通信情報保護學會論文誌
第9卷 第1號, 1999. 3

에러 정정을 위한 Viterbi 알고리즘의 FPGA 구현

조 현숙*, 한 승조**, 이 상호***

The FPGA Implementation of The Viterbi Algorithm for Error Correcting

Hyun-sook Cho*, Seung-jo Han*, Sang-ho Lee*

요 약

통신 및 컴퓨터 시스템의 처리 속도가 높아짐에 따라 에러 정정을 위해서 고속의 데이터 처리가 필요하게 된다. 본 논문에서는 무선 통신 시스템에서 적용 가능한 복호 알고리즘을 제안하고, 이를 이용하여 부호기 및 복호기를 설계한다. 부호기와 복호기를 VHDL로 설계한 후, V-system을 이용하여 관련 회로를 시뮬레이션 한다. 설계된 알고리즘은 SYNOPSYS 툴을 사용하여 합성하고, XILINX XC4010EPC84-4를 이용하여 one chip화하여, 입력 클락으로 20MHz를 사용하였을 때 data arrival time은 29.20ns였고, data require time은 48.70ns였다.

Abstract

As the processing speed of communication and computer system has been improved, high speed data processing is required to correct error of data. In this paper, decoding algorithm which is applicable to the wireless communication system is proposed and encoder and decoder are designed by using the proposed decoding algorithm. We design the encoder and decoder by using the VHDL(VHSIC Hardware Description Language) and simulate the designed encoder and decoder by using V-system. Designed algorithm is synthesized by using synopsys tools and is made to one chip by means of XILINX XC4010EPC84-4. When 20MHz was used as the input clock, data arrival time was 29.20ns and data require time was 48.70ns.

* 한국전자통신연구원
* 조선대학교 전자·정보통신공학부
* 충북대학교 컴퓨터과학과

Ⅰ. 서 론

현대 사회는 매스미디어의 시대라 할 정도로 통신 매체가 발달하였으며, 통신 시스템은 개인 생활의 필수품으로서 자리잡고 있다. 통신 서비스의 이용자들은 단순히 메시지만을 주고 받는 차원을 넘어서 음성, 영상 신호 등의 실시간 전송을 원하고 있으며, 신뢰성 있는 정보의 전송을 원하고 있다. 선로를 통해 전송되는 신호는 데이터의 양이 증가하면 할 수록 외부로부터의 영향에 의한 오류 발생 확률이 높아진다. 따라서 잡음이 있는 환경에서 에러 제어를 위한 부호기와 복호기에 관한 연구는 통신 시스템의 신뢰성 있는 정보 전송을 위하여 더욱 중요하게 되었으며, 선로상에서 에러 제어를 위한 오류 정정 부호의 사용은 통신 시스템에서 중요한 위치를 차지하게 되었다[1].

1948년에 부호화 이론이 처음 제기된 이후에 부호 이론은 크게 블록부호와 길쌈부호로 발전하였으며, 이를 복호하기 위한 방법이 제시되었다. viterbi 복호화 방식은 viterbi에 의해 최초로 알고리즘이 개발되었으며, 헬러(Heller)에 의해 실제적인 기법이 제시되었다. viterbi 복호화는 최대 근사 복호화기법의 하나로서, 오류 정정은 부호기에서 출력된 모든 부호어 중 가장 근접한 부호어를 선택함으로서 이루어진다[2]. 통신 시스템의 변조부와 복조부에 사용되어질 부호기와 복호기는 정보의 오류 정정뿐만 아니라 신속한 정보의 전송을 필요로 하기 때문에 S/W로 구현할 경우에는 정보 처리 속도의 지연이 가장 큰 문제가 된다. 따라서 통신 시스템의 속도 향상을 위하여 하드웨어 구현이 반드시 필요하다.

따라서 본 논문에서는 시스템 구현을 위해 VHDL을 이용하여 모델링하고 EDA Tool을 사용하여 시뮬레이션 및 합성을 하여 FPGA에 구현하여 성능을 평가 및 분석한다.

본 논문의 구성은 다음과 같다. 제 Ⅱ장에서는 부호기와 복호기의 동작 원리를 제 Ⅲ장에서는 복호기의 모델링을 한다. 그리고 제 Ⅳ장에서는 분석 및 고찰을 하며, 제 Ⅴ장에서 결론을 맺고자 한다.

Ⅱ. 컨벌루션 부호기와 viterbi 복호기

통신 시스템에서 정보 전송을 위하여 입력 데이터는 부호기에 의해 부호화 되어 전송되고 전송된 신호는 다시 복호기를 통하여 원래의 신호로 복원된다. 본 논문에서는 오류율이 비교적 적은 컨벌루션 부호기를 사용하였으며, 복호를 위하여 viterbi 복호기를 사용하였다. viterbi 복호기는 입력단에 단계별로 연속해서 통과하는 데이터를 받아들여 오류 발생의 가능성이 있는 모든 상태 값과 비교한 뒤에 오류 발생 확률이 가장 낮은 근사 값을 구하고 이 비교 값에 의해 구해지는 값을 선택하는 방법을 말한다. viterbi 복호기는 통신 채널에서 불규칙 오류 정정 능력이 있으므로 위성 통신에 적합한 방식이다.

2.1 컨벌루션 부호기

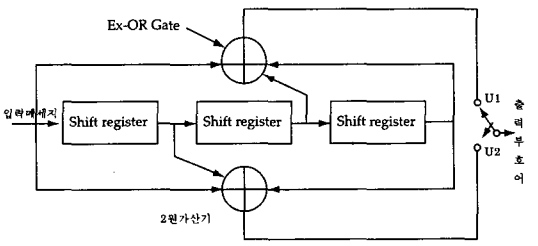

그림 1. 컨벌루션 부호기의 구성도
Fig 1. Block diagram of convolutional encoder

부호기는 그림 1과 같이 3개의 시프트 레지스터와 2원 가산기 그리고 멀티플렉서로 구성되어 있고, 메시지 비트가 입력될 때마다 오른

쪽으로 1개 비트씩 이동시킨 뒤 출력을 얻게 된다. 2원 가산기에 입력되는 값은 이동 중인 데이터 비트열의 중간 값을 취하여 연산을 하며, 전환 스위치에 의해서 U1, U2를 출력시킨다. 멀티플렉서에 U1의 출력을 A1이라 하고, U2의 출력을 A2라 하면 그림 1의 부호기에서 식 (2.1)과 식 (2.2)과 같은 다항식을 얻을 수 있다.

$$A_1(x) = 1 \oplus 1x^2 \oplus 1x^3 \quad (2.1)$$

$$A_2(x) = 1 \oplus 1x \oplus 1x^3 \quad (2.2)$$

A_1, A_2 다항식의 차수는 레지스터단에 의한 것이며, A_1의 다항식은 입력단과 두 번째, 세 번째 레지스터가 2원 가산기에 접속되어 있는 것을 나타내고, A_2의 다항식은 입력단과 첫 번째, 세 번째 레지스터단이 2원 가산기에 접속되어 있는 것을 나타낸다. 입력 비트열을 $m1$, $m2$, $m3$, $m4$, ···, mn이라 하면 메시지 벡터를 식 (2.3)과 같은 다항식으로 나타낼 수 있다.

$$A_m(x) = m_1x^0 \oplus m_2x^1 \oplus m_3x^2 \oplus m_4x^3, \cdots, m_nx^{n-1} \quad (2.3)$$

이와 같이 $A_1(x)$, $A_2(x)$, $A_m(x)$가 정해지면 출력 부호어 시퀀스를 계산할 수 있다. 출력 부호어 시퀀스를 V라 하고, $A_1(x)$에 의한 출력 부호어 시퀀스를 $V_1(x)$, $A_2(x)$에 의한 출력 부호어 시퀀스를 $V_2(x)$라 하면 식 (2.4)와 식 (2.5)와 같이 출력 다항식을 나타낼 수 있다.

$$V_1(x) = A_m(x) \cdot A_1(x) \quad (2.4)$$

$$V_2(x) = A_m(x) \cdot A_2(x) \quad (2.5)$$

두 식을 동일 차수 항으로 정리하면 출력 부호어 시퀀스 V값이 정해진다[3].

2.2 viterbi 알고리즘

viterbi 복호기는 서로 다른 경로를 가진 2개의 신호에 대한 경로길이를 비교해서 오류 발생 확률이 낮은 하나의 신호만 선택하고, 경로길이가 긴 것은 지워 버림으로서 최적의 경로를 얻게 된다. 이때 유지 경로의 결정은 각 노드 점에서 매번 수행되는데 이때마다 경로길이가 긴 것은 기억 장치에서 삭제시킨다. 이러한 과정은 복호기의 기억 용량의 증가를 방지할 뿐만 아니라 복호화 과정의 복잡도를 피할 수 있다[4]. 이렇게 하여 맨 처음 수신된 부호어에 해당하는 값은 현재 남은 각 경로에 공유시키고, 이때 최초 부호어를 복호화 하여 메시지 비트의 최초 비트를 출력시키게 된다. 여기서 최적의 경로를 선택하는 것은 최대 근사 값을 가진 부호어를 선택함으로서 이루어진다. viterbi 복호기의 복호화 과정은 그림 2와 같이 나타낼 수 있다.

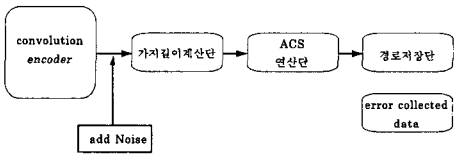

그림 2. Viterbi 복호기의 구성도
Fig 2. Block diagram of Viterbi decoder

viterbi 복호기에서 각 경로를 기억하기 위한 기억 용량은 시프트 레지스터단의 수에 따라 지수적으로 증가한다. 그러므로 부호화 이득을 증가시키기 위해 레지스터의 수를 증가시키면 경로를 기억하기 위한 회로와 경로 선택을 위한 회로의 증가를 가져오게 된다. 기억량을 Mem이라고 하고, B_n를 메시지 비트수라 하면 경로의 기억량과 레지스터 k와의 관계는 식 (2.6)과 같다[5].

$$Mem = B_n \cdot 2^{k-1} \quad (2.6)$$

이 식에서 k 값을 증가시키면 시프트 레지스터의 수가 증가해서 비교해야 할 경로가 증가하고, 기억 장소가 증가하기 때문에 경로 기억량의 증가를 가져오게 된다. 위 식에서 B_n 값은 최초의 메시지 비트를 복호화해서 출력시킬 때까지 수신된 부호어의 비교 수를 나타낸다. 최초로 수신된 부호어는 복호화 된 다음 각 수신 부호어의 가지 길이를 계산해서 유지 경로를 선택하며, 항상 2^{k-1}에 해당하는 경로만을 기억하게 된다. 선택된 유지 경로 2^{k-1}은 다음 메시지 비트의 복호화 때 영향을 미치므로, 유지 경로의 선택 과정을 진행해 가고 진행 상태를 복호기에서 기억시키면 어느 시점에서 기억된 가지 길이의 하나가 2^{k-1}의 유지 경로 모두에게 공유된다. 이때 공유된 가지 길이를 복호화 하여 메시지 비트를 출력시킨다. 메시지 비트를 출력시키기 위해서는 복호화 과정의 시작부터 2^{k-1}의 유지 경로를 기억하고 있어야 하며, 유지 경로의 기억은 메시지 시퀀스 비트가 모두 복호화 될 때까지 계속해야 하는데 k 값이 증가하면 기억해야 될 유지 경로의 수가 지수적으로 증가해서 기억량이 커지게 된다. 보통 적당한 시프트 레지스터 값을 감안해서 B_n값은 4~5회로 하는데 B_n값은 10이하로 제한한다[6, 7, 8, 9].

Ⅲ. 하드웨어 설계

본 논문에서 설계한 시스템의 최상위 레벨 블록도는 그림 3과 같다. 전체 시스템은 경로 길이를 계산하기 위한 Cal_Branch 블록, 계산된 경로를 선택하기 위한 ACS(Adder Comparator Selector) 블록, 경로를 기억하였다가 출력하기 위한 Path_Mem 블록의 세 부분으로 나누어 설계하였다. 부호기의 입력으로 8비트 데이터가 enc_in에 입력되고, clk에 동기화 되어 동작한다. 부호기의 동작은 reset 단자에 의해 활성화되며 부호기의 출력으로는 22비트의 데이터가 출력되고, 각 비트에 잡음신호인 add_noise 신호가 부가된다. 잡음신호가 부가된 22비트의 신호는 복호기 블록의 입력으로 사용되고, Cal_Branch 블록, ACS 블록, Path_Mem 블록의 수행 과정을 거쳐 오류가 정정된 8비트의 원 신호를 출력한다. 복호기는 clear 신호에 의해 초기화되고, enable 신호가 "1"되면 블록들은 동작을 한다.

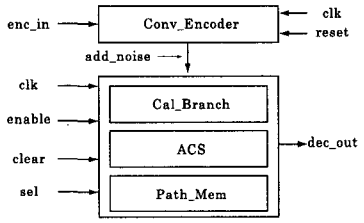

그림 3. 최상위 레벨 블록도
Fig 3. Top level block schematic

3.1 Cal_Branch 블록

Cal_Branch 블록은 그림 4와 같이 가지 길이 값을 기억하기 위한 RAM 부분과 Compare, Counter, Decoder, Addr_Gen로 구성되어 있다. clk의 입력으로 20MHz의 클락을 사용하고, enable 신호가 "1"되면 RAM이 동작하게 되며 add(1), add(2) 신호를 생성하게 된다. clear 신호는 RAM, Counter, Compare, Decoder의 값을 초기화시키는 역할을 하며 data_in 신호는 컨벌루션 부호기를 통해 출력된 22비트의 데이터에 잡음 신호가 부가된 값이 사용되며, 수신된 값은 패턴 값을 비교하여 가장 짧은 부호어 값만을 기억하고 나머지는 삭제한다. Counter를 통하여 출력된 cout은 Decoder와 Addr_Gen의 입력으로 사용되며,

RAM의 어드레스를 지정하는 역할을 한다.

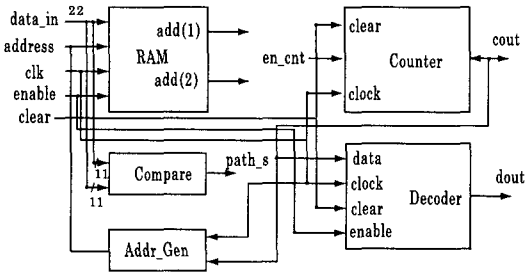

그림 4. Cal__Branch 블록의 구조
Fig 4. Structure of Cal__Branch Unit

3.2 ACS 블록

ACS 블록은 그림 5와 같다. ACS 블록은 해밍거리를 산출하는 Hamming 부분과 이 경로 길이를 서로 더하여 비교 출력하는 부분으로 이루어져 있다. 해밍 거리를 계산하는 블록에 2진 데이터인 wsig 신호와 decin 신호가 입력으로 가해지면 두 신호의 거리를 계산한다. Hamming 블록은 XOR 게이트로 이루어져 있으며, 거리를 계산하기 위해 XOR 연산을 수행한다. 이때 high 값을 가지는 부분의 개수가 해밍거리로 인식한다. Hamming 블록에는 2비트의 데이터가 입력으로 사용되고, 이때 출력값은 '0' '1' '2' 세 개의 값을 가질 수 있다. 예를 들어 wig 신호로 "11"입력되고, decin 신호로 "00"입력되면 두 신호의 해밍 거리는 "11", 즉 2가 된다. 해밍 거리를 구하는 식은 식 (2.7)과 같이 나타낼 수 있다.

$$hamming(1) = (wig(1) \oplus decin(1)) \text{and } (wig(0) \oplus decin(0))$$

$$hamming(0) = (wig(1) \oplus decin(1)) \oplus (wig(0) \oplus decin(0)) \quad (2.7)$$

해밍 거리 값은 Adder의 입력으로 사용되며 state값과 가산 연산을 수행하게 되고, 결과값은 비교기에 입력으로 사용되어 값이 작은 것만을 출력한다. 이러한 과정은 각 단에서 매번 실행되게 되며 모든 노드 점에서 계산을 수행하려면 8개의 ACS 블록을 필요로 한다. 각각의 노드 점에 ACS단을 설치하여 사용하게 되면 칩 면적의 효율성을 떨어뜨리게 되므로 2개만을 설치하여 각각의 노드를 공유하여 사용하도록 한다.

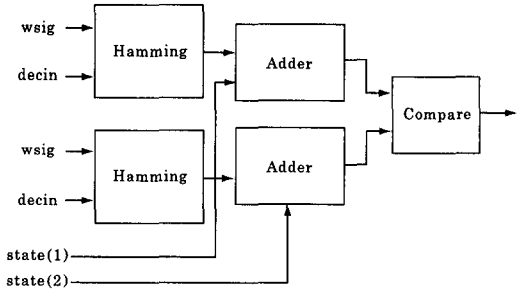

그림 5. ACS 블록의 구조
Fig 5. Structure of ACS Unit

3.3 Path__Mem 블록

Path_Mem 블록은 그림 6과 같이 ACS단에서의 선택된 신호를 임시로 저장하는 Register 부분과 Mem_Ram, MUX1, MUX2, Buffer로 구성되어 있다. MUX1에서 read, write 신호를 입력으로 받고, sel 신호에 의해서 하나를 선택하게 된다. Mem_Ram 블록에서는 read 신호가 활성화 되면 ACS단의 정보가 저장되어 있는 Register에서 출력 data를 입력받게 되고, write 신호가 활성화되면 저장되어 있는 부호어 값을 r_out 단자를 통하여 출력한다. Mem_Ram 블록은 clk의 상승 에지에서 동작하며, 할당된 어드레스에 데이터를 저장하거나, 저장된 데이터를 출력하는 역할을 한다. Buffer에는 전 단계의 경로 기억 값이 저장되어 있고, 이 값은 MUX2의 입력으로 사용된다. MUX2에서는 입력 신호와 Mem_Ram 블록의

출력 신호와 비교해서 가장 작은 값을 출력 단자를 통해서 출력한다.

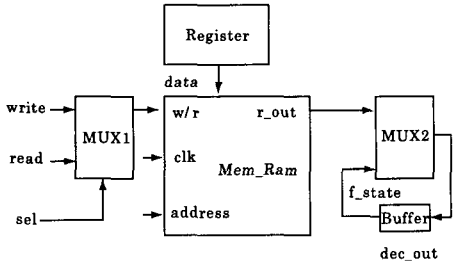

그림 6. Path__Mem 블록의 구조
Fig 6. Structure of Path__Mem Unit

Ⅳ. 설계 흐름 및 분석

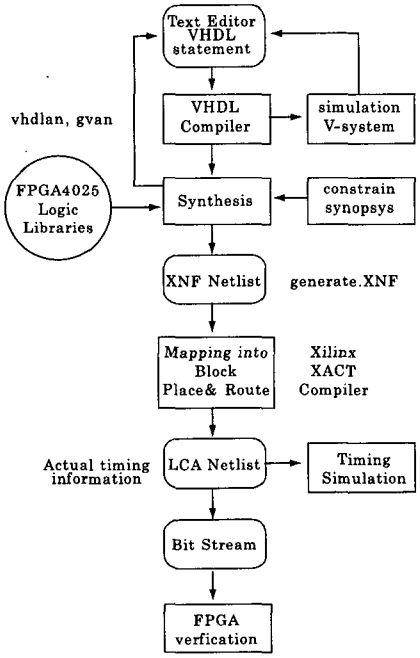

그림 7. 설계 흐름
Fig 7. Design flow

viterbi 복호기의 VLSI 구현을 위해 top-down 방식으로 설계하고, VHDL을 사용하여 모델링 한다. VHDL을 이용하면 회로 설계의 효율화 및 설계 시간을 단축시킬 수 있으며, EDA 소프트웨어나 ASIC 라이브러리에 상관없이 객관성 있게 설계를 할 수 있다. 또한 반도체 내부 회로에 대한 검증부터 시스템 설계까지 할 수 있으며, synthesis 소프트웨어를 이용하여 실제 반도체 회로를 쉽게 얻을 수 있다는 장점을 가지고 있으므로 본 논문에서 설계 언어로서 채택한다[10, 11, 12].

설계 초기 단계에서 그림 7과 같이 VHDL 모델을 작성하여 Function 시뮬레이션을 수행하여 기능적 동작의 타당성을 검증한다. 이들 모델을 RTL로 구체화한 후, SYNOPSYS 툴을 사용하여 논리합성을 한다. 합성이 끝난 회로는 SDF 파일을 생성시켜 타이밍 시뮬레이션을 하며, 타이밍 시뮬레이션을 거쳐 얻은 결과는 hold time violation 등 시간에 관련된 오류를 검사하는데 사용한다. 타이밍 시뮬레이션을 수행한 후 XC4010EPC84-4에 구현한다. FPGA를 이용한 구현은 미리 설계된 칩에 프로그래밍을 함으로써 원하는 기능의 테스트용 회로를 구현하는데 목적이 있으며, IC 설계 제작에 비하여 소요되는 비용과 시간을 절감할 수 있을 뿐 만 아니라, 원하는 회로에 대한 신속한 실험 제작이 가능하다는 장점이 있다.

4.1 시뮬레이션 결과

설계된 시스템의 동작을 검증하기 위하여 V-system을 사용하여 시뮬레이션을 수행한 결과는 그림 8과 그림 9와 같다.

부호기의 입력으로는 8비트의 데이터를 사용하였으며, 부호화된 22비트의 출력을 얻었다. 부호기에서 출력된 22비트의 데이터는 잡음 신호를 부가하여 다시 dec_in의 입력으로 사용하였으며, 각 블록들의 연산을 수행한 뒤 12 클락이 소요된 뒤에 출력 값을 얻었다. 복호기의 입력으로는 부호기의 출력에 잡음을 부가하여 사용하였으며, 12 클락이 지난 후에 에러가 정정된 신호가 출력되었다. 시뮬레이션 수행시에 사용된 채널에 발생하는 잡음은 각 비트에 임

으로 값을 부가하여 사용하였다. dec_in 신호로 각 비트에 2, 4, 6개의 잡음을 부가한 22비트 신호를 입력으로 사용하였으며, 그 출력 값은 표 1과 같다. 이 출력은 viterbi 알고리즘을 이용하여 오류를 정정한 값이며, 잡음 신호가 첨가되지 않은 원 신호의 복호 결과와 비교하여 오류 정정 여부를 확인하였다.

표 1. 부호기와 복호기의 시뮬레이션 결과
Table 1. Simulation result of encoder and decoder

원 신 호	출 력	잡음채널	잡음보상 출 력
1101011001100110101100	10101010	0101011001100110101101	10101010
1110110100101101110000	11001100	1110110100101101110011	11001100
1101100001100010110111	10010011	1011100101110010110101	10010011
0000000011101101110000	00001100	0101010011011011100000	00001100

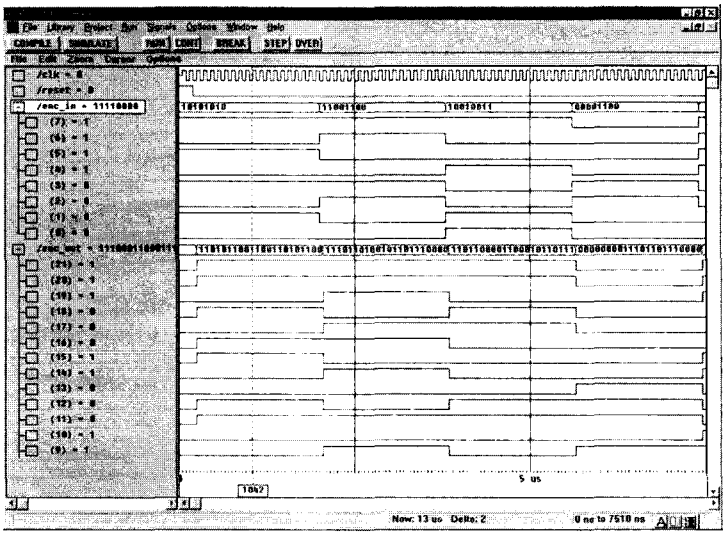

그림 8. 부호기의 시뮬레이션 결과
Fig 8. Simulation result of encoder

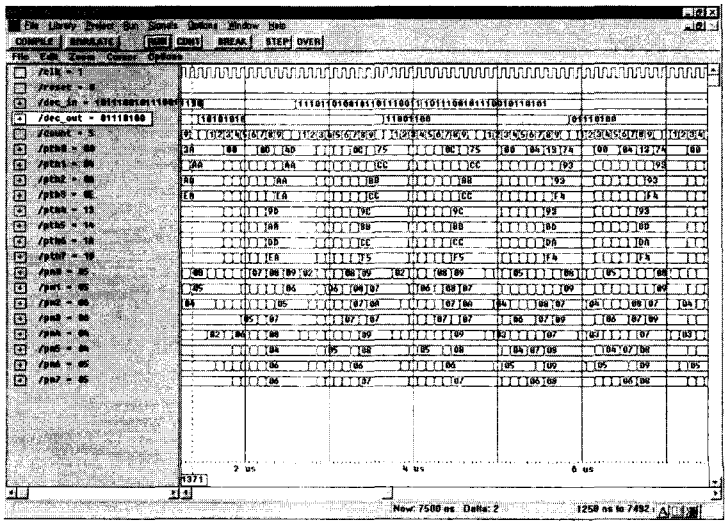

그림 9. 복호기의 시뮬레이션 결과
Fig 9. Simulation result of decoder

4.2 합성 결과

본 논문에서 구현한 알고리즘은 모뎀부에 사용할 목적으로 설계하였으므로 타이밍보다는 칩 면적 감소에 중점을 두었으며, 합성된 결과는 그림 10과 그림 11과 같다.

XILINX의 XDM을 사용하여 XC4010EPC84-4에 Place & Route한 결과는 그림 12와 같다. Place & Route후 back-annotated VHDL netlist와 SDF 파일을 이용하여 타이밍 시뮬레이션을 수행한 후 XC4010EPC84-4에 다운로딩하여 테스트하였다.

복호기의 XC4010EPC84-4를 이용한 디바이스 사용도는 표 3과 같고 data arrival time은 29.20ns가 걸렸으며, data required time 48.70ns가 걸렸다.

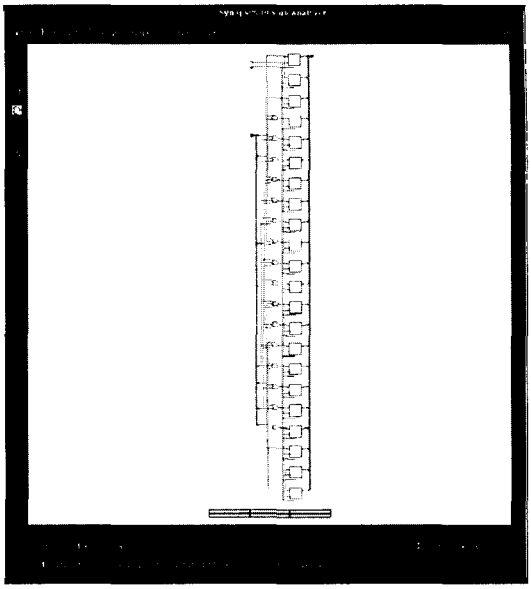

그림 10. 부호기 블록의 합성결과
Fig 10. Synthesis of encoder block

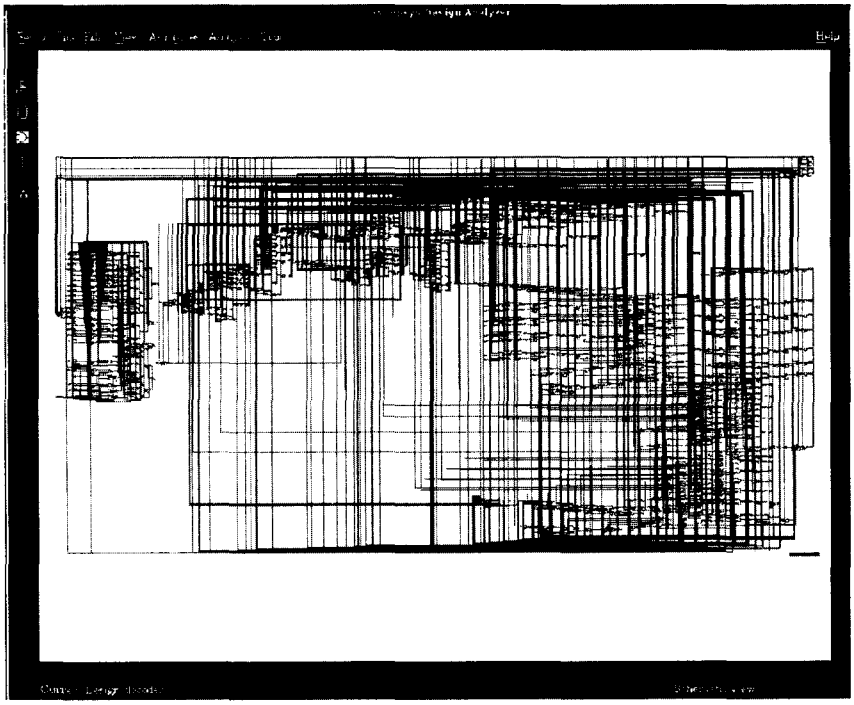

그림 11. 복호기 블록의 합성결과
Fig 11. Synthesis of decoder block

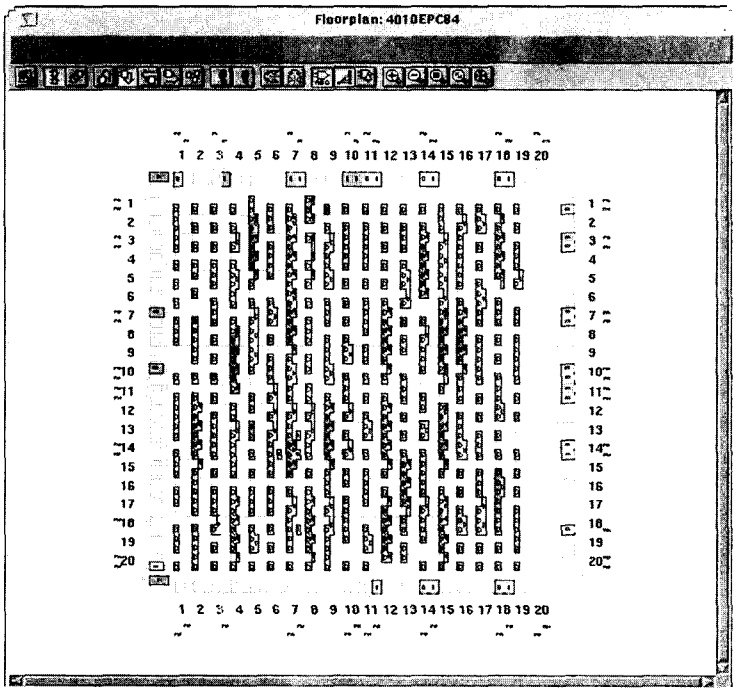

그림 12. Place & Route 결과
Fig 12. Result of Place and Route

표3. XC4010EPC84-4을 이용한 디바이스 사용도
Table 3 Device utilization using XC4010EPC84-4

	Used	Usable	Used(%)
Number of CLBs	279	400	69
CLB Flip Flops	4		
4 input LUTs	426		
3 input LUTs	167		
Number of global buffers	2	8	25
Number of bonded IOBs	32	61	52
Number of secondary CLKs	2	4	50
Data arrival time	29.20 ns		

V. 결 론

본 논문에서는 잡음이 있는 채널상에서 신뢰성 있고 신속한 정보 전송을 위하여 에러 정정 능력을 갖는 부호기와 복호기를 설계 및 구현하였다. 설계 언어로는 하드웨어 설계언어인 VHDL을 사용하였고, 전체적인 설계 흐름은 top-down 방식을 채택하였다. 설계된 알고리즘은 V-system을 사용하여 function 시뮬레이션을 수행하였으며, 동작 검증이 끝난 시스템은 SYNOPSYS 툴을 사용하여 합성하였다. 논리검증을 통하여 오동작 여부를 미리 검증하여 합성 때 나타나는 전체적인 오류를 줄일 수 있었으며, XILINX사의 XC4010EPC84-4에 구현하였다. 입력 클락으로는 20MHz의 클락을 사용하였으며, 데이터 출력 시간으로 29.20ns가 소모되었다. 향후 연구 과제로는 소프트웨어, 하드웨어 혼합 설계 방식을 이용하여 전체적인 설계 비용을 줄이는 것과 복호기 전체의 병렬 처리 및 파이프라인 처리가 남아 있다.

참 고 문 헌

[1] S. Lin and D. J. Costello, Error Control Coding : Fundamentals and Applications, Englewood Cliffs, NJ: Prentice-Hall, 1983.

[2] A. J. Viterbi, "Error Bound for Convolutional Codes and an Asynptotically Optimum Decoding Algorithm," IEEE Trans. Inf. Theory, IT-13, pp.260-269.

[3] A. J. Omura, "On the Viterbi algorithm, " IEEE Trans. Inform. Theory, pp. 177-179, Jan. 1969.

[4] K. L. Jordan, "The Performance of the Sequential Decoding Computation Time," IEEE Trans. Inf. Theory, pp.143-147.

[5] G. Fettweis and H. Meyr, "High-Speed Parallel Viterbi Decoding: Algorithm and VLSI-Architecture," IEEE Communications Magazine, pp. 46-55, May. 1991.

[6] P.Elias, "Coding for Noisy Channels," IRE Conv. Rec., Part 4, 37-47, 1955.

[7] C. E. Shannon, " A Mathematical Theory of Communication," Bell System Technical Journal, 27, pp. 379-423, July 1948.

[8] P. J. Black and T. H. Meng, "A140-Mb/s, 32-State, Radix-4 Viterbi Decoder," IEEE J. of Solid-State Circuits, Vol. 27, No.12, pp. 1877-1885, Dec 1992.

[9] J. A. heller and I. M. Jacobs, "Viterbi Decoding for Satellite and Space Communication," IEEE Trans. Commun. Technol., Vol. COM-19, No. 5, pp. 835-848, Oct 1971.

[10] S. Lin and D. J. Costello, Error Control Coding: Fundamentals and Applications, Englewood Cliffs, NJ: Prentice-Hall, 1983.

[11] Z. Navadi, "Using VHDL for model and design of processing unit," Proceed of the 1992 ASIC Conf. and Ex., pp.315-326. 1992.

[12] N. G. Einspruch, J. L. Hilbert, Application Specific Integrated Circuit(ASIC) Technology, Academic Press, Inc. San Die, pp. 2-3, 1991.

□ 著者紹介

조 현 숙(종신회원)

전남대학교 수학과 졸업(학사)
충북대학교 대학원 전자계산학과 졸업 (석사)
1982년 ~ 현재 한국전자통신연구원 책임연구원
정보보호시스템연구부장

※ 주관심분야 : Network Security, Conditional Access

한 승 조

1980년 조선대학교 전자공학과 학사
1982년 조선대학교 대학원 전자공학과 석사
1994년 충북대학교 대학원 전자계산학과 박사
1997년 조선대학교 전자·정보통신공학부 교수
1986년 6월 ~ 1987년 3월 Univ. of New Oleans 객원 교수
1995년 2월 ~ 1996년 1월 Univ. of Texas 객원 교수

※ 주관심분야 : 통신보안시스템설계, 네트워크 보안, ASIC 설계, 음성합성

이 상 호

1976년 숭실대학교 전자계산학과 졸업(공학사)
1981년 숭실대학교 대학원 전자계산학과 졸업(공학석사)
1989년 숭실대학교 대학원 전자계산학과 졸업(공학박사)
1981년 ~ 현재 충북대학교 컴퓨터과학과 교수
1992년 ~ 1993년 캐나다 UBC초청교수
1994년 ~ 1998년 충북대학교 전자계산소 소장

※ 주관심분야 : 프로토콜 공학, 통신보안, 시뮬레이션