멀티미디어 모바일 시스템에서의 효율적인 H.264 움직임 보간법

(Effective Motion Compensation Method of H.264 on Multimedia Mobile System)

정 대 영 † 지 신 행 † 박 정 욱 † 김 신 덕 ††
(Dae-Young Jeong) (Shin-Haeng Ji) (Jung-Wook Park) (Shin-Dug Kim)

요 약 데이타 전송이 전력 소비 중 가장 많은 부분을 차지하는 멀티미디어 모바일 시스템에서 전력 감지 설계는 가장 중요한 분야 중 하나이다. 이 논문에서는 H.264/AVC의 MC(움직임 보정)에서 데이타 전송을 감소시킴으로써 전력 소비를 줄이는 새로운 구조를 제안한다. 이러한 목적으로 성능 저하 혹은 시스템의 지연 없이 MC(Motion Compensation) IP(Intellectual Property)에 효과적으로 접목되는 재구성 가능한 마이크로 아키텍처를 제안한다. 하나 또는 두 1/2 보간법과 한 평균법으로 이루어진 이전 1/4 보간법 공식은 메모리 접근을 줄이고 시스템 효율성을 유지하는 다른 수행 제어 형식을 가지도록 새로이 설계되었다. 모의 시험의 결과로써, MC 모듈에서 이전 방법을 사용했을 때 데이타 전송으로 인해 일어나는 전력 소모의 87%가 제안된 방법을 이용해서 줄어든 것을 보여준다.

키워드 : H.264/AVC, 움직임 보간, 1/4 픽셀 보간법, 저전력, 메모리 접근, 멀티미디어 SoC, 시스템 아키텍처

Abstract Power-aware design is one of the most important areas to be emphasized in multimedia mobile systems, in which data transfers dominate the power consumption. In this paper, we propose a new architecture for motion compensation (MC) of H.264/AVC with power reduction by decreasing the data transfers. For this purpose, a reconfigurable microarchitecture based on data type is proposed for interpolation and it is mapped onto the dedicated motion compensation IP (intellectual property) effectively without sacrificing the performance or the system latency. The original quarter-pel interpolation equation that consists of one or two half-pel interpolations and one averaging operation is designed to have different execution control modes, which result in decreasing memory accesses greatly and maintaining the system efficiency. The simulation result shows that the proposed method could reduce up to 87% of power consumption caused by data transfers over the conventional method in MC module.

Key words : H.264/AVC, motion compensation, quarter-pel interpolation, low-power, memory access, multimedia SoC, system architecture

1. 서 론

모바일 시스템을 위한 저전력 고성능 컴퓨팅 아키텍처에 대한 수요는 멀티미디어 처리와 비디오 압축의 독보적인 유행으로 인해 급상승해왔다. 그리고 한 칩에 많은 구성 부품들을 통합하는 SoC(System on Chip) 기술은 짧은 시간 내에 임베디드 시스템 설계를 가능하게 하였고, 또한 응용프로그램 군을 목표로 하는 SoC 플랫폼 기술도 성장시켰다[1]. 최근, 모바일 장치들의 작은 저장공간과 제한된 무선 인터넷 전송 용량으로 인해 데이타/비디오 코딩은 매우 중요한 기능이 되었다. 가장 최근의 비디오 코딩 표준인 H.264/AVC는 고화질의 동영상을 적은 비트 전송 속도로도 만들 수 있는 능력 때문에 현재 가장 주목할만한 코덱이다. 그러므로, H.264/AVC는 화상회의, 화상전화, 교육용 솔루션, 그리고 디

· 이 연구는 Seoul Research and Business Development Program (10698)의 지원 하에 수행되었음
† 비 회 원 : 연세대학교 컴퓨터과학과
dangbang@parallel.yonsei.ac.kr
twoze@parallel.yonsei.ac.kr
pjppp@parallel.yonsei.ac.kr
†† 정 회 원 : 연세대학교 컴퓨터과학과 교수
sdkim@cs.yonsei.ac.kr
논문접수 : 2006년 2월 24일
심사완료 : 2007년 6월 8일

지털 방송(DMB)과 같은 많은 멀티미디어 응용프로그램들에서 사용이 되고 있다. 중요한 멀티미디어 응용으로써, 이전 코덱들과 비교했을 때 H.264/AVC에는 더욱 고화질/고압축의 비디오를 만들어내기 위하여 복잡한 수행이 요구된다. H.264/AVC 디코더에서 가장 두드러지는 작업은 많은 메모리 접근을 수행하고 많은 연산들을 필요로 하는 움직임 보간(Motion Compensation, MC)이다.

보통의 경우, 멀티미디어 응용프로그램들은 많은 양의 데이타와 복잡한 수식 처리를 필요로 하는 데이타 우위적인 알고리즘을 포함하는데, 이것은 데이타 전송과 메모리 구성이 실제 구현상의 면적과 전력 소비에 현저한 영향을 미칠 것이라는 것을 내포한다[2]. 이것은 멀티미디어 시스템들에서 왜 저전력에 대한 구조적 및 알고리즘적인 접근이 기술, 레이아웃, 회로, 게이트와 같은 다른 부분에 대한 접근들과 비교했을 때 더욱 커다란 향상을 가지고 올 수 있는지를 알려준다[3]. 그러므로, 응용 특화된 메모리 구성과 제어 플로우를 설계함으로써 시스템 단계에서의 전력 소비를 줄이기 위해 알고리즘의 효율적인 구현 방법이 필요하다. 참고논문 [4]는 I/O 전력이 칩의 총 전력 소비의 80%만큼 높다는 것을 보여줌으로써 이러한 접근이 중요하다는 것을 알려주고 있고, 이러한 문제점을 해결하는 저전력 비디오 및 신호처리 응용프로그램들에 대한 몇몇 방법들이 제시되어왔다[5-8].

대부분의 H.264의 저전력 연구는 인코더의 움직임 추정(Motion Estimation, ME)과 낮은 레벨의 설계에 대한 것들이다. 디코더의 움직임 보간법을 수정함으로써 성능 향상과 저전력을 얻기 위한 연구가 수행되기는 했으나 이는 메모리 컨트롤러를 직접 바꾼 것이기 때문에 구현이 그만큼 힘들다[12].

이 연구는 H.264/AVC 코덱에서 메모리 효율적인 움직임 보간법을 통하여 전력 소비가 적은 새로운 계산 아키텍처를 제안한다. 이것은 어떤 성능의 저하 없이 메모리 접근을 감소시킴으로써 H.264/AVC 디코더의 중요한 부분의 시스템 전력소모 중 많은 부분을 감소시킬 수 있다. 이러한 목적을 위해 두 가지 기법이 제시되는데, 첫 번째 기법은 1/2 픽셀 보간법과 평균법을 하나로 합치는 기법이고, 두 번째 기법은 보간법의 실행 순서를 재배열하는 기법이다. 이것들은 작은 중간 메모리를 가진 SIMD(Single Instruction Multiple Data) 스타일의 아키텍처에 매핑되어서, 보간법 단계가 수행되는 도중 발생하는 메모리 접근들을 줄여주는 역할을 하게 된다.

다음 장에서는 H.264/AVC의 움직임 보간 알고리즘을 설명하고, 3장에서는 제시하는 SoC 플랫폼의 기본 아키텍처를 소개할 것이다. 4장에서는 1/4 픽셀 보간법에 대한 제안된 방법을 자세히 설명하고, 5장에서는 전력 모델과 시뮬레이션 결과들이 보여질 것이다. 마지막으로 6장에서 결론이 지어진다.

2. H.264/AVC의 움직임 보간

비디오 코딩은 가공되지 않은 비디오 순열로부터 중복되는 정보를 제거함으로써 이루어진다. 보통의 경우, 픽셀 값들은 연속된 프레임이나 같은 프레임에서 그 이웃된 픽셀 값들과 관계가 있다. 이것들은 각각 시간적/공간적 중복성이라 불리운다. 이 중복성들은 사각형 블록(MxN 혹은 NxN) 단위의 움직임 추정과 움직임 보간을 사용하여 줄일 수 있다. 현재 비디오 코딩 표준의 움직임 보간에서는 매크로블록(Macroblock)이라 지칭되는 16×16 픽셀 공간이 기초적인 데이타 유닛이다.

2.1 블록 기반의 ME/MC 그리고 H.264/AVC

H.264는 블록 단위의 움직임 추정과 움직임 보정과 이전 표준에서부터 있어왔던 다른 기술들에 기반하고 있지만, 이전 표준들보다 훨씬 많은 코딩 효율성을 얻을 수 있다. 이 효율성은 움직임 추정과 움직임 보정들의 중요한 특징들로부터 얻어진다. 중요한 특징들은 다음과 같다. (a) 작은 블록 크기들을 이용한 다양한 블록 크기의 움직임 보간, (b) 1/4픽셀만큼의 정확도를 가진 픽셀 움직임 추정, (c) 다양한 참조 프레임 선택. 그러나 이와 동시에 복잡도도 늘어나게 된다. 그림 1은 움직임 보정의 핵심 부분인 6-tab FIR(Finite Impulse Response) 필터를 이용한 반복문이다. 그리고 조건 분류문들이 반복문 마지막에 들어간다.

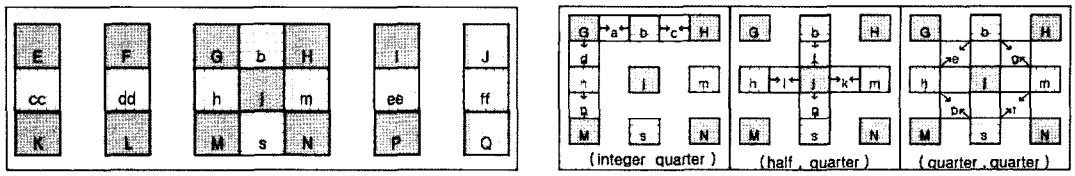
```
for ( j = 0 ; j < block_size ; j++ )        // block_size = 4
    for ( i = 0 ; i < block_size ; i++ )
        for (result = 0, x = -2 ; x < 4 ; x++)
        // 6_tab FIR filter
         resut += imgY[max(0,min(y,y_pos+j))][max(0,max(x,x_pos+i+x))]*COEF[x+2];
```

그림 1 움직임 보간 단계의 픽셀 보간법

2.2 H.264/AVC의 1/2와 1/4 픽셀 보간법

H.264에서 1/4 픽셀 보간법은 움직임 추정을 정확하게 하는데 이것은 1/2 픽셀을 만드는 표준 1/2 픽셀 보간법과 1/2 픽셀과 정수 픽셀의 평균을 내는 평균법으로 구성되어있다. 1/4 픽셀 보간법과 1/2 픽셀 보간법은 다양한 블록 크기들과 1/4 해상도의 움직임 벡터들(Motion Vector)을 지원한다.

이러한 움직임 벡터는 X와 Y값인 두 좌표로 구성되어있다. 현재 매크로블록의 움직임 벡터의 X, Y값이 모두 1/2의 소수점 자리를 가지고 있다면, 이 매크로블록의 움직임 보간을 위해서는 1/2 픽셀 보간법 만이 필요하게 된다. 그러나 움직임 벡터가 하나 또는 두 개의



(a) half-pel interpolation (b) quarter-pel interpolation

그림 2 1/2 픽셀과 1/4 픽셀의 보간법

1/4 소수점 자리를 가진 값들로 이루어져있다면, 움직임 보간을 위해서 1/2 픽셀 보간법과 평균법 둘 다 필요하게 된다. 1/2 픽셀 보간법의 공식은 다음과 같다(그림 2(a) 참고).

$$b = [(E - 5F + 20G + 20H - 5I + J)/32] \quad (1)$$

b값을 구하기 위해서는 몇몇의 ADD, MUL, SHIFT 연산들이 필요하다. 이 공식은 매우 간단하지만, 수행 회수가 많고, 많은 데이타 전송을 필요로 하는 공식이다.

그림 2(b)에서와 같이, 평균법의 결과 픽셀 값들은 정수 픽셀값과 1/2 픽셀 보간법으로 계산된 1/2 픽셀값의 평균을 냄으로써 얻어진다. 평균법의 공식은 다음과 같다.

$$a = (G + b)/2 \quad (2)$$

메모리 접근 측면에서, 각 보간법을 수행할 때 시스템은 외부 메모리로부터 내부 메모리로 참조될 매크로블록을 가지고 와야한다. 또한, 1/2 픽셀 보간법 수행 중에 발생하는 중간 값들을 저장해야 하기 때문에 내부 메모리 접근이 또 다시 필요하다. 표 1은 각 비디오 해상도에서의 반복되는 보간법의 수행 횟수를 나타낸다. 여기에서는 1/2 픽셀 보간법 수행 횟수가 1/4 픽셀 보간법 수행 횟수의 2배로 나타나지만, 1/4 픽셀 보간법은 1/2 픽셀 보간법을 포함하고 있고, 여기에 또 다른 1/2 픽셀 보간법이나 평균법과 같은 또 다른 처리가 수행되기 때문에, 전체적으로는 1/2 픽셀 보간법보다 더 많은 메모리 접근을 수행하게 되기 때문에 1/4 픽셀 보간법이 움직임 보간 중 가장 복잡한 부분이 된다. 표 2는 매크로블록 크기에 따른 1/4 픽셀 보간법의 메모리 접근 수를 나타낸다. 빈번하게 일어나는 메모리 접근들은 각자 독립적인 보간법에서 일어나게 되기 때문에, 메모리

표 1 해상도에 따른 보간법 횟수

Resolution	Max. half-pel interpolation count per frame	Max. quarter-pel interpolation count per frame
QCIF(176*144)	50688	25344
CIF(352*288)	202752	101376
VGA(640*480)	614400	307200
1080i(1920*1080)	4147200	2073600

표 2 매크로블록 크기에 따른 메모리 접근 횟수

Macroblock size	Necessary memory accesses for quarter pel interoplation	Max. count of memory accesses per frame in VGA resolution
16×16	4921	5905200
16×8, 8×16	2456	5894400
8×8	1368	6566400
8×4, 4×8	684	6566400
4×4	412	7910400

접근 간에는 특별한 연관성이 없다. 메모리 접근은 실행 시간을 늘릴 뿐만 아니라 다른 시스템 부분으로부터의 다른 메모리 접근 요청들이 허락되지 않게 함으로써 내부 버스의 독점현상을 일으키기 때문에, H.264/AVC 디코더 시스템에서 심각한 문제가 될 수 있다.

3. 기본 시스템 아키텍처

그림 3에서 보여지듯이, 기본 아키텍처는 시스템 수준에서 응용과 구성 요소에 적합한 여러 IP들로 구성되어 있다. 여기에는 DSP, 메모리 시스템, 내부 시스템 버스 등의 다른 모듈들 간의 데이타 전송을 수행하는 디코딩 연산들을 실행하기 위한 32 비트 RISC 기반의 주 프로세서가 있다. 그리고 다양한 I/O 주변장치들은 시스템 버스를 통하여 설정될 수 있다. 모든 메모리들은 각각 독립적이고 동시 다발적으로 접근될 수 있다.

여기서 검토될 응용프로그램은 H.264/AVC의 움직임 보간 단계이다. 위 그림에서 회색 모듈은 움직임 보간 IP인데, 이것은 SIMD 형식으로 동작하는 여러 처리 유닛(Processing Element, PE)들과 응용 특화 되고 모든

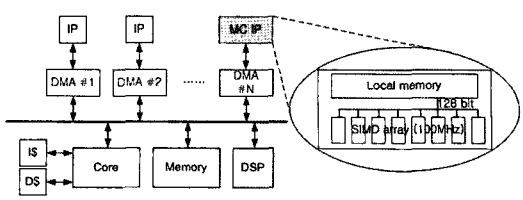


그림 3 전체 시스템 아키텍처

PE들에게 공유되는 작은 중간 메모리로 이루어져있다. 이 IP에서 각 PE는 많은 에너지가 소비되는 큰 외부 메모리에 접근하는 대신에 내부 공유 메모리로부터 사용된 데이타를 읽을 수 있다. 연산이 독립적이고 공유 내부 메모리로부터 128 비트 넓이의 내부 버스를 통해 매 사이클에 데이타가 공급되기 때문에 각 PE는 필요한 데이타가 오기를 기다리지 않고 병렬적으로 동작할 수 있다.

그리고 메모리 체계는 외부 메모리와 내부 메모리로 구성되어있다. 외부 메모리는 1 메가 바이트, 내부 메모리는 1 키로 바이트로 가정하였다. 외부 메모리는 6개의 프레임 전체를 저장하기 충분해야 하고, 내부 메모리는 움직임 보간 단계에서 생겨나는 중간 값들을 저장하기 충분해야 하기 때문이다. H.264의 첫 디코딩 단계인 엔트로피 코딩(Entropy Coding)의 결과값은 외부 메모리에 저장된다. 움직임 보간 단계는 이 값들을 외부 메모리로부터 내부 메모리로 가지고 와서 보간법을 수행하고 이 결과로써 발생하는 값들을 다시 외부 메모리에 저장한다. 이런 방식으로 진행되기 때문에 보간의 정확도를 줄이지 않고 외부 메모리 접근을 줄이는 것은 거의 불가능하다. 줄일 수 있는 메모리 접근은 내부 메모리에서 중간에 일어나는 load/store 뿐이다.

4. 픽셀 보간법에 대한 방법론 제안

여기서 제안하는 1/4 픽셀 보간법은 메모리 접근을 줄이는 두 가지 기법으로 구성된다. 하나는 그림 4와 같이 움직임 벡터가 (정수, 1/4)일 경우, 두 보간법 단계를 하나로 합치는 것이다. 이 그림은 그림 2에서의 픽셀 표현과 연관되어 원래의 1/4 픽셀 보간법과 제안되는 1/4 픽셀 보간법의 다이어그램을 보여준다. 다른 하나의 방법은 그림 5와 같이 움직임 벡터가 (1/2, 1/4) 또는 (1/4, 1/4)일 경우 보간법의 수행 순서를 재배열하는 것이다. 제안하는 두 가지 방법은 중복되는 메모리 접근을 제거하는 시간적 지역성을 내포하고 있다.

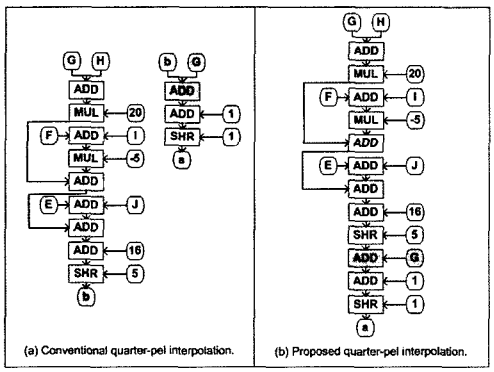


그림 4 두 다른 보간법에 대한 세부 묘사

4.1 움직임 벡터가 (정수, 1/4)일 경우

1/4 픽셀 보간법은 먼저 매크로블록 전체에서 1/2 픽셀 보간법을 수행한다. 그 후에 움직임 벡터 값에 따라서, 바로 평균법 단계를 수행하거나 한번의 1/2 픽셀 보간법을 다른 방향으로 더 수행한 후에 평균법 단계를 수행하게 된다. 움직임 벡터가 (정수, 1/2), (정수, 1/4)와 같은 경우일 때에는 오직 한번의 1/2 픽셀 보간법이 필요하게 되고, 그렇지 않은 경우에는 움직임 벡터가 (1/2, 1/4)나 (1/4, 1/4)의 경우일 때와 같이 또 다른 1/2 픽셀 보간법이 더 필요하다.

이전의 1/4 픽셀 보간법은 이 두 단계가 따로 계산되지만, 움직임 벡터가 (정수, 1/4)인 경우, 이 두 단계는 따로 계산될 필요가 없다. 두 보간 단계를 하나로 묶음으로써 메모리 효율적인 계산이 가능하다. 이 제안된 아이디어에 따라 수정된 공식은 아래와 같다.

$$a = \left[\left(\left[(E - 5F + 20G + 20H - 5I + J)/32\right] + G\right)/2\right] \quad (3)$$

그림 4의 회색 블록들은 SIMD 모듈에 제안된 아이디어를 매핑할 경우 수정되는 부분이다. 이 방법에서, 간단한 제어를 통하여 중간에 일어나는 load/store를 제거함으로써 메모리 접근을 줄일 수 있다. 이에 대한 효과는 5장에서 자세히 설명될 것이다.

4.2 움직임 벡터가 (1/2, 1/4) 혹은 (1/4, 1/4)일 경우

두 번째 기법은 그림 6(b)에 나와 있듯이, 이전의 보간법에서 17번 수행되는 단위 동작들을 재배열하는 새로운 수행 절차이다. 이것은 내부 메모리 접근을 줄이기 위해 SIMD 어레이에 매핑된다. 그림 6(a)에서는 먼저 1/2 픽셀 보간법을 수행한 후(1~9단계), 이 값들을 다시 1/2 픽셀 보간법 수행하여 (1/2, 1/2)를 계산한 후(10~13단계), 평균법 단계를 통하여 (1/4, 1/4)를 만들어내는 이전의 수행 절차를 보여준다(14~17단계). 그림 6(b)에서는 제안하는 수행 절차를 나타나는데, 화살표들은 재배열되는 위치를 알려준다. 아래의 숫자는 각 절차의 읽고 쓰는 횟수이다. 이 그림에서, 재배열 기법과 시간적 지역성 기법을 모두 적용했을 경우에는 이해하기 매우 어렵기 때문에, 시간적 지역성 기법은 포함되어있지 않고 오직 재배열 기법만 사용된 경우를 보여주었다. 재배열된 수행 방식은 아래와 같다.

1. 가로 방향으로 1/2 픽셀들을 만들기 위해 5줄의 가로 보간법을 수행하고 이 결과를 내부 메모리에 저장한다. (그림 6(b)의 1~5단계)
2. 새로운 1/2 픽셀을 만들기 위해 1번의 가로 보간법을 수행한다. (그림 6(b)의 6단계)
3. 1의 결과값들을 불러들이고, <공식 3>을 이용하여 세로방향으로 보간법을 사용한다. (그림 6(b)의 7,8단계)
4. 2와 3을 반복한다.

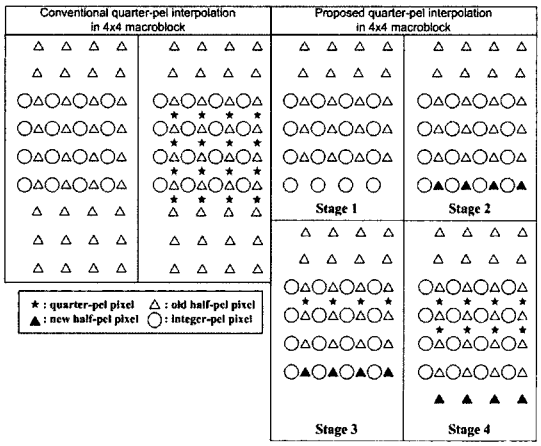


그림 5 보간법 수행의 재배열

그림 5, 그림 6은 동일한 의미를 갖지만, 그림 5는 픽셀이 어떤 위치에 순서대로 생겨나는 지를 보여주고, 그림 6은 픽셀들을 계산하는데 단위 명령(instruction)들을 어떤 순서로 수행하는지를 보여준다.

이전의 1/4 픽셀 보간법은 그림 5의 왼쪽과 같다. 우선 세로 방향 보간법을 수행하기 위한 중간 값들을 만들기 위해 모든 가로 방향 보간법들을 한꺼번에 수행한 후(그림 6(a)의 1~9단계), 이 중간 값들을 불러들여 세로 방향 보간법을 한꺼번에 수행한다(그림 6(a)의 10~13단계). 여기에서 불필요한 메모리 접근들이 발생하게 된다. 제안한 보간법은 그림 5의 오른쪽과 같다. 이것은 보간법들의 순서를 변화시킴으로써 필요없는 중간 메모리 접근을 최소화 할 수 있다. 그림 5 오른쪽 그림의 1단계(그림 6(b)의 1~5단계)는 이전의 방법에서 바꿀 수 없는 부분이다. 2단계는 세로 방향 1/2 픽셀 보간법을 수행하기 위해서 6개의 세로 방향 픽셀들이 필요하기 때문에 이에 쓰이는 보간법의 결과로써 한 줄의 가로 1/2 픽셀들을 만든다(그림 6(b)의 6단계). 이 6개의 세로 방향 픽셀들은 3단계에서 보간된다. 그리고 3단계에서 4.1절에서 제안된 두 공식을 합치는 기법을 사용해서, 중간 메모리 접근 없이 1/4 픽셀을 만들 수 있다(그림 6(b)의 7,8단계). 4.3절에서 소개 될 시간적 지역성을 사용함으로써 세로 방향 1/2 픽셀 보간법에 필요한 5개의 세로 방향 픽셀들을 읽어 들일 필요가 없기 때문에, 검은색 삼각형을 만들기 위한 가로 방향 1/2 픽셀 보간법에서 쓰일 6개의 가로 방향 픽셀들과 검은색 별을 만들기 위한 1/4 픽셀 보간법에서 쓰일 1개의 픽셀 값만이 4단계에서의 각 보간법에서 읽혀지게 된다.

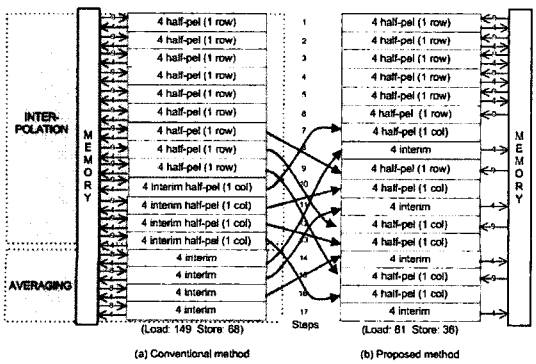


그림 6 재배열 순서와 메모리 접근 횟수

4.3 보간법 단계에서의 시간적 지역성

보간법 단계의 시간적 데이타 지역성은 이전 보간법 단계에서 읽어진 변수들을 지우지 않고 가지고 있음으로써 중간 load/store를 줄이기 위해서 사용된다. 그림 7과 같이, 첫 번째 보간법은 A부터 F까지의 픽셀값들이 필요하고, 두 번째 보간법은 B부터 G까지의 픽셀값들이 필요하다. 계속 이런 식으로 불필요하게 겹치는 메모리 접근들이 일어나게 되어서 전력 소모를 증가시키게 된다. 이러한 접근들은 시간적 지역성 기법을 통하여 제거될 수 있는데, 이것은 하드웨어적인 설정이 아닌 코드 최적화를 통하여 실현 가능하다. 그림 8은 재구성 된 코드를 보여준다. 이전 보간법 코드들이 합쳐지고, 겹치는 데이타 읽기가 제거되었다. 이 방법은 보간법 단계 전체에서 사용된다.

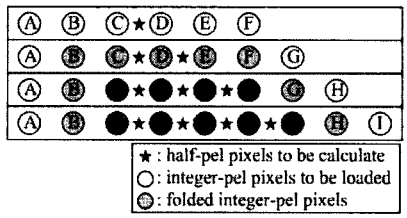


그림 7 보간법 간의 겹쳐진 메모리 접근

5. 전력 모델과 성능 측정

Salesman, Carphone, Container, 그리고 Mthr_dotr, 이렇게 비디오 코딩 시뮬레이션에 자주 쓰이는 4개의 동영상을 선택하여 시뮬레이션을 수행하였다. 이 동영상들은 모두 QCIF 포맷으로 작성되었고 총 178 프레임으로 이루어졌다. 시뮬레이션을 위해 그림 3에서 보여지는 것과 같은 하드웨어 아키텍처에 기반하여 C++로 두 가지의 동작 모델을 구현하였다.

결과는 제안한 방법이 1/2 픽셀 보간법과 1/4 픽셀 보간법의 수정을 통하여 메모리 접근을 매우 많은 부분 감소시킨 것을 보여준다. 이것은 특정한 동작을 제거한 것이 아니기 때문에 어떠한 성능 감소나 시스템 지연을 일으키지 않았다. 모든 시뮬레이션에서, 제안한 방법은

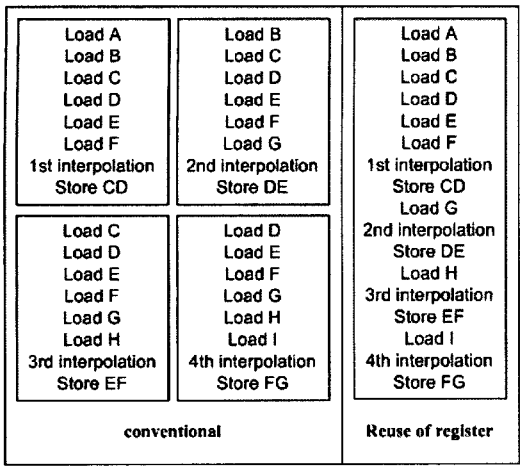


그림 8 이전 방법과 시간적 지역성을 이용한 방법 간의 차이점

내부 메모리 접근을 87%까지 제거하였다.

위에서 언급한대로, 이 제안한 방법들은 오직 내부 메모리 접근만 줄일 수 있다. 그림 6에서 외부와 내부 메모리 접근이 모두 보여지기 때문에, 약 46%의 메모리 접근만이 줄어드는 것으로 나와 있지만, 그림 9에서는 오직 내부 메모리 접근만을 보여준다.

이러한 내부 메모리 접근 수 감소는 저전력 효과도가 지고 온다. 높은 시스템 단계에서 전력 측정을 위해서, [2]와 비슷한 방법론을 참고하여, 가장 가망있는 후보를 선택하는 상대적 비교 방법을 사용하였다. 이 논문에서 쓰이게 된 응용은 데이타가 많이 전송됨으로써 메모리 전송으로 인한 전력 소비가 가장 큰 응용프로그램인 H.264/AVC이다. 그리고 이전 알고리즘과 제안한 알고리즘 둘 모두 같은 아키텍처에 매핑되었다. 그러므로 조작이나 제어 등이 전력 소비에 미치는 영향을 무시할 수 있다. 이 방법에 따라, 메모리 접근들에서 데이타 전송이 줄어듦으로써 얻어진 전력 소비 감소를 측정하였다.

데이타 전송에 드는 전력은 메모리의 크기와 접근의 빈도, 그리고 기법에 따라 결정되는 함수이다[2]. 기법은 메모리에 관련된 내용이기 때문에 같은 시스템을 사용한다는 가정을 통하여 제외시킬 수 있다. 그래서 나올 수 있는 전력 모델 함수는 다음과 같다.

$$P_{Transfers} = E_{Tr} \times \frac{\#Transfers}{Second} \quad (4)$$

$$E_{Tr} = f(\#words, \#bits) \quad (5)$$

[9]에서 함수 f는 단어 개수와 비트 개수의 관점에서 전송당 소모 에너지인 E_{tr}를 측정한다. 총 시스템 전력은 데이타 전송의 개수에 부분적으로 비례한다고 할 수 있다. 전력 절감 비율을 같은 시스템에서 두 개의 다른 동작 모델을 비교한다고 가정한다면, 한 데이타 전송에 얼마만큼의 에너지가 필요한지와 두 동작 모델들에서의 비디오 총 데이타 전송 개수를 알아야 한다. [9]의 전력 모델에 기반하여, 8 비트의 256×256 블록을 메모리로부터 읽어오는데 걸리는 에너지는 1.17uJ이다.

위에서 제안한 방법이 내부 메모리 접근을 87% 제거한 것으로 결과가 나왔기 때문에 전력 소모도 같은 비율로 감소되고, 이러한 저전력 효과는 그림 9에 나와 있다.

6. 결 론

H.264/AVC의 움직임 보간을 위한 데이타 전송을 줄임으로써 전체 시스템의 성능을 향상시키고, 전력을 감소시키는 새로운 아키텍처를 제시하였다. H.264/AVC와 같은 데이타 이동이 많은 응용프로그램들에서, 데이타 전송은 내부 버스의 load와 전력 소모의 대부분을 차지한다. 이것을 줄이기 위해, 움직임 벡터의 값에 따른 마이크로 아키텍처 수준의 재구성 가능한 시스템을 사용하였고, 이로 인해 1/4 픽셀 보간법을 수행하는 도중에 시스템에 다른 제어를 가능하게 하였다. 결과로써, 시스템 수준에서 메모리 접근 수와 전력 소모가 현저하게 줄어들었다. 시뮬레이션 결과는 제안한 방법을 사용하였을 때 어떠한 성능 하락도 없이 이전 방법을 사용했을 경우보다 87%의 메모리 접근 및 전력 소모가 줄어든 것을 보여준다.

Video image	Interpolation method	Off-chip memory read count	Off-chip memory write count	Local memory READ count	Local memory WRITE count	Consumed energy to READ from local memory (μJoule)	READ power saving ratio (1-proposed /conventional)	Consumed energy to WRITE from local memory (μJoule)	WRITE power saving ratio of (1-proposed /conventional)
salesman	conventional	9972288	4432128	1466940	900780	26.19	87.93%	16.08	86.94%
	proposed	9972288	4432128	177216	117760	3.16		2.10	
carphone	conventional	10008000	4448000	5951480	3795928	106.25	86.85%	67.77	88.15%
	proposed	10008000	4448000	782231	449780	13.97		8.03	
container	conventional	9505152	4224512	930084	483892	16.60	97.65%	8.64	98.03%
	proposed	9505152	4224512	21951	9720	0.39		0.17	
mthr_dotr	conventional	9828288	4368128	3447044	2070564	61.54	88.61%	36.97	90.70%
	proposed	9828288	4368128	392697	192820	7.01		3.44	

그림 9 시뮬레이션 결과

참 고 문 헌

[1] Vahid, F., Givargis T.: Platform Tuning for Embedded Systems Design. Computer, Vol. 34 N. 3, Mar. (2001) 112-114.

[2] Smith, R., Fant, K., Parker, D., Stephani, R., Ching-Yi, W.: System-Level Power Optimization of Video Codecs on Embedded Cores: A Systematic Approach. Processing of Journal of VLSI Signal, 18 (1998) 89-109.

[3] Kakerow, R.: Low Power Design Methodologies for Mobile Communication Computer Design. Proceedings of 2002 IEEE International Conference on VLSI in Computers and Processors, Sep. (2002) 8-13.

[4] Musoll, E., Lang, T., Cortadella, J.: Exploiting the Locality of Memory References to Reduce the Address Bus Energy. Proceeding of 1997 International Symposium on Low Power Electronics and Design, Aug. (1997) 202-207.

[5] Kim, H., Park, I. C.: High-Performance and Low-Power Memory-Interface Architecture for Video Processing Applications. IEEE Transactions on Circuits and Systems for Video Technology, Vol. 11, Issue 11, Nov. (2001) 1160-1170.

[6] Kapoor, B.: Low Power Memory Architectures for Video Applications. Proceedings of the 8th Great Lakes Symposium on VLSI, Feb. (1998) 2-7.

[7] Brockmeyer, E., Nachtergaele, L., Catthoor, F.V.M., Bormans, J., De Man, H.J.: Low Power Memory Storage and Transfer Organization for the MPEG-4 Full Pel Motion Estimation on a Multimedia Processor. IEEE Transactions on Multimedia, Vol. 1, Issue 2, June (1999) 202-216.

[8] Nachtergaele, L., Catthoor, F., Kapoor, B., Janssens, S., Moolenaar, D.: Low Power Storage Exploration for H.263 Video Decoder. Workshop on VLSI Signal Processing IX, 30 Oct.-1 Nov. (1996) 115-124.

[9] Landman, P.: Low-Power Architectural Design Methodologies. PhD thesis, U.C. Berkeley, Aug. (1994).

[10] ISO/IEC 14496-10:2003: Coding of Audiovisual Objects-Part 10: Advanced Video Coding, 2003, also ITU-T Recommendation H.264: Advanced Video Coding for Generic Audiovisual Services.

[11] Sato, K.; Yagasaki, Y.: Adaptive MC Interpolation for Memory Access Reduction in JVT Video Coding. Proceedings of Seventh International Symposium on Signal Processing and Its Applications, Vol. 1, July (2003) 77-80.

[12] Jiahui Zhu; Ligang Hou; Wuchen Wu: High Performance Synchronous DRAMs Controller in H.264 HDTV Decoder. Solid-State and Integrated Circuits Technology, 2004. Proceedings. 7th International Conference on, Vol. 3, Oct. (2004) 1621-1624.

정 대 영

2004년 연세대학교 컴퓨터과학과(학사) 2006년 연세대학교 컴퓨터과학과(공학석사). 2007년~현재 LIG Nex1 시스템연구소 재직 중. 관심분야는 내장형 시스템 구조, 비디오 코덱, 시스템 통합, 전자전 분야 등

지 신 행

2003년 경상대학교 컴퓨터과학과(학사) 2006년 연세대학교 컴퓨터과학과(공학석사). 2006년~현재 LG전자 정보통신연구소 재직 중. 관심분야는 내장형 시스템 구조, 고성능 메모리 계층구조

박 정 욱

2003년 연세대학교 컴퓨터과학과(학사) 2005년 연세대학교 컴퓨터과학과(공학석사). 2005년~현재 연세대학교 컴퓨터과학과 박사과정. 관심분야는 프로세서 및 내장형 시스템 구조, 고성능 메모리 계층구조

김 신 덕

1982년 연세대학교 공과대학 전자공학과(학사). 1987년 University of Oklahoma 전기공학(공학석사). 1991년 Purdue University 전기공학(공학박사). 1993년~1995년 광운대학교 컴퓨터공학과 조교수. 1995년~현재 연세대학교 공과대학 컴퓨터과학과 교수. 관심분야는 고성능 컴퓨터 구조, 지능형 캐쉬 메모리, 병렬처리 시스템, 그리드 컴퓨팅 등