논문 00-25-4A-11

한국통신학회논문지 '00-4 Vol.25 No.4A

Radix-4 방식의 64-state Viterbi 복호기 구조 설계 및 구현

정회원 정 지 원*, 김 진 호**, 김 명 섭**, 오 덕 길**

A Design and Implementation of 64-state Viterbi Decoder with Radix-4 Method

Ji-Won Jung*, Jin-Ho Kim**, Myung-Sup Kim**, Duk-Gil Oh** *Regular Members*

요 약

본 논문에서는 Radix-4 방식의 64-state, R=1/2, 3비트 연판정 Viterbi 복호기를 설계하고 FLEX10K CPLD 칩으로 제작하였다. Viterbi 복호기 동작을 고속화하기 위해 Viterbi 복호기를 구성하고 있는 모듈인 ACS, BMU, TB 구조를 제시하였다. 실제 ASIC 설계시, CPLD 칩으로 제작한 것 보다 6~7 배의 속도를 빨리할 수 있으므로 본 연구에서 제시한 40Mb/s급 Viterbi 복호기 구조는 200Mb/s급 무선멀티미디어통신에서 적용할 수 있다.

ABSTRACT

A 40-Mb/s, 64-state, R=1/2, 3 bit soft decision Viterbi decoder based on Radix-4 method has been designed and fabricated using a FLEX10K CPLD chip in this paper. In order to implement the high-speed Viterbi decoder , the architectures of adder-compare-select(ACS), branch metric calculation(BMC), trace back(TB) are present. In practical designed by ASIC, the speed is faster than that of CPLD by 6~7 times. Therefore, 40 Mb/s Viterbi decoder architecture can be used for high-speed wireless multimedia communications with 200 Mb/s.

Ⅰ. 서 론

무선통신시스템은 무선채널의 특성으로 비트오류가 발생하기 쉬우며 이를 정정하기 위해 사용되는 채널부호는 무선통신시스템에서 매우 중요한 요소기술이다. 위성통신 및 이동통신 등에서 사용되는 채널부호는 일반적으로 연판정이 가능한 길쌈부호를 적용한다. PCS 및 이동통신 시스템에서 사용되고 있는 길쌈부호는 서비스대상인 데이터 및 원천 부호화된 음성의 속도가 수~수십[kbps] 급으로 다소 저속이므로, 길쌈부호의 부호기 및 복호기의 구현에 있어서 동작속도 보다는 칩셋의 소모전력과 소요면적을 줄이는 것이 더욱 중요하였다[1]. 그러나 기술의 발전에 따라 고도의 통신서비스를 제공하기 위해 전송하여야 할 데이터 양이 급격히 증가할 전망이며, 무선으로 멀티미디어를 제공하는 것은 망의 유연성, 신속성, 특히 이동성으로 발전성 등으로 인한 이용자가 요구하는 편리성 측면에서 큰 이점을 가진다. 따라서 현재의 무선 통신 시스템은 고속 데이터 전송 및 동영상 등을 포함한 무선 멀티미디어 전송에 기반을 두고 있고 또한 2000년대의 정보통신은 가입자들에게 더욱더 다양한 서비스를 제공하기 위해 방송과 통신의 융합, 지상망 및 위성망을 통합한 혼합망, 혼합망의 근간을 이룰 무선 ATM 전송, 무선 CATV(LMDS)등의 다양한 형태로 발전해 나갈 것이 예상되므로, 고속 데이터 전송에 따른 고속 부/복호방식의 개발이 절실히 요구된다. 이러

* 한국해양대학교 전파공학과 위성통신 연구실(jwjung@hanara.kmaritime.ac.kr)
** 한국전자통신연구원 무선방송기술연구소 초고속위성통신연구팀
논문번호 : 99362-0908, 접수일자 : 1999년 9월 8일

한 요구에 따라 현재, 국내외적으로 Viterbi 복호기를 고속으로 구현하고자 하는 연구가 활발히 진행되고 있다[2].

길쌈부호의 복호기인 Viterbi 복호기는 비터비 복호기의 입력 비트인 연판정된 복조기 출력 신호와 비터비 복호기의 트렐리스 상의 부호화 비트와 XOR를 함으로써, 수신신호와 부호화 비트간의 해밍거리를 구하는 부분인 BMU(Branch Metric Unit)와 한 상태에 입력되는 두개의 경로중 누적된 해밍거리가 작은 경로를 선택하는 ACSU(Adder Compare Select Unit), ACSU에서 선택된 경로의 해밍거리 누적값을 저장하는 PMU(Path Metric Unit), ACSU에서 선택된 경로 정보를 이용하여 역추적하여 데이터를 복호하는 TBU(Trace Back Unit)등의 모듈로 구성되고 있다.

고속으로 구현하기 위해서는 각 모듈을 고속으로 구현할 수 있는 알고리즘을 제시해야 하며, 또한 1비트를 복호하기 위해서 필요한 클럭수를 최소화 시키는 구조로 변경할 필요가 있다.

따라서 본 논문에서는 길쌈부호의 복호기인 Viterbi 복호기를 고속으로 구현할 수 있는 방안을 주요 모듈별로 검토하고, 도출하여 설계하고 Altera Tool을 이용하여 CPLD로 Viterbi 복호기를 구현한 결과를 제시한다.

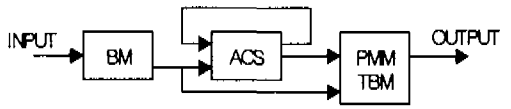
Ⅱ. 고속화를 위한 Viterbi 복호기 모듈 분석

Viterbi 복호기는 그림 1과 같이 크게 3부분으로 나눌 수 있다. BM에서 입력신호와 Trellis diagram의 Branch word와의 비트차 즉, 해밍거리를 구하고 ACS에서 과거 해밍거리와 더한다음, 가장 작은 해밍거리 누적치를 갖는 생존경로를 선택한다. 이렇게 선택된 생존경로들을 PMM에 저장하고 TBM에서 PMM에 저장된 생존경로를 역추적 하여 복호한다.

1. BM(Branch Metric) unit

Viterbi 복호기의 트렐리스상의 부호화 비트인 BCW(Branch Coded Word) 값과 Viterbi 복호기의 입력 비트인 연판정된 복조기 출력 신호 $R_k=(r_{k1}, r_{k0})$를 XOR함으로써, 수신신호와 부호화 비트간의 해밍거리를 구하는 부분이다.

그림 2는 BMU를 나타내고 있다. 수신신호가 3비트 연성판정 되었으므로 수신신호와 BCW (Branch Coded Word) 와의 해밍거리를 구하기 위하여 cw_0 및 cw_1을 각각 3비트로 표현하여 BCW를



BM (Branch Matric)
ASC (Add Compare Select)
PMM (Path Metric Memory) : store path value
TBM(Trace Back Memory) : store survival path

그림 1. Viterbi 복호기의 구조

("000", "000"), ("000","111"), ("111","000"), ("111", "111") 중의 하나가 된다 .예를 들어 수신신호를 ("100", "101")이라 하고 현재 BM을 계산하는 가지의 BCW가 ("111","000") 이라 하면 해밍거리는 |7-4| + |0-5| = 9, "1001"이 된다. 즉 cw_i 가 "000" 일때는 가산기 입력 A_i를 r_M 로 하고, cw_1 가 "111" 일때는 가산기 입력 A_i를 $not(r_M)$ 로 취하여 가산한다. 이 때 가산기의 출력이 BM값에 해당한다. BMU의 출력은 0~14사이의 값을 가지므로 4비트로 표현할 수 있다.

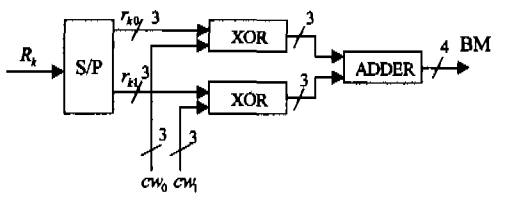


그림 2. BMU의 구조

2. ACS(Adder Compare Select) unit

ACS unit는 Viterbi 복호기가 처리할 수 있는 최대 데이터 속도를 결정하는 가장 중요한 블럭이다. 복조기의 출력신호는 3비트 soft decision 되어 복호기에 입력되는데, BM unit의 출력(PM0,PM1)과 과거 누적된 PM(Path Metric)값인 PMi를 합하여 각 상태 노드에 있는 두 경로의 PM값을 비교하여 PM값이 작은경로를 선택하여, 선택된 PM의 경로정보

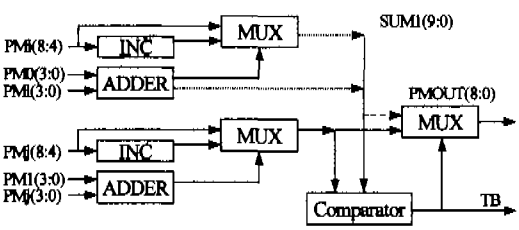


그림 3. ACSU의 구성

를 trace back 복호부에 정보를 준다. ACSU의 구조는 그림 3과 같다.

현재의 BM값과 과거 누적된 PM값을 더하는 과정을 단순한 전가산기를 이용하면 carry 발생으로 인한 시간 지연이 있으므로 본 논문에서는 현재의 PM값을 구하기 위해 다음과 같은 step으로 구현한다.

Step1. PM 비트를 5비트와 4비트로 분할한다.

Step2. 하위 4비트와 BM 4비트를 더하는 동시 에 상위 5비트를 1씩 증가시킨다.

Step3. 하위 4비트 합의 결과가 carry 발생하면 1씩 증가한 상위 5 비트와 멀티플렉싱 (multiplexing) 시킨다. carry가 발생하지 않으면 1씩 증가 시키지 않은 원래의 상 위 5비트와 멀 티플렉싱한다.

Step1~Step3에서 알 수 있듯이 9비트의 PM을 4비트와 5비트로 분할하여 동시에 계산하면 9비트 덧셈보다 고속화로 구현 할 수 있다. 각 노드에서 최소 PM을 선택하는 comparator는 일반적으로 MUX와 뺄셈으로 구성되는데 이는 많은 시간을 요구한다. 따라서 고속화를 위해 Metric rescaling 방식으로 comparator를 구성하였다. comparator의 구조는 그림 4와 같다. Metric rescaling 방식은 뺄셈 대신에 1의 보수를 취하여 더하는 방식이데 더한 결과의 MSB(Most Significant Bit)가 경로정보인 동시에 MUX의 select단자의 입력이다. 경로정보는 ACSU로 입력되는 각 노드의 두개의 상태중에서 위에서 입력되는 상태를 0, 아래서 입력되는 상태를 1 로 선택하여 TBU로 보낸다.

3. PM(Path Metric) unit

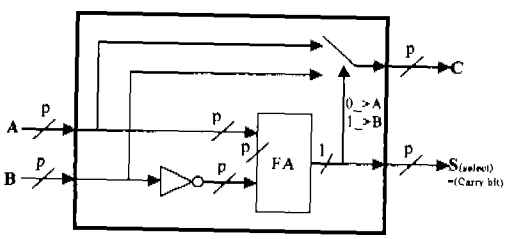
다음 상태의 ACS동작을 위해 현 상태까지의 누적된 해밍거리값을 나타내며 이는 PMM(PM Memory)에 저장한다. PM을 구현하는데 가장 큰 문제점은 한정되어있는 PM비트로 인하여 발생하는 overflow를 어떻게 처리하느냐에 있다. 여기서는 고속화를 위하여 Modulo Arithmetic 방법을 사용한다. Modulo Arithmetic 방법은 overflow를 무시하고 계속 더하는 방법으로 ACSU에 입력되는 각 상태노드에서의 두 PM값간의 차이만 비교할 수 있다면 Viterbi 알고리즘은 올바른 동작을 할 수 있고, 또한 임의의 시간에서 두 PM값의 차이에는 최대값이 존재하고 더 이상 커지지 않는다는 성질을 이용한 것이다. 이러한 성질을 이용하기 위해서는 PM의 비트수인 WL(Word Length) 정확히 규정하여야 한다.

$$WL \geq \lfloor \log_2(2nB) \rfloor + 1 \qquad (1)$$

여기서 $n = K-1$(K는 구속장 수), B는 BM의 상한치이다. 따라서 본 논문에서는 PM에 할당된 비트 수를 $WL \geq \lfloor \log_2(2\times6\times14) \rfloor + 1 = 9$ bits로 할당하였다.

각 상태에서 최소값을 찾을 시, TBU의 생존 경로를 역추적 하기 위한 초기 역방향 포인터인 최소 메트릭 값을 가진 상태를 선택하는 최소 상태 결정 구조 블럭도는 그림 5와 같다.

그림 5는 (2,1,4) 길쌈부호기 예를 들었다. 구속장 수가 4이므로 상태수가 8개가 되며 각 상태에서 PM은 "6","7","4",1","5","3","0","2" 라 가정하고 최소 메트릭을 가지는 상태는 S_6 (110)이라 가정한다. "110" 이라는 상태를 찾기 위해서는 그림 4에서 S단자를 이용하면 쉽게 찾을 수 있으며, S단자의 비트 값을 역순으로 읽으면 최소 PM값을 가지는 상태임을 알 수 있다.



p : 입외의 비트수

그림 4. Cmparator의 구성

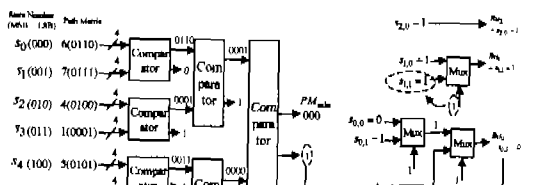


그림 5. 최소 PM값을 갖는 상태 결정구조

4. TB(Trace Back) unit

ACS에 의하여 결정되어진 경로정보가 저장된 메모리(TBM) 이용하여 복호하는 부분이다. TBM의

경로정보를 이용하여 복호하는 방식은 크게 RE(Register)방식과 TB방식으로 구분할 수 있는데, RE방식은 VLSI로 구현할 때 쓰기 동작이 매우 복잡하고 전력소모가 크다는 단점이 있어,대부분의 경우 TB방식을 사용한다. TB동작은 식(2)을 이용하여 역방향으로 수행되며, 이 때 식(2)에 의하여 송신비트를 복호해 낸다.

$$X_{k-1(1)} = X_k(0) = flag \quad d_k = X_k(1) \tag{2}$$

여기서, $X_k(0), X_k(1)$은 각각 t=k인 시점에서 노드상태의 LSB(Least Significant Bit) 및 MSB (Most Significant Bit)를 의미하고 flag는 ACS에서 TBM에 저장한 경로정보이다. d_k는 복호된 비트로 상태정보의 MSB가 복호되는 비트이다. 이러한 TB 동작을 구현하기 위해서는 그림 6에서와 같이 TBM 메모리 구조는 세가지 동작(Read, DC, Write) 을 필요로 한다.

Read : TBM에 기록되어 있는 경로정보를 역방 향 포인터로 해석하여 이전 단계의 상 태를 계산하는 과정이다.

DC(DeCode) : Read 과정과 동일 하지만 읽어낸 비트가 복호된 비트이다.

Write : ACS에서 출력되는 경로정보를 TBM에 쓰는 과정이다.

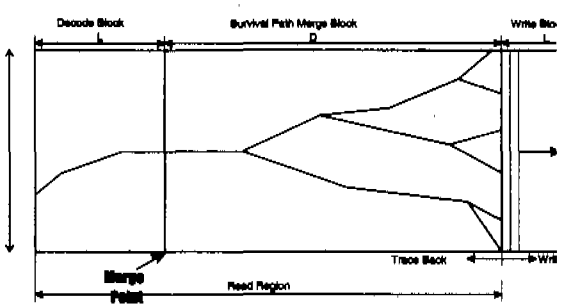


그림 6. TBM 구조 및 동작별 메모리 영역

Viterbi 복호기는 1비트를 복호할 때마다 Decoding_Depth만큼 역추적 해야하기 때문에 구속장이 클수록 Decoding_Depth가 커져, 잦은 Read동작으로 인해 전력소모가 커지게 된다. Viterbi 복호 구조를 보면 역추적 경로가 D컬럼만큼 경로정보를 누적시키고 trace back 하는데, D를 어느 정도 크게 하면 Merge Point가 발생한다. 즉 어느 시점부터 어느 상태에서 trace back하여도 일치(merge)되는 영역인 DC영역이 존재하기 때문에, 이 DC 영역의 비트를 한꺼번에 복호할 수 있다. 이때 D의 최소치는 D_{min} 은 구속장의 4배 또는 5배 정도이다. 따라서 이 성질을 이용하면 TB과정의 오버 헤드를 분산시킬 수 있으며, 복호 속도도 고속으로 구현 가능하다. 오버헤드를 분산시켜 처리하는 알고리즘에는 대표적으로 One Point 알고리즘과 k Point 알고리즘이 있다.

1) One Point 알고리즘

ACSU의 경로정보를 TBM에 쓰는 동안에 하나의 읽기용 포인터를 이용해서 TB 동작과 DC 동작을 수행해야하므로, 읽기클럭이 쓰기클럭보다 수배 빨라야하므로 고속을 요하는 시스템에서는 구현하기 어렵다.

2) k point 알고리즘

복호시 TBM을 여러 개의 bank로 나누어 쓰는 동안 각 bank에서 병렬로 읽기 때문에 쓰기클럭과 동시에 읽기클럭이 수행되므로, 고속처리가 가능한 알고리즘이다. 본 논문에서는 k=3 Point 알고리즘을 적용하여 총 6개의 bank로 구분하여 읽기클럭과 쓰기클럭에 대한 요구속도가 동일하므로 복호과정을 고속화 시켰다. 적용한 bank 구조는 그림 7과 같다.

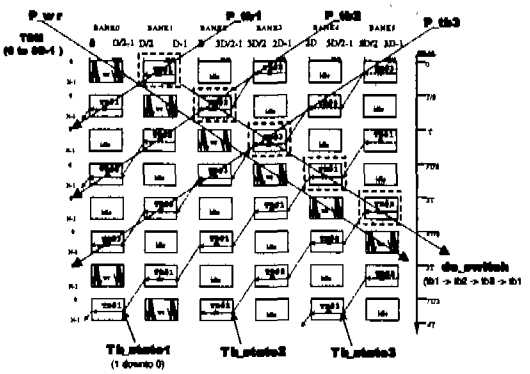


그림 7. 3-Point 방식에서의 메모리 관리 구조

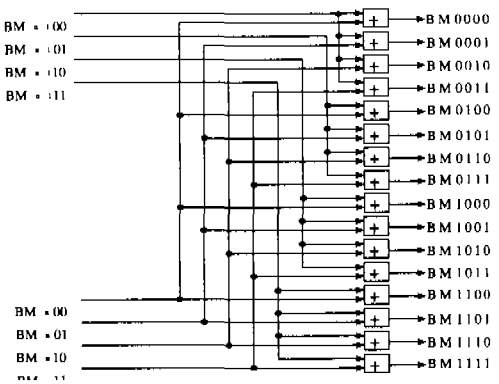
요구되는 메모리 요구량은 bank number와 bank size의 곱이며 요구되는 bank수는 2k이고 bank size는 D/(k-1) 이다. 적용한 메모리 관리 방식에서는 TBM 메모리 번지0~3D-1를 번지 단위의 6개의 bank(idle bank 2개, TB bank 2개, DC bank 1개, WR bank 1개)로 분할하여 관리한다.

5. Radix-4 방식을 이용한 복호방식

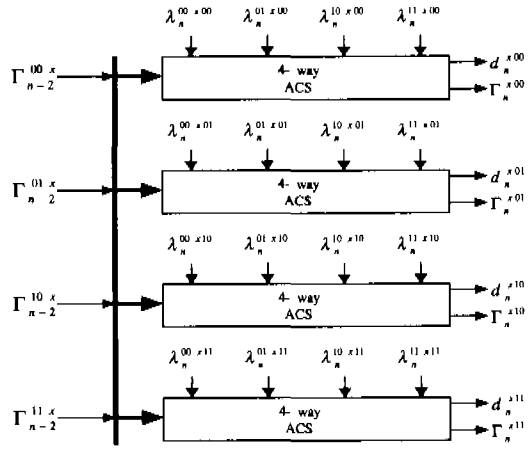
Radix-2 방식은 복호시 trace back할 때, 바로 그 이전의 상태를 trace back하는데 반해 radix-4 방식은 두 상태 이전을 trace back하기 때문에 복호 비트의 출력속도가 radix-2 방식보다 빠르다. Radix-4 방식은 과거 2단의 연판정된 수신 비트를 입력 받아 한꺼번에 BMU, ACSU를 처리하기 때문에 trellis 구조는 Radix-2 방식의 2개 시점을 하나의 시점으로 간주하여 처리하며 복호시에도 2비트를 동시에 복호한다. Radix-4 구조로 바뀐 trellis 상에서 BM을 계산하기 위해서는 두 시점을 합친 BCW를 고려해야 한다. 즉 Radix-2 구조에서 고려되는 BCW는"00", "01","10","11"의 네가지 경우이고 다음 상태에서의 BCW 역시 동일하므로 두 시점을 합친 BCW는 "0000", "0001".....“1111"의 16가지 경우가 존재한다. 따라서 Radix-4 구조에서 BM값을 계산하기 위해서는 그림 8과 같이 임의의 n-1 시점에서 수신되는 연판정된 비트와 Radix-2 구조의 BM값을 구하고 n 시점에서 수신되는 연판정된 비트와 Radix-2 구조의 BM값을 구하여 각각을 서로 더하여 Radix-4 구조의 BM값을 구한다. 각각 구해진 BM값을 ACSU로 분배한다. ACSU는 BMU에서 계산된 BM값을 이전의 PM값과 더하여 새로운 NPM(New PM)값을 저장하고 선택된 경로에 대해 경로 정보를 TBU에 저장하는 부분이다.

Radix-2 구조에서는 경로를 선택하기 위해 비교되는 가지 수는 2개이며, 경로 정보 비트는 1 비트인데 반해 Radix-4 구조에서는 비교되는 가지 수는 4개이며 경로 정보 비트는 2비트가 할당되어야한다.

이상에서 분석한 바와 같이 고속화를 위한 각 모듈별 알고리즘을 기존의 저속 동작으로 구현된 모듈별 알고리즘과 비교하여 표 1에 나타내었다.



(a) Radix-4 BMU 구조



(b) Radix-4 ACS 구조

그림 8. Radix-4방식의 BMU, ACS 구조

표 1. 고속화를 위한 각 모듈 파라미터

복호기 unit	Unit별 알고리즘	기존 저속 알고리즘	고속화를 위한 알고리즘
BM unit	◦ 3bit Full Adder ◦ 3bit CSA	◦ 3bit Full Adder	◦ 3bit CSA
ACS unit	◦ 직렬/병렬연산 ◦ Full Adder ◦ CSA 연산 ◦ Incrementer	◦ 직렬 연산 ◦ Full Adder	◦ 병렬연산 ◦ CSA 알고리즘과 Incrementer를 이용한 고속 ACS 방식
TB unit	◦ REM방식 ◦ TraceBack 방식 - one pointer - k pointer	◦ Trace Back 방식 - one pointer 알고리즘	◦ Trace Back 방식 - k pointer 알고리즘 (k=3)
복호 방식	◦ Radix-2 방식 ◦ Radix-4 방식	◦ Radix-2 방식	◦ Radix-4방식

Ⅲ. 구현 결과

앞에서 언급한 고속화 방안들을 적용하여 고속 (2,1,7)Viterbi 복호기의 스키메틱도를 그림 9에 나타내었다. 본 설계에서는 고속 동작을 위하여 각 상태마다 계산에 할당된 ACSU를 따로 두어 완전 병렬구조로 설계하였으며, PMM은 레지스터를 단순히 hard_wiring하는 구조로 설계하였다. 복호되는 bit는 TBU에서 trace back의 역추적으로 복호되므로 LIFO(Lase In First Out)가 필요로 한다.

Viterbi 복호기의 VHDL(Very high speed intergreted circurt Hardware Description Languige) 코드를 Alter사의 Design compiler를 이용하여 compile하고 합성하였다. Viterbi 복호기는 FLEX

10K70RC240-3 칩으로, TB memory는 외부 RAM을 사용하지 않고 FLEX10 칩 안의 내부 EAB (Embedded Array Block)을 사용하여 구현하였다. 그림 10은 FLEX10K70RC240-3 칩으로 구현되어진 (2,1,7)Viterbi 복호기의 report file이며, 메모리는 약 7만 게이터 기준으로 87%가 차지하였음을 알 수 있다.

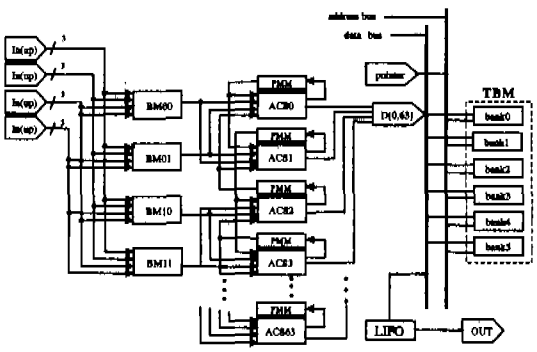


그림 9. 완전 병렬구조의 Viterbi 복호기에 대한 스키메틱도

```
** DEVICE SUMMARY **

Chip/                Input Output Bidir Memory  Memory

LC's Device          Pins  Pins  Pins  Bits    %Utilized LCs %Utilized
EPF10K70RC240-3      4     1     0     16384   88%       3283 87%
User Pins            4     1     0
```

그림 10. FLEX1070RC240-5 report file

(2,1,7)Viterbi 복호기의 타이밍도의 출력은 그림 4-3에 나타냈다. source data는 data port로 입력되고, 복호 되어진 bit는 LIFO를 거쳐서 out_bit로 출력된다. 클럭주기는 약50[ns]이고, 전체 복호지연은 약 33[ns]였다. BM port는 branch metrics를 , p_tb port와 p_wr port는 TB unit에서 읽기 bank와 쓰기 bank의 시작점을, TB_state port는 trace back 과정을 지시한다. 그리고 memory_addr port는 FLEX10K70RC240-3 칩에서 EAB memory의 주소를 지시한다.

그림 12는 본 연구에서 모델 chip을 제작하여 실제 VHDL 코드를 FLEX10K100RC chip으로 download하기 위해 설계한 보드이다. 컴퓨터에 서 VHDL코드를 시뮬레이션하고, 결과를 검증하고 난 후, 시리얼 포트를 이용하여 그림 12 보드로 VHDL 시뮬레이션 결과인 sof 파일을 down-loading하면 chip의 출력포트를 통해 로직 분석기를 이용하여 확인한다. 또는 DAC를 이용해 파형을 확인한다.

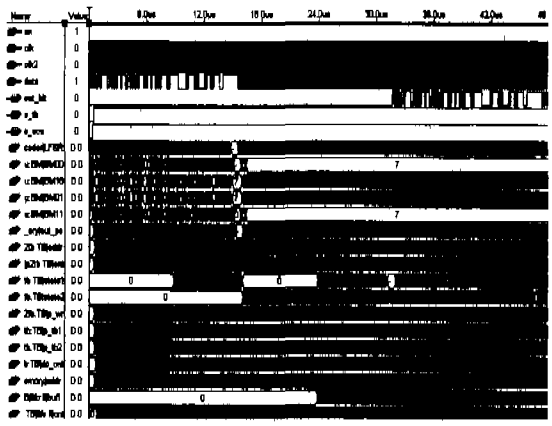


그림 11. (2,1,7)Viterbi 복호기의 타이밍도의 출력

그림 12. (2,1,7) Viterbi 복호기 제작 보드

Ⅲ. 결 론

본 논문에서는 무선 멀티미디어 전송을 위한 고속 Viterbi 복호기를 BMU, ACS, TB 모듈별로 구조설계 하였다. 복호속도를 좌우하는 ACS 구조는 병렬구조로써 comparator회로, multiplexer회로, 5비트 incrementer 회로 적용을, TB unit는 k pointer 알고리즘의 적용을, 전체적인 복호 방법은 Radix-4 알고리즘 적용이 고속화를 구현하는데 이상적인 방법이다.

Altera사의 Design Compiler를 이용하여 FLEX 10K 칩에 합성한 Viterbi 복호기 구현결과 FLE X1070RC240-5 칩으로 구현가능하였으며, 최고 40[Mbps]급 전송속도를 갖고 있다. CPLD 속도보다 ASIC 설계시의 속도가 약 5~6배 이상 고속화가 가능하므로 200Mbps 이상의 속도가 가능한 ASIC 칩을 설계할 수 있으며, 고속 무선멀티미디어통신 시스템의 오류정정부호로 적용될 수 있다

참 고 문 헌

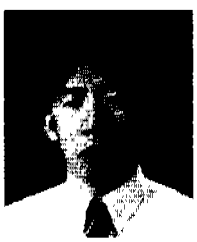
[1] S. Kubota, K. Ohtani, and S. Kato, "High-speed and high-coding-gain Viterbi decoder with low power consumption employing SST scheme," *IEE Electorn. Lett.*, vol.22, no. 9, pp. 491, Apr. 1986.

[2] G. Fettweis and H. Meyr, "High-Speed Viterbi Processor: A Systolic Array Solution," *IEEE Journal on Selected Area in Commun.* , vol SAC-8, pp. 1520-1534, OCT.,1990.

[3] T.K. Truong, Ming-Tang Shih and E.H. Satorius, "A VLSI Design for a Trace-Back Viterbi Decoder" *IEEE Trans. on Commun.*, vol. 40, no. 30, pp. 616-624, March 1992.

[4] P.J. Black and T. H.-Y. Meng, "A 140Mb/s, 32-state, Radix-4 Viterbi Decoder," *IEEE Journal of Solid-State Circuits,* vol. 27, no. 12, pp. 1877-1885, Dec., 1992.

정 지 원(Ji-Won Jung) 정회원



1989년 2월 : 성균관대학교 전자공학 학사
1991년 2월 : 성균관대학교 전자공학 석사
1995년 2월 : 성균관대학교 전자공학 박사
1991년 1월~1992년 2월 : LG 정보통신연구소 연구원
1995년 9월~1996년 8월 : 한국통신 위성통신연구실 선임연구원
1997년 3월~1998년12월 : 한국전자통신연구원 초빙연구원
1998년 9월~현재 : 한국해양대학교 전파공학과 조교수
<주관심 분야> 위성통신, 이동통신, 변.복조기술, 채널코딩, FPGA 기술 등

김 진 호(Jin-Ho Kim) 정회원
1988년 2월 : 경북대학교 전자공학과 학사
1990년 2월 : 한국과학기술원 전기 및 전자공학과 석사
1990년 3월~현재 : ETRI 무선방송연구소 초고속위성통신팀 선임연구원
<주관심 분야> 영상처리, 디지털 위 성방송수신기, Viterbi 복호기

김 명 석(Myung-Suk Kim) 정회원
1986 : 한양대학교 전자공학과 학사
1991 : 충남대학교 전자공학과 석사
1999 : 한국과학기술원 정보및통신공학과 공학박사
1986. 1~현재 : 한국전자통신연구원 초고속위성 통신연구팀 팀장
<주관심 분야> 위성 통신, 이동 통신, 통신 프로토콜

오 덕 길(Duk-Gil Oh) 정회원
1980 : 서울대학교 공과대학 전자공학과 학사
1984 : 서울대학교 공과대학 전자공학과 석사
1996 : 서울대학교 공과대학 전자공학과 박사
1982~현재 : 한국전자통신연구원 지상시스템연 구부장
<주관심 분야> 무선멀티미디어 전송 기술 및 서비스 시스템