VHDL을 이용한 고속 움직임 예측기 설계

김 진 연, 박 노 경, 진 현 준, 윤 의 중, *박 상 봉
호서대학교 전기공학부, *세명대학교 정보통신학과
전화 : 041-549-5085

Design of Fast Search Algorithm for The Motion Estimation using VHDL

Jean-Youn Kim, No-Kyung Park, Hyun-Jun Jin, Eu-Jung Yoon, *Sang-Bong Park
School of Electrical Engineering , Hoseo University
E-mail : esdamars@asic.hoseo.ac.kr

Abstract

Motion estimation technique has been used to increase video compression rates in motion video applications. One of the important algorithms to implement the motion estimation technique is search algorithm. Among many search algorithms, the H.263 adopted the Nearest Neighbors algorithm for fast search. In this paper, motion estimation block for the Nearest Neighbors algorithm is designed on FPGA and coded using VHDL and simulated under the Xilinx foundation environments. In the experiment results, we verified that the algorithm was properly designed and performed on the Xilinx FPGA(XCV300Q240)

I. 서론

오늘날 통신 기술이 나날이 발전하고 있지만, 디지털 영상신호가 방대한 자료를 가지고 있기 때문에 저장, 처리 및 전송하기 위해서는 보다 많은 데이터 압축이 요구되고 있다. 영상처리에서는 픽쳐간 상관 관계를 이용하여, 픽쳐간의 움직임 예측을 통한 시간적 중복성을 제거하여 많은 데이터를 압축한다.

대부분의 비디오 코딩 시스템에서 움직임 예측/보상(Motion Estimation/compensation)으로는 화소 순환 알고리즘(Pel Recursive Algorithm), Frequency - domain 알고리즘, Gradient 알고리즘, 블록 정합 알고리즘(Block Matching Algorithm)들이 쓰이고 있다.[1] 이 중에 블록 정합 알고리즘은 현재 프레임의 블록 움직임을 과거 프레임의 탐색 영역에서 예측하는 것으로 일반적으로 많이 사용되고 있다. 블록 정합 알고리즘을 이용할 때 많은 계산량이 필요하다. 많은 계산량을 줄이기 위해 고속 탐색 알고리즘이 사용되는데, 이런 요구에 의해 ITU-T에서는 해결책으로 H.263에서는 고속탐색 알고리즘으로는 Nearest Neighbors 탐색법이 제안되었다.[3],[4] 그림 1은 H.263 엔코더 블록을 보여주고 있다. 그림에서 선택된 블록은 움직임 예측을 하는 부분을 보여주고 있다. 즉 화상통신 같은 실시간을 요구하는 분야에서는 움직임 예측을 고속으로 처리하는 것이 필요하다.

본 논문은 H.263과 같은 동영상 코덱에 적용할 수 있는 움직임 예측기를 하드웨어로 설계하기 위해서, Nearest Neighbors 탐색 알고리즘을 이용한 고속 움직임 예측기를 FPGA로 설계하였다. 또한 설계한 움직임 예측기의 function과 timing 시뮬레이션을 위해 VHDL로 코딩하고 Xilinx Foundation을 이용하여 설계 및 검증하였다.

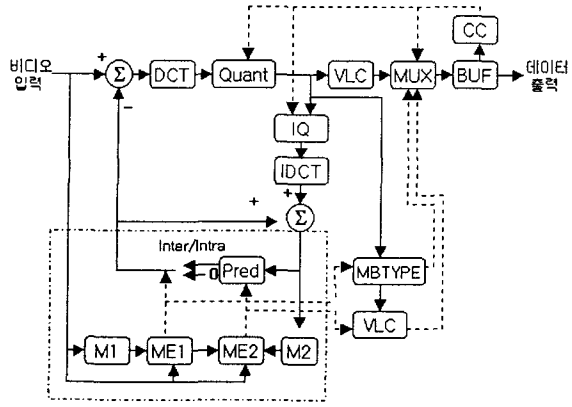

그림 1. H.263 엔코더 블록도

Ⅱ. 본론

2.1 Nearest Neighbors 탐색법

저비트율 코딩을 목적으로 고안된 탐색법이 Nearest Neighbors 탐색법이다. 그림 2에서처럼 기준점에서 마름모 형태의 상하좌우 네 점에 대해서 MB(Macro Block)내의 화소들의 차의 절대값들의 합인 SAD(Sum of absolute difference) 값을 계산한다. 그 중에서 최소 SAD 값을 갖는 점을 찾아 이 점을 새로운 기준점으로 설정하여, 다시 그 점의 상하좌우 네 점에 대해서 최소 SAD 값을 찾는다. 이와 같은 과정은 이전 단계에서 구한 최소 SAD 값이 현재 단계에서 구한 최소 SAD 값보다 작을 때까지 반복된다. 그림 2의 우측은 2단계까지 탐색하는 예를 보여준다.

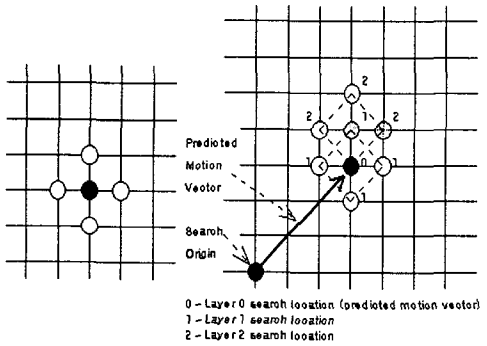

그림 2. Nearest Neighbors 탐색창

또한 Nearest Neighbors 탐색법은 시작점으로 영벡터(zero vector) 대신에 먼저 구한 인접 매크로 블록들의 동작 벡터들의 중간 값(median value)을 예측 동작 벡터(predicted motion vector)로 사용함으로써 그 정확도를 더욱 높일 수 있다.

그러나, Nearest Neighbors 탐색법은 주위 네 개의 점들로만 구성된 작은 후보화소 영역을 고려함으로써 움직임이 빠른 영상의 경우 지역적 최소값에 빠질 가능성이 커지게 되어 그 성능이 떨어질 것으로 생각된다. 그러나 화상통신같은 움직임이 적은 영상에서는 큰 성능저하 없이 적은 계산량으로 사용할 수 있다.

2.2 고속 움직임 예측기 블록

고속 움직임 예측기 시스템 구성은 그림 3과 같이 현재 프레임과 과거 프레임을 저장하고 있는 메모리와 이를 인터페이스 할 수 있는 Memory Interface 블록, SAD연산을 하는 SAD 블록, 움직임 벡터를 생성하는 Vector Generate 블록, 이것들을 제어하는 Control Unit 블록으로 구성되어 있다. .

Memory Interface 블록은 Memory 에 저장되어 있는 현재 영상(Cur_Image)와 참조 영상(Ref_Image)을 가져오는 블록이다. SAD 블록은 MB간의 SAD 값을 계

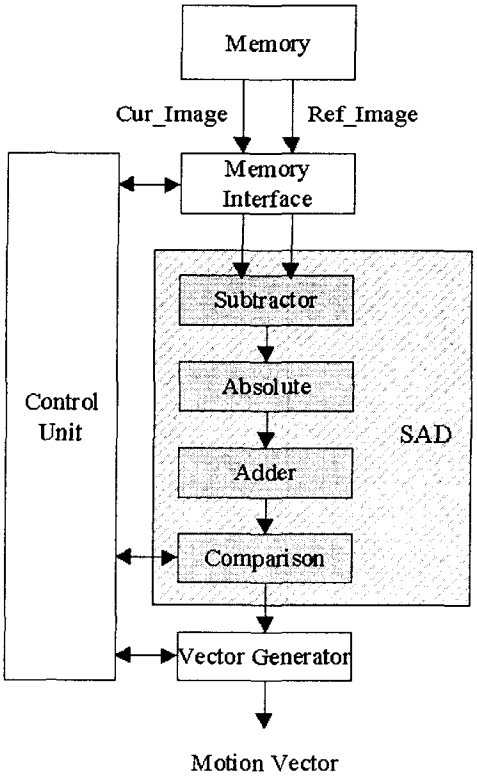

그림 3. 고속 움직임 예측기 블록도

산하고, 최소 SAD 값을 넘겨주는 역할을 한다.

그리고 Vector Generator 블록은 SAD 블록으로부터 최소 SAD값을 가지는 블록의 메모리 어드레스를 벡터로 변환하는 블록이다. 마지막으로 Control Unit 블록은 앞의 블록들을 제어할 수 있는 신호를 처리하는 블록이다.

Ⅲ. FPGA 구현 및 모의실험

이 장에서는 2장에서 간단히 설명한 H.263 엔코더에 사용되는 Nearest Neighbors 탐색을 이용한 움직임 탐색 예측기의 VHDL 구현에 대하여 설명한다. 소단원에서는 각각의 블록 동작 설명과 시뮬레이션을 보여준다.

3.1 Memory Interface 블록 설계

Memory Interface 블록은 현재 영상이 저장된 RAM의 어드레스와 이 어드레스로부터 상하좌우 네 개의 후보화소를 가리키는, 참조 영상이 저장된 RAM의 어드레스를 결정한다. 제어 블록으로부터 네 개의 후보화소 중 최소 SAD를 갖는 화소가 결정되면 다시 이 어드레스를 갱신하여 기준 화소를 정하고 주위의 네 개의 후보화소의 어드레스를 RAM으로 넘겨준다. 그리고 어드레스가 가리키는 RAM의 화소 데이터를 받아서 SAD 블록으로 넘겨주게 된다. 이 과정을 현재 MB에 대한 움직임 벡터를 결정할 때까지 계속하게 된다. 본 논문에서는 FPGA의 구조가 RAM을 구현하는 것에 매우 비효율적이기 때문에 RAM을 직접 구현하지 않는 대신에 미리 설계된 RAM 모듈을 가지고 시뮬레이션 하였다.

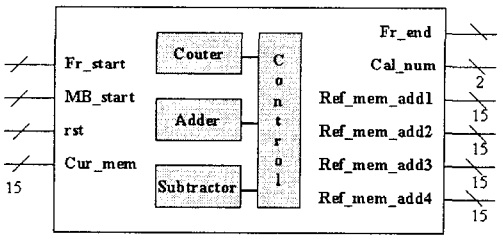

그림 4. Memory Interface 블록

3.2 SAD 블록 설계

SAD 블록은 아래와 같은 SAD식을 처리하도록

$$SAD(x,y) = \sum_{i=1,j=1}^{16,16} | CurImage_{i,j} - RefImage_{x+i,y+j} |$$

Memory Interface 블록으로부터 화소값을 받은 후 네 개의 화소순대로 Subtractor, Absolute, Adder블록에서 처리하도록 구현하였다. 그리고 Comparison에서는 네 개의 후보화소에 대한 SAD값 들을 비교하여 그 중 최소 SAD값을 결정하여 Control Unit 블록으로 출력하도록 구현하였다.

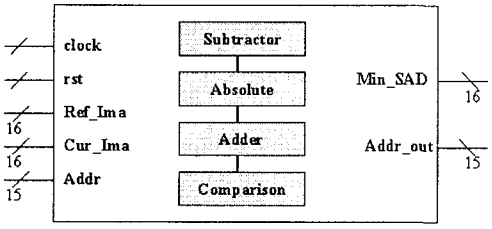

그림 5. SAD 블록

3.3 Vector Generator 블록 설계

Vector Generator 블록은 현재 MB에서 최소 SAD값을 가지는 화소를 결정하게 되면 기준 화소에서부터의 어드레스와 최소 SAD값을 가지는 화소의 어드레스의 차를 가지고 MB의 크기와 비교를 해서 움직임 벡터의 크기와 방향을 X좌표와 Y좌표의 벡터로 출력하는 블록이다. H.263 규격과 사용 가능하도록 ±15 화소 크기의 탐색영역 내에서 움직임 벡터를 탐색한다.

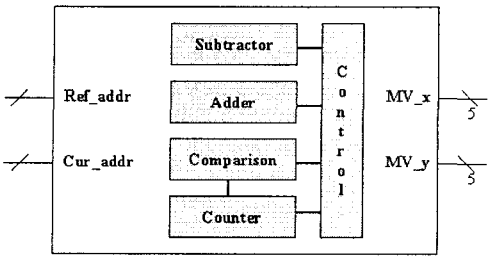

그림 6. Vector Generator 블록

3.4 고속 움직임 예측기 구현 및 시뮬레이션

하드웨어는 VHDL을 이용하여 설계하고, Xilinx Foundation의 FPGA Express을 이용하여 설계를 검증하였다.

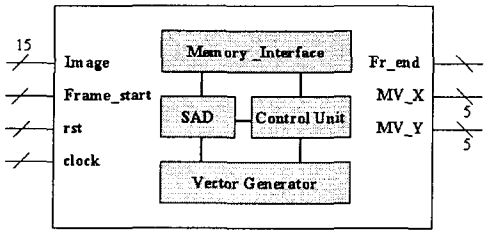

그림 7. 고속 움직임 예측기 블록

이 시스템의 입력 영상 포맷은 각 화소 값들은 16bit값을 가지고, QCIF 포맷을 처리할 수 있도록 설계하였다. 그리고 출력은 움직임 벡터의 크기가 ±15로 제한되도록 signed 5bit 값을 출력한다.

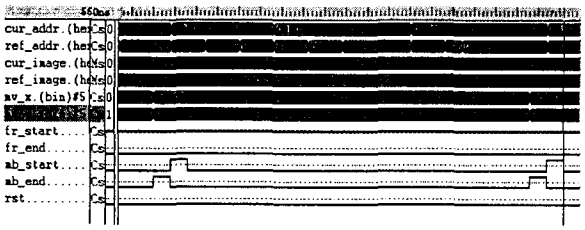

그림 8. 고속 움직임 예측기 Function 시뮬레이션

그림 8에서는 고속 움직임 탐색 전체 블록에 대한 시뮬레이션이다. MB에서 처음 시작하게 되면 현재 영상의 화소값과 참조 영상의 화소값을 가지고서 최소 SAD를 찾을 때까지 수행 후에 벡터값을 출력하는 것을 볼 수가 있다. 그림 9에서는 고속 움직임 탐색 예측기를 구현한 VHDL 코드를 보여준다. 그림 10은 타켓 디바이스인 Xilinx FPGA(XCV300PQ240)에서 구현 한 것을 보여준다.

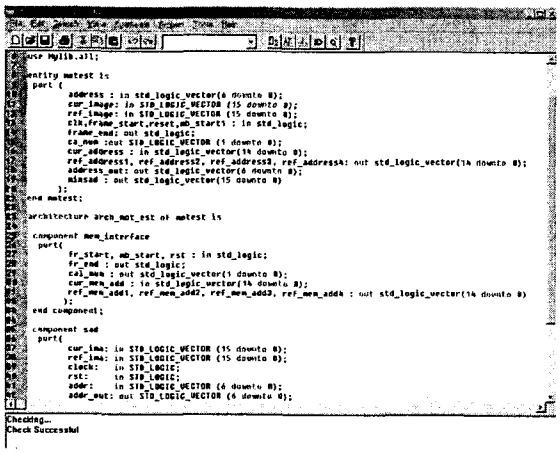

그림 9. 고속 움직임 예측기 VHDL 코드

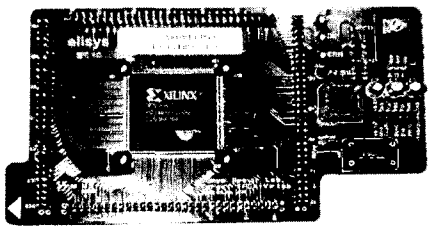

그림 10. Xilinx FPGA(XCV300PQ240) 보드

Ⅳ. 결론

본 논문에서는 블록 정합 알고리즘에서 움직임 벡터를 고속으로 찾는 알고리즘 중에서 H.263 규격에 사용되고 있는 Nearest-Neighbors 탐색 알고리즘을 VHDL을 이용하여 구현하였다. 전체적인 구성 방식은 Top-down 방식으로 구현하여 설계 및 시뮬레이션 하였고 움직임 예측이 Xilinx FPGA(XCV300PQ240)에서 이상없이 동작함을 확인할 수 있었다. 구현된 블록은 H.263 엔코더뿐이 아니라 MPEG-4와 같은 동영상 코덱 분야에도 적용할 수 있으리라 사료된다.

Ⅴ. 참고 문헌

[1] Borko Furht, Joshua Greenberg, and Ray mond Westwater, "Motion Estimation Algorithms for Video Compression", Kluwer Academic Publishers, 1997

[2] K. R. Rao and J. J. Hwang, "Techniques & Standards for Image · Video & Audio Coding", Prentice Hall, 1996

[3] Guy Cote, Michael Gallant and Faouzi Kossentini, "Efficient Motion Vector Estimation and Coding for H.263-Based Very Low Bit Rate Video Compression"

[4] Guy Cote, Michael Gallant and Faouzi Kossentini, "An Efficient Computation-Constrained Block-Based Motion Estimation Algorithm for Low Bit Rate Video Coding"

[5] David Van Den Bout, "The Practical XILINX Designer Lab Book", Prentice-Hall

[6] Charles H. Roth, Jr. "Digital System Design Using VHDL", PWS Publishing Company