# MPEG-4 AVC를 위한 고속 인터 예측기의 하드웨어 구현

준회원 임 영 훈*, 이 대 준**, 정회원 정 용 진***

# Hardware Implementation of a Fast Inter Prediction Engine for MPEG-4 AVC

Young hun Lim* Dae joon Lee** *Associate Member*, Yong jin Jeong*** *Regular Member*

요 약

본 논문에서는 MPEG-4 AVC 부호화기를 위한 고속 인터 예측기에 대한 하드웨어 구조를 제안한다. 동영상 압축기술인 MPEG-4 AVC 부호화기의 전체 구성 중에서 핵심 부분인 인터 예측기를 1/4화소 단위로 움직임 추정을 수행 할 수 있도록 하드웨어 구조를 설계하였으며 이를 위해 블록 조각화, 움직임 추정, 움직임 보정의 기본적인 구조를 구성하고 실시간 동영상 부호화를 할 수 있도록 인터 메모리와 1/4화소 단위 고속 블록 계산기 등을 이용하였다. 구현된 전체 모듈은 Altera Excalibur 디바이스와 Xilinx Virtex2 디바이스를 이용한 FPGA 구성을 통해 검증하고 삼성 STD130 0.18um CMOS Cell Library를 이용하여 합성 및 검증을 하였다. 이렇게 검증된 구조의 성능은 ASIC으로 구현할 경우 최대 동작 주파수가 약 62.5MHz이며 성능은 QCIF크기의 영상데이터를 기준으로 초당 약 88프레임의 인터 예측을 수행할 수 있다. 본 성능은 하드웨어 기반의 MPEG-4 AVC 실시간 부호화기를 설계하기에 적합한 구조임을 보여준다.

Key Words : H.264, MPEG4 AVC, Inter Prediction, Motion Estimation, Motion Compensation

ABSTRACT

In this paper, we propose an advanced hardware architecture for the fast inter prediction engine of the video coding standard MPEG-4 AVC. We describe the algorithm and derive the hardware architecture emphasizing and real time operation of the quarter_pel based motion estimation. The fast inter prediction engine is composed of block segmentation, motion estimation, motion compensation, and the fast quarter_pel calculator. The proposed architecture has been verified by ARM-interfaced emulation board using Excalibur & Virtex2 FPGA, and also by synthesis on Samsung 0.18 um CMOS technology. The synthesis result shows that the proposed hardware can operate at 62.5MHz. In this case, it can process about 88 QCIF video frames per second. The hardware is being used as a core module when implementing a complete MPEG-4 AVC video encoder chip for real-time multimedia application.

* 엠텍 비젼(limyh14@chol.com),
** 광운대학교 전자통신공학과 실시간구조 연구실(foot286@explore.kw.ac.kr), *** 광운대학교 부교수(yjjeong@daisy.kw.ac.kr)
논문번호 : KICS2004-11-291 , 접수일자 : 2004년 11월 25일
※본 연구는 광운대학교 2004 교내학술연구 및 IDEC/SIPAC 과 IT-SoC 사업단의 지원으로 이루어졌습니다.

## Ⅰ. 서 론

최근 디지털 멀티미디어 방송을 위한 지상파 및 위성파 DMB 표준이 제정되었는데, 고화질의 동영상 서비스를 제공하기 위한 비디오 표준으로 MPEG-4 AVC가 채택되었다[1][2][3]. 이러한 MPEG-4 AVC 부호화기 설계에서 가장 핵심이 되고 있는 기술 중에 하나는 인터 예측(Inter Prediction)의 움직임 추정기와 움직임 보정기의 설계이다. 이러한 인터 예측기를 설계하기 위해서는 움직임 추정 알고리즘을 필요로 하게 되는데 움직임 추정 알고리즘들은 여러 가지 방안으로 연구되어왔다.

많은 움직임 추정 알고리즘 중에서 본 논문에서 사용되어진 다해상도 움직임 추정 방식은 다른 움직임 추정 알고리즘들과 비교하여 우수한 예측 성능과 빠른 계산 속도를 가진다[4]. 이러한 움직임 추정 알고리즘에 블록 조각화, 움직임 보정을 추가하여 인터 예측기를 설계한다. 또한 고속의 인터 예측기를 구현하기 위해 추가적인 하드웨어적 기법을 사용하여 하나의 매크로블록을 4화소x4화소부터 16화소x16화소의 가변 블록들로 조각화하고 1/4화소 단위의 움직임 추정을 하여 MPEG-4 AVC 부호화기의 성능을 높일 수 있는 구조를 설계한다.

이후 본 논문의 구성은 다음과 같다. 먼저 2장에서는 인터 예측의 기본 기능에 대해서 설명한다. 3장에서는 고속 인터 예측기의 구조에 대해 설명하고 4장에서는 구현된 고속 인터 예측기의 전체 모듈에 대한 동작 검증 및 성능을 분석하며 마지막으로 5장에서 결론을 맺는다.

## Ⅱ. 인터 예측의 기본 기능

인터 예측의 기본 기능은 영상 데이터의 공간적 중복성을 줄여서 압축 효율을 높이는 인트라 예측과 달리 시간적 중복을 줄임으로써 동영상 압축 효율을 높이는데 있다. MPEG-4 AVC 이전의 동영상 압축기술에서는 기본적으로 매크로블록단위의 움직임 추정 및 보정을 이용하여 인터 예측을 하지만, MPEG-4 AVC에서는 이전의 매크로블록 단위의 움직임 추정뿐만 아니라 블록과 서브블록 단위로 조각화를 하여 움직임 추정을 할 수 있도록 표준화되어 있다. 이러한 방법을 이용하여 인터 예측에서는 조각화되어진 블록 단위의 영상 데이터와 기준 영상간의 시간적 움직임의 차이를 계산하고 영상이 움직인 위치를 추정하게 되는데 이것을 움직임 벡터라 하고 이렇게 추정되어진 움직임 벡터를 이용하여 기준 영상과의 차분을 취하여 영상을 보정하게 되는데 이것을 움직임 보정이라 한다. 그러므로 인터 예측을 위해서는 크게 블록 조각화, 움직임 추정, 움직임 보정과 같은 세 가지의 구조가 필요하게 된다.

### 2.1 블록 조각화

MPEG-4 AVC에서는 기본적으로 매크로블록(16화소x16화소) 단위로 인터 예측을 수행하게 된다. 그러나 더욱더 정확한 예측을 위해서 예측 범위를 분할하여 움직임 추정을 하는데 이것을 위해 먼저 블록(8화소x8화소)을 16화소x16화소에서 4화소x4화소 단위까지 조각화를 하여야한다. 이때 사용하는 기법이 블록 조각화이다. 하나의 매크로블록은 먼저 16화소x16화소, 8화소x16화소, 16화소x8화소, 8화소x8화소 단위의 4가지 방법으로 블록 조각화가 수행되며, 8화소x8화소로 조각화가 될 시에는 다시 8화소x8화소, 4화소x8화소, 8화소x4화소, 4화소x4화소 단위의 4가지 방법으로 서브블록 조각화가 수행되어진다. 아래 그림 1은 각각의 방법으로 조각화 되는 단위를 나타낸다. 이렇게 하나의 매크로블록에 대한 블록과 서브블록(4화소x4화소) 단위로 조각화를 하기 위해서는 모드 선택 알고리즘을 필요로 한다.

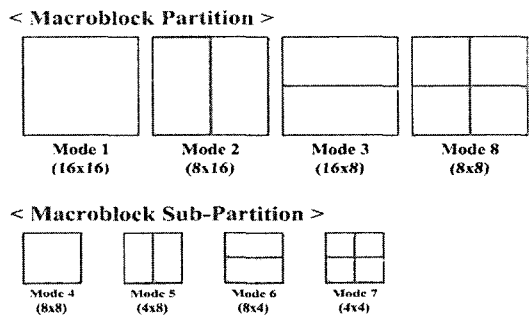



그림 1. 블록 조각화를 위한 각 모드

### 2.2 고속 움직임 추정

일반적으로 움직임 벡터의 정밀성은 움직임 추정 성능에 주요한 영향을 미치게 된다. 본 논문에서는 해상도 변화된 레벨에서 단일 모션 벡터 후보가 아닌 다중 모션 벡터 후보들과 모션 벡터 영역에서 공간적인 관계에 대한 움직임 벡터 후보를 사용하여 움직임 추정을 하고, 또한 인터 메모리와 고속 움직임 추정을 위한 다양한 방법을 적용하여 인터 예측에 대한 정밀성과 계산 속도를 높이고자 한다.

2.2.1 다해상도 움직임 추정 알고리즘

다해상도 움직임 추정의 기본 기능은 전체를 하나의 방법으로 움직임 추정을 하는 전역 움직임 추정과 달리 영상 데이터의 해상도를 구분하여 움직임 추정을 하게 된다[4]. 움직임 추정을 위해서 입력되어진 영상데이터는 세 개의 해상도 변화된 레벨로 구성되는데 낮은 해상도 레벨에서 최소 SAD (Sum of Absolute Difference)값을 기준으로 중간 해상도 레벨에서 계산 되어질 두개의 모션 벡터 후보들을 얻는다. 중간 해상도 레벨에서는 낮은 해상도 레벨에서 선택된 두개의 움직임 벡터 후보와 이전 영상 프레임의 원래 해상도 레벨에서 공간적 움직임 벡터 4개 값의 평균값으로 얻어진 다른 하나의 후보를 계산하여 원래 레벨의 마지막 최종 벡터의 후보로서 사용한다. 마지막으로 움직임 벡터 후보를 이용하여 원래 해상도 레벨에서 최종 움직임 벡터의 선택과 함께 최종적인 SAD값도 계산하게 된다. 그림 2에서는 최종적인 움직임 벡터와 SAD 값을 계산하기위한 다해상도 움직임 추정 순서를 보여준다.

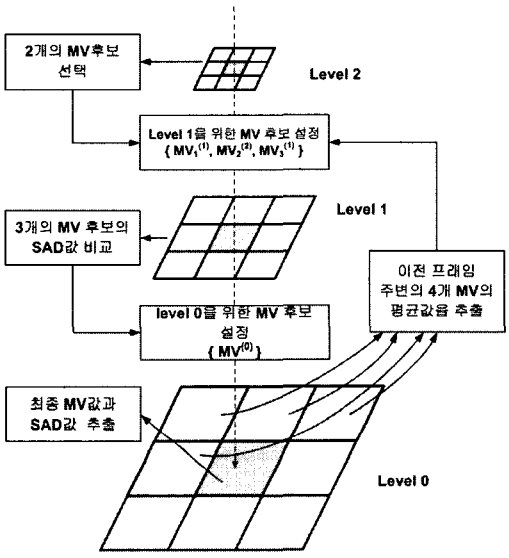



그림 2. 다해상도 움직임 추정 흐름도

2.2.2 모드별 움직임 추정

인터 예측에서의 움직임 추정은 앞서 설명한 블록 조각화에서 이루어진 16화소x16화소부터 4화소x4화소까지의 선택된 모드를 이용하여 각 모드별 움직임 추정을 하게 된다. 그러므로 추정되어지는 움직임 벡터와 SAD값의 수는 각각 최대 16개가 필요하게 되며 최종적으로 움직임 보정에서 사용되는 정보가 된다.

2.2.3 1/4화소 단위의 움직임 추정

영상 데이터에 대한 움직임 추정을 할 때의 기본적인 추정단위는 정화소단위로 움직임 추정을 하게 된다. 하지만 좀더 나은 동영상 압축을 위해 반화소 단위, 1/4화소 단위의 움직임 추정을 하게 되는데 MPEG-4 AVC Baseline Profile 이상에서는 1/4화소 단위의 움직임 추정을 표준으로 정하고 있다. 또한 기본적으로 1/4화소 단위의 움직임 추정 시에 6단 FIR 필터(6-tap FIR Filter)를 사용하여 움직임 추정에 대한 성능을 높이게 된다. 그러나 1/4화소 단위 움직임 추정기의 구조는 정화소 단위 움직임 추정기보다 복잡한 구조로 구성되어있기 때문에 하드웨어 구현의 어려움을 해결하기 위해서 특수한 레지스터 구조와 전처리 모듈, 블록 계산기 등을 이용하여 움직임 추정기의 구조를 설계하고 구현한다.

### 2.3 움직임 보정

움직임 보정이란 움직임 추정에 의해서 생성된 움직임 벡터를 사용하여 영상 데이터의 움직인 위치만큼 기준영상을 옮겨서 움직임을 보정하는 것을 말한다. MPEG-4 AVC에서의 움직임 보정은 블록 조각화와 움직임 보정을 통틀어 이르고 있으나 본 논문에서는 블록 조각화와 움직임 보정을 분리하여 구조를 설계하여 인터 예측기의 성능을 향상시켰다. 또한 움직임 추정에서 사용한 전처리 모듈과 블록 계산기를 공유하여 별도의 하드웨어 모듈의 추가 없이 모드별 움직임 보정을 수행한다.

## Ⅲ. 고속 인터 예측기의 구조

고속 인터 예측기의 구조는 크게 인터 메모리 (Inter Memory), 블록 조각화(Block Segmentation), 움직임 추정(Motion Estimation), 움직임 보정(Motion Compen -sation)의 4개의 구조로 나누어져 있다. 가장 먼저 수행되어지는 것은 기준영상 데이터와 원영상 데이터를 인터 메모리에 저장하는 것이다. 이렇게 저장되어진 데이터들을 이용하여 각 모드별 움직임 추정을 위해 블록 조각화를 하게 되고 조각화된 각 모드값들은 움직임 추정과 움직임 보정에서 사용되어질 데이터로 입력되게 된다. 움직임 추정은 입력된 모드값과 인터 메모리로부터의 원영상, 기준영상을 이용하여 실제 움직임 벡터를 추정

하게 된다. 움직임 추정을 위해서는 SAD값의 계산을 위한 BSU(Basic Search Unit)와 1/4화소 단위 고속 블록 계산기가 필요하게 된다.

움직임 추정이 이루어 진 후 추정된 움직임 벡터 값과 SAD값은 블록 조각화에서 생성된 모드값과 함께 움직임 보정기의 입력으로 들어간다. 움직임 보정에서는 입력되어진 정보와 인터 메모리로부터의 기준영상 데이터를 이용하여 움직임이 보정된 데이터를 생성하게 되는데 이 움직임 보정된 데이터는 이 후 MPEG-4 AVC 전체 모듈에서 원영상값과 차분되어 예측값으로 사용되어 진다. 그림 3은 인터 예측기의 구조를 나타내고 있다.

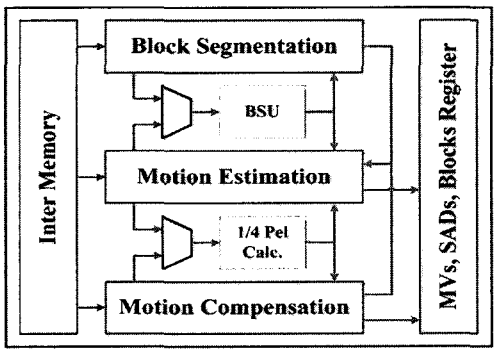



그림 3. 인터 예측을 위한 기본 구조

### 3.1 인터 메모리

고속의 인터 예측기를 구현하기 위해서는 영상 데이터의 읽고 쓰는 속도가 무엇보다도 중요하다. 이러한 문제를 해결하기 위하여 크게 세 가지의 방법을 사용하였다. 첫 번째는 하나의 매크로블록 단위의 영상데이터를 메모리에 저장 시에 추가의 클록 사용 없이 해상도 변화된 값을 저장하는 기법을 사용하였다. 두 번째로는 블록 조각화, 움직임 추정을 위해 인터 메모리로부터 데이터를 읽어 올 때 한번에 BSU 단위인 128비트 단위로 읽어 올 수 있도록 데이터 추출 방식을 사용하였다. 16개의 영상 데이터를 동시에 추출하기 위해서는 각각 메모리 블록에 저장된 영상 데이터를 출력 주소 생성기를 사용하여 한 클록에 출력할 수 있도록 구성하였다. 세 번째로는 가상 주소 방식을 사용하였는데 해상도 변화된 레벨 2에서 추정되어진 움직임 벡터 후보들을 사용하여 레벨 1과 레벨 0에서 움직임 추정을 할 때에 영상 데이터의 주변 8개의 매크로블록의 끝부분에서의 검색 오류를 피하고 검색력을 높이기 위한 방식이다. 이 방식을 사용하면 실제 메모

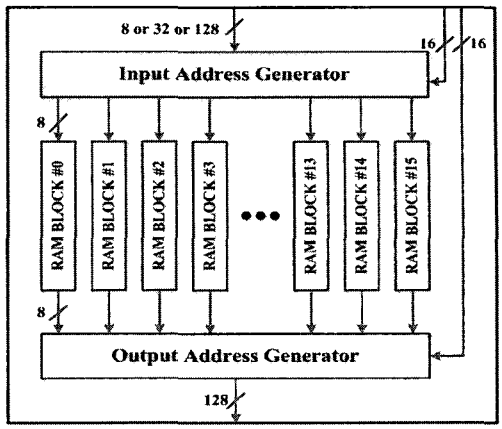



그림 4. 인터 메모리의 구조

리의 증가 없이 각 레벨간의 움직임 추정위치에 따른 검색능력이 증가하게 된다[5]. 그림 4는 인터 메모리의 구조를 나타낸다.

### 3.2 블록 조각화

블록 조각화를 위해서는 모드 선택 알고리즘이 필요하게 되는데 임계점(Threshold)값과 매크로블록의 4분할에 대한 영상 데이터의 합을 사용하여 모드를 선택할 수 있도록 하였다. 아래의 알고리즘에서 알 수 있듯이 블록단위와 서브블록단위의 모드 선택을 위해 같은 방법을 사용하였으며 블록 조각화를 수행한 후에 각각의 단위별로 8개의 모드가 생성이 되어진다.

블록 조각화를 위한 모드 선택 알고리즘

```
if ( |X0 - X2| < TH ) {
    if ( |X1 - X3| < TH ) {
        if ( |X0 - X1| < TH ) Mode 1 or Mode 4
        else Mode 2 or Mode 5
    }
    else Mode 8 or Mode 7
}
else {
    if (( |X0 - X1| < TH ) &&
        ( |X2 - X3| < TH )) Mode 3 or Mode 6
    else Mode 8 or Mode 7
}
X0~X3 : High or Low Block Position, TH : Threshold
```

### 3.3 고속 움직임 추정

고속 움직임 추정기는 다해상도 움직임 추정 알고리즘을 사용하고 제안된 기능들을 추가하여 고속의 동영상 부호화가 가능하도록 하드웨어 구조를

개선하였다. 추가된 기능으로는 모드별 움직임 벡터 검색 및 SAD값 추출 기능과 6단 FIR 필터를 이용한 1/4화소 단위의 움직임 추정, 블록 계산기를 위한 십자형 레지스터 구조, 1/4화소 단위의 움직임 추정을 위한 고속 블록 계산기, 공유 비교기를 사용한 각 모드별 검색, 움직임 추정 레벨 1, 0을 위한 위치 생성기가 있다. 이러한 기능들이 추가됨으로서 고속의 동영상 부호화가 가능하게 할 수 있는 고속 움직임 추정을 할 수 있게 된다.

움직임 추정의 방법은 우선 블록 조각화에 의하여 생성된 모드에 따라 움직임 추정의 비교기를 통하여 움직임 벡터와 SAD값이 추출되어진다. 비교기는 정화소단위의 움직임을 추정하는 레벨 2, 1의 비교기와 1/4화소 단위의 움직임을 추정하는 레벨 0의 비교기로 나누어지는데 이러한 비교기를 통하여 16개로 나누어진 서브블록 단위를 각각의 모드별로 움직임 추정하게 된다. 그림 5는 움직임 추정을 위한 비교기 구조를 나타낸다.

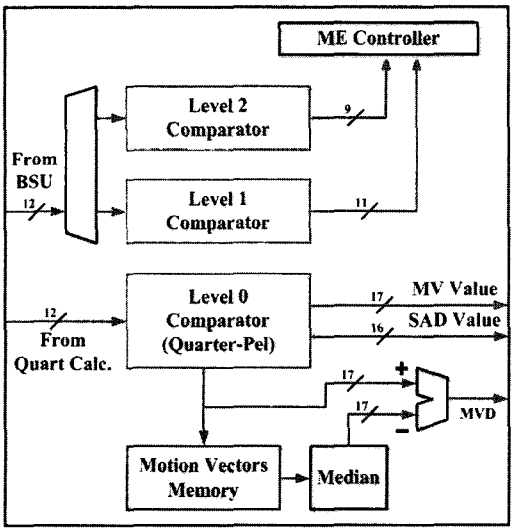



그림 5. 움직임 추정의 비교기 구조

### 3.3.1 Basic Search Unit(BSU)의 구조

고속 인터 예측에서 사용되는 BSU는 16개의 PE (Process Element)로 구성되어 있으며 현재 프레임과 이전 프레임의 화소값에 대한 차이값을 절대값으로 변환하여 16개의 값을 모두 더하는 방식으로 계산되어진다. BSU의 구조는 최장 지연 시간을 최소화하기 위해 CPA를 사용하지 않고 CSA와 CLA를 사용하여 구성하였다. 그림 6은 BSU의 구조를 나타내고 있다.

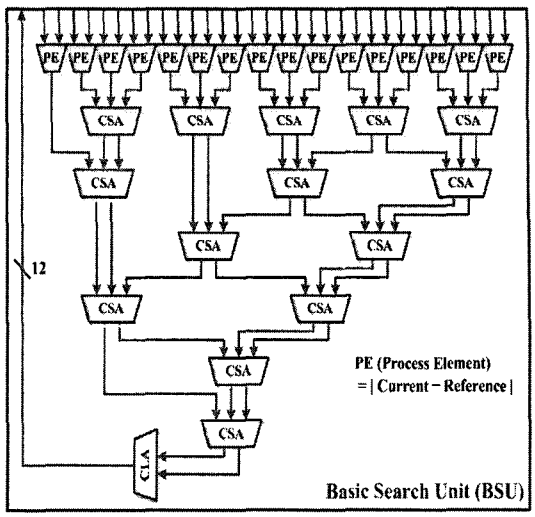
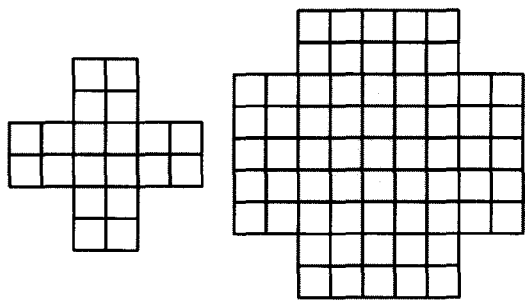



그림 6. BSU의 구조

### 3.3.2 블록 계산기를 위한 십자형 레지스터 구조

6단 FIR 필터를 이용하여 1화소에서의 1/4화소 단위의 움직임을 추정할 때는 움직임 추정할 위치의 영상 데이터와 주변의 19개의 영상 데이터가 추가로 필요하게 된다. 그러나 모드별 움직임 추정을 수행할 시의 기본적인 단위는 서브블록이므로 16화소에서의 1/4화소 단위의 움직임 추정이 필요하다. 만약 1화소를 기준으로 움직임 수행을 하고자 한다면 20개의 영상데이터 입력이 필요하며 복잡한 블록 계산들이 16번 중복 사용되어지게 된다. 이러한 방법은 하드웨어 사이즈 대비 동작 속도가 느려지는 단점이 되게 된다. 그러나 16화소를 동시에 입력받아서 1/4화소 단위의 움직임을 추정하게 되면 하드웨어 사이즈는 증가하게 되나 동작 속도는 1/4화소 단위 움직임 추정에서 16배 감소하게 된다. 그림 7은 1화소를 위해 필요한 데이터양과 16화소를 위해 필요한 데이터양을 나타내었다.



(a) 1화소 (b) 16화소

그림 7. 6단 FIR 필터계산을 위한 데이터양

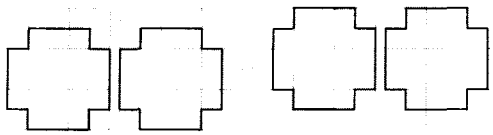


그림 8. 십자형 구조를 위한 영상 데이터 입력 수

이렇게 16화소를 위해 필요한 데이터의 구조를 살펴보면 십자형의 구조를 이루고 있다. 십자형 레지스터 구조는 1/4화소 단위의 움직임 추정 시에 블록 계산에 대한 중복성을 줄여줄 뿐만 아니라 움직임 추정 전에 영상 데이터의 입력 속도에 대한 장점으로도 쓰인다. 본 논문에서 제안된 인터 예측의 기본 데이터 버스의 크기는 16화소 단위의 128비트로 구성되어 있으므로 이러한 장점을 이용하여 그림 8과 같이 단지 각 위치별로 8개 서브블록에 대한 데이터만 필요로 한다.

### 3.3.3 1/4화소 단위 고속 블록 계산기의 구조

1/4화소 단위의 계산을 할 때에는 실제 영상데이터가 존재하는 것이 아니라 가상의 화소를 생성하여 움직임 추정을 하게 되므로 각 16개 위치에 따는 위치 생성기와 계산기가 필요하게 된다. 또한 계산기의 구조 또한 일정한 구조로 이루어져 있지 않고 크게 4가지의 계산기가 필요하게 되는데 논문에서 제안하는 방식은 4가지 방식의 계산기를 통합하고 거기에 따른 중복성을 이용하여 고속 블록 계산기를 설계하였다.

고속 블록 계산기는 단일 클록에 1/4화소 단위 계산을 할 수 있도록 구조를 구성하였다. 520비트의 십자형 레지스터에 입력받아진 영상데이터는 1/4화소 단위 위치 생성기에 의해 각 위치에 데이터를 분류하여 입력하게 되고, 6단 FIR 필터구조, 4개의 데이터 평균값, 2개의 데이터 평균값 계산구조를 사용하여 각 위치에 따른 1/4화소 단위 영상 데이터로 재 생성하게 된다. 그림 9는 십자형 레지스터의 위치에 따른 영상데이터의 위치와 6단 FIR 필터를 위한 계산식을 나타낸다.

이와 같이 십자형 레지스터 구조를 사용한 이유는 6단 FIR필터를 이용하여 1화소 단위로 움직임 추정을 하게 된다면 160비트의 레지스터가 필요하게 되고 그에 따른 1/4화소 단위의 추정을 위해 55개의 덧셈기가 필요하게 된다. 그러나 본 논문에서 제안한 방법처럼 16화소의 움직임을 동시에 추정하게 되면 1/4화소 단위의 움직임 추정으로 위해서 병렬 처리 방식을 사용하여 640개의 덧셈기를 필요로 하게 된다. 움직임 추정에서 16화소의 움직임 추정은 1화소의 움직임 추정보다 많은 수의 덧셈기를 사용하지만 16배 이상의 속도를 감소시키는 성능과 약 240개의 덧셈기를 공유할 수 있는 성능을 가지고 있다. 이러한 성능은 하드웨어 사이즈가 증가하지만 사이즈 대비 속도 면에서 인터 예측을 보다 빠르게 수행할 수 있도록 할 수 있다. 그림 10은 520비트의 레지스터에 재배열 된 영상데이터를 이용하여 동시에 1/4화소 단위 16개 위치의 움직임을 추정하는 연산기와 SAD값을 계산할 수 있는 BSU

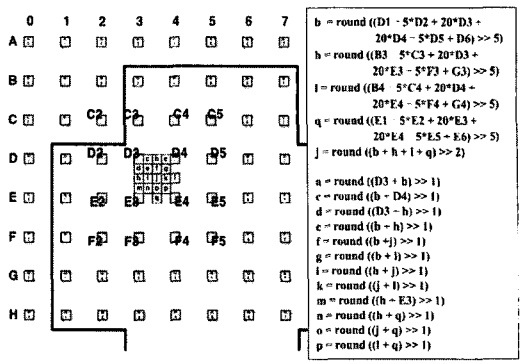



그림 9. 십자형 레지스터의 위치에 따른 영상 데이터

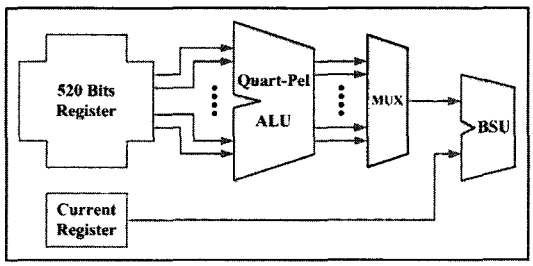



그림 10. 1/4화소 단위 고속 블록 계산 흐름도

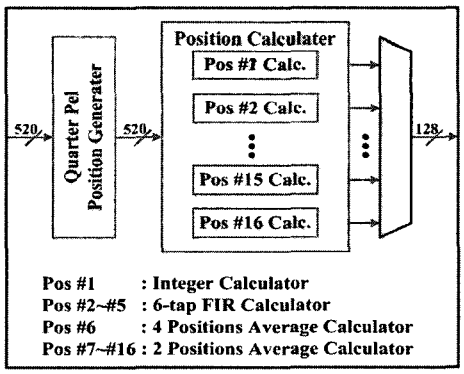



그림 11. 위치별 계산이 가능한 블록 계산기

간의 데이터 흐름을 나타내고 있으며, 그림 11는 블록 계산기의 내부 구조를 나타내었다.

3.3.4 레벨 2,1에서의 움직임 추정 구조

레벨 2,1에서는 기본적으로 인터 메모리로부터 입력되어지는 16개의 8비트 영상 데이터는 BSU를 통하여 정화소 단위의 SAD값이 계산되어진다. 이렇게 계산된 값을 이용하여 움직임 추정을 하게 되는데 레벨 2에서의 움직임 추정은 최소의 SAD값을 찾기 위하여 모든 추정 영역을 검색하는 전역 검색 방식을 사용하여 총 64개의 위치를 추정하게 되며 이중에 최소 SAD값을 가진 두개의 위치를 검색하여 중간 해상도인 레벨 1에서의 움직임 벡터 후보로 사용하기 위해 2개의 레지스터에 저장한다. 그림 12는 레벨 2를 위한 비교기 구조이다.

레벨 1에서의 움직임 추정은 레벨 2에서 추정된 2개의 움직임 후보의 위치를 레벨 1의 좌표에 해상도 변화 후 같은 위치 값을 찾아 새로운 추정 영역으로 입력되며, 또 하나의 후보는 이전 프레임의 주변 움직임 벡터값의 평균값을 취하여 3번째 후보로서 입력되게 된다. 모두 3개의 움직임 벡터 후보들 중에 최종 움직임 벡터 후보를 찾기 위해서는 4개

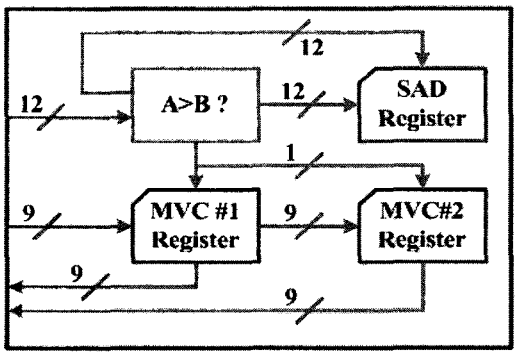



그림 12. 레벨 2를 위한 비교기 구조

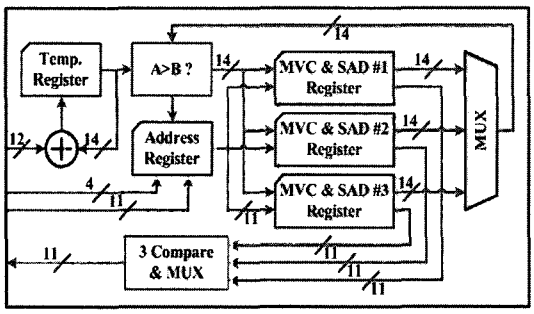



그림 13. 레벨 1을 위한 비교기 구조

의 위치에 대한 SAD값을 모두 합하여 저장한 후 3개의 모션 벡터 후보 값들을 비교하게 되며 최소 SAD값으로 추정되어진 움직임 벡터는 최종후보로 사용하게 된다. 그림 13은 레벨 1을 위한 비교기 구조를 나타낸다.

3.3.5 레벨 0에서의 움직임 추정 구조

레벨 0에서의 움직임 추정을 위한 구조는 레벨 2, 1에서의 계산과 달리 1/4화소 단위의 움직임 추정을 위해 블록 계산기를 통해 나오는 데이터값을 BSU로 입력하여 계산된 SAD값을 사용함으로서 최종적인 움직임 벡터와 SAD값을 구하게 된다. 기본적인 추정영역은 16화소x16화소의 크기를 가지고 있으며 모든 모드별 서브블록의 영역은 총 16개이므로 16개의 레지스터를 이용하여 부분으로 나누어 계산하게 된다. 이와 같은 방법은 가변 블록에 대한 각 모드별 움직임 추정을 분리하는 것이 아니라 저장될 레지스터만 분리하고 공유 비교기를 사용하기 때문에 하드웨어 사이즈뿐만 아니라 추정 시간에서도 높은 성능을 나타낼 수 있다. 그림 14는 레벨 0에서의 움직임 추정을 위한 비교기 구조를 나타낸다.

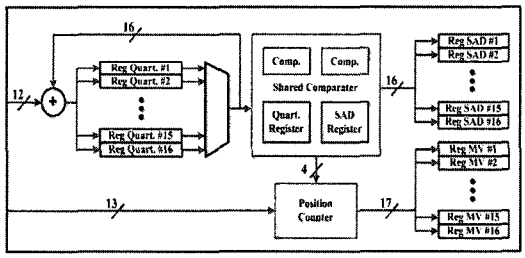



그림 14. 레벨 0을 위한 비교기 구조

## 3.4 움직임 보정

본 논문에서 제안한 움직임 보정은 각 모드별로 16개의 서브블록 단위 보정을 수행하는 구조로 이루어져있다. 이미 블록 조각화와 움직임 추정에 의해서 16개의 서브블록으로 나누어진 데이터를 이용하는 구조를 사용하게 되면 추가적인 모듈의 필요없이 4화소x4화소부터 16화소x16화소까지의 어떠한 모드에 상관없이 움직임 보정을 수행할 수 있게 된다.

움직임 보정을 하기위해서 필요한 데이터는 블록 조각화에서 나누어진 모드값들과 움직임 추정에서 생성된 최종적인 움직임 벡터와 SAD값이 있다. 이렇게 입력된 데이터들은 움직임 추정과 공유된 고속 블록 계산기를 통하여 움직임 보정된 16개의 서

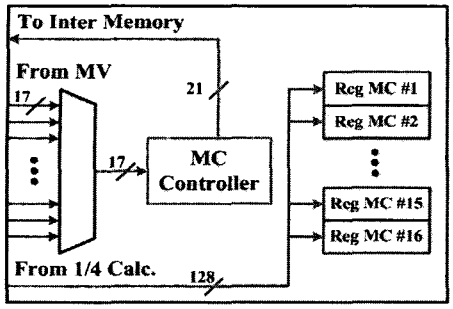



그림 15. 움직임 보정의 구조

브블록 단위의 영상데이터가 생성되게 되고 이것은 하나의 움직임 보정된 매크로블록단위의 영상데이터로 사용되어진다. 그림 15는 움직임 보정을 위한 구조를 나타낸다.

## Ⅳ. 동작 검증 및 성능 분석

### 4.1 동작 검증

인터 예측을 위해 사용되는 클록의 수를 살펴보면 크게 블록 조각화, 움직임 추정, 움직임 보정의 3부분으로 나눌 수가 있다. 이중에서 인터 예측의 속도면에서 가장 큰 비중을 차지하고 있는 부분은 동영상 부호화기의 핵심은 움직임 추정부분이다. 각 모듈별 세부적인 클록의 수를 살펴보면 블록 조각화를 위한 클록의 수는 16화소x16화소부터 8화소x8화소의 모드를 결정하기 위한 클록과 8화소x8화소부터 4화소x4화소의 모드를 결정하기 위한 22클록이 소요된다. 움직임 보정을 위한 클록의 수는 1/4화소 단위의 움직임 보정을 위해 각 서브블록 단위로 8클록의 메모리 입력과 2클록의 보정이 필요하다. 그러므로 16개의 서브블록에 대한 움직임 보정으로 위해서는 160클록이 소요되게 된다.

움직임 추정을 위한 클록의 수는 세부적으로 움직임 추정 레벨 0, 1, 2로 나눌 수가 있는데, 가장 낮은 해상도 레벨인 레벨 2가 먼저 수행되며 그 다음 중간 해상도 레벨인 레벨 1, 원 해상도 레벨인 레벨 0의 순서로 움직임 추정이 이루어진다. 먼저 레벨 2를 위한 클록의 수를 살펴보면 전역 탐색 방식으로 이용하여 총 64클록이 소요되며, 중간 해상도인 레벨 1의 클록의 수는 3개의 움직임 벡터 후보에 대한 추정을 하게 되므로 244클록이 소요된다. 원 해상도인 레벨 0에서의 움직임 추정은 레벨 2, 1과 달리 1/4화소 단위의 움직임 추정을 하기 때문에 많은 클록을 소요하게 된다. 우선 십자형 레지스터에 기준영상 데이터를 입력하기 위한 8클록, 1/4화소 단위의 추정으로 위한 16클록, 추가의 2클록의 26클록을 기본으로 하여 16번의 나선형 방식의 움직임 추정을 하게 되고 16개의 서브블록에 대한 움직임 추정을 하게 되므로 총 6656 (=(8+16+2)*16*16) 클록이 소요되게 된다. 이때 16화소x16화소부터 4화소x4화소까지의 각 모드별 움직임 추정에 대한 움직임 벡터의 추출시간은 동일하다.

이러한 고속 인터 예측기의 각 기능별 수행 클록수를 순차적으로 수행한 합은 블록 조각화, 정화소 단위의 움직임 추정 레벨 2, 레벨 1, 1/4화소 단위의 레벨 0, 움직임 보정의 총 7146(=22+64 +244+6656+160)클록으로서 하나의 매크로블록에 대한 인터 예측 시간이 된다. 그림 16에서는 인터 예측을 위해 전체 소요되는 클록의 수를 나타내었다.

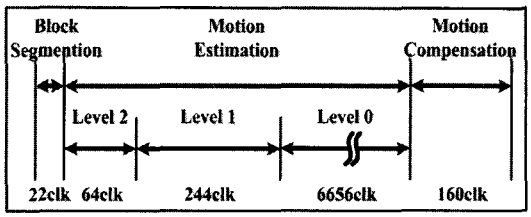



그림 16. 움직임 추정을 위한 전체 클록 분포도

### 4.2 성능 분석

구현된 모듈들의 설계 검증을 위해 FPGA와 ASIC의 두 가지 방법을 사용하여 결과 값을 분석하였다. 첫 번째로는 Altera사의 ARM922T를 내장한 FPGA 디바이스인 EPXA10F1020와 Xilinx사의 8백만 게이트의 FPGA 디바이스인 XC2V8000을 결합하여 인터페이스를 구성하고 리눅스 환경에서 테스트 프로그램을 작성하여 구현된 고속 인터 예측기의 동작을 검증하였다. 이렇게 FPGA로 구현 시 XC2V8000 디바이의 기준으로 결과 값에 따른 최대 동작 주파수는 15MHz이며 대략 20%의 로직 리소스와 23.6Kbit의 메모리 비트가 사용되어진다.

두 번째로는 Samsung STD130 0.18um CMOS Cell Library와 Synopsys사의 Design Analyzer를 이용하여 합성 결과를 분석하였다. 구현된 전체 모듈을 ASIC Library를 이용하여 합성한 결과, 측정된 최장 지연 경로(Critical Path)는 15.99ns이며 최대 동작 주파수는 약 62.5MHz가 된다. 이러한 동작 속도에 따라서 구현된 하드웨어 사이즈는 로직 리소스는 약 135K게이트가 사용되며 메모리는 23.6Kbit가 사용되었다. 표 1은 그림 3의 전체 모듈

표 1. 구현된 모듈 성능

(삼성 STD130 0.18um공정)

| Module | Area[gates] (Percent) |
|---|---|
| Block Segmentation | 2,030 (1.5%) |
| Motion Estimation | 17,731 (13.3%) |
| Motion Compensation | 19,997 (15%) |
| Basic Search Unit | 4,587 (3.5%) |
| Fast 1/4 Pixel Calculator | 72,127 (54.3%) |
| Inter Memory + Ram | 8,428 (6.3%) + 23.6Kbit |
| etc | 8,045 (6.1%) |
| Total | 132,945 +23.6Kbit |
| Critical Path of the Inter Prediction : 15.99ns | |

(최적화 옵션에 따라 최장 지연 시간 및 게이트량 변동)

에서 각 모듈별로 구현된 게이트 사이즈와 전체 인터 예측기의 최장 동작 지연시간을 나타내었다.

구현된 모듈들의 성능에 의해서 MPEG-4 AVC 인터 예측을 수행 시 계산되어지는 속도를 살펴보게 되면, 하나의 매크로블록의 움직임 벡터와 SAD 값을 추출하기 위한 인터 예측의 수행 속도는 그림 16에서 나타낸 바와 같이 총 7146클록이 된다. QCIF(176화소 x 144화소) 크기의 영상을 기준으로 했을 경우 하나의 프레임은 99개의 매크로블록으로 이루어져 있으므로 총 707,454클록이 소요되며 시스템 동작 클록을 62.5MHz로 사용하여 동작을 수행 시에 약 11.3ms가 소요된다. 이러한 성능은 초당 88프레임의 QCIF크기의 영상 데이터를 인터 예측할 수 있는 성능이며 MPEG-4 AVC 부호화기에서 영상 데이터를 실시간으로 부호화할 수 있기에 충분한 성능을 보여준다.

## V. 결 론

본 논문에서는 MPEG-4 AVC를 위한 고속 인터 예측기의 구조를 기술하고 하드웨어 설계를 위한 구조를 제안 및 구현하였다. 기본적으로 다해상도 움직임 추정 알고리즘을 사용하여 움직임 추정 구조를 구성하고 실시간 부호화를 할 수 있도록 블록 구조를 이용한 인터 메모리, 모드 선택 알고리즘을 이용한 블록 조각화, BSU, 1/4화소 단위를 위한 십자형 레지스터 구조와 고속 블록 계산기, 블록 계산기의 공유를 통한 간단한 움직임 보정 등과 같은 기술을 사용하여 성능향상을 꾀하였다. 이렇게 설계된 고속 인터 예측기는 하드웨어 사이즈나 처리량 면에서 우수한 성능을 나타내고 있으므로 MPEG-4 AVC 부호화기의 핵심 모듈로서 우수한 구조라고 할 것이다. 현재는 전체 MPEG-4 AVC의 부호화기 부분 중에서 인트라 예측, 루프 필터 등의 각 모듈들을 설계 중에 있으며 향후 지상파 및 위성파 DMB 시스템에 적합한 저전력 및 실시간 처리구조의 MPEG-4 AVC 부호화기를 하드웨어로 구현할 예정이다.

## 참 고 문 헌


[1] 한국정보통신기술협회, "초단파 디지털라디오 방송(지상파 DMB) 비디오 송수신 정합 표준", Doc. TTAS/KO_07.0026., 2004년 8월.

[2] 한국정보통신기술협회, "위성 디지털 멀티미디어 방송 송수신 정합표준", Doc. TTAS.KO_07.0027., 2004년 9월.

[3] ISO/IEC 14496-10:2003, "Coding of Audio-visual Objects-Part 10: Advanced Video Coding," 2003, also ITU-T Recommendation H.264 "Advanced video coding for generic audiovisual services."

[4] B. Song and J. Ra, "A fast multi-resolution block matching algorithm for motion estimation", Signal Processing : Image Communication, vol.15, pp. 799-810, 2000.

[5] Y. Lim, D. Lee and Y. Jeong, "Hardware Implementation of Fast Multi-resolution Modtion Estimator for MPEG-4 AVC", Proc. of IEEK Summer Conference, June 2004.



**임 영 훈**(Young hun Lim) 준회원

2001년 2월 대진대학교 전자공학과 졸업

2001년 3월~2003년 2월 동서전자 기업부설연구소 연구원

2005년 2월 광운대학교 전자통신공학과 석사

2005년 3월 엠텍비젼 연구원

<관심분야> 영상처리, 제어 시스템 설계, SoC 설계

**이 대 준**(Dae joon Lee) 준회원

2004년 2월 광운대학교 전자공학부 졸업

2004년 3월~현재 광운대학교 전자통신공학과 석사과정

<관심분야> 영상처리, SoC 설계

**정 용 진**(Yong jin Jeong) 정회원

1983 2월 서울대학교 제어계측공학과 졸업

1983년 3월~1989년 8월 한국전자통신연구원

1995년 2월 미국 UMASS 전자전산공학과 박사

1995년 4월~1999년 2월 삼성전자 반도체 수석 연구원

1999년 3월~현재 광운대학교 전자공학부 부교수

<관심분야> 무선 통신, 정보보호, SoC 설계