는 반대로 위아래가 뒤집어져 있는 방식이다. 따라서 각 브랜치는 인터리버와 디인터리버를 합하였을 때 총 지연되는 길이는 187바이트로써 모두 같게 된다.

3.3 RS 디코더(204,188, t = 8)

RS 디코더는 RS 인코딩된 데이터를 이용하여 채널 상에서 발생 할 수 있는 오류를 정정하기 위해 사용한다. 본 논문에서는 유클리드 알고리즘은 덧셈과 곱셈만으로 구성하는 수정된 유클리드 알고리즘을 적용하여 회로의 구성을 간략화 시켰다.

$g(x)$: 생성다항식

$$g(x) = x^{16} + \alpha^{120}x^{15} + \alpha^{104}x^{1} + \alpha^{107}x^{13} + \alpha^{109}x^{12} + \alpha^{102}x^{11} + \alpha^{161}x^{10} + \alpha^{76}x^{9} + \alpha^{3}x^{8} + \alpha^{91}x^{7} + \alpha^{191}x^{6} + \alpha^{147}x^{5} + \alpha^{169}x^{4} + \alpha^{182}x^{3} + \alpha^{194}x^{2} + \alpha^{225}x + \alpha^{120}$$

$r(x)$: RS 디코더에 수신된 데이터 다항식

$$r(x) = r_0 + r_1x + r_2x^2 + \cdots + r_n - 1x^{n-1}$$

$c(x)$: codeword 다항식

$$c(x) = c_0 + c_1x + c_2x^2 + \cdots + c_{n-1}x^{n-1}$$

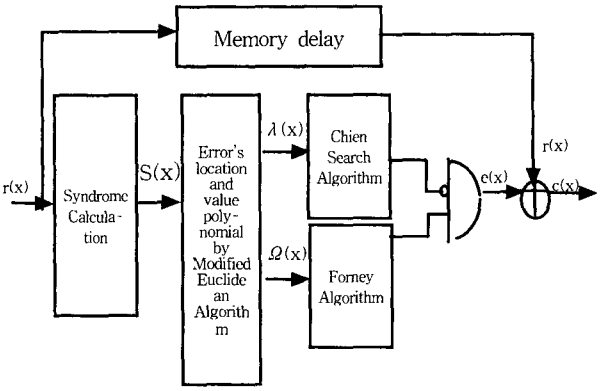
$e(x)$: 에러 정정 다항식

$$e(x) = e_0 + e_1x + e_2x^2 + \cdots + e_{n-1}x^{n-1}$$

일 때,

$$r(x) = c(x) + e(x) \tag{3.1}$$

식 (3.1)에서, RS 디코더 에 수신된 데이터에 에러가 없다면, $r(x)$를 $g(x)$로 나누었을 때 나머지는 '0' 이 된다. 반대로 $r(x)$ 값이 '0'이 아니라면 디코더에 수신된 데이터에는 에러가 있다고 판단할 수 있다.



(그림 9) RS-디코더의 구조

RS 디코더는 에러가 있는 위치와 그 위치에서의 크기를 계산하고, 이를 $r(x)$와의 연산으로 에러가 있는 $r(x)$를 $c(x)$로 정정하는 기능을 갖는다. 디코딩을 위한 몇 가지 알고리즘이 사용되는데, 간단히 블록화하면 (그림 9)와 같다. 그리고 각각의 블록들은 다음과 같다.

3.3.1 신드롬 연산

S_i : 신드롬($r(x)$ 에 $g(x)$의 근을 대입했을 때 $r(x)$의 값)

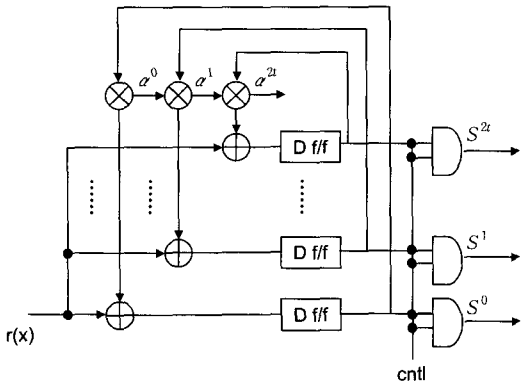
$$S_0 = r_0(\alpha^0)^0 + r_1(\alpha^0)^1 + \cdots + r_{n-1}(\alpha^0)^{n-1}$$
$$S_1 = r_0(\alpha^1)^0 + r_1(\alpha^1)^1 + \cdots + r_{n-1}(\alpha^1)^{n-1}$$
$$\vdots$$
$$S_{2t-1} = r_0(\alpha^{2t-1})^0 + r_1(\alpha^{2t-1})^1 + \cdots + r_{n-1}(\alpha^{2t-1})^{n-1}$$

$S(x)$: 신드롬 다항식(신드롬 계수로 형성된 다항식)

$$S(x) = S_0 + S_1x^1 + S_2x^2 + \cdots + S_{2t-1}x^{2t-1} = \sum_{i=0}^{2t-1}\sum_{j=0}^{n-1} r_jx^i \tag{3.2}$$

Horner's rule에 의해 표현하게 되면 아래와 같고, 회로는 (그림 10)과 같이 표현 할 수 있다.

$$S_1 = r_0(\alpha^1)^0 + r_1(\alpha^1)^1 + r_2(\alpha^1)^2 + \cdots + r_{n-1}(\alpha^1)^{n-1}$$
$$= (((\cdots(r_{n-1}\alpha + r_{n-2})\alpha + r_{n-3})\alpha + r_{n-4})\alpha + \cdots + r_0)$$



(그림 10) 신드롬 연산의 구조

3.3.2 수정된 유클리드 알고리즘

① GCD(a,b) = sa + tb로 정의

② r_{-1} = a, r_0 = b, s_{-1} = 1, s_0 = 0, t_{-1} = 0, t_0 = 1로 초기화

③ $r_{i-1} \neq 0$이면, $r_i = r_{i-2} - q_i r_{i-1}$

$s_{i-1} \neq 0$이면, $s_i = s_{i-2} - q_i s_{i-1}$

$t_{i-1} \neq 0$이면, $t_i = t_{i-2} - q_i t_{i-1}$

$r_i = 0$일 될 때까지 반복

④ $r_i = 0$이면 GCD(a, b) = r_{i-1}이고, $r_{i-1} = s_{i-1}$ a + t_{i-1}b

3.3.3 유클리드 키 방정식

신드롬 연산에서 언급된 식 (3.2)를 r_j가 L_i(에러위치)와 V_i(에러 값)의 값을 갖고 있다고 가정할 때

$$S(x) = \sum_{i=0}^{2t-1}\sum_{j=0}^{n-1} r_jx^i = \sum_{i=0}^{2t-1}\sum_{j=0}^{n-1} V_jL_j^iz^i \tag{3.3}$$

와 같이 표현할 수 있다.

이때 L_i의 역수를 근으로 갖는 에러 위치 다항식 $\lambda(x)$는 다음과 같다.

$$\lambda(x) = \prod_{i=0}^{n-1}(1 - L_ix) \tag{3.4}$$

식 (3.3)과 식 (3.4)을 이용하여 유클리드 키 방정식을 도출하면 다음과 같다.

$$S(x)\cdot\lambda(x) = \sum_{i=0}^{2t-1}\sum_{j=0}^{n-1} V_j L_j^i z^i \cdot \prod_{k=0}^{n-1}(1-L_k x) \\ = \sum_{j=0}^{n-1} V_j L^i \cdot \prod_{k=0,k\neq j}^{n-1}(1-L_k x)(1-L_j^{2t}x^{2t}) \quad (3.5)$$

식 (3.5)에서 x^{2t}를 곱한 부분과 곱하지 않은 부분으로 나누어서 $\Omega(x)$와 $\omega(x)$로 정의하면 다음과 같다.

$$\Omega(x) = \sum_{j=0}^{n-1} V_j L^i \prod_{k=0,k\neq j}^{n-1}(1-L_k x) \\ = S(x)\cdot\lambda(x) \bmod x^{2t}$$

$$\omega(x) = \sum_{j=0}^{n-1} V_j L^i \cdot \prod_{k=0,k\neq j}^{n-1}(1-L_k x)(L_j^{2t})$$

유클리드 키 방정식은 다음과 같다.

$$\therefore\ \Omega(x) = S(x)\cdot\lambda(x) + x^{2t}\omega(x) \quad (3.6)$$

3.3.4 에러위치와 에러 값 다항식

식 (3.6)을 수정된 유클리드 알고리즘에 적용하여 에러 값 다항식 $\Omega(x)$와 에러위치 다항식 $\lambda(x)$를 구한다.

3.3.5 치엔 탐색(Chien Search) 알고리즘

$\lambda(x) = (1-\lambda_0 x)\times(1-\lambda_1 x)\times(1-\lambda_2 x)\times\cdots\times(1-\lambda_{n-1}x)$

이므로 $\lambda(x)$의 근이 '0'이 될 때가 에러의 위치가 된다.

오류위치다항식 $\lambda(x)$는 8비트 레지스터 9개를 이용하여 계산하며 초기 값은 α^0, α^{52}, α^{104}, α^{156}, α^{208}, α^5, α^{57}, α^{109}, α^{161}의 값을 갖게 된다.

3.3.6 포니(Forney) 알고리즘

포니 알고리즘 계산식은

$$e_i = \frac{Li\cdot\Omega(Li^{-1})}{\lambda^{-1}(\alpha^{-1})} \quad (0\le i\le n-1) \quad (3.7)$$

으로 식 (3.7)을 이용하여 $e(x)$를 구한다.

에러값 다항식은 8개의 레지스터로 구성이 되며 초기 연산값으로 α^0, α^{52}, α^{104}, α^{156}, α^{208}, α^5, α^{57}, α^{109}의 값을 갖는다.

3.3.7 에러 수정

에러 수정 계산식은

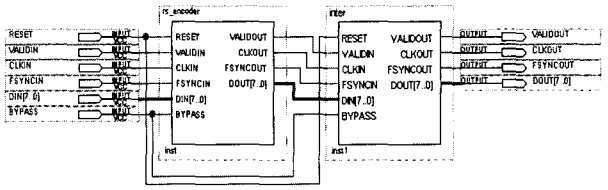
$$c'(x) = r(x) \text{ xor } e(x) \quad (3.8)$$

으로 입력된 코드 워드와 생성된 에러 값를 Exclusive-OR 연산에 의해 수정을 한다.

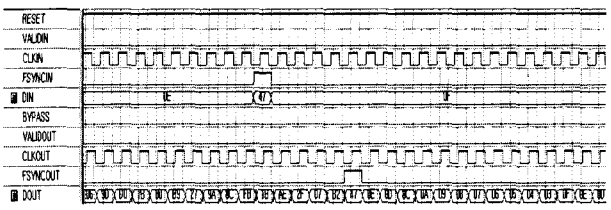
4. 실험결과

본 연구에서 VHDL기술 및 시뮬레이션은 반도체 설계교육센터(IDEC)에서 지원받은 Quartus와 Modelsim툴을 사용하였다. RTL 시뮬레이션은 ModelSim을 이용하였으며, P&R은 Altera사의 QuartusII 툴을 이용하여 검증하였다. 그리고 FPGA 테스트는 Altera사의 Stratix 디바이스를 이용하여 자체 제작한 보드를 이용하여 검증하였다.

(그림 11)과 (그림 12)는 Outer 인코더의 전체 블록도와 타이밍도로 입력되는 MPEG2-TS데이터는 인코더를 거쳐 분산된 형태의 데이터를 송신하게 된다. (그림 12)의 타이밍 도에서처럼 입력된 데이터의 마지막 16바이트는 패리티 바이트로 정정이 인터리빙 되어서 출력되는 것을 알 수 있다.



(그림 11) Outer 인코더 블록도



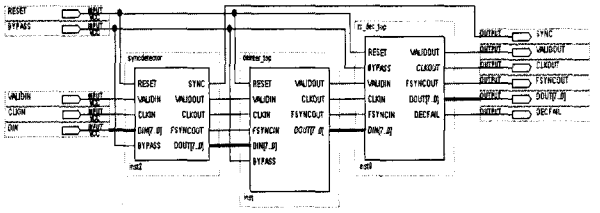
(그림 12) Outer 인코더 타이밍도

<표 2>는 Outer 인코더의 컴파일 결과로 QuartusII 툴을 이용하여 얻은 결과이다.

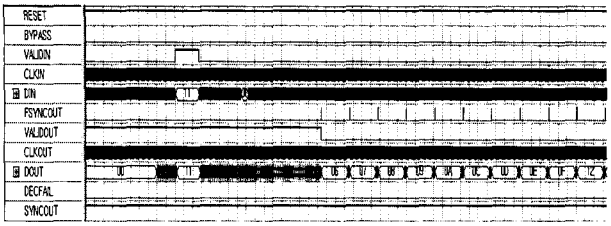
〈표 2〉 Outer 인코더 특성

디바이스	EP1S10F484C7(Stratix)
Total logic elements	1,004/10,570
Total pins	24/335
Total memory bit	16,384/920,448
최대 동작 주파수	100MHz

(그림 13)과 (그림 14)는 Outer 디코더의 블록도와 타이밍도 부분으로 입력된 분산된 데이터를 원래형태로 수정하고 중간에 발생(그림 14)에서 보면 처음 데이터가 입력되었을 때 동기를 찾고 디인터리빙을 시켜 에러를 수정한 정상적인 데이터가 출력되는 것을 볼 수 있다.할 수 있는 에러를 수정하는 역할을 한다.



(그림 13) Outer 디코더 블록도



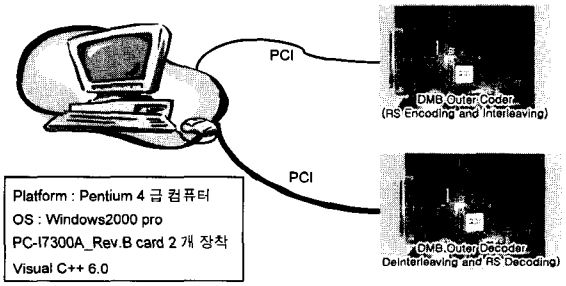
(그림 14) Outer 디코더 타이밍도

<표 3>은 Outer 디코더의 컴파일 결과로 QuartusII 툴을 이용하여 얻은 결과이다.

〈표 3〉 Outer 디코더 특성

디바이스	EP1S10F484C7(Stratix)
Total logic elements	5,858/10,570
Total pins	25/335
Total memory bit	24,576/920,448
최대 동작 주파수	80MHz

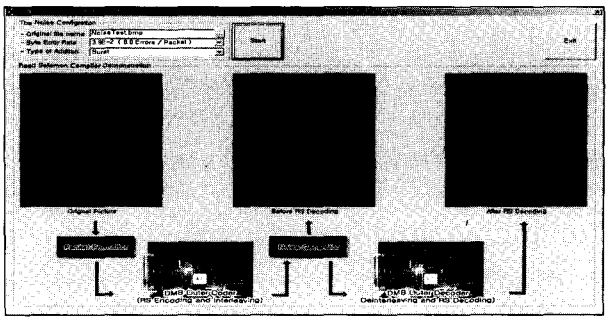
보드 실장 테스트는 (그림 15)에서처럼 PC에 고속 PCI 장착 후 PCI 인터페이스를 이용해 보드에 데이터를 주고받았다. Test한 동작속도는 고속 PCI의 최대 성능 10MHz까지 테스트 하였다.



(그림 15) PCI를 이용한 Test 환경

(그림 16)는 검증하기 위한 프로그램으로 비주얼 C++를 이용하여 작성하였으며, 24비트 BMP파일을 이용하였다. 노이즈 삽입방식은 채널 상에서 발생할 수 있는 에러의 형태를 군집과 비군집으로 설정하고 이를 임의의 위치에 삽입하여 실질적인 채널상에서 발생할 수 있는 노이즈를 테스트 하였다. (그림 16)에서 좌측이 입력된 원 영상이며, 이 영상을 인코딩하고 노이즈를 첨가한 영상이 중앙의 영상이고, 이를 디코딩한 영상이 오른쪽의 영상으로 입력된 노이즈가 수정되어 출력되는 것을 확인 할 수 있었다.

<표 4>는 노이즈의 크기를 나타낸 것으로 테스트한 이미지의 크기(249,696바이트)를 기준으로 작성하였다. 위 표에서 EPP가 8일때 즉 BER이 3.9×10^{-2}이고 군집에러 일때 첨가된 노이즈가 모두 수정이 되어 출력되는 것을 확인하였다. 단, 노이즈의 삽입 단위는 204(바이트)×12(패킷)으로 설정하고 테스트하였다.



(그림 16) 실장 테스트 결과 화면

〈표 4〉 노이즈의 크기

Errors(byte)	BER(Byte Error Rate)	EPP(Errors Per Packet)
24	1.0E-4	1.9E-2
249	1.0E-3	1.9E-1
1224	4.9E-3	1.0
2448	9.8E-3	2.0
2496	1.0E-2	2.039
3672	1.4E-2	3.0
4896	1.9E-2	4.0
6120	2.4E-2	5.0
7344	2.9E-2	6.0
8569	3.4E-2	7.0
9792	3.9E-2	8.0
11016	4.4E-2	9.0
12240	4.9E-2	10.0
24969	1.0E-1	20.399

5. 결 론

본 논문에서는 최근 방송국의 증가로 인한 주파수 자원의 고갈과 아날로그 FM방송의 혼선 및 이동시 음질저하 문제의 해결책으로 새롭게 발전하고 있는 차세대 디지털 방송규격인 지상파 DMB에서 Outer 인코더/디코더를 설계하고 ALTERA의 FPGA를 이용하여 구현하고 검증하였다.

FPGA보드와 PC를 고속 PCI를 이용하여 연결하고 여러 가지 데이터를 삽입하여 검증하였다. 또한 채널상에 발생할 수 있는 노이즈의 형태를 군집과 비군집으로 나누어 다양한 크기의 노이즈를 삽입하면서 결과를 관찰하고 정상적으로 노이즈가 수정되는 것을 확인하였다.

본 시스템을 활용함으로써 지상파 DMB를 송·수신하는데 있어서 보다 품질이 좋은 서비스를 가능하게 할 것이라고 사료된다. 그리고 본 연구에서는 DMB Outer 부분만을 작성하였지만 더 나아가 DAB칩과 튜너부분을 하나의 원칩으로 제작하는 지속적인 연구가 필요할 것이다.

참 고 문 헌

[1] I. S. Reed, In-Shek Hsu외, "The VLSI Design of a Reed-Solomon Decoder Using Berlekamp's Bit-Serial Multiplier Algorithm", IEEE Tran.on Computers Vol.33, No.10, pp.906-911, Oct., 1984.

[2] Howard M. Shao, T.K. Trunong, Irving S. Reed외 "A VLSI Design of a Pipeline Reed-Solomon Decoder," IEEE Tran.on Computers Vol.34, No.5, pp.393-403, May, 1985.

[3] J. L. Massey, "Shift Register Synthesis and BCH Decoding," IEEE Tran.on Information Thoery, Vol. IT-15, No.1, pp122-127, Jan., 1969.

[4] H. M. Shao and I. S. Reed, "On the VLSI design of a pipeline Reed-Solomon decoder using systolic arrays," IEEE Tran.on Compututers, Vol.C-37, No.10, pp.1273-1279, Oct., 1988.

[5] S. Kwon and H. Shin, "An area efficient VLSI architecture of a Reed-Solomon decoder/encoder for digital VCR's," IEEE Trans.on Consumer Electron, pp.1019-1027, Nov., 1997.

[6] 이만영, "BCH부호와 Reed-Solomon부호", 민음사, 1990.

[7] 이만영, "Error Correction Coding Theory," McGraw-Hill Publishing Company, 1989.

[8] 한국전자통신연구원, "지상파 DMB 시장전망 및 경제적 파급효과", 2003.

[9] 한국정보통신기술협회, "DMB 표준화", TTA Journal 86호, pp.51-58, 2003.

[10] 한국정보통신기술협회, "위성 DMB 표준화 및 서비스", TTA Journal 87호, pp.141-148, 2003.

원 지 연

e-mail : jywon@hanbat.ac.kr

2002년 한밭대학교 컴퓨터공학과학사)

2002년~현재 한밭대학교 정보통신 전문대학원 컴퓨터공학과(석사과정)

관심분야 : VLSI설계, 임베디드 시스템

이 재 흥

e-mail : jhlee@hanbat.ac.kr

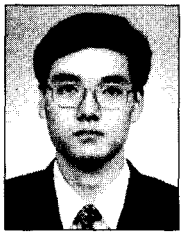
1983년 한양대학교 전자공학과(공학사)

1985년 한양대학교 대학원 전자공학과 (공학석사)

1994년 한양대학교 대학원 전자공학과 (공학박사)

1989년~현재 한밭대학교 정보통신·컴퓨터공학부 교수

관심분야 : CAD 및 VLSI 설계, Embedded 시스템, SoC 시스템



김 건

e-mail : kimgeon@etri.re.kr

1997년 중앙대학교 전자공학과(학사)

1999년 중앙대학교 전자공학과(공학석사)

1999년~현재 한국전자통신연구원 방송 시스템연구부 연구원

관심분야 : 디지털 오디오 방송, 디지털 통신, VLSI 설계