위성통신을 위한 (204, 188) Reed-Solomon Decoder 설계 및 합성

신수경* · 최영식** · 이용재***

*동의대학교

The Design and Synthesis of (204, 188) Reed-Solomon Decoder for a Satellite Communication

Su-Kyoung Shin* · Young-Shig Choi** · Yong-Jae Lee***

*Dongeui University

E-mail : mrssk@hanmail.net* · yschoi@dongeui.ac.kr** · yjlee@dongeui.ac.kr***

요 약

본 논문에서는 위성방송용으로 제안되고 있는 GF(2^8)상의 8중 오류정정 (204,188) Reed-Solomon 복호기를 설계하고 CMOS 라이브러리를 이용하여 합성하였다. Reed-Solomon 부호의 복호 알고리즘은 오증을 계산하고, 오류위치 다항식을 구한 후, 오류를 판단하여, 오류치를 구하는 4단계로 이루어지는데, 본 논문에서는 Modified Euclid 알고리즘을 사용하여 설계가 이루어졌다. 먼저, 알고리즘과 회로의 동작을 확인하기 위해 C++로 프로그램을 작성하여 검증을 한 후, 이를 바탕으로 VLSI 설계를 위해서 Verilog HDL로 하드웨어를 기술하였다. 또한, 각 블록에 대한 로직 시뮬레이션을 거친 후, 최종적으로 Synopsys사의 합성 툴을 이용해서 회로를 합성하였다.

ABSTRACT

This paper describes the 8-error-correction (204,188) Reed-Solomon Decoder over GF(2^8) for a satellite communication. It is synthsized using a CMOS library. Decoding algorithm of Reed-Solomon codes consists of four steps which are to compute syndromes, to find error-location polynomial, to decide error-location, and to slove error-values. The decoder is designed using Modified Euclid algoritnm in this paper. First of all, The functionalities of the circuit are verified through C++ programs, and then it is designed in Verilog HDL. It is verified through the logic simulations of each blocks. Finally, The Reed-Solomon Decoder is synthesized with Synopsys Tool.

I. 서 론

최근 정보의 양과 종류가 다양해짐에 따라 통신화망은 고속화 되고 영상과 음성등의 멀티미디어 자료와 같은 실시간 데이터 전송에 대한 수요가 증가하는 추세이다. 따라서 고속 정보 전송 과정에서 발생할수 있는 에러의 복구도 고속으로 진행 된다. RS 부호는 적은 redundancy로도 강력한 에러 정정 능력을 갖는 에러 정정 부호로 비2진 형태이어서 산발 에러뿐만 아니라 연집 에러에도 강한 특성을 갖으며 현재 위성통신, 무선통신, 데이터 통신, 컴퓨터의 저장 시스템 등에 널리 이용되고 있다. 본 논문은 유럽의 디지털 위성방송 표준인 ETSI(European Telecommunication Standards Iustitute)에 호환하는 위성 방송 송수신 시스템 중 Reed-Solomon Decoder를 Verilog HDL로 기술한 후, Synopsys사의 합성툴로 전반부 설계까지 하였다.

II. RS 복호기의 알고리듬과 설계

오류 제어를 중심으로 본, 전체 디지털 위성통신 방송 송수신 시스템은 아래 그림 1과 같다. 이 중, Reed-Solomon Decoder는 (204,188) t=8 로서 아래 식 (1), (2)와 같은 원시 다항식과 생성 다항식을 갖는데, 송신단에서의 Randomzie 되어진 188 바이트의 데이터가 204 바이트로 Reed-Solomon 부호화 되어지며, 수신단에서는 부호기의 역순으로 204 바이트 단위로 복호하여, 최대 8 바이트의 에러까지 정정할수 있는 능력을 보유하고 있다.

생성다항식 : $g(x) = x^{15} + x^{14} + 1$ (1)

원시다항식 : $p(x) = x^8 + x^4 + x^3 + x^2 + 1$ (2)

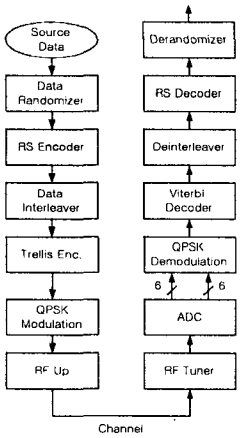

그림1. 디지털 위성 방송 수신기 오류 제어 시스템의 구성

이러한 Reed-Solomon Decoder의 전체 구조는 그림 2와 같으며, 신드롬 생성기와 Modified Euclid 알고리듬 블럭, Chien Search 블럭을 거쳐 검출된 에러가 지연된 입력과 Exclusive-OR 연산되어 최종적으로 에러가 정정되어지는 구조로 구성되어진다.

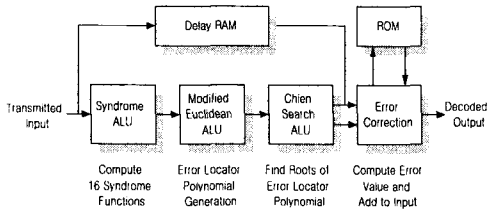

그림 2. MEA을 이용한 Reed-Solomon Decoder의 구조

2.1 신드롬 계산

채널의 불완전한 특성 때문에 수신된 데이터에는 에러가 발생 할 확률이 매우 크게 된다. 이를 보상해 주기 위해 수신단에서는 송신단에서와 반대 과정을 거쳐서 에러를 정정하고 원래의 메시지를 얻을 수 있다. Reed-Solomon 부호 복호를 하기 위해서는 수신한 데이터에 대한 신드롬을 계산한다. 신드롬은 블록 부호의 복호에 필수적으로 요구되는 정보이며, 수신된 데이터는 g(x)의 근을 대입해 얻어지는 결과이다. 모든 부호어들은 생성다항식 g(x)로 나누면 나누어 떨어진다. 따라서 수신된 데이터에 오류가 없다면 모든 신드롬은 '0' 이 된다. 그러나 만일 전송과정에서 오류가 발생했다면 생성다항식보다 한 차수가 낮은 신드롬 다항식이 생성된다. 신드롬을 구하기 위해서는 다항식 계산회로가 필요하므로 아래의 식 (3)과 같이 적용하여 이를 하드웨어로 구현하면 편리하다.

$$\begin{aligned} S_i &= S(a^i) = r(a^i) = e(a^i) = c(a^i) + e(a^i) \quad 0 \le i \le 2t-1 \\ &= r_0 + r_1 a^i + r_2 a^{2i} + \cdots + r_{n-1} a^{(n-1)i} \\ &= r_0 + a^i(r_1 + a^i(r_2 + a^i(r_3 + \cdots + a^i(r_{n-2} + r_{n-1}a^i)\cdots))) \end{aligned} \quad (3)$$

본 논문의 (204,188) Reed-Solomon Decoder에 이를 적용하여 식 (4)에 보였고, 신드롬 S_0, S_1, $\cdots, S_{14}$, S_{15}을 병렬적으로 동시에 구할수 있도록 설계하였다. 신드롬 계산 기본 단위 블럭을 그림 3에 표시하였다.

$$\begin{aligned} S_i &= r(a^i) \qquad \text{for } i = 0,1,2,\cdots,14,15 \\ &= r_{203}(a^i)^{203} + r_{202}(a^i)^{202} + \cdots + r_1 a^i + r_0 \\ &= (\cdots((r_{203}a^i + r_{202})a^i + \cdots + r_1)a^i + r_0) \end{aligned} \quad (4)$$

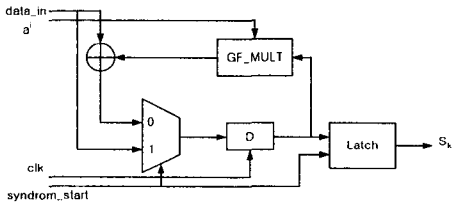

그림 3. 신드롬 계산 기본 단위

2.2 Modified Euclid 알고리듬

본 논문에서는 Modified Euclid 알고리듬을 사용하였는데, 이 방법으로 회로를 구현하게 되면, 결과로 에러 평가 다항식과 에러 위치 다항식을 손쉽게 얻을수 있다. 이 부분에서 필요한 레지스터열들은 R(x) 레지스터열, Q(x) 레지스터열, start 레지스터, $\lambda(x)$ 레지스터열, 그리고 $\mu(x)$ 레지스터열 들을 갖는다. 따라서 Modified Euclid 알고리듬은 에러 위치 다항식을 계산해 내기 위해 다음 두 식 (5), (6)의 최대공약수를 구하는 방법이다.

$$A(x) = x^{2t} \quad (5)$$

$$S(x) = \sum_{k=1}^{2t} S_k x^{2t-k} \quad (6)$$

(5), (6)식의 최대공약수를 다음 식 (7)을 만족하도록 i를 반복시켜, i번째 나머지 $R_i(x)$, $r_i(x)$와 $\lambda_i(x)$를 찾도록 반복하는데, 나머지 $R_i(x)$의 차수가 t보다 작으면 멈춘다. 그러면 $\lambda(x)$가 원하는 다항식 $\lambda(x)$가 된다.

$$r_i(x)A(x) + \lambda_i(x)S(x) = R_i(x) \quad (7)$$

이 알고리듬을 위한 초기화 조건은 아래 식 (8)

과 같다.

$$R_0(x) = A(x),\ Q_0(x) = S(x),\ \lambda_0(x) = 0$$
$$\mu_0(x) = 1,\ r_0(x) = 1,\ \eta_0(x) = 0 \qquad (8)$$

위 초기조건에서 아래식을 반복시켜 원하는 다항식을 구한다.

$$R_i(x) = [\sigma_{i-1}\, b_{i-1}\, R_{i-1}(x) + \sigma_{i-1}\, a_{i-1}\, Q_{i-1}(x)] - x^{|l_{i-1}|}[\sigma_{i-1}\, a_{i-1}\, Q_{i-1}(x) + \sigma_{i-1}\, b_{i-1}\, R_{i-1}(x)]$$
$$\lambda_i(x) = [\sigma_{i-1}\, b_{i-1}\, \lambda_{i-1}(x) + \sigma_{i-1}\, a_{i-1}\, \mu_{i-1}(x)] - x^{|l_{i-1}|}[\sigma_{i-1}\, a_{i-1}\, \mu_{i-1}(x) + \sigma_{i-1}\, b_{i-1}\, \lambda_{i-1}(x)]$$
$$Q_i(x) = \sigma_{i-1}\, Q_{i-1}(x) + \sigma_{i-1}\, R_{i-1}(x)$$
$$\mu_i(x) = \sigma_{i-1}\, \mu_{i-1}(x) + \sigma_{i-1}\, \lambda_{i-1}(x) \qquad (9)$$

여기서, a_{i-1}과 b_{i-1}은 각각 $R_{i-1}(x)$와 $Q_{i-1}(x)$의 최고차항의 계수이다. 그리고 다음 식 (10)이 성립해야된다.

l_{i-1} = $R_{i-1}(x)$의 차수 - $Q_{i-1}(x)$의 차수

$$\sigma_{i-1} = \begin{cases} 1 & \text{if } l_{i-1} \geq 0 \\ 0 & \text{if } l_{i-1} < 0 \end{cases} \qquad (10)$$

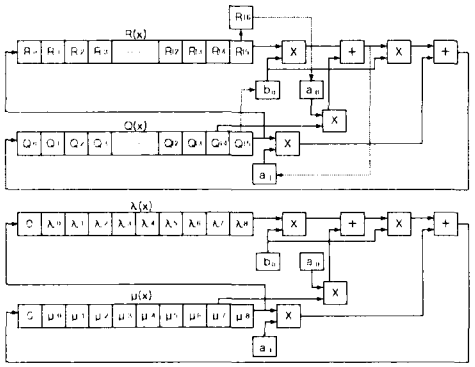

그림 4. Modified Euclid 알고리듬 블럭도

2.3 Chien Search

이 블럭에서는 Modified Euclid 알고리듬을 수행해서 구한 에러 위치 다항식과 에러 평가 다항식들로부터 에러의 위치와 크기를 찾는다. 우선 에러의 위치는 에러 위치 다항식의 근을 구함으로써 알 수 있다. 에러의 위치가 결정되면 그 위치에서 에러의 크기는 식 (11)으로부터 구할수 있다.

$$e^{i} = -\frac{\omega(X_i^{-1})}{\sigma'(X_i^{-1})X_i^{b-1}} \quad i = 1,2,\cdots,\nu \qquad (11)$$

이렇게 해서 찾아진 에러에 입력으로 받은 부호 단어를 더해 주면 에러가 정정된다. 에러의 위치는 식 (12)와 같은 에러 위치 다항식에 a^{-i}를 대입해서 0이 나오는지를 보면 된다.

$$\sigma(x) = \sigma_0 + \sigma_1 x + \cdots + \sigma_t x^t \qquad (12)$$

a^{-i}를 대입해서 다항식의 값이 0이 나왔다면 i+1 번째 부호단어에서 에러가 발생한 것이고, 그렇지 않다면 에러가 발생하지 않은 것이다. 그런데 입력 부호 단어의 순서가 N번째부터 하나씩 감소하면서 첫 번째 부호 단어까지 들어오게 되므로 에러의 위치를 찾을 때 에도 처음에는 $a^{-(N-1)}$을 대입해서 N번째 부호 단어에서의 에러 발생 여부를 확인하고, 그 다음부터는 이전 값에 a를 곱하면 원하는 위치에서 에러 발생 여부를 확인할수 있다. 이렇게 대입해서 각각의 항의 값을 계산하는 것을 그림 5에 나타내었다.

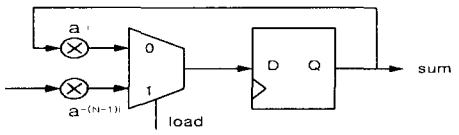

그림 5. Chien Serch의 기본 블럭

III. RS 복호기 회로의 시뮬레이션 및 합성

Reed-Solomon Decoder는 C++ 프로그램으로 기능적 동작 특성을 검증한 후, Verilog HDL로 하드웨어를 기술하였다. Verilog RTL code는 다음 그림 6과 같이 6개의 블럭으로 구성하였다.

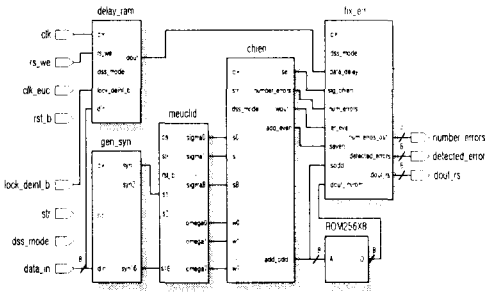

그림 6. Reed-Solomon Verilog HDL의 구성

먼저, 본 Reed-Solomon Decoder의 Module 이름은 rs_dec이며, 복호를 위한 8비트의 입력신호(data_in)가 신드롬 생성기(gen_syn)와 입력 신호 지연 소자(delay_ram)으로 인가되어, 신드롬 생성과 입력 신호 지연을 수행한다. 생성된 신드롬은 Modified Eucild 알고리듬에 해당하는 meuclid를 거쳐 sigma와 omega 신호를 생성하여, Chien Search에 해당하는 chien 블럭을 통과하게 된다. chien 블럭에서 생성된 신호는 마지막으로 에러

정정 블럭인 fix_err블럭을 거쳐 최종 복호된 8비트의 신호(dout_rs)가 출력되어 진다. 이 밖에도, 에러의 수를 나타내는 4비트 신호 number_error와 에러 값을 나타내는 8비트 신호 detector_error 신호도 출력되어진다. 따라서 입력 포트는 총 15비트, 출력 비트는 총 20 비트로서 각 입출력 포트를 표 1에 표시하였다.

Port name	I/O	bits	Function
clk	I	1	Main clock
rs_we	I	1	write enable of delay_ram
clk_euc	I	1	clock of modified euclid block (meuclid)
rst_b	I	1	reset (active low)
lock_deint_b	I	1	de-interleaver lock signal (active low)
str	I	1	start signal of reed solomon block
dss_mode	I	1	dvb-s/ dss mode selection (dss='b1)
data_in	I	8	input signal
number_error	O	4	error number (up to 8)
detected_error	O	8	error value
dout_rs	O	8	decoded output

표 1. Reed-Solomon Decoder의 입출력 포트

이렇게 구성되어진 복호기의 Verilog HDL 모델링을 Mentor Graphics사의 Model-Sim으로 전산 모의 실험을 거쳐 최대 8바이트의 에러가 정정됨을 확인 할 수 있었다. 시뮬레이션 결과를 그림 7에 표시하였다.

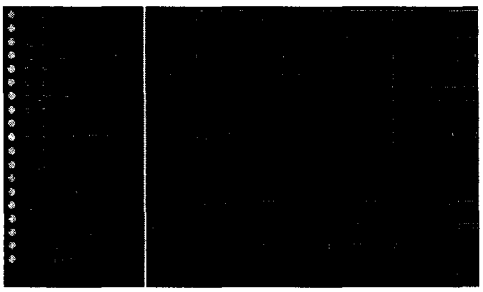

그림 7. Reed-Solomon Decoder 시뮬레이션 결과

이와 같이 논리 시뮬레이션을 완성한 Verilog HDL 프로그램을 Synopsys사의 회로 합성 툴을 이용해서 Reed-Solomon Decoder에 대한 회로합성을 수행하였다. 그 결과 복호기의 합성된 게이트의 수는 약 16000개이고, 약 22.5Mhz의 속도까지 동작하는데, 이는 22.5Mbyte/sec로 데이터를 처리 할수 있다는 것을 보여준다. 그림 8은 합성된 Reed-Solomon Decoder의 구성도를 보인 것이다.

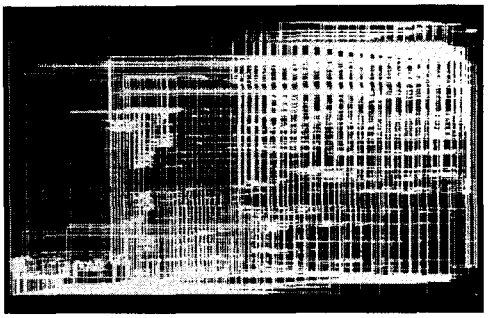

그림 8. Reed-Solomon Decoder 의 합성 결과

IV. 결 론

본 논문은 고속의 위성방송에서 요구하는 효율적인 대역폭의 사용과 안정적인 정보 전달을 하기 위해서 Modified Euclide 알고리듬에 바탕을 둔 (204,188) t=8 Reed-Solomon Decoder이다. 기능적 동작 검증을 위해 C++ 프로그램으로 검증했으며, Verilog HDL로 시스톨릭 어레이 형태를 사용한 파이프라인 구조로 회로를 설계하여 논리 시뮬레이션을 통해 회로의 동작을 검증하였고, 최종적으로 회로 합성까지 실시하였다. 구현된 Reed-Solomon Decoder는 파이프라인 구조를 취함으로써 게이트 수의 증가를 방지함과 동시에 속도 개선을 얻을 수 있었다. 그래서 0.6μm CMOS 라이브러리를 사용한 결과, 총 게이트 수는 약 16,000개이고, 동작 속도는 22.5Mhz이다.

참고문헌

[1] SHU LIN, DANIEL J. COSTELLO, JR. "Error Control Coding-Fundamentals and Applications" 1983 by Prentice-Hall.

[2] HOWARD M. SHAO. T. K. TRUONG "A VLSI Design of a Pipeline Reed-Solomon Decoder", IEEE Trans. Computers, Vol. C-34, NO. 5, pp. 393~403, 1985

[3] HOWARD M. SHAO. IRVING S. REED "On the VLSI Design of a Pipeline Reed-Solomon Decoder Using Systolic Arrays" IEEE Trans. Computers, Vol. 37, NO. 10, pp. 1273~1280, 1988

[4] PETER SWEENEY. 박상규 역 "오류제어부호" January, 1998

[5] 이만영 "BCH부호와 Reed-Solomon부호" March, 1990

[6] 이문호 " 갈루이스필드 · 리드솔로몬 · 비터비 · 터보 부호기의 설계" July, 2000