역추적 예견 알고리즘을 적용한 파이프라인 비터비 복호기의 효율적인 Polling 구조 제시

유기수*, 송오영*
*중앙대학교 전자전기공학부
e-mail:song@jupiter.cie.cau.ac.kr

Efficient Polling Structure for Pipeline Viterbi Decoder Using Backtrace Prediction Algorithm

Ki-Soo You*, Ohyoung Song*
*School of Electrical & Electronic Engineering, Chung-Ang University

요 약

본 논문은 역추적 예견 알고리즘을 사용한 비터비 복호기에서의 TB단의 Polling 구조의 단순화 방법을 제시한다. 비터비 복호기의 3대 Unit중 하나인 Trace Back에서 역추적 예견 알고리즘을 사용할 경우 복호화 시점에서의 최소 State Metric 값을 찾아야 하는 번거로움을 줄일 수 있다. 하지만 복호 신호의 신뢰도 분산에 따라 Polling Unit 이 추가되어야 함에 따라 실제 하드웨어 복잡도에서의 이득은 미미한 것으로 알려져 있다. 제시된 구조에서는 Polling Unit을 단순화 할 수 있는 방법을 적용하였다. 기존 하드웨어와의 비교 평가를 위하여 IEEE802.11a의 표준에 따른 부호화율 1/2, 구속장 7을 갖는 비터비 디코더에 대하여 역추적 예견 알고리즘과 파이프라인 구조만을 갖는 경우와 제안된 단순화한 Polling Unit을 적용한 구조와의 비교에서 Trace Back Unit에서 약 45%의 감소 효과를 보였다.

I. 서 론

길쌈 부호기와 비터비 복호기는 데이터의 전송 시에 발생하는 채널상의 에러를 자동으로 수정해주기 위하여 통신 시스템의 채널 코덱에 활용되는 기술로서 무선 통신, 위성을 이용한 통신 등의 고속 멀티미디어 통신시대로 변화하고 있는 요즘 많이 사용되어지고 있다.

길쌈부호(Convolutional Code)의 부호기는 일반적으로 Shift Register를 연결하여 만든 Sequential회로로 구성된다. 또한 복호기는 임의의 블록에 대한 관찰 없이 임의의 시점에서의 최적의 경로로 입력을 복호화한다. 길쌈부호의 복호에는 Sequential Decoding과 Viterbi Decoding 두가지 방법이 있으나 Sequential Decoding 방법은 Sub-Optimal한 방법이므로 주로 Viterbi Decoding 방법을 이용하여 복호화 한다. Viterbi 복호 이론은 부호기의 플립플롭에 저장되어있는 Data를 State로 간주하여 State간의 천이 개념으로 Trellis Diagram을 이용한 복호 개념이다.

디지털 통신의 발전에 따라 최신의 비터비 복호기는 보다 많은 양의 데이터 전송을 위하여 고속화를 요구받고 있으며 이에 따라 파이프라인 구조를 적용하여야 한다. 하지만 파이프라인 구조는 많은 메모리를 필요로 하고 있으며 또한 비터비의 3개의 주요 Unit인 Branch Metric Unit, Add-Compare-Select Unit, Trace Back Unit의 세 Unit의 소요 시간의 균형을 유지하기 위하여 많은 Combinational Logic을

필요로 하게되었다. 본 논문에서는 역추적 예견 알고리듬을 적용한 Trace Back Unit의 설계에서 Polling 구조를 단순화하여 Trace Back Unit의 Combinational Logic의 크기를 약 45% 가량 줄이게 되었다.

II. Register-Exchange Algorithm과 Trace Back Algorithm

ACS에서의 Survival Path를 저장하는 대표적인 두 가지 방법으로 Register-Exchange 방법과 Trace Back 방법을 들 수 있다.[6]

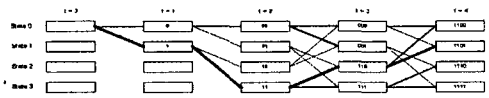

그림 1 Register Exchange 방식의 Trace Back

[그림 1]은 Register-Exchange 방법을 사용하여 Trace Back을 하는 예이다. Register-Exchange Approach 는 지난 Time Frame의 Survival Path 정보를 Initial State부터 현재 State 까지 Shift 시키면서 유지해오는 방법이다.

[그림 2]는 Trace Based 방법을 사용하여 Trace Back을 하는 예이다.

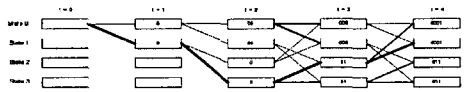

그림 2 Trace Based 방법의 Trace Back

Trace Based Approach 는 State를 기준 주소로 하는 Register에 Information을 저장하는 방법으로서 해당주소지의 Register 값을 보면 해당 State의 Survival Path 변화를 볼 수 있다. Decoding 작업이 진행될 수록 새로운 Survival Path의 Information이 이미 존재하고 있는 Information 뒤에 추가되게 된다. 본 논문에서는 Butterfly를 응용하여 Time Frame의 변경시의 Data Copy부의 복잡도를 줄이도록 하였다. 또한 Register-Exchange의 특성을 분석하여 Survival Path Register의 해당 State를 보고 Decoding을 하지 않고 확률론적 관점에서 Decoded Data를 모든 State의 Survival Path Register에서 결정하는 방법인 Register Exchange Approach의 Trace Back 방식을 사용하였다.

III. Register-Exchange Algorithm을 적용한 Viterbi 복호기의 기존 구조

Trace Back 부에서 할 일은 ACS에서 발생한 Survival Path Information을 해당 Survival Path Register에 Update하고 일정한 길이의 Trace Back Window Length 후의 값들을 이용하여 최종 Decoded의 출력을 정하는 일이다. Register Exchange 방법을 기본으로 하여 매 Time Slot마다 Survival Path Register의 값을 Copy해 나가도록 하고 있으며 이에 대한 Combinational Logic의 복잡도를 줄이기 위하여 ACS에서의 Butterfly 개념을 적용하여 단순화하였다. 또한 구속장 길이의 4배 이상인 32 bit의 Trace Back Window Size를 가지도록 하였다.

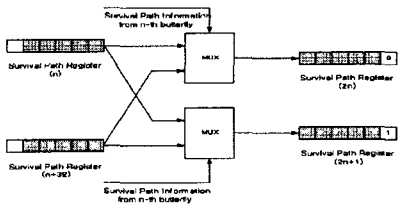

그림 3 Butterfly 구조를 응용한 TB Update Unit

[그림 3]는 위에서 설명한 Butterfly 개념을 적용한 Track Back Unit 이다. 먼저 Survival Path Register는 State의 개수와 동일한 64개(0번지 - 63번지)를 가지고 있으며 매 번지수에는 Trace Back Window Size인 32bit을 가지게 된다. Survival Path Register의 Update시에는 이전 State의 Survival Path Register값으로 0-30 Bit을 Copy 하고 짝수번째 State에는 최 상단 Bit에 '0'을, 홀수번째 State에는 최 상단 Bit에 '1'을 추가해주게 된다.

Decoded Bit은 Trace Back Window Size 만큼의 Time Slot이 지난 후부터 출력되며 이는 Survival Path Register의 LSB의 64개의 Bit중 '0'의 개수와 '1'의 개수를 Count하여 결정하게 된다.

IV. 제안된 Polling Unit을 추가한 구조

기존의 Trace Back 방법에서 State Metric의 최소값을 찾아서 이에 해당하는 State의 값을 Decoding 신호로 정하는 방법은 최소의 State Metric을 찾기 위하여 64개의 State Metric 값을 비교평가 하여야 하기에 이를 위하여 총 63개의 8Bit 비교기가 추가되어 지며 이는 비터비 디코더의 Trace Back Unit에서의 커다란 부담이 된다. 또한

[그림 2]에서의 64개의 Input에서 0과 1중 Majority를 Polling하는 방법은 기존의 State Metric의 최소값을 찾는 방법보다 약 3dB의 이득을 준다[2]. 하지만 이 방법에서도 64개의 Input으로부터 1, 0의 Majority를 찾는 작업을 위하여 64 Input의 1 counter를 구현해야 하는 부담이 있으며 이또한 Trace Back Unit의 구현에 있어서 심각한 부담으로 존재한다. 이를 위하여 64개의 Metric중 임의의 n개를 선택하여 선택한 정보 내에서만 Decoding을 하게되는 방법을 취한다면 Polling Unit을 최소한으로 줄이면서도 동일한 성능의 Decoder를 구현할 수 있다. Polling Unit은 구현 방식에 따라서 64개의 Full Counter를 사용할 수도 있고 부분만을 선택하여 Count하게되는 로직을 구현할 수도 있다. 우선 64 Full Counter와 부분적인 선택형 Counter간의 성능의 차이는 Software Simulator 상에서 다음과 같이 계산되었다.

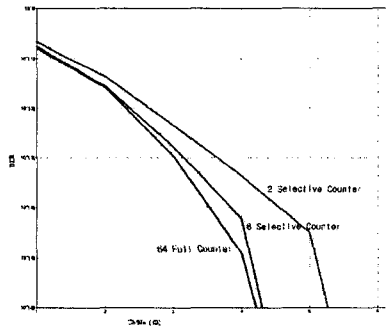

그림 4 Counter별 BER 성능 비교 그래프

[그림 4]에서 보듯이 임의의 2개의 Bit을 선정하여 Trace Back을 하였을 경우와 64 Bit을 모두 계수하여 Trace Back을 하였을 경우는 약 1dB가량의 차이를 내었다. 차지하는 Chip의 Area 와 성능과의 비교 판단에서 8Bit Selective Counting 방법이 가장 효율적인 것으로 판단되었다. 또한 이는 AWGN 채널에서의 측정치이다.

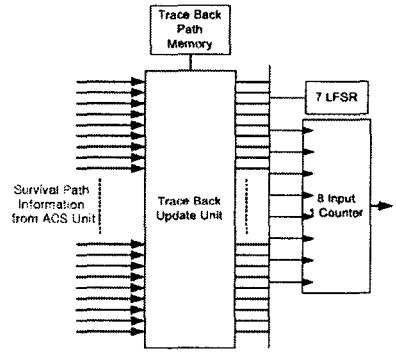

그림 5 단순화된 Polling Unit을 적용한 Trace Back Unit

새로운 Trace Back Unit은 [그림 5]에서 보는 것과 같이 64 개의 Trace Back Data 중에 8개의 임의의 선택을 위하여 LFSR을 이용한 Pseudo Random Selector를 사용하였고 나머지 8 Bit은 Counter를 사용하여 1의 갯수를 계수하였다. 8 Bit Counter의 경우는 일반적인 ripple carry adder를 사용하여 구현하였다. 이는 Viterbi Decoder 전체 구성 중 Critial Path가 ACS Unit에 있으므로 TB부에서는 충분한 여유시간이 존재하므로 크기를 줄이는데 집중하기 위함이다.

V. 하드웨어 크기 비교

기존의 구조와 새로 제안된 Polling Unit을 적용한 구조 사이의 하드웨어 크기 비교를 위하여 다음과 같은 공통된 전제사항을 마련한 후 크기를 비교하였다. 비교 대상의 두 하드웨어는 모두 IEEE 802.11a에서 표준으로 제정되어있는 부호화율 1/2, 과 구속장 7의 비터비 복호기를 사용하였으며 Trace Back Window의 크기는 공히 구속장의 약 4 내지 5 배인 32로 하였다. [표 1]은 비터비 복호기의 Trace Back Unit에 소요되는 Trace Back Update Unit과 Polling Unit의 설계에 소요되는 Comparator의 수를 비교한 것이다. 기존의 63개의 1Bit Comparator의 숫자가 7개로 줄어들었음을 알 수 있다.

구분	일반적 비터비 복호기	제안된 비터비 복호기
Trace Back Update Unit	32개	32개
Trace Back Unit의 polling Unit에 소요되는 Comparator의 수	63개	7개
Polling 입력 선택용 LFSR의 Register 수	-	7개

표 1 역추적 예견 알고리듬을 적용한 기존의 Trace Back Unit 크기와 제안된 방식의 Trace Back Unit비교

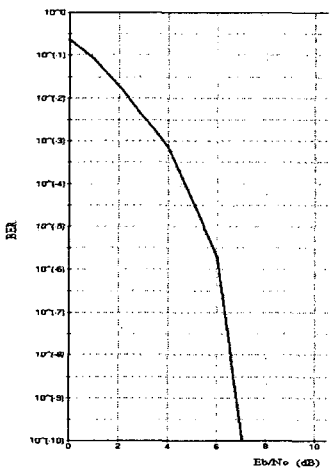

그림 6 구현된 Viterbi Decoder의 BER 특성곡선

또한 [그림 6]는 구현된 비터비 복호기의 BER 특성을 보여준다. 이는 하드웨어와 동일한 구조의 Software Simulator를 통해서 얻어진 결과이다. 이전의 역추적 예견 알고리듬이 적용된 기존의 비터비 복호기의 성능과 동일함을 알 수 있다.

Ⅵ. 결 론

본 논문은 일반적인 비터비 디코더에서 사용되는 방법인 TB Unit에서의 상태값 저장과 ACS Unit에서의 최소 State Metric 계산을 통한 Trace Back Memory에서의 최종 복호값 추적이 아닌 각 상태에서의 Survival Path의 값을 천이시켜 이의 Polling을 통하여 최종 복호값을 구하는 Register-Exchange Algorithm에 그 기반을 두고 있다.[6]

TB로 전해지는 Survival Path값을 메모리에 저장시키지 않고 Pipeline 의 특징을 이용하여 천이시키면 추가로 TB Memory 가 필요치 않게된다. 또한 각 State 별 Survival Path의 값을 Polling을 통하여 복호할 경우 추가적인 3dB의 성능향상을 가져오는데 반해 Polling Unit의 크기로 인하여 구현상 부담이 증가하게 된다. 이런 Polling Unit을 단순화하여 기존 Viterbi Decoder의 성능을 유지하는 동시에 구현상 문제되지 않는 크기로 Polling Unit을 단순화하였으며 이를 통하여 Trace Back Unit에서만 약 45%의 Combinational Logic의 크기를 감소시켜 하드웨어 복잡도를 줄이게 되었다.

Ⅶ. 참고문헌

[1] Bernard Sklar, "Digital Communications Fundamental and Application," Prentice-Hall, 1988.

[2] Peter J. Black, "A 140-MB/s, 32-State, Radix-4 Viterbi Decoder," IEEE Jounal of Solid-State Circuit, 1992.

[3] Shu Lin, Marc Fossrier, "A Diferential Viterbi Algorithm for Decoding Rate 1/n Doubly Complementary Convolutional Codes and their High-Rate Punctured Codes Based on Compare-Select-Add Principle," ISIT, pp. 365, 1998.

[4] S. Lin, D. J. Costello, Jr, "Error Control Coding : Fundamentals and Applications," Prentice Hall, 1983.

[5] S. B. Wicker, "Error Control Systems for Digital Communication and Storage," Prentice Hall, 1995.

[6] Shu Lin, Marc Fossrier, "Trellises and New Trellis-Based Decoding Algorithms for Codes," ITW, pp. 79-80, 1998.

[7] 장대익, 김대영, "연속 및 버스트모드 통신을 위한 길쌈부호기와 비터비복호기 ASIC 설계," 전자공학회지, pp. 984-995, 1996.

[8] 전북대학교, "CDMA용 Viterbi 복호기의 최적 구조 제시 및 FPGA 구현에 관한 연구," 최종 연구 보고서, 1995.

[9] 진익수, "IMT 2000을 위한 채널 코덱 설계," 한국 전자통신연구원 ASIC 지원센터, 2000.