파이프라인 기반 다중윈도방식의 비터비 디코더를 이용한 채널 코딩 시스템의 구현

서영호* · 김동욱**

Implementation of Channel Coding System using Viterbi Decoder of Pipeline-based Multi-Window

Young-Ho Seo* · Dong-Wook Kim**

이 논문은 2005학년도 광운대학교 교내 학술연구비 지원에 의해 연구되었음.

요 약

본 논문에서는 시분할 방식을 확장하여 윈도를 통해 비터비 복호화되는 단위를 다중으로 버퍼링하고 병렬적으로 처리하는 비터비 복호화기를 구현한다. 연속적으로 입력되는 신호를 복호화 길이의 배수로 버퍼링한 후 이를 고속의 비터비 복호화기 셀을 이용하여 병렬적으로 복호화를 수행한다. 비터비 복호화기 셀의 사용수에 비례하여 데이터 출력율을 얻을 수 있는데 입력 버퍼의 프로그래밍 및 수정에 따라서 이러한 동작을 만족시킬 수 있다. 구현된 비터비 복호화기 셀은 해밍 거리 계산을 위한 HD 블록, 각 상태의 계산을 위한 CM 블록, 비교를 위한 CS 블록, 그리고 trace-back을 위한 TB 블록 및 LIFO 등으로 구성된다. 비터비 복호화기 셀은 ALTERA의 APEX20KC EP20K600CB652-7 FPGA에서 1%(351 cell)의 LAB(Logic array block)를 사용하여 최대 139MHz에서 안정적으로 동작할 수 있었다. 또한 비터비 복호화기 셀과 입출력 버퍼링을 위한 회로를 포함한 전체 비터비 복호화기는 약 23%의 자원을 사용하면서 최대 1Gbps의 데이터 출력율을 가질 수 있도록 설계하였다.

ABSTRACT

In the paper, after we propose a viterbi decoder which has multiple buffering and parallel processing decoding scheme through expanding time-divided imput signal, and map a FPGA, we implement a channel coding system together with PC-based software. Continuous input signal is buffered as order of decoding length and is parallel decoded using a high speed cell for viterbi decoding. Output data rate increases linearly with the cell formed the viterbi decoder, and flexible operation can be satisfied by programming controller and modifying input buffer. The cell for viterbi decoder consists of HD block for calculating hamming distance, CM block for calculating value in each state, TB block for trace-back operation, and LIFO. The implemented cell of viterbi decoder used 351 LAB(Logic Arrary Block) and stably operated in maximum 139MHz in APEX20KC EP20K600CB652-7 FPGA of ALTERA. The whole viterbi decoder including viterbi decoding cells, input/output buffers, and a controller occupied the hardware resource of 23% and has the output data rate of 1Gbps.

키워드
Channel Coding, Viterbi Decoder, VLSI, FPGA

*유한대학교 전자정보과
**광운대학교 전자재료공학과 디지털 설계 및 테스트 연구실

접수일자 : 2005. 1. 20

Ⅰ. 서 론

최근 다양한 형태의 통신방식이 발전함에 따라서 여러 잡음원에 의해 발생되는 에러에 대해 신뢰성 있는 통신이 요구되고 있다. 이러한 에러 정정을 위한 기술로써 길쌈 부호화(Convolution coding)가 있는데 이를 복호하는 대표적인 방식이 비터비 복호화(Viterbi decoding)이다. 비터비 알고리듬[1],[2]은 크게 데이터를 저장하거나 통신을 할 때 발생할 수 있는 데이터의 오류를 정정하는데 사용된다. 전자의 경우에 제한길이 4 이하를 가지면서 Gbps 급의 고성능이 요구되고[3] 후자의 경우에는 9정도의 비교적 큰 제한 길이를 가지면서 통신방식이 요구하는 정도의 성능(2Mbps)을 가져야 한다[4].

비터비 복호화기를 하드웨어(Hardware, H/W)로 구현할 때에는 크게 두 가지 관점에서 주로 다루어지는데 ACS(add-compare-select)와 TB (trace-back)의 구현이 이에 해당한다. 최근 ACS의 경우에는 큰 제한길이를 요구하는 CDMA등의 통신을 위해서 Bit-serial(BS) 방식의 구조를 통해 동작속도를 높이는 연구가 이루어졌다. 이러한 구조는 state의 수가 클 경우 효율적이고 FA의 delay만을 가지므로 고속의 동작이 가능하지만 최종 bit(MSB)의 연산이 끝날 때까지 selection이 유보되므로 이를 위한 부가적인 레지스터가 요구되고 덧셈과 뺄셈시 중간 데이터의 저장을 위한 저장소가 요구되는 단점이 있다[5]. 데이터 저장을 위한 고속처리 방식으로는 Radix-4의 ACS 구현이 좋은 결과를 보이고 있는데 coding pass를 반으로 줄임으로써 성능이 향상되지만 radix 증가에 따라 각 state에서의 compare와 select 및 BM을 위한 H/W 및 복잡도 증가가 발생하여 성능이 감소하고 radix 증가에 따라서 전체 하드웨어(BM, trace-back)도 변형하는 복잡성을 가진다[3]. 또한 time-multiplexed ACS (BP folded ACS) [6] 방식도 연구되었는데 H/W의 양을 줄이면서 적당한 속도를 내기 위해 일정 크기의 ACS 모듈을 구현한 뒤 반복적으로 사용하는 것으로 제어의 복잡성과 routing의 오버헤드가 크고 [7] 시간적인 차이를 두기 위해 중간 데이터의 저장을 위한 부가적인 저장 장소가 요구되는 단점이 있다.

또 하나의 핵심적인 연구대상은 trace-back (TB)인데 가장 대표적인 방식이 Register exchange 방식이다[8]. 이 방식은 shift register array에 모든 state의 survivor path를 저장해야 하고 모든 코딩 패스마다 모든 레지스터가 decision에 의해서 exchange 해야 하므로 power 소모가 큰 단점이 있다. 또한 survivor path가 긴 경우 H/W가 거대해지고 register의 exchange를 위한 MUX가 커져서 전체적인 성능이 감소한다. 이에 대한 대안으로 메모리와 pointer를 이용한 방식이 연구되었다[9].

본 논문에서는 고속의 복호화가 요구되는 분야를 위해 시분할 방식을 확장하여 윈도를 통해 비터비 복호화 되는 단위를 다중으로 버퍼링 하고 병렬적으로 처리하는 고 데이터율의 비터비 복호화기를 구현한다.

Ⅱ. 다중 윈도를 통한 입력 분할처리

본 장에서는 비터비 복호화에 대해서 설명하고 연속적으로 입력되는 신호에 대해서 다중 윈도를 적용하고 버퍼링을 거쳐서 병렬적으로 처리되는 방식을 나타낸다.

1. 비터비 복호화

비터디 복호화기는 그림 1과 같은 길쌈 부호기를 이용한다. 그림 1은 제한길이 K=3, 부호화율 1/2인 길쌈 부호기이다. 입력은 레지스터에 차례로 저장되고 각 레지스터의 데이터를 XOR하여 두 개의 출력 데이터를 만들어준다.

콘볼루션 복호화 방법은 최대유사디코딩을 이용한다. 최대유사디코딩 방식은 주어진 입력에 대해 가장 유사한 형식을 찾아 이를 디코딩하는 방식으로 최적의 복호화 방법으로 알려져 있다. Viterbi 알고리즘은 BM(branch metric)과 PM(path metric)이라는 변수를 통해 최대유사디코딩을 수행한다. BM은 상태천이도에 의해 생성된 기준 데이터와 송신된 데이터와의 차를 구하는 것으로 HD(Hamming Distance)로 정의된다. PM은 생존자 경로를 통해 전달된 연속적인 BM의 합을 저장한다. 복호화과정은 상태천이도의 각 상태에 대해 PM과 BM을 더해서 작은 값을 가지는 부분을 선택해 다시 PM에 저장하고 그 경로를 저장한다. 이런 과정을 통해 모여진 경로 데이터를 일정시간 후부터 다시 거슬러 올라가(traceback) 최소 경로를 찾아가게 된다. 이때 선택된 경로가 복호화 결과로 출력

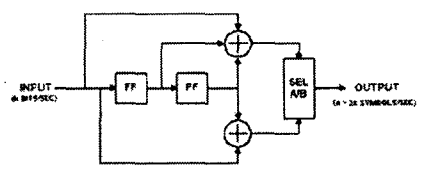


그림 1. 제한길이 3의 길쌈 부호화기
Fig 1. Convolution encoder of constraint length 3.

된다[1].

2. Correlation 기반의 복호화

각 단계의 상태들에서 이전상태들의 PM과 현재 상태의 BM을 이용하여 연산된 현재 상태의 PM 중에서 작은 PM을 선택하는 방식을 "Distance" 기반의 비터비 복호화라 하고 큰 PM을 선택하는 방식을 "Correlation" 기반의 비터비 복호화라고 한다. 이들 중에서 본 구현에서는 후자의 방식을 선택한다.

길쌈 복호화 방법은 최대유사 복호화를 이용한다. 최대유사 복호화 방식은 주어진 입력에 대해 가장 유사한 형식을 찾아 이를 복호화하는 방식으로 최적의 복호화 방법이다. 비터비(viterbi) 알고리즘은 BM(Branch Metric)과 PM(Path Metric)이라는 변수를 통해 최대유사 복호화 수행한다. BM은 상태천이도에 의해 생성된 기준 데이터와 송신된 데이터와의 차를 구하는 것으로 HD(Hamming Distance)를 정의하고 PM은 생존자 경로를 통해 전달된 연속적인 BM의 합을 저장한다.

본 논문에서의 사용하는 복호화 과정은 상태천이도의 각 상태에 대해 PM과 BM을 더해서 작은 값을 가지는 부분을 선택해 다시 PM에 저장하는 "Distance" 기반의 비터비 복호화가 아니라 그림 2와 같이 상태천이도의 각 상태에 대해 PM과 BM을 더한 후에 작은 값을 선택하지 않고 큰 값을 선택하여 그 경로를 저장하는 "Correlation" 기반의 비터비 복호화를 수행한다. 그림에서 PM과 BM의 연산에 의한 결과들 중에서 각 상태마다 큰 값을 선택(순서대로 4, 4, 5, 그리고 4)한 것을 볼 수 있다. 따라서 Depth가 16인 복호화 단위 윈도우(window)를 사용한 비터비 복호기의 경우에 16번째 상태에서 최대 PM을 가지는 상태를 찾고 Decision을 이용해 처음 단계까지 다시 거슬러 올라가는 과정(traceback)을 거쳐 출력을 생성한다. 다음 절에 예를 나타냈다.

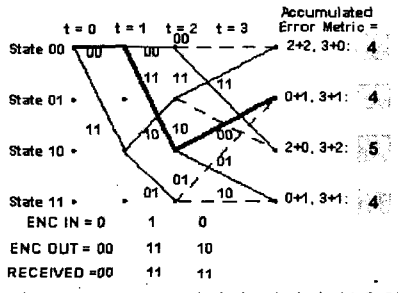
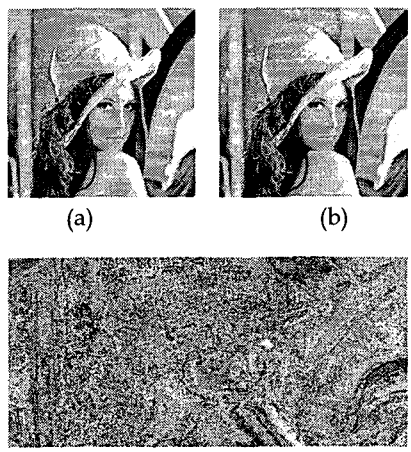


그림 2. Correlation 방식의 비터비 복호화
Fig 2. Correlation-based Viterbi decoding.

약 70,000의 비터비 복호화 실험을 통해서 PM의 마지막 상태에서 최대 PM의 값은 항상 32임을 확인하였다. 따라서 하드웨어 구현 시 최대 상태를 구하기 위한 비교기를 사용하지 않고 (32)10, 즉 (10000)2를 찾는 회로를 구성하면 된다. 이 경우에 단순히 5비트 위치와 AND 연산을 통해서 최대 상태 발견에 대한 신호를 발생시킬 수도 있다. 따라서 최대 PM을 이용하는 경우에 실험에 의해서 최대값의 범위 혹은 최대값이 수렴하는 정도를 구한다면 하드웨어의 간략화를 가져올 수 있을 것이다. 그러나 본 구현에서는 최대 상태를 직접 찾는 방식을 선택하였다.

그림 3에 길쌈 코딩과 "Correlation" 방식의 비터비 복호화를 수행한 영상을 나타냈다. 검증은 256×256 크기의 그레이 영상을 대상으로 C++ 언어를 이용하여 수행되었다. 일반적인 H/W 검증방식과 마찬가지로 C 언어와 같은 상위 레벨 프로그래밍 언어를 이용하여 하드웨어 구현에 필요한 각 단계별 테스트 벡터를 추출하고 골든 유닛(Golden Unit)으로 설정한다.



(a) (b)

(c)

그림 3. 길쌈 부호 및 비터비 복호화의 영상예 (a) 원영상, (b) 복원영상, (c) 길쌈 부호화된 영상
Fig. 3. Example of convolution encoding and viterbi decoding (a) original, (b) reconstructed, (c) convolution-encoded image.

3. Correlation 기반의 복호화

그림 3에 나타난 8 비트 기반의 그레이 영상을 길쌈 부호화하기 위한 비트정렬 방식을 그림 4에 나타내었는데 비터비 복호화의 경우에도 동일한 방식으로 8 비트 그레이 영상에 적용된다. 그림 3에 보이는 것과 같이 일반적인 그레이 영상(Gray

image)의 경우에 8비트로 영상의 밝기 정보를 표시한다.

먼저 길쌈부호화를 수행하기 위해 영상의 각 화소들을 비트 단위로 입력한다. 길쌈부호화기는 어떠한 길이 제약이 없기 때문에 그림 4와 같이 단순히 MSB(Most significant bit)부터 LSB(Least significant bit)의 순서로 각 화소들을 비트열 형태로 입력시키면 된다. 길쌈 부호화를 거친 영상은 현재 1/2의 출력율을 가지기 때문에 2배의 크기로 데이터양이 증가한다. 부호화된 데이터는 다시 8비트 단위로 묶어져서 하나의 영상을 이룬다. 이때에도 데이터 취급의 용이성을 위해서 입력되는 순서대로 화소의 MSB부터 채운다. 그림 4는 영상에서 비트열을 만들어내는 방식에 대한 예를 보인 것일 뿐 길쌈 부호화나 비터비 복호화 알고리즘과는 아무런 상관이 없다.

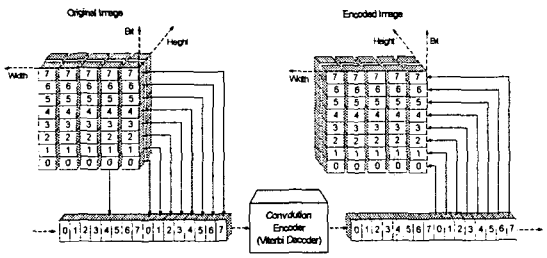


그림 4. 8비트 영상의 길쌈 부호화(비터비 복호화)
Fig 4. Encoding(viterbi decoding) 8bits image.

III. 제안한 비터비 복호화기의 설계

본 장에서는 비터비 복호화기 셀의 구조에 대해서 설명하고 수정된 Register exchange 방식의 trace-back구조를 제안한다. 또한 확장이 가능한 전체적인 비터비 복호화기의 동작을 설명한다.

1. 비터비 복호화기 셀

비터비 복호화기 셀은 일반적인 비터비 복호화기를 의미하고 본 논문에서는 전체적인 비터비 복호화기와 구분하고자 셀로써 명명했다. 그림 5와 같이 제안한 비터비 복호화기 셀은 해밍거리 계산을 위한 HD(Hamming distance), Branch metric 연산을 위한 CM(Compute Metric), 비교를 위한 CS(Compare select), metric의 정규화를 위한 Normalize, 이전 metric의 저장을 위한 PM(Path metric), 그리고 Traceback을 위한 TB(TraceBack) 및 LIFO(Last input first output) 블록으로 구성된다.

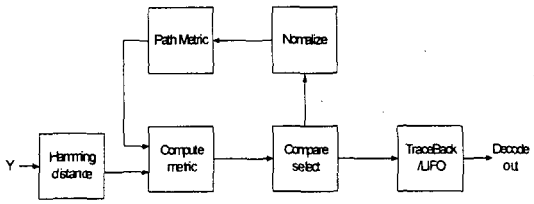


그림 5. 비터비 복호화기 셀의 구조
Fig 5. Architecture of viterbi decoder cell.

2. 제안한 Trace-Back의 구조

Register exchange 방식이나 메모리 pointing 방식들은 제한 길이가 작을 경우 효율적인 방법이 되지 못한다. 특히 메모리를 이용하는 방식의 경우에 작은 메모리를 잘게 나누어서 매크로 셀로 구현하기 어렵기 때문에 적합하지 않다. 따라서 본 논문에서는 register exchange 방식을 수정한 형태의 TB를 제안하는데 구조를 그림 6에 나타냈다. Register exchange 방식은 각 레지스터 앞에 MUX를 가지고 있어 한 상태가 천이할 때마다 전체 레지스터의 내용이 바뀌어야한다. 따라서 많은 H/W이 요구되고 전력 소비도 크다. 그러나 제안된 방식은 상태를 천이할 때마다 사용하는 MUX를 전부 없애고 그 대신에 LIFO의 끝에서 decision을 결정하는 제어기를 통해서 그 역할을 대신하게 하였다. 또한 복호화의 효율을 높이기 위해서 최소 상태를 찾아서 traceback 동작의 시작점으로 삼을 수 있는 Min-state Detector를 포함하여 구현하였다.

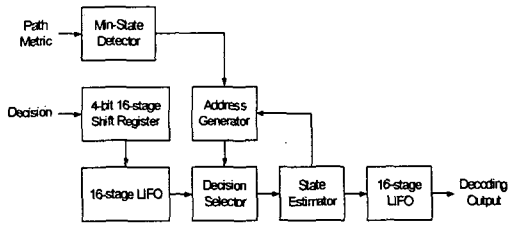


그림 6. 제안한 Trace-Back의 구조
Fig 6. Proposed architecture og Trace-Back.

3. 병렬처리를 위한 전체구조

그림 7에 다중 윈도를 이용해서 병렬처리를 수행함으로써 고 데이터율을 가지는 비터비 복호화기에 대한 전체 구조를 개략적으로 나타냈다. 그림 4에서 제시된 것과 같은 동작을 수용하기 위해 비터비 복호화기 셀(Viterbi Cell)의 병렬적인 확장형태를 가진다. 입력과 출력 FIFO 및 버퍼는 연속적인 동작에 대해서 데이터 및 시간적 손실을 없애고자 쌍을 이루고 있다.

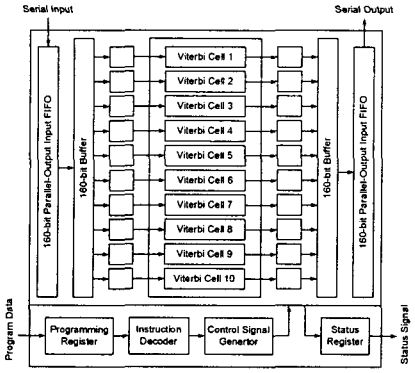


그림 7. 전체 비터비 복호화기의 구조
Fig 7. Global architecture of viterbi decoder.

4. 입력 분할 및 처리

다중 윈도를 버퍼링 하여 병렬처리 하는 방식을 그림 8에 나타냈다. 그림 8에 나타낸 것과 같이 한 번에 병렬적으로 처리할 수 있는 최대 입력열의 수를 10개로 하였다. 만일 더 많은 병렬처리를 요구할 경우에는 버퍼링의 수를 증가시키고 비터비 복호화기 셀을 추가시킴으로서 성능을 만족시킬 수 있다.

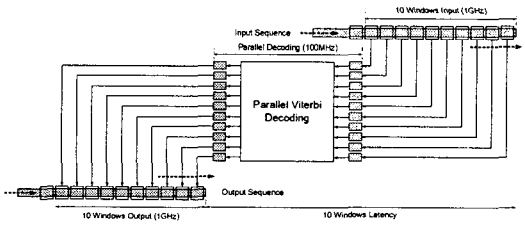


그림 8. 다중 윈도를 이용한 복호화 방식
Fig 8. Decoding method using multi-window.

1GHz로 입력되는 직렬입력을 16 비트 단위로 10개의 윈도를 취한 후 모든 입력에 대한 버퍼링이 완료되면 동시에 병렬적으로 비터비 복호화가 수행되고 그와 함께 새로운 입력에 대한 버퍼링이 수행된다. 복호화가 끝난 데이터는 일정 대기지연 시간 후에 다시 직렬로 출력된다. 이러한 세부적인 동작들도 전체적으로는 파이프라인화되어 그림 9와 같은 시간적인 손실이 없는 효율적인 동작 구조를 이룬다.

IV. 구현 및 검증

본 장에서는 하드웨어와 소프트웨어(S/W)의 구현을 비롯하여 전체적인 검증 및 응용을 위한 시스템의 구현 결과에 대해서 설명한다.

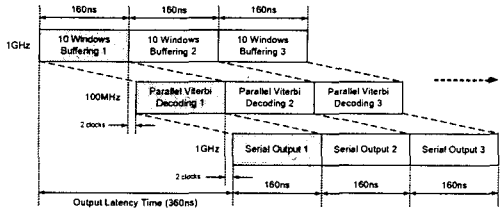


그림 9. 파이프라인을 통한 분할처리
Fig 9. Segment processing using pipeline.

1. 하드웨어 구현

2, 3장을 통해 소개한 H/W구현을 위해 Quartus II와 Synplify pro 환경을 사용하고 VHDL(VHSIC Hardware Description Language)을 이용하여 설계하였다[10]. 타겟 플랫폼으로 이용된 FPGA는 Altera의 APEX20KC EP20K1000CB652-7칩이다. 비터비 복호화기 셀은 1%(351 cells)의 LAB를 사용하여 최대 139MHz에서 안정적으로 동작할 수 있었다. 또한 비터비 복호화기 셀과 입출력 버퍼링을 위한 회로를 포함한 전체 비터비 복호화기는 23%(5679 cells)의 자원을 사용하면서 최대 1Gbps의 데이터 출력율을 가질 수 있었다.

비터비 복호화기 셀은 Synplicity의 Synplify pro를 통해 FPGA 합성되었고 RTL수준의 합성 결과를 그림 10에 보였다. 또한 그림 11에서는 합성된 비터비 복호화기 셀의 검증 결과를 나타내고 있는데 ModelSim 환경에서 수행되었다. 현재 그림에서 보이는 시뮬레이션 결과는 "0101110010100010"로 주어진 입력을 "00111000011001111110001011001110"으로 길쌈 부호화하고 "00111100011001111110000011001110"으로 에러를 삽입한 뒤 원래의 입력으로 제대로 비터비 복호를 수행하는 과정을 보이고 있다.

그림 12에 전체 비터비 복호화기를 RTL 수준으로 합성하여 나타냈고 그림 13은 시뮬레이션 결과를 나타낸다.

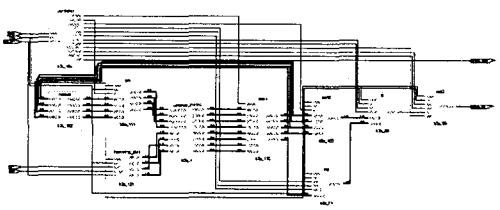


그림 10. 비터비 복호화기 셀의 RTL수준 합성도
Fig 10. RTL synthesis of viterbi decoder cell.

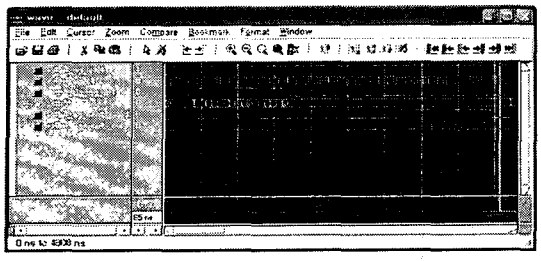


그림 11. 비터비 복호화기 셀의 시뮬레이션 결과
Fig 11. Simulation result of viterbi decoder cell.

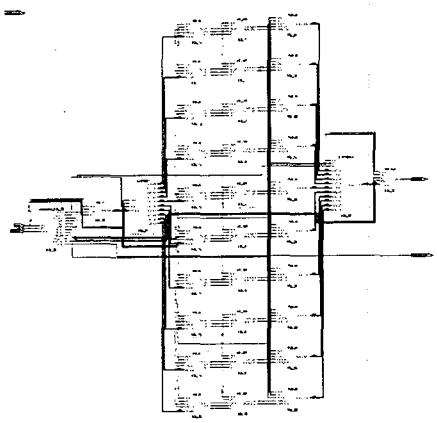


그림 12. 전체 비터비 복호화기의 RTL수준 합성도
Fig 12. RTL synthesis of global viterbi decoder.

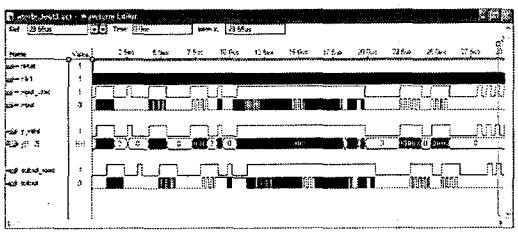


그림 13. 전체 비터비 복호화기의 시뮬레이션 결과
Fig 13. Simulation result of global viterbi decoder.

2. S/W 구현

HDL을 이용하여 구현된 비터비 복호기의 검증을 위한 상위 레벨 언어로써 LabVIEW를 이용하였다. 사용된 LabVIEW의 User Interface는 그림 14와 같다.

그림 14에서 오른쪽 위쪽에 나타난 그림은 Pepper 영상의 길쌈 부호화 결과이다. Raw형태의 파일로부터 입력받아 화면에 출력한 결과로써 이 그림의 정보를 FPGA 칩 내의 하드웨어 비터비 복호화기를 통해 복호화한 결과가 오른쪽 아래에 나타난 그림이다. 또한 이 그림은 전송 채널(Channel)에 의한 잡음이 없다고 가정하여 실험했을 경우에 한 비트의 오차도 발생하지 않고 완벽히 복호화를 수행한다.

3. 전체 시스템의 구조

본 장에서는 구현된 하드웨어 비터비 복호기의 검증을 위한 전체 시스템의 구성도를 설명한다.

컴퓨터, FPGA 보드, S/W, 그리고 PCI 인터페이스를 이용하여 전체 시스템을 구성하였다. 그림 15에 비터비 복호기의 검증을 위한 전체 시스템을 나타냈다. LabVIEW를 이용한 응용 S/W부, HDL 기반으로 구현된 비트비 복호화기를 맵핑한 FPGA 보드부, 그리고 둘 사이의 인터페이스를 위한 디지털 입출력(Digital I/O)부로 나누어진다. LabVIEW 프로그램은 파일형태의 길쌈 부호화된 영상을 입력받아서 TB의 Depth 단위로 영상을 재배열하고 Digital I/O를 거쳐서 FPGA에 입력시킨다. 이 모든 제어과정을 LabVIEW 기반의 프로그램이 담당한다. 입력된 길쌈 부호화 정보는 FPGA 내의 비터비 복호기에 의해서 복호화 과정을 거치고 복호화 결과가 완료되면 그 시점을 검출하여 LabVIEW 프로그램은 결과를 컴퓨터의 일정 메모리에 저장시킨다. 이 과정을 입력의 크기만큼 반복한 후에 전체 영상을 재구성하고 LabVIEW 프로그램을 이용하여 복호화 된 영상을 화면에 출력한다.

V. 결 론

본 논문에서는 시간적으로 다중 윈도를 통해 비터비 복호화되는 단위를 다중으로 버퍼링하고 병렬적으로 처리하는 비터비 복호화기를 구현한 후에 FPGA에 사상하고 응용 S/W와 함께 채널 코딩 시스템을 구현하였다. 연속적으로 입력되는 신호를 복호화 길이의 배수로 버퍼링하여 병렬적으로 복호화를 수행하는 방식으로 고속의 복호화를 이루었다. 비터비 복호화기 셀의 사용수에 비례하여 데이터 출력율을 얻을 수 있었고 입력 버퍼의 프로그래밍을 통해서 가변적인 데이터율을 얻을 수 있었다. 구현된 비터비 복호화기 셀은 ALTERA의 APEX20KC EP20K600CB652-7 FPGA에서 1%의 LAB를 사용하여 100MHz 이상에서 안정적으로 동작할 수 있었고 비터비 복호화기 셀과 입출력 버퍼링을 위한 회로를 포함한 전체 비터비 복호화기는 23%의 자원을 사용하면서 최대 1Gbps의 데이터 출력율을 가질 수 있었다. 구현된 비터비 복호화기는 고속 처리를 위한 다양한 통신 환경에서 사용될 수 있을 것으로 사료된다.

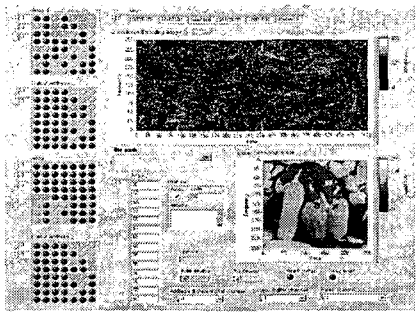


그림 14. LabView기반 비터비 복호화의 S/W 검증
Fig 14. Software verification of viterbi decoding using LabView.

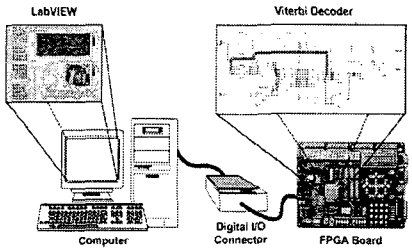


그림 15. 비터비 복호기의 전체 시스템
Fig 15. System of viterbi decoding.

감사의 글

본 연구는 2005학년도 광운대학교 교내 학술연구비 지원에 의해 수행한 연구로서, 관계부처에 감사드립니다.

참고문헌

[1] G. D. Forney, Jr., "The Viterbi algorithm," Proc. IEEE, Vol. 61, No. 3, pp. 268-278, Mar. 1973.

[2] A. J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," IEEE Trans. Inform. Theory, vol. IT-13, pp. 260 - 269, Apr. 1967.

[3] A. K. Yeung and J. Rabaey, "A 210-Mb/s, radix-4 bit-level pipelined Viterbi decoder," ISSCC Dig. Tech. Papers, pp. 88 - 89, Feb. 1995.

[4] J. K. Hinderling et al., "CDMA mobile station modem ASIC," IEEE J. Solid-State Circuits, vol. 28, pp. 253 - 260, Mar. 1993.

[5] Y. N. Chang, H. Suzuki, and K. K. Parhi, "A 2-Mb/s 256-State 10-mW Rate-1/3 Veterbi Decoder", IEEE Journal of Solid-State Circuits, Vol. 35, No. 6, June. 2000.

[6] H. Dawid, S. Bitterlich, and H. Meyr, "Trellis pipeline-interleaving: A novel method for efficient Viterbi decoder implementation," in Proc IEEE Int. Symp. Circuits and Systems, vol. 4, San Diego, CA, May 1992, pp. 1875 - 1878.

[7] J. Sparso, H. N. Jorgensen, E. Paaske, S. Pedersen, and T. Rubner-Petersen, "An area-efficient topology for VLSI implementation of Viterbi decoders and other shuffle-exchange type structures," IEEE J. Solid-State Circuits, vol. 26, pp. 90 - 96, Feb. 1991.

[8] C. M. Rader, "Memory management in a Viterbi decoder," IEEE Trans. Commun., vol. 29, pp. 1399 - 1401, Sept. 1981.

[9] R. Cyper and C. B. Shung, "Generalized trace-back techniques for survivor memory management in the Viterbi algorithm," Proc. GLOBECOM, vol. 2, pp. 1318 - 1322, Dec. 1990.

[10] IEEE Std. 1076-1993, "VHDL language reference manual", IEEE Press, New Jersey 1994.

저자소개

서영호(Young-Ho Seo)

1999년 2월 광운대학교 전자재료공학과 공학사
2001년 2월 광운대학교 공학석사
2004년 8월 광운대학교 전자재료공학과 공학박사
2000년 3월~2001년 12월 인티스닷컴(주) 연구원
2003년 6월~2004년 6월 한국전기연구원 연구원
2004년 10월~현재 유한대학교 연구교수
※관심분야 : Image Processing/Compression, 워터마킹, 암호학, FPGA/ASIC 설계

김동욱(Dong-Wook Kim)

1983년 2월 한양대학교 전자공학과 공학사
1985년 2월 한양대학교 공학석사
1991년 9월 Georgia공과대학 전기공학과 공학박사
2000년 3월~2001년 12월 인티스닷컴(주) 연구원
1992년 3월~현재 광운대학교 전자재료공학과 교수
※관심분야 : 디지털 VLSI Testability, VLSI CAD, DSP 설계, Wireless Communication