1997 年度 大韓電子工學會 夏季綜合學術大會 論文集 第 20 卷 第 1 號 97/6

실시간 처리가 가능한 구조로 구현한 Viterbi 디코더의 설계

한정일, 최종문, 최우영, 김봉열
연세대학교 전자공학과
120-749 서울특별시 서대문구 신촌동 134

Design of A Viterbi Decoder for Real-Time Processing Capability

Jung-Il Han, Jong-Moon Choi, Woo-Young Choi, Bong-Ryul Kim
Dept. of Electronic Engineering, Yonsei University
134, Shinchon-Dong, Sudaemoon-Ku, Seoul 120-749, Korea

요약

본 논문에서는 DBS, CDMA등의 이동통신 시스템에 사용되는 Viterbi 디코더를 제시하였다. 설계된 Viterbi 디코더는 제한길이 7, 부호율이 1/2인 콘벌루션 인코딩에 대한 채널 디코딩을 수행한다. Viterbi 디코더의 에러정정 수행시 실시간으로 역추적이 가능한 구조로 설계하였다. 또한 VHDL 및 게이트로 설계, 검증 및 레이아웃을 하여 향후 에러 정정 기능을 사용하는 주문형 반도체에 사용이 용이하도록 하였다.

1. 서론

통신 시스템에 있어서 데이터의 전송시 기상의 변화, 비선형 감쇄, 간섭 요인 등의 통신 채널 환경에 따른 비트 오류 발생으로부터 데이터를 보호할 필요가 있다. 이와 같은 시스템에서 신뢰성 있는 데이터 복호를 위하여 강력한 오류정정부호 기술인 Viterbi 알고리즘을 채택하고 있다[1-5]. 최근 초고집적회로 기술의 급속한 발전은 이러한 알고리즘의 하드웨어 구현을 가능하게 하고 있다[7].

현재 미국에서는 디지털 이동통신 방식의 하나로 CDMA 이동통신 표준안이 검토 중에 있고 차세대 개인휴대통신(PCS : Personal Communication Service) 및 무선 데이터 통신망에도 CDMA의 사용이 제안 또는 검토되고 있다.

Viterbi 디코더는 콘벌루션 코드의 최대유사발견(maximum-likelihood detection)을 보장하는 Viterbi 알고리즘을 구현한 하드웨어로서 위성통신 분야의 CDMA chip set, Set-top Box의 에러정정부분, HDTV 신호 전송부분의 모듈 등에서 에러 정정을 수행하는데 널리 사용되고 있다.

Viterbi 디코더가 에러를 정정할 때 어느 시점에서나 가능하도록 실시간 처리가 되어야 실제 응용에 적용 가능하다[6]. 본 논문에서는 이러한 기능이 가능하도록 하드웨어 구조 및 논리회로를 설계하였다. 먼저 Viterbi 알고리즘에 대해 살펴본 후, Viterbi 디코더의 설계, 시뮬레이션 결과 및 레이아웃 구현에 대하여 기술하였다.

2. Viterbi 알고리즘

(1) 에러정정 인코딩

통신을 할 때, 잡음이 있는 채널을 통해 데이터가 전달되기 때문에 전달받은 데이터에 오류가 생길 확률이 존재한다. 이때 오류를 발견하여 올바른 데이터로 정정하도록 하기 위해 에러정정 코딩이 필요하게 된다.

에러정정 코딩은 메모리를 필요로 하지 않는 블럭

코딩(block coding)과 메모리를 필요로 하는 콘벌루션 코딩(convolution coding)으로 나눌 수 있다[2]. 블럭 코딩은 통신하고자 하는 정보를 k개 비트의 블럭으로 나누어 적절한 변환을 통해 n개의 비트로 표현되는 코드워드(codeword)로 매핑(mapping)시키는 코딩 방법으로 n개의 비트로 표현되는 인코더의 출력은 k개의 입력 메시지에만 의존하기 때문에 조합 회로(combinational circuit)로 구현될 수 있다. 콘벌루션 코딩은 k 비트의 입력 데이터를 받아들여 n 비트의 출력을 내는 인코딩 방법으로 m 비트의 과거 데이터와 함께 코딩하므로 순차 회로(sequential circuit)로 구현된다. 이때 인코딩 과정에서 사용된 전체 비트수를 제한길이(constraint length)라 하며 K로 나타낸다[1,2]. 입력 데이터의 크기가 1인 경우 레지스터의 수는 (K-1)이 된다.

(2) Viterbi 알고리즘

Viterbi 알고리즘은 에러정정 코딩 기법 중 콘벌루션 인코딩된 디지털 정보에 대한 디코딩 방법으로, 메모리가 없는 가우시안 채널을 통한 통신에서 주로 사용되는 방식이다. Viterbi 알고리즘은 트렐리스도(trellis diagram)에서 최단 거리를 찾는 알고리즘으로 콘벌루션 코드의 최대유사 디코딩을 효율적으로 수행하는 알고리즘이다[1-5].

Viterbi 알고리즘에서 사용되는 디코딩 방식은 동적 프로그래밍 기법으로 트렐리스도에서 각 상태에 들어오는 입력값에 대해 그 시간까지의 패스 메트릭과 각 시간에서의 브랜치 메트릭 값을 더해서 작은 값을 그 상태의 패스 메트릭으로 결정하는 알고리즘으로 생존자가 되는 패스는 일정한 시간이 지난 후에 결정되어 이를 디코딩 결과로 출력하는 디코딩 방식이다. 알고리즘에서 사용하는 브랜치 메트릭은 주어진 시간에서 현재의 입력과 인코더에 의해 인코딩된 기준값과의 차이를 구하여 결정된다. Viterbi 알고리즘은 트렐리스의 각 노드(node)에 해당하는 상태들이 전 시간축까지의 패스 메트릭을 저장하고 있고 새로 들어온 브랜치 메트릭과 인코더 트렐리스의 상태로부터 얻어진 브랜치 메트릭을 더해 작은 값을 갖는 패스 메트릭을 선택한다. 일정한 시간이 지난 뒤 디코더는 각 노드의 패스 메트릭 중에서 가장 작은 값을 갖는 것을 선택한다. 이때 선택된 패스 메트릭을 생존자 패스라 한다. 선택된 패스는 현재 시간축에서 다시 거슬러 올라가서, 즉 역추적(trace back) 과정을 거쳐 원래의 데이터로 디코딩된다.

3. 실시간 처리를 위한 Viterbi 디코더의 구현

본 논문에서는 코딩율이 1/2이고, 제한길이가 7인 콘벌루션 인코더를 포함하는 Viterbi 디코더를 설계하였다. 콘벌루션 인코더는 연결 벡터로 표시하였을 때, G1 = 171(octal), G2 = 133(octal)인 출력을 갖도록 구현하였다. 인코딩된 데이터를 디코딩하기 위해 3비트 연판정(soft decision) 방식을 사용하였다. 이 방식을 사용하면 경판정(hard decision) 방식보다 약 2 dB 정도의 프로세싱 이득을 얻을 수 있다. 인코딩된 데이터 G1과 G2는 디코더 클럭에 동기되어 입력 데이터로 들어온다. 그림 1에 Viterbi 디코더의 전체적인 구조를 나타내었다.

Viterbi 디코딩을 수행하기 위해 우선 모든 입력 데이터에 대한 유클리디언 거리를 언급하고자 한다. 유클리디언 거리(Euclidean distance)는 양자화된 두 신호의 차이를 나타내며 각 가중치(weight)별로 모듈로 2 연산을 행한 것과 같다. 유클리디언 거리를 계산하는 기능 구현은 게이트로 구성된 로직으로 할 수 있으나, 게이트 수 및 각 게이트 간의 연결로 인한 하드웨어의 증가로 칩 크기가 커지는 단점이 있어 ROM을 사용하여 하드웨어의 오버헤드를 줄이고자 하였다. vit_rom에 이러한 정보가 저장되어 있다. 현시점의 각 상태별 총 유클리디언 거리값에 새로 계산된 유클리디언 거리를 더하여 각 상태마다 2개의 패스에서 들어온 총 유클리디언 값을 구하고 이중에서 작은 패스 메트릭을 갖는 패스를 선택하는 vit_acs 블럭이 이어진다. 마지막으로 이 패스를 패스 메모리 블럭에 기억시키고, 역추적할 시점이 되면 패스 메모리에 들어 있는 상태 정보를 추적하여 디코딩된 데이터를 내보내는 tr_logic 블럭이 있다.

vit_acs 블럭은 add, compare, select 기능을 수행한다. vit_acs 블럭은 acs_cal 블럭 32개와 lmt_cntl 블럭 1개가 모여 이루어져 있는데, 각각의 출력은 상태값인 경우 저장을 하고, 각각의 상태에서 패스 메트릭 값은 다른 acs_cal 블럭의 입력으로 들어가는 구조를 하고 있다.

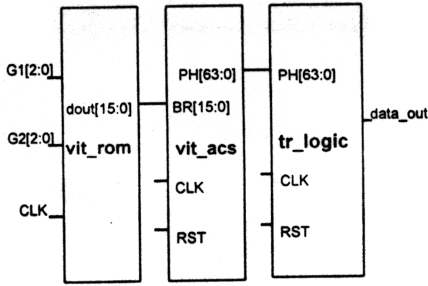

그림 1. Viterbi 디코더의 블럭 다이어그램

Forney에 의하면, 제한 길이의 약 5~6배의 역추적 깊이(trace-back depth)를 가질 때 모든 생존자 패스는 한 곳으로 수렴함을 나타내었다[4,5,8,9]. 본 논문에서는 이 성질을 이용하여 역추적 깊이를 128로 하여 생존자 패스를 찾아 내도록 설계하였다. 역추적 깊이를 128로 한 이유는 향후 코딩율을 높이기 위한 punctured code를 사용하는 시스템에서 depuncturing할 때 필요하기 때문이다.

tr_logic 블럭은 RAM 4개를 제어하여 처음 디코딩된 데이터가 나올 때까지의 지연 시간을 제외하고는 중단없이 계속 데이터가 나올 수 있도록 한다. 즉, 데이터의 실시간 처리를 구현하는 블럭이다[11].

그림 2는 tr_logic 블럭 다이어그램을 나타낸다. 전체적으로 tr_logic의 동작을 나타내면 다음과 같다. t0에서는 RAMA에 vit_acs의 출력인 64개의 상태값을 쓴다. t1에서는 RAMB에 상태값을 계속하여 쓴다. t2에서는 RAMC에 상태값을 쓰고 RAMB에 쓰여져 있는 값을 읽어낸다. t3에서는 RAMD에 상태값을 쓰고 RAMA의 데이터를 읽어 tr_back 블럭에서 디코딩한 다음 디코딩한 데이터를 LIFO(Last In First Out)의 기능을 갖는 TB_RAM0에 쓴다. 동시에 RAMC의 데이터를 읽어 RAMB을 디코딩하기 위한 수렴값을 구한다. t4에서는 RAMA에 새로이 데이터를 쓰고, RAMB의 데이터를 읽어 tr_back 블럭에서 디코딩한 다음 디코딩한 데이터를 TB_RAM1에 쓴다. 이와 동시에 RAMC를 디코딩하기 위한 수렴값을 구하기 위해 RAMD의 데이터를 읽어 내려가고 TB_RAM0에 저장되어 있던 데이터를 읽어 들인다. t4에서 디코딩된 데이터가 전체 Viterbi 디코더의 출력으로 나오게 된다. t5에서는 RAMB에 데이터를 쓰고 RAMC의 데이터를 읽어 tr_back 블럭에서 디코딩한 다음 디코딩한 데이터를 TB_RAM0에 쓴다. 동시에 RAMD을 디코딩하기 위한 수렴값을 구하기 위해 RAMA의 데이터를 읽어 들이고 TB_RAM1에 저장되어 있던 디코딩된 데이터를 출력한다. t6에서는 동일한 과정을 거쳐 RAMD의 데이터를 디코딩하여 TB_RAM1에 쓰고, TB_RAM0에 저장되어 있던 데이터를 출력한다. 이후 t3에서 t6까지의 과정을 반복하게 된다. 한편, t4부터 디코딩된 데이터가 연속적으로 나오게 되어 실시간 처리가 이루어진다.

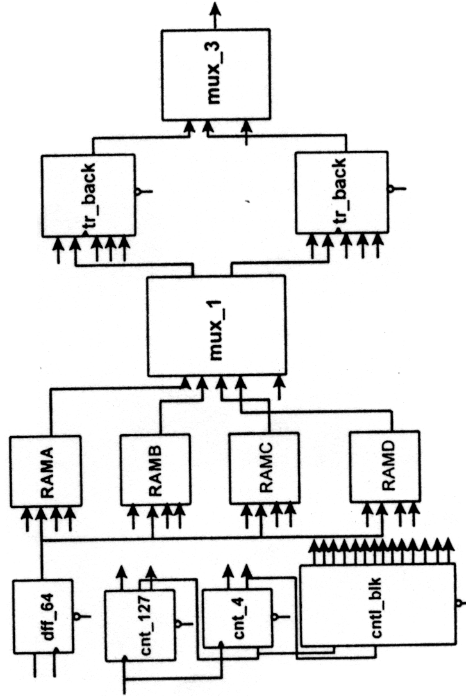

그림 2. tr_logic 블럭 다이어그램

4. 시뮬레이션 및 레이아웃 구현

전체적인 회로 설계는 이미 언급되었는데, 이를 VHDL로 기술하여 동작을 확인하였다. 회로 동작을 확인하기 위하여 콘벌루션 인코더에 입력으로 '1', '1', '0', '0', '1'을 반복하여 넣어주었으며 각각에 대한 인코더의 출력을 얻을 수 있었다. 전송한 데이터에 에러가 있을 때 설계한 Viterbi 디코더가 에러정정 기능을 수행하는 것을 보이기 위해 두 번째 전송 데이터에 에러를 가하였다. 시뮬레이션을 수행한 결과 예상대로 에러가 정정된 데이터를 얻을 수 있었다.

실제적인 칩을 설계하기 위하여 0.6 ㎛ CBIC(Cell-Based IC)의 3.3 V용 라이브러리를 사용하

여 게이트 레벨 시뮬레이션을 수행하였다. 이를 위하여 VHDL 소스 코드로부터 합성을 하였고 합성된 회로로 시뮬레이션을 수행하였다. 게이트 레벨 시뮬레이션의 입력은 VHDL 시뮬레이션에서 사용한 테스트 벤치(test bench)의 입력과 동일한 벡터를 사용하였다.

본 논문에서 제안된 Viterbi 디코더는 0.6 ㎛ CMOS 표준셀 기법과 이층 금속배선(2-layer metal) 기술로 설계되었다. 이것은 15,887 게이트의 논리 회로와 128 ×64 비트의 RAM 4개, 128×1 비트의 RAM 2개, 64 ×16 비트의 ROM 1개로 이루어져 있다. 동작 전압은 3.3 V이고 동작 주파수는 20 MHz이며 이때 예상되는 소모전력은 약 120 ㎽이다. 또한, 본딩 패드를 포함한 칩 크기는 4 ㎜ × 4 ㎜이다. 그림 3은 설계된 Viterbi 디코더의 레이아웃 결과를 보여준다.

5. 결론

본 논문에서는 실제 상용화를 위한 Viterbi 디코더를 설계하였으며 데이터의 오류정정을 실시간에 처리되도록 하는 구조로 설계하였다. 칩 설계를 위하여 전체적인 설계를 마친 다음 VHDL로 기술하여 이것의 시뮬레이션 결과로 동작을 확인하였고 이를 합성(synthesis)하여 게이트 레벨의 회로를 얻어냈다. 게이트 레벨의 시뮬레이션을 수행하여 동작을 확인한 다음 이로부터 레이아웃을 수행하였다.

이것은 상용화를 위해 실시간 처리에 주안점을 두었고 또한 설계를 VHDL 및 논리회로로 구성하였기 때문에 향후 여러 에러정정 기능이 필요한 PCS, FPLMTS, DBS, Set-top Box 분야 주문형 반도체의 메가 라이브러리로 사용이 용이할 것이다.

참고문헌

[1] A. J. Viterbi, "Convolutional Codes and their Performance in Communication Systems," IEEE Trans. Commun., Vol. 19, pp. 751-772, Oct. 1971.

[2] Bernard Sklar, "Digital communications fundamentals and applications", Prentice-Hall, 1989.

[3] A. J. Viterbi, "Error Bounds for Convolutional Codes and Asymptotically Optimum Decoding Algorithm," IEEE Trans. Information Theory, Vol. IT-13, pp. 260-269, April 1967.

[4] G. D. Forney, "The Viterbi Algorithm," Proc. IEEE, Vol. 61, pp. 268-278, Mar. 1973.

[5] G. D. Forney, "Convolutional Codes II : Maximum Likelihood Decoding," Inform. Theory, Vol. 25, pp. 222-226, July 1974.

[6] C. Y. Chang and K. Yao, "Systolic Array Processing of the Viterbi Algorithm," IEEE Trans. Information Theory, Vol. 35, No. 1, pp. 76-86, Jan. 1989.

[7] G. Fettweis, H. Meyr, "A 100 Mbit/s Viterbi Decoder chip : Novel Architecture and its realization," IEEE International Conference on Communications, Atlanta, No. 307, 4, pp. 463-467, April 1990.

[8] T. K. Truong, "A VLSI Design for a Trace-Back Viterbi Decoder," IEEE Transactions on Communications, Vol. 40, No. 3, March 1993.

[9] C. M. Radar, "Memory Management in a Viterbi Algorithm," IEEE Transactions on Communications, pp. 1399-1401, Sep. 1981.

[10] Jens Sparso, "An Area-Efficient Topology for VLSI Implementation of Viterbi Decoders and other Shuffle-Exchange Type Structures", IEEE Journal of Solid-State Circuits, Vol. 26, No. 2, Feb. 1991.

[11] G. Feygin, P. G. Gulak, "Architectural Tradeoffs for Survivor Sequence Memory Management in Viterbi Decoders", IEEE Trans. on Commun., pp. 425-429, Vol. 41, No. 3, Mar. 1993

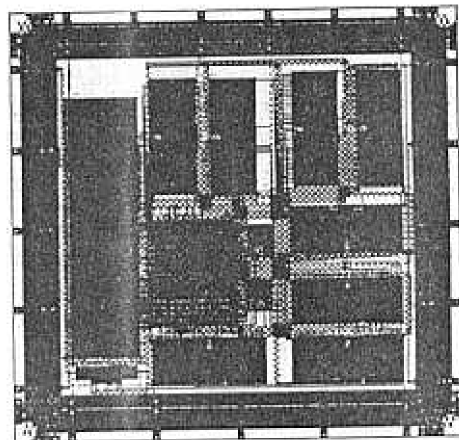

그림 3. 설계된 Viterbi 디코더의 레이아웃