3-Bit Soft Decision Viterbi 복호기의 VLSI 설계

김 기 명, 송 인 채
숭실대학교 정보통신공학과
전화 : (02) 816-6073 / 팩스 : (02) 821-7653

VLSI Design of 3-Bit Soft Decision Viterbi Decoder

Ki-Myoung Kim, Inchae Song
Department of Information & Telecommunication Engineering, Soongsil University
E-mail : jazz@hanul.soongsil.ac.kr

Abstract

In this paper, we designed a Viterbi decoder with constraint length K=7, code rate R=1/2, encoder generator polynomial $(171, 133)_8$. This decoder makes use of 3-bit soft decision. We designed the Viterbi decoder using VHDL. We employed conventional logic circuit instead of ROM for branch metric units(BMUs) to reduce the number of gates. We adopted fully parallel structures for add-compare-select units(ACSUs). The size of the designed decoder is about 200,000 gates.

I. 서론

현대는 디지털 통신 시스템이 보편화되어있다. 호출기나 cellular phone이나 PCS등은 이미 일반화되어 널리 사용되고 있다. 이런 디지털 통신 시스템에서는 에러가 없는 데이터의 전송을 요구하며 신뢰성 있고 효율적인 시스템을 추구한다. 신뢰성 있고 효율적인 데이터의 전송을 위해 에러 정정 코딩방식이 필연적으로 사용되었다. 이 에러 정정 코딩방식은 redundancy를 첨가하여서 에러를 제어할 수 있다[1].

본 논문에서는 3 bits Soft Decision code을 가지는 Viterbi 복호기를 반도체설계교육센터(IDEC)에서 지원된 Altera와 Mentor를 사용하여 VHDL로 설계하였다. 설계된 Viterbi 복호기의 code rate(R)은 1/2이고, constraint length(K)는 7이며 encoder의 generator polynomial은 $(171, 133)_8$이다. branch metric을 계산하는 BMU는 ROM으로 구성하지 않고 간단한 logic으로 구성하여 전력소모와 면적의 낭비를 막았다. 그리고 path metric을 계산하는 ACSU는 완전 병렬구조를 택하여 고속처리가 가능하도록 하였다.

II. Viterbi Algorithm

Viterbi 알고리즘은 트렐리스 도(trellis diagram)에서 송신단으로부터 noise가 존재하는 channel을 통해 전송된 에러 섞인 데이터의 path와 가장 유사한 path를 찾는 알고리즘으로 convolution code의 최대유사 디코딩(MLD; maximum likelihood decoding)을 효율적으로 수행하는 알고리즘이다.

Viterbi 알고리즘에서 가능한 trellis path들과 수신된 신호 사이에 발생한 차이를 branch metric이라고 하고 이를 각 상태에 따라 천이가 가능한 path에 따라 계속

더해나간다. 이 상태들의 값을 path metric이라고 한다. 또 각 상태로 들어오는 path들 중 작은 path metric을 갖는 path를 선택하여 새로운 path metric을 생성하게 된다. 그리고 이 때의 선택 정보를 저장하게 된다. 가장 작은 path metric을 갖는 path를 survivor path라고 하고 어느 정도 시간(trace back depth, TBD)이 지난 후에 survivor path를 역추적(trace back)하여 decoding 출력을 결정하게 된다[2].

Ⅲ. Viterbi Decoder의 설계

convolution code의 복호기 중에서 가장 효율적인 복호기로 알려진 Viterbi 복호기는 크게 4개의 블록으로 나눌 수 있다. branch metric을 계산하는 branch metric unit(BMU)와 이전 path metric에 현재 branch metric을 더해 현재 path metric을 계산하는 add compare select unit(ACSU)와 가장 작은 path metric을 갖는 state를 찾아 그에 따른 제어신호를 출력하는 control 블록 그리고 시간에 따른 path 선택정보를 역추적해 디코딩하는 trace back unit(TBU)로 구성된다. 그림1은 Viterbi 복호기의 전체 블록도이다.

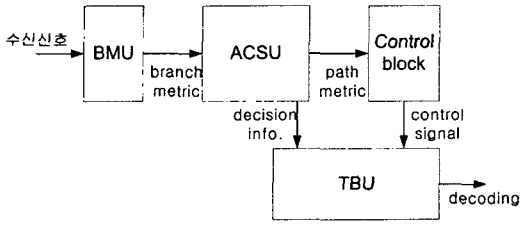

그림 1. Block Diagram of Viterbi Decoder

1. Branch Metric Unit (BMU)

BMU는 수신된 신호와 주어진 trellis diagram을 따라 천이 가능한 path에 따라 distance를 구하는 블록이다. BMU의 계산 방식으로는 hard decision(HD)과 soft decision(SD)이 있다. SD는 HD보다 약 2dB의 이득을 가지고 있기 때문에 이 논문에서는 3-bit soft decision을 선택하였다.

3-bit soft decision을 sign magnitude 표현방식을 사용해 1/2 code rate의 출력이 00, 01, 10, 11일 경우의 branch metric을 계산하면 다음과 같이 표현할 수 있다. 여기서 J_1과 J_2는 soft decision되는 입력이다.

00, $BM_0(t) = (J_1 \odot 011) + (J_2 \odot 011)$
01, $BM_1(t) = (J_1 \odot 011) + (J_2 \odot 111)$
10, $BM_2(t) = (J_1 \odot 111) + (J_2 \odot 011)$
11, $BM_3(t) = (J_1 \odot 111) + (J_2 \odot 111)$

⊙에 대한 연산방법은 표1에 주어진다.

표1. ⊙ operation

Jx	011	010	001	000	100	101	110	111
011	0	1	2	3	4	5	6	7
111	7	6	5	4	3	2	1	0

그림 2는 BMU의 블록도이다. BMU의 구성은 ROM table로 구현하는 방법과 logic을 이용해 구현하는 방법이 있다. ROM table로 구현하는 방식을 보면, 각각 3-bit인 J_1과 J_2를 ROM의 address로 하고 각각 4-bit인 branch metric 값 BM_0, BM_1, BM_2, BM_3을 16-bit data word로 하는 ROM을 구성하게되면 $2^6 \times 16$-bit의 ROM이 생성되고 그 ROM의 디코더의 크기만도 만만치가 않다. 또 한가지의 방법은 logic으로 구성하는 방법이 있다. 본 논문에서는 면적과 전력소모 면에서 유리한 logic으로 BMU를 구성하였다.

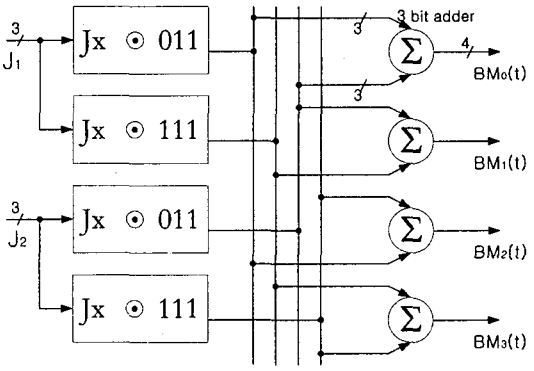

그림 2. branch metric unit

2. Add Compare Select Unit (ACSU)

ACSU는 이전 path metric과 현재 수신된 신호의 branch metric계산 결과를 더해 새로운 path metric을 만들어내며 trellis node에 중첩하는 path를 비교하여 더 작은 path metric을 갖는 path를 선택하는 블록이다. K=7인 Viterbi 복호기의 state는 64개이고, 그림 3과 같은 butterfly가 32개가 생긴다. 그림 4는 ACSU의 기본 cell을 보여준다. K=7인 복호기는 32개의 ACS 기본 cell을 병렬로 연결하여 구성한다. 각 butterfly 구

조에서 ACS 연산은 다음과 같이 나타낼 수 있다.

$PM_i(t) = \min\{PM_{2i}(t-1) + BM_x(t),\ PM_{2i+1}(t-1) + BM_y(t)\}$

$PM_{2^{(m-1)}+i}(t) = \min\{PM_{2i}(t-1) + BM_y(t),\ PM_{2i+1}(t-1) + BM_x(t)\}$

(여기서 m은 encoder의 memory 개수이고, i=0,1,2,3……31)

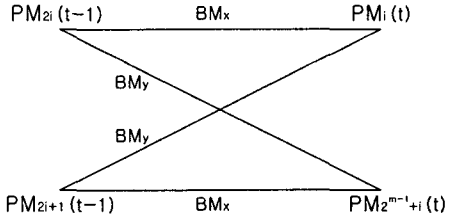

그림 3. ACSU의 butterfly 구조

ACSU는 Viterbi 복호기의 고속화의 걸림돌이 되는 병목현상에 해당한다. 이 병목현상을 해결하기 위해 여러 가지 계산방법이 고안되었다. 특히 ACS 모듈을 이용한 병렬계산방법은 하드웨어 크기가 큰 반면에 ACS 모듈에 대한 제어가 간단하고 고속처리가 가능하므로 고속통신 시스템 구성에 용이하다. 그러나 병렬 구조를 구현할 경우에 큰 면적과 각 상태간의 이전 path metric정보를 전달하기 위한 복잡한 신호연결로 인해 배선에 의한 지연이 커질 수 있으므로 Layout시 배선에 의한 지연을 최소화 할 수 있도록 각 상태에 대한 ACS 모듈을 배치해 주어야 한다[3].

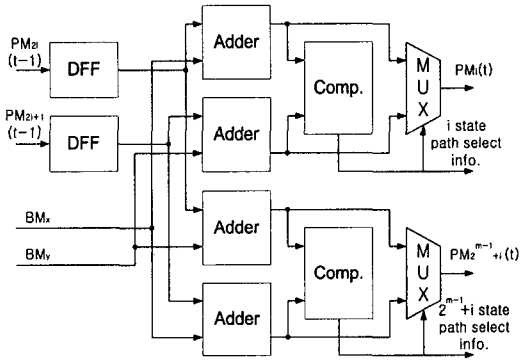

그림 4. ACSU 기본 cell

path metric값은 이전 path metric에 현재 branch metric값을 더하므로 시간에 지남에 따라 증가하게 되고, overflow가 발생했을 때 올바른 디코딩이 이루어지지 않을 수 있게된다. 이를 위해 path metric의 word size를 크게 키워 사용할 수 있으나 word size증가에 따른 하드웨어 증가와 지연증가로 연산속도가 저하되는 단점이 있다. 그래서 실험상의 에러 발생율을 고려하여 K=7일 때 path metric word size를 8 bit로 사용하면 에러율이 아주 적어져 무난하게 설계할 수가 있다[4].

3. Control Unit

TBU에서는 ACSU에서 계산되어진 path metric중 가장 작은 path metric을 가지는 상태로부터 역추적을 시작하게 되는데 이를 위해서는 매번 가장 작은 path metric을 가지는 상태를 찾아야한다. control unit은 TBU가 가장 작은 path metric을 갖는 상태로부터 역추적할 수 있도록 가장 작은 path metric을 갖는 상태를 찾아 그에 따른 제어신호를 내보내는 역할을 하는 블록이다.

4. Trace Back Unit (TBU)

TBU는 ACSU에서 만들어진 path select 정보를 저장해두었다가 Trace Back Depth(TBD) 후에 역 추적하여 디코딩 하는 블록이다. 원래 전송된 값을 추측하는 디코딩을 위해서 역추적을 하게되는데 이전의 ACSU에서 계산되어진 각 state마다의 path정보를 저장해두었다가 역추적 시 사용하게 된다. R=1/2의 ACSU에서는 각 state마다 2개의 path가 모이게된다. 그 2개의 path중에 작은 path metric을 가지는 path를 선택하게되는데, 윗쪽 path를 선택할 경우에는 '0'을 저장하고 아래쪽 path를 선택할 경우에는 '1'을 저장한다. 그리고 그 정보를 이용하여 각 state에 따른 역추적 천이를 이용해 역추적 하게된다. TBD후에 역추적을 할 때 path metric값이 가장 작은 state가 survivor path의 끝점이 된다. 이 점에서부터 역추적을 하여 처음의 한 bit를 디코딩하게 되는데 마지막 역추적이 S_i로부터 역추적 되어왔다면 '0'을, $S_{2^{m-1}+i}$로부터 역추적 되어 왔다면 '1'을 디코딩 하게된다.

그림 5는 역 추적하는 기본 cell을 나타내며 cntl_state 블록은 S_i에서 path select정보가 '0'일 때에는 S_{2i}로 역추적 하게 하고, '1'일 때에는 S_{2i+1}로 역추적 하게 하는 역할을 하도록 한다[5].

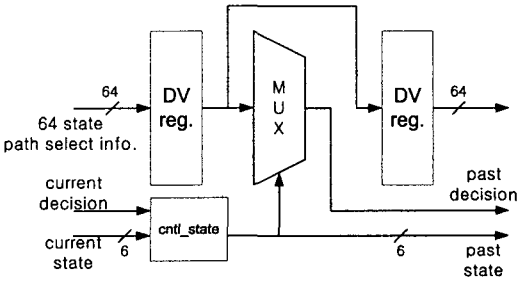

그림 5. trace back 기본 cell

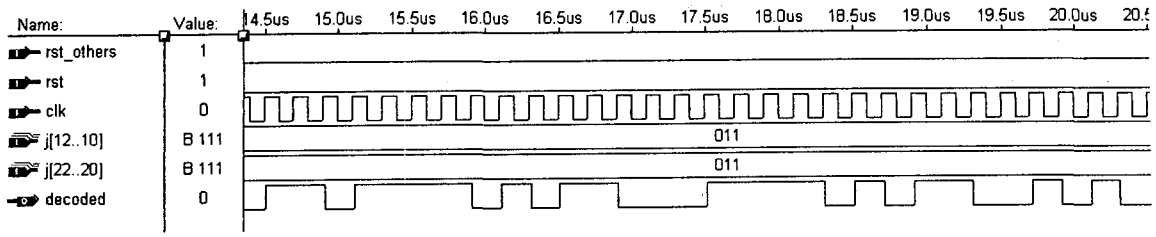

그림 6. Viterbi Decoder 전체 Simulation

Ⅳ. Simulation 및 검토

앞에서 살펴본 모든 block들을 합성한 전체적인 회로의 simulation 결과는 그림 6에 나타나 있다.

Viterbi 복호기의 전체 블록의 simulation은 "1101111010110011110101100101 00"이 순차적으로 encoder에 입력되어 그 결과 값이 노이즈가 있는 채널을 통과하게되고 그 값을 3 bits quantization을 통해 SD의 값 상태로 변환하게된다. 이 SD 값 상태로 변환된 값이 복호기의 입력에 들어가게 되어 에러를 correct하게 된다. encoder 블록과 노이즈 채널은 C code로 모델링하여 테스트 벤치 값을 뽑아내었다.

여기서는 TBD를 35(=7×5배)로 하여 설계하였다. 역추적 시에 TBD의 2배를 결쳐야하므로 70 clock후에 출력이 나오게 된다. 도중에 control 블록으로 인해 2 clock이 지연되어 총 72 clock후에 입력에 의한 출력이 나오게 된다.

Ⅴ. 결론

본 논문에서는 Code Rate R=1/2, Constraint Length K=7, Encoder Generator polynomial $(171, 133)_8$, 3 bits Soft Decision code을 가지는 비터비 복호기를 VHDL을 사용하여 설계하였다. 설계된 Viterbi 복호기의 크기는 약 20만 gate이다.

참고문헌

[1] Peter Sweeney, *Error Control Coding an introduction*, Prentice Hall, 1991.

[2] G. D. Forney Jr., "The Viterbi algorithm," Proc. IEEE, vol. 61, pp. 268-278, Mar. 1973.

[3] P. Gular and T. Kailath, "Locally Connected VLSI Architecture for the Viterbi Algorithm," IEEE J. Sel. Areas Comm., vol. 6, no.3, pp. 527-537, Apr. 1988.

[4] 김식, "Design of Area-Efficient Viterbi Decoder Supporting Punctured Coding", 서강대학교 석사 학위 논문, 1995.

[5] T. Truong, M. Shih, I. Reed, and E. Satorius, "A VLSI Design for a Trace-Back Viterbi Decoder", IEEE Trans. on Commun., vol. 40, no. 3, pp. 616-624., Mar. 1992.