

역추적 방식의 시스톱릭 어레이 구조를 가진 연판정 비터비 복호기의 설계

김기보, 김종태

성균관대학교 전기전자 및 컴퓨터 공학부 컴퓨터 시스템 합성연구실

VLSI Design of Soft Decision Viterbi Decoder Using Systolic Array Architecture

Ki Bo Kim, Jong Tae Kim

School of Electrical and Computer Engineering, SungKyunKwan University

Abstract - Convolutional coding with Viterbi decoding is known as a powerful method for forward error correction among many kinds of channel coding methods. This paper presents a soft decision Viterbi decoder which has systolic array trace-back architecture[1]. Soft decision is known as more effective method than hard decision and most of digital communication systems use soft decision. The advantage of using a systolic array decoder is that the trace-back operation can be accomplished continuously in an array of registers in a pipe-line fashion, instead of waiting for the entire trace-back procedure to be completed at each iteration. Therefore it may be suitable for faster communication system.

We described operations of each module of the decoder and showed results of the logic synthesis and functional simulation.

1. 서 론

채널 코딩 방식의 하나인 길쌈부호의 복호 방법 중 비교적 오류 정정 능력이 우수한 것으로 알려진 비터비 복호기는 현재의 이동통신 시스템에 널리 사용되는 복호 방식이다. 비터비 복호기는 좀더 나은 복호 성능과 고속 동작을 위하여 다양한 구조가 제시되어 있다. 본 논문에서는 최근 이동 통신 시스템에서 채용하고 있는 연판정법과 Systolic array 구조의 역추적 방식을 결합한 비터비 복호기를 제안한다.

본문에서는 먼저 연판정법과 Systolic array 구조의 특징에 대해 서술을 한 다음 제안된 복호기의 구조에 대해 설명을 한 후 논리 합성과 검증결과를 제시하였다.

2. 본 론

2.1 비터비 복호기

길쌈부호의 복호기에 사용되는 기억소자의 수를 m 이라 하면 부호기의 상태 수는 2^m 이 된다. 이 상태는 부호기에 입력되는 정보원에 따라서 순차적으로 변화하게 되고 부호기로부터 출력되는 부호어도 달라진다. 이 부호어는 연속적으로 통신채널을 통해 전송되는데 이 전송과정에서 전송되는 부호어 열에는 오류성분이 포함되게 된다. 복호기에서는 이 오류를 포함하는 부호어 열에서부터 부호기의 상태 전이를 유추한 다음 이 정보로부터 부호어를 알아내게 된다.

부호기의 상태가 변화되는 순서를 나타내기 위해서는 복잡한 가지구조가 필요하다. 즉 시간변수 T_1 일 때 2가지였던 부호기의 상태변화 순서는 T_2 일 때는 4가지 T_3 일 때는 8가지로 늘어나게 되는데 이 복잡한 가지구조는 일정하게 반복되는 중복부분을 가지고 있다. 이 중복부분을 간단화하여 시간과 상태의 표현으로 나타낸 것을

격자선도라고 한다. 길쌈 부호의 복호를 위해서는 이 격자선도에서 부호기의 상태전이 순서를 알아내야 하는데 이를 위한 방법으로 여러 가지가 제안되고 있다. 그 중에서 최우복호법에 의한 비터비 알고리즘은 가능한 탐색 경로를 짧고 간단하게 하기 때문에 필요한 연산량을 크게 감소시킬 수 있어 길쌈부호의 복호 알고리즘으로 많이 사용되고 있다.

발생가능성이 가장 높은 경로를 선택하는 방법에는 여러 가지가 있을 수 있는데, 크게 경판정법과 연판정법으로 구분할 수 있다. 경판정법의 경우, 복호기의 수신단에서 수신신호를 단지 '0'과 '1'로 양분하여 부호어로 인식한다. 이렇게 수신한 부호열들과 가장 작은 해밍거리를 갖는 부호어열을 만들어 내는 격자선도 내의 경로를 최우복호 경로로 선택한다.[2]

반면, 연판정법에서는 수신된 신호를 신호의 품질에 따라서 여러 단계로 세분한다. 복호기에는 여러 단계로 구분된 신호가 각각 '0'이 될 확률과 '1'이 될 확률 값을 저장하고 있다. 해밍거리를 구하는 경판정법과는 달리 연판정법은 수신된 신호들의 전송중 오류발생 확률값을 비교하여 최우복호 경로를 결정하므로, 복호성능이 경판정법에 비해 우수한 것으로 알려져 있다. 보통 8단계(3bit)의 연판정인 경우 경판정에 의한 방법보다 약 2dB의 성능 향상이 있어 대부분의 디지털 통신 시스템에서 3, 4 비트의 연판정을 주로 사용하고 있다.[3]

이렇게 최우 복호경로가 결정되면 이 경로에 대해 복호를 수행하게 되는데, 복호방법에는 register-exchange 법과 trace-back법이 있다[1]. 본 논문에서는 trace-back 방식의 복호법을 변형한 systolic array 구조를 갖는 역추적 방법을 선택하여 복호기를 구성하였다.

Trace-back 방식의 복호법은 하나의 부호어를 얻기 위해서는 매번 복호깊이(decoding depth, L)만큼의 클럭이 필요하다. 하지만, systolic array 구조의 역추적법은 trace-back 방식보다 메모리 소자를 2배 이상 필요로 한다는 단점이 있으나, 매 클럭마다 하나의 부호어를 얻을 수 있다. 따라서 현재의 고속 통신 환경에 적합한 방식이라고 할 수 있다. 본 논문에서는 이상의 연판정 복호법과 systolic array 구조를 결합한 비터비 복호기를 제안한다.

2.2 복호기의 구조

표 1과 같이 구속장과 부호율 그리고 연판정 비트수 등 설계 사양이 다른 비터비 복호기를 본 논문에서 제안된 구조로 설계하였다.[4]

그림 1에는 구속장 5, 부호율 1/2인 4비트 연판정 비터비 복호기의 블록 구조도를 나타내었다. 그림 1과 같이 복호기는 크게 연판정법으로 경로를 선택하는 경로 결정부와 역추적 과정을 통해서 복호를 행하는 역추적부(TBU)로 나눌 수 있다.

| 구속장 (K) | 부호율 (R) | 생성 다항식 | 연관정 비트수 | 역추적 깊이(L) |
|---------|---------|-------------------------|---------|-----------|
| 3 | 1/2 | (5,7) ₈ | 4 | 10 |
| 3 | 1/3 | (5,7,7) ₈ | 3 | 10 |
| 4 | 1/2 | (15,17) ₈ | 3 | 15 |
| 4 | 1/3 | (13,15,17) ₈ | 4 | 15 |
| 5 | 1/2 | (23,35) ₈ | 4 | 20 |
| 5 | 1/3 | (25,33,37) ₈ | 4 | 20 |

표 1 비터비 복호기의 설계사양

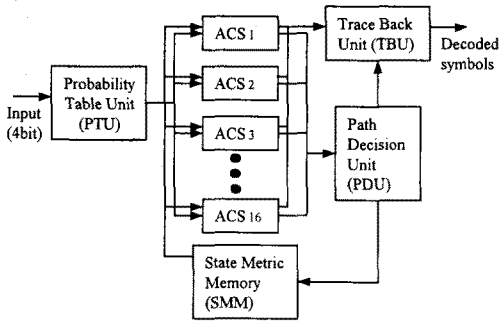


그림 1 복호기의 구조도

2.2.1 경로결정부

경로결정부는 확률값저장부(Probability Table Unit, PTU)와 ACS부, SMM부(State Metric Memory), 경로선택부(Path Decision Unit, PDU)등으로 구성되어 있다. 확률값 저장부에는 수신한 신호의 품질에 따라 각 단계별로 확률값이 저장되어 있다. 연관정 비트수가 4bit이면 수신 신호는 4bit의 16단계로 나눌 수 있고 각 단계에 해당하는 신호가 각각 '0'이 될 확률과 '1'이 될 확률값을 저장해야 하므로 16×2 가지의 확률값을 저장하고 있어야 한다.

본 논문에서는 전송 채널에서 10%의 오류 발생가능성이 있다고 가정하고 이때에 해당되는 확률값을 표준 정규 분포표에서 산출해내어 확률값저장부의 데이터로 사용하였다. 확률 연산시 사용되는 곱의 연산대신 합이 연산으로 간단화하기 위해서 확률값에 상용log를 취한 다음 소수점의 위치를 적절히 변경시켜서 설계시 하드웨어가 복잡해지지 않도록 간단화 하였다. 확률값은 30 - 500사이의 값을 갖는 10bit 정수형으로 되어 있다.

수신된 신호는 이 확률값 저장부를 거쳐 확률값으로 변환되고 변환된 값들은 ACS부의 입력으로 들어간다.

ACS부는 격자선도의 일정 상태로 천이 가능한 두 경로에 대해서 SMM에서 받은 10bit의 상태값(State metric)값과 확률값저장부에서 받은 10bit의 가지값(Branch metric)을 각각 더하여 새로운 상태값을 구한다. 이 중에서 작은 상태값을 갖는 경로를 선택한 후 이 선택된 경로의 상태값을 SMM부로 전달하고 복호에 필요한 추적정보를 역추적부(TBM)에 기억시켜 놓는다. 그림 1의 블럭도에서 예시한 (5, 1/2) 길쌈부호는 16가지 상태가 가능하므로 16개의 ACS부가 필요하다. 격자선도의 각 상태마다 한 개의 ACS부를 할당해 놓은 완전병렬형 ACS구조를 사용하므로서 복호속도를 빠르게 해줄 수 있다.

SMM부는 각각 10bit로 구성된 기억소자로서 ACS부에서 받은 각 경로별 상태값을 저장한다. SMM도 ACS부와 마찬가지로 격자선도의 각 상태에 대응해야 하기

때문에 격자선도의 상태개수와 같은 수의 SMM이 필요하다.

SMM은 다른 모든 기억소자들 처럼 reset신호와 함께 초기화 되는데 상태0에 해당되는 SMM은 0으로 초기화 되고, 이외의 SMM은 0이상의 일정 값으로 초기화된다. 이렇게 하므로써 상태0의 상태값이 가장 작게되어 격자선도의 경로가 항상 상태 0에서부터 시작하게 된다. 상태값은 지로값을 계속 누적한 값인데 고정된 메모리에 무한정 하게 상태값을 누적시킬 수 없으므로 SMM의 모든 상태값들 중 가장 작은 값으로 다른 모든 상태값을 빼주는 정규화를 해주어야 한다. SMM부는 이 정규화를 한 뒤 다음차례로 수신한 부호어의 ACS 연산을 위해 정규화된 상태값을 ACS부로 전달한다.

선택부는 ACS부로부터 받은 상태값들 중 가장 작은 값을 선택해서 복호경로를 결정하고, 결정된 복호 경로를 TBM에 전달한다. 여기서 결정된 최소 상태값은 SMM에서의 정규화를 위해서도 사용된다.

이상의 경로 결정부에서 선택된 최우복호경로의 상태천이 정보와 추적정보는 역추적부의 메모리 열에 차례로 저장된다.

2.2.2 역추적부

역추적부(TBU)는 앞단의 경로 결정부에서 결정된 경로를 따라 역추적하여 복호어를 구한다. 그림 2는 앞서 예시한 (5, 1/2) 길쌈 부호기의 역추적부의 블록 구조도 이다.

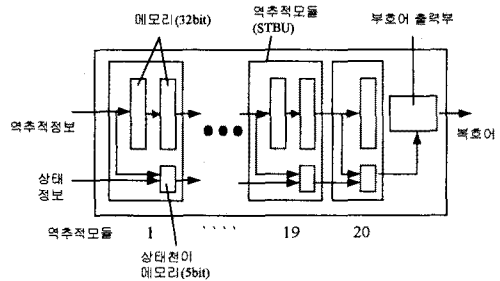


그림 2 역추적부의 구조도

Systolic array 구조의 역추적부는 그림 2와 같이 직렬 연결된 역추적 모듈(STBU)들로 구성되어 있다.

이 역추적 모듈들은 파이프라인 형태로 된 레지스터의 배열로 구성되어 있다. 이 역추적 모듈의 연결 개수는 역추적 깊이와 같은데 이 역추적 깊이는 오류의 정정능력과 하드웨어 구현시의 효율을 동시에 고려하여 결정되어야 한다. 역추적 깊이는 보통 부호기 레지스터수의 다섯배 크기가 많이 사용되고 있다. 구속장이 5인 경우 역추적 깊이를 20으로 하기 때문에 그림 2와 같이 20개의 역추적 모듈이 직렬로 연결된다.

한 개의 역추적 모듈은 2개의 메모리 소자의 1개의 상태천이 메모리로 구성되어 있다. 구속장이 5인 길쌈부호의 경우 격자선도의 상태가 총 16가지이므로 추적정보를 저장할 메모리 소자도 16비트로 되어 있어야 하고 상태천이 메모리는 16가지 격자선도 상태를 모두 표현할 수 있도록 4비트로 구성되어야 한다. 다른 역추적 모듈들과 달리 마지막 20번째 모듈은 더 이상 추적정보를 전달할 필요가 없기 때문에 16bit 메모리 소자를 1개만 사용하였다.

역추적부로 입력되는 최초의 상태정보는 복호 시작 후 역추적깊이(L) 만큼의 클럭이 지난 후에 입력되고 두 번째 이후의 상태정보는 매 클럭마다 입력된다. 상태정보가 첫 번째 역추적 모듈에 입력된 후 다시 L 만큼의 클럭이 지나면 입력된 상태정보로부터 복호를 한 후 복호어를 출력하게 된다. 그래서 복호 시작 후 최초상태정보

가 복호시까지 걸리는 시간인 2L 클럭의 초기지연시간이 지나면 매 클럭마다 복호어를 연속적으로 얻을 수 있는 것이다. 리셋신호와 함께 역추적부의 모든 기억소자들은 '0'으로 초기화된다. 따라서 최초의 복호어가 출력되기 전까지의 출력 값은 모두 '0'이다.

역추적깊이가 L일 때 역추적부의 전체 메모리 소자는 $(2L - 1)$ 개가 필요하고 상태전이 메모리는 총 L개가 필요하다. 메모리 소자는 상태전이 메모리 보다 대략 2배 정도 더 필요한데 이는 상태정보가 역추적 정보보다 L클럭 만큼 더 늦게 출발하게 되면서 생기는 시간차를 만회하게 하기 위해서이다.

역추적부의 마지막 블록인 부호어출력부는 상태전이 메모리 중 필요한 복호어만 선택해서 출력한다.

2.3 논리합성 및 검증결과

표 2는 본 논문에서 제안된 구조를 갖는 비터비 디코더를 표 1에서 제시한 사양으로 논리 합성한 결과이다. 논리합성을 위해서 SynopsysTM의 Design Compiler를 사용하였으며 Altera Flex 10K 라이브러리를 사용하였다.

| 구속장 (K) | 부호율 (R) | 생성 다항식 | 연판정 비트수 | 블럭수 | | 게이트수 |
|---------|---------|-------------------------|---------|-----|------|-------|
| | | | | ACS | STBU | |
| 3 | 1/2 | (5,7) ₈ | 4 | 4 | 10 | 12000 |
| 3 | 1/3 | (5,7,7) ₈ | 3 | 4 | 10 | 12500 |
| 4 | 1/2 | (15,17) ₈ | 3 | 8 | 15 | 24300 |
| 4 | 1/3 | (13,15,17) ₈ | 4 | 8 | 15 | 28500 |
| 5 | 1/2 | (23,35) ₈ | 4 | 16 | 20 | 60200 |
| 5 | 1/3 | (25,33,37) ₈ | 4 | 16 | 20 | 61600 |

표 2 설계된 비터비 복호기의 논리합성결과

연판정방식은 경판정방식의 복호기보다 블록사이에 전달해야할 정보량이 많기 때문에 일반적으로 크기가 크고 복잡하다. 그림 3은 구속장 3, 부호율 1/2인 4bit 연판정 비터비 복호기의 합성결과를 나타내었다.

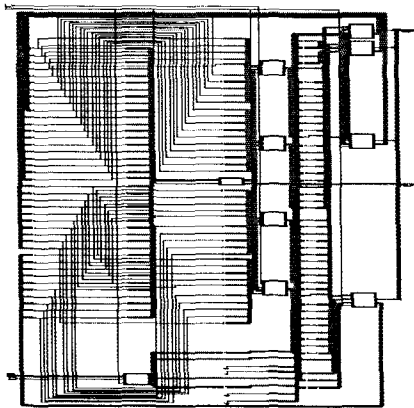


그림 3 논리 합성한 결과

본 논문에서 구현한 복호기의 검증을 위해서 function level 시뮬레이션을 수행하였다. 본 시뮬레이션을 위해서 SynopsysTM의 VHDL System Simulator (VSS)를 사용하였으며 Sun ultra sparc 60 환경에서 실행하였다. 그림 4는 구속장 3, 부호율 1/2인 4비트 연판

정 비터비 복호기를 시뮬레이션한 결과이다.

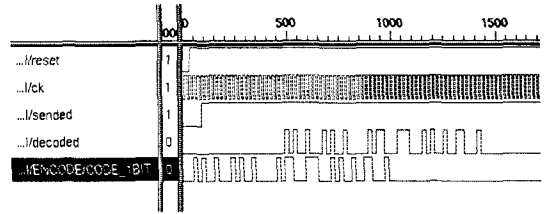


그림 4 시뮬레이션 결과

3. 결 론

본 연구에서는 이동통신 시스템에서 사용되는 비터비 복호기의 향상된 구조를 제안하였다.

제안된 비터비 복호기는 연판정방식을 사용하여 같은 구속장 크기를 갖는 경판정법에 비해 높은 오류정정능력을 갖으며 systolic array 구조를 한 역추적방식을 사용하여 고속 복호가 가능하다.

제안된 구조의 복호기는 구속장이 커질수록 역추적정보를 저장할 메모리 량과 메모리에서 소비되는 전력량이 문제가 된다. 이러한 문제를 효율적으로 개선하기 위해서 앞으로 보다 많은 연구가 진행되어야 할 것이다.

(참 고 문 헌)

- [1] T.K. Truong, M.T. Shih, I.S. Reed, E.H. Satorius, "A VLSI Design for a Trace-Back Viterbi Decoder", IEEE Trans. Comm., vol. 40, No. 3 pp. 616-624, March 1992.
- [2] Rorabaugh, C. Britton, "Error coding cookbook : practical C/C++ routines and recipes for error detection and correction" McGraw Hill, 1996.
- [3] M.S. Kong, S.I. Bae, J.S. Kim, "Automated Design of Viterbi Decoder using Specification Parameters" Journal of IEK, Vol 36-c, No 1. pp Jan. pp. 1-10. 1999.
- [4] Man Young Rhee, "Error-Correction Coding Theory" Hee-JungDang pp. 277-321. Sept. 1984.