메모리 최적화를 위한 Viterbi 디코더의 설계

신동석*, 박종진*, 김은원**, 조원경*
*경희대학교 전자공학과, **대림대학 전자통신학과
dsshin@csvlsi.kyunghee.ac.kr

A design of Viterbi decoder for memory optimization

Dongsuk Shin*, Jongjin Park*, Eunwon Kim**, WonKyung Cho*
*Dept. of Electronics Engineering, Kyunghee University
**Dept. of Electric communication Engineering, Dealim College
dsshin@csvlsi.kyunghee.ac.kr

Abstract

Viterbi decoder is a maximum likelihood decoding method for convolution coding used in satellite and mobile communications. In this paper, a Viterbi decoder with constraint length of K=7, 3-soft decision and traceback depth of Γ=96 for convolution code is implemented using VHDL. The hardware size of designed decoder is reduced by 4 bit pre-traceback in the survivor memory.

1. 서 론

디지털 통신에서 데이터의 전송시 오류가 발생하여 데이터의 손실을 가져오므로 오류 정정 부호화(error correcting coding)는 데이터의 전송시 필수가 된다. 이는 크게 블록 부호화(block code)와 콘볼루션 부호화(convolutional code)등 크게 두 종류가 있다.[1] 블록 부호화는 블록별로 부호화와 복호화가 수행됨으로 한 블록이 모두 입력이 끝난 후에 복호화 할 수 있다. 콘볼루션 부호화는 일정 길이의 메모리를 이용하여 이전과 현재의 입력 값을 연산하여 부호화를 수행한다. 그리고 복호화는 입력 후에 어느 정도의 지연이 있으나 한 클럭에 하나의 데이터를 출력한다. 콘볼루션 부호화의 디코더로 Viterbi 알고리즘이 이용되고 있다. 이러한 Viterbi 디코더는 위성통신이나 디지털 셀룰라 시스템에 널리 사용되고 있는데 이는 연성판정(soft decision)을 사용할 경우 백색 잡음 채널(additive white Gaussian noise channel)에 대해 오류 정정 능력이 향상되기 때문이다. [2]

본 논문에서는 제한 길이(constraint length) K=7, 3비트 연성판정, 부호화율 1/2인 콘볼루션 인코더의 디코딩을 수행하는 Viterbi 디코더를 역추적길이 96으로 하여 VHDL로 설계하였다. BMG 구조에서 BM(branch metric)의 값을 최소값으로 출력하도록 하여 PM(path metric)의 overflow가 일어날 확률을 줄이는 구조를 가지며, SM(survivor memory)에서 4비트 선행 역추적(4bit pre-traceback)을 수행하여 메모리 양을 줄이고 이에 적합한 메모리 구조를 채택하였다.

2. 콘볼루션 부호와 viterbi 알고리즘

그림 1은 제한 길이 k=3, 부호화 율(code rate) 1/2인 콘볼루션 인코더이다. 입력은 레지스터에 차례로 저장되며, 각 레지스터의 데이터를 XOR하여 하나의 입력 데이터에 대해서 두 개의 출력 데이터가 생성된다. 이때 인코더의 생성방정식은 식(1)과 같다. 그림 2는 그림 1의 상태 천이를 나타내는 트렐리스도 이다.

$$g_1 = 1 + D^2, \quad g_2 = 1 + D^1 + D^2 \qquad (1)$$

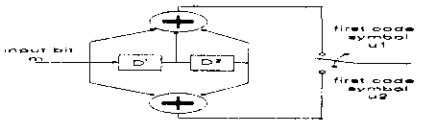

그림 1.콘볼루션 부호화의 부호기

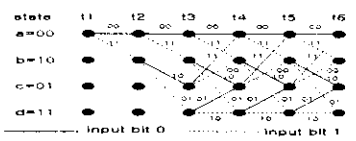

그림 2.트렐리스도 (rate 1/2,K=3)

최대유사디코딩(MLD: Maximum Likelihood Decoding)은 주어진 입력에 대해 가장 유사한 형식을 찾아 이를 디코딩의 출력으로 하는 방식으로 주어진 입력에 대해 가장 가능성이 높은 디코딩 결과를 나타내므로 최적의 디코딩 방법으로 알려져 있다.[3] 이러한 최대유사디코딩 방법으로 Viterbi 알고리즘을 사용한다.[2] 이러한 Viterbi 알고리즘은 부호기에 의해 생성되는 트렐리스와의 차이를 나타내는 척도로 BM과 PM을 사용한다. BM은 부호기의 트렐리스에 의해 생성된 기준 데이터와 송신된 데이터와의 차를 Hamming distance 또는 Euclidean distance값으로 정의하며 PM은 생존자 경로를 통해 전달된 연속적인 BM의 합으로 정의한다. Viterbi 알고리즘의 디코딩은 트렐리스의 각 상태에 연속적인 입력 값에 대해 PM과 BM을 가산하여(add) 작은 값의 (compare) PM을 선택하고(select), 이에 대한 경로 정보(decision bit)를 저장하고 일정시간 후에 최소의 PM을 가지는 경로에서 다시 거슬러 올라가(traceback) 선택된 경로가 디코딩 결과로 출력하는 방식이다.

통신에서 사용되는 채널로는 그림 3과 같이 '0'과 '1'의 두 가지 값만으로 나타내는 이진 대칭 채널(binary symmetric channel)과 가질 수 있는 값의 범위가 가우시안 확률을 갖는 가우시안 채널로 나눌 수 있다.[1] 가우시안 채널을 8단계로 양자화 할 경우에 입력되는 신호의 크기에 따라 값이 결정되고, 이 값은 '0'과 '1'에 대해 weight를 가지게 되는 유클리드 거리이다.

전송된 입력 데이터로 전자를 이용한 경우는 경성판정(hard decision), 후자는 연성판정(soft decision)이라 한다. Viterbi 디코더에서 BM를 3-bit 연성판정한 경우 경성판정보다 S/N비에서 2dB의 성능향상을 가진다.[2]

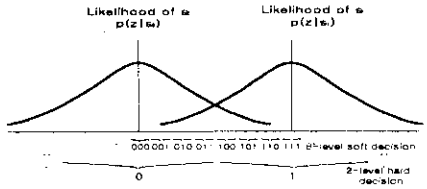

그림 3. Hard and soft decisions.

3. Viterbi 디코더의 구조

부호화율 1/2이고, 제한길이 K = 7인 콘볼루션 인코더는 식(2)의 생성방정식과 그림 4의 구조를 가진다. 그림 6은 이 인코더의 상태천이를 보여 준다.

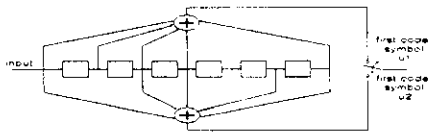

그림 4. 인코더의 구조

$$g_1 = 1 + D^1 + D^2 + D^3 + D^6$$
$$g_2 = 1 + D^2 + D^3 + D^5 + D^6 \qquad (2)$$

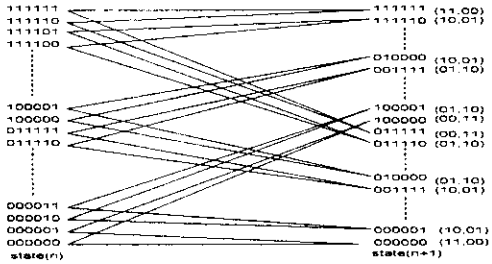

그림 5. K=7,R=1/2인 convolution code의 격자도

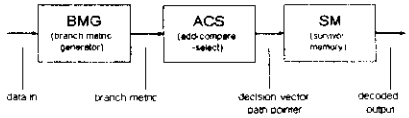

그림 6. Viterbi decoder의 전체 구조

Viterb 디코더는 그림 6과 같이 크게 BMG부(branch metric generator), ACS부(add-compare-select), SM (survivor memory)부로 구성된다.

3.1 BMG(Branch Metric Generator)부

연성판정의 BMG의 구조는 sign-magnitude, offset binary notion을 이용하는 두 가지가 방법이 있다. 본 논문에서는 구조가 간단한 3비트 offset-binary notion을 이용하였다. BMG에서는 들어오는 데이터에서 (00 10 01 11)등 4개의 BM을 만들어 이를 ACS에 전달한다. 이때 출력되는 값에서 최소값을 찾아 모든 출력에서 빼준다. 이는 PM에서 overflow가 발생할 확률을 줄여 오류를 어느 정도 적게 해 줄 수 있다. 본 논문에서는 하드웨어의 오버헤드를 줄이고자 ROM에 이러한 정보를 저장하였다.

3.2 ACS(Add-Compare-Select)부

ACS 연산은 PM과 BM을 더해주고 같은 상태로 천이하는 두 PM을 비교하여 작은 값을 가지는 PM을 선택하고 상태천이 정보를 SM에 출력한다. 그림 5는 상태천이를 나타내며 이는 ACS연산기들의 연결관계를 결정하는 요소가 되며, 나비 구조를 가지는 것을 알 수 있다. 그리고 이를 하나의 연산기로 구성하여 라우팅 와이어의 길이를 줄일 수 있다. 상태수가 64개이므로 두 개를 한 쌍으로 하여 총 32개의 동일한 기능을 하는 ACS 연산기로 구성된다.[5][8] 이와 같은 구조을 완전 병렬구조라 하며 고속의 데이터 전송에 사용된다. 그림 7은 하

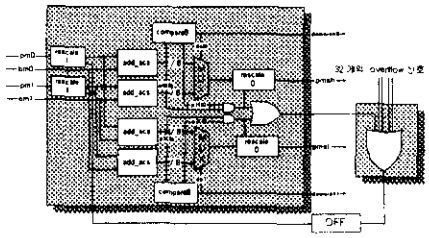

그림 7. ACS 연산기

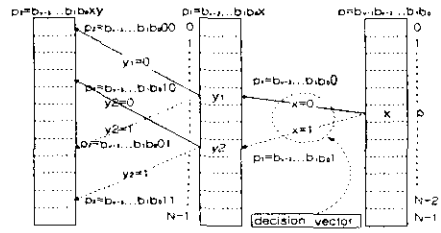

그림 8. 2bit traceback

나의 ACS 연산기 구조로 두 7비트 PM과 두 4비트 BM을 입력으로 하는 ACS 연산기이다. 비교기는 두 PM의 합에서 작은 값의 정보인 decision bit들을 SM에 출력하고 작은 값을 가지는 PM을 선택하는 MUX의 입력으로 사용한다. 그리고 한 상태에서 천이하는 두 개의 PM과 BM의 합이 동시에 overflow가 발생할 경우에 해당 PM을 rescale0에서 "1111111"로 값을 입력하고 다음 클럭 입력동안 rescale1에서 전체 PM에 대하여 1/2로 나누어 rescale함으로써 overflow에 대해 보상해 주었다. 위에서 rescale를 두 부분으로 나눈 것은 이 부분이 가장 시간 지연이 크기 때문이다. ACS 전체 블록의 구성은 32개인 ACS 연산기부외에 "000000"에서 "011111" 상태 중에 4단계씩 8개의 상태에서 최소 PM을 가지는 상태를 찾는 부분이 있다. 이 상태 값은 SM에서의 traceback 과정에서 MUX의 출력을 선택하는 시작 벡터로 사용한다.

3.3 SM(Survivor Memory)부

SM부에서는 ACS부에서의 출력인 decision bit들을 메모리에 저장(write)하고, 일정 길이를 역추적(traceback)한 후에 전송된 데이터를 출력(decoding)한다.[4][6]

표 1은 역추적 알고리즘에 대한 메모리의 양을 나타내고 있다. 여기에서 Γ는 역추적길이, K는 제한길이를 나타낸다. 표 1에서 알 수 있듯이 k값에 따라 많은 양의 메모리나 특별한 구조의 메모리가 필요하게 되고 지연이 발생하게 된다.[6] 역추적길이(Γ)는 보통 제한 길이의 5,6배 크기로 구성되나, 본 논문에는 depuntring을 지원하기 위해 Γ=96의 크기로 설계하였다.[7] 이와 같은 긴 역추적길이 가지는 SM은 많은 메모리 양이 필요하다. 따라서 본 논문에서는 ACS에서 출력되는 decision bit를 선행 역추적[8]을 수행하여 이를 SM에 저장하는 방식을 채택하여 메모리의 양을 줄이고, 이에 적합한 SM 구조로 설계하였다. 그림 8은 2비트 역추적 과정을 보여주고 있다. 그림 9는 그림 8로부터 고안된 2비트 선행 역추적기이다.

표 1.열추적 알고리즘과 메모리양의 비교[6]

역추적 알고리즘	필요한 메모리 크기
k-Pointer Even Algorithm	$\frac{2k\Gamma}{k-1} \times 2^{K-1}$
k-Pointer Odd Algorithm	$\frac{(2k-1)\Gamma}{k-1} \times 2^{K-1}$
One-Pointer Algorithm	$\frac{(k+1)\Gamma}{k-1} \times 2^{K-1}$

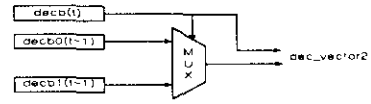

그림 9. 2비트 선행 역추적기

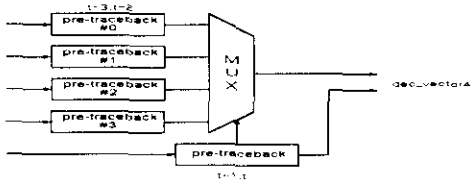

그림 10. 4비트 선행 역추적기

본 논문에서는 그림 10과 같은 4비트 선행 역추적기 이용하여 역추적 속도를 가속하여 메모리의 양을 줄일 수 있었다. 그림 11은 SM의 구조를 나타내고 있다. 선행 역추적에 의해 한 클럭에 4개를 역추적하므로 Γ=96을 모두 역추적하기 위해서 걸리는 시간은 24클럭이 필요하다. 이는 하나의 메모리 블록의 크기를 24, 즉 선행 역추적을 하므로 6×4로 구성할 수 있다. 표 2는 임의의 시간 t 이후의 각 주기(24클럭)별로 메모리의 동작을 보

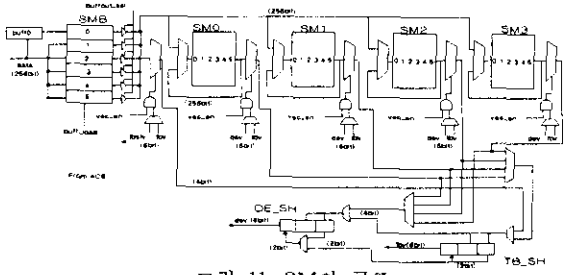

그림 11. SM의 구조

표 2. 설계된 SM의 동작 순서

(TB: traceback, W: write, DC: decoding)

주기	SMB	SM0	SM1	SM2	SM3
t	TB0/W	TB1	TB2	TB3	W/DC
t+1	TB0/W	TB2	TB3	DC/W	TB1
t+2	TB0/W	TB3	DC/W	TB1	TB2
t+3	TB0/W	DC/W	TB1	TB2	TB3
t+4	TB0/W	TB1	TB2	TB3	W/DC
t+5	TB0/W	TB2	TB3	DC/W	TB1

여 준다. SMB의 Write동작은 24클럭동안 4클럭마다 6번 이루어진다. 그 외의 동작은 모두 6클럭 동안 각 메모리 블럭에서 이루어 진다. SMB에서는 ACS부에서 역추적 시작 벡터를 받아 역추적를 시작하고, 4클럭 마다 선행 역추적 된 256(64×4)비트가 입력된다. 두 동작은 시간적으로 독립되어 이루어지나 다만 처음 입력되는 데이터는 역추적시 충돌하므로 이를 임시 저장하여 SMB의 역추적이 끝나면 이를 SMB0에 저장한다. 역추적은 TB_SH의 내용을 변경하면서 디코딩되는 SM블럭을 제외하고 순차적으로 진행된다. 이는 4개의 메모리 블럭을 지나므로 96(=(4×6)×4)의 역추적이 이루어진다. 다음으로 디코딩은 TB_SH의 내용을 DE_SH에 저장하고 6클럭 동안 이루어진다. 이때 디코딩으로 생기는 공백을 역추적되는 SMB의 데이터로 채우게 된다. 이와 같은 동작은 24클럭 마다 역추적되는 메모리 블럭과 디코딩 되는 메모리 블럭을 변경하면서 연속적으로 이루어진다.

4. 실험 및 고찰

본 논문에서 설계한 Viterbi 디코더는 VHDL를 이용하여 모델링하고 CAD tool은 V-system을 사용하여 기능을 시뮬레이션 하였고, Compass 0.8um CMOS 공정으로 합성하였다. 시뮬레이션에 사용된 데이터는 "000","110",...등 연속된 데이터를 사용하였다. 이는 3비트 연성판정을 수행하기 때문이다.

BMG을 최소 BM의 값으로 출력하도록 구성하여 PM의 overflow 수를 최소화하였다. 이 것은 PM의 값는 단지 BM의 연속적인 합에 의해 이루어지므로 BM 값이 작을수록 overflow의 수가 줄어든다. 그리고 BMG는 ROM으로 구성되어 196게이트로 이루어진다. ACS부는 총 17530개의 게이트로 이루어진다. 이는 연산기와 7×64비트의 메모리로 구성된다. 4비트 선행역추적을 하여 역추적 속도를 4배로 향상시켜 SM부에는 24×64비트인 메모리 5개와 4×64비트의 메모리(1.29×Γ×64:Γ=96)로 출일 수 있었다. 보통 표 1이나 다른 경우[9]는 2×96×64비트 이상의 면적을 가진다. k=3일 경우에 even algorithm은 3×96×64비트, odd agorithm은 2.5×96×64비트, one point algorithm은 2×96×64비트[6], 참고문헌[9]는 4×96×64비트의 RAM필요하다. 그림 11과 같이 크기가 큰 64×4 MUX와 256×2 MUX가 9개 필요하다. 여기에 선행 역추적기는 중복되는 상태의 MUX를 제거하여 2745개의 게이트로 이루어진다.

5. 결론

본 논문에서의 viterbi decoder는 제한 길이 K=7, R=1/2로 고속의 데이터 처리를 위해 완전병렬 구조로 설계하였다. BMG는 기존의 출력 값에서 최소값을 이용하여 이 값들을 빼주어 PM의 증가를 최소화하였다. 그리고 SM에서는 4 비트 선행 역추적기를 사용하여 RAM의 크기를 줄일 수 있었다.

향후 상용화 칩에서 지원하는 depuncturing의 추가에 대한 연구가 진행 중이다.

참고문헌

[1]. Bernaed Skalar,"Digital communications Fundamentals and Applications", Prentice hall international edition.

[2]. Heller, J.L.,and jacobs, I.W., "Viterbi Decoding for Satellite and Space Communication" IEEE trans. Com-mun. Technol. ,COM19, no.5,October 1971.

[3]. g.Forney, "Convolutional Codes II: Maximum Likelihood Decoding." Inform. Theory. vol.25, July 1974

[4].Gennady Feygin and P.G. Gulak P.Chow "A VLSI Imolementation of a Cascade Viterbi Decoder With Traceback" Department of Electrical Engineering University of Toronto, Canada July 31,1996

[5]Alexandre Giulietti."A 250 kb/s,K=7,3-bit soft decision programable code rate customized viterbi decoder." Laboratory of Microelectronics University of Sao Paulo

[6]Gennaday Feygin and P.G Gulak, "Architectural tradeoffs for Survivor Sequence Memory Management in Viterbi Decoders" IEEE transactions on Communications, VOL 41, NO. 3, MARCH 1993

[7] MC92300 Viterbi Decoder for Digital TV, Motorola semiconductor Technical data

[8]. Montse Boo,F. rancisco Arguello, Javier D. Bruguera, Ramon Doallo, "High-performance VLSI Architecture for the Viterbi Algorithm" IEEE Transaction on Communications VOL.45, NO. 8, February 1997

[9]한정일,최종문,최우영,김봉열 "실시간 처리가 가능한 구조로 구현한 Viterbi 디코더의 설계" 1997년도 대한 전자공학회 하계종합학술대회 논문집 제 20권 제 1호 p679~682