

저전력 에너지 관리 알고리즘 적용을 위한 하드웨어 움직임 추정기 구조 설계 및 특성 분석

김응섭, 이찬호

숭실대학교 전자공학과 VLSI System 연구실
전화: 02-820-0710

Design and Analysis of Motion Estimation Architecture Applicable to Low-power Energy Management Algorithm

Eung-Sup Kim, Chanho Lee

VLSI System Laboratory
department of electronic engineering, the Soongsil University
E-mail: congeal@hanmail.net, chlee@ssu.ac.kr

Abstract

The motion estimation which requires huge computation consumes large power in a video encoder. Although a number of fast-search algorithms are proposed to reduce the power consumption, the smaller the computation, the worse the performance they have.

In this paper, we propose an architecture that a low energy management scheme can be applied with several fast-search algorithm. In addition, we show that ECVH, a software scheduling scheme which dynamically changes the search algorithm, the operating frequency, and the supply voltage using the remaining slack time within given power-budget, can be applied to the architecture, and show that the power consumption can be reduced..

I. 서론

멀티미디어 응용에서 파워를 가장 많이 소모하는 곳은 비디오 인코딩(video encoding) 부분이다. 그 중에서도 동영상 압축에 필수적으로 사용되는 움직임 추정(motion estimation) 부분이, 그것의 엄청난 연산량으로 인해 인코더(encoder)가 소모하는 파워의 대부분을 차지하고 있다[1]. 이런 움직임 추정의 연산량을 줄이고자 여러 가지 고속 탐색 알고리즘(fast-search algorithm)이 제안되어 왔다. 하지만 일반적으로 연산량이 적은 알고리즘일수록 성능은 떨어진다.

S. Lee 와 T. Sakurai 가 제안한 ECVH(Energy-

Constrained Vdd Hopping)[4]는 소프트웨어 태스크 스케줄링(software task scheduling)방법의 하나로, task 의 완료 시간이 대부분 WCET(worst case execution time)을 넘지 않는다는 점을 이용하여 수행 시간 중에 동적으로 사용하는 알고리즘 및 동작 주파수, 공급 전압을 선택한다. 이렇게 함으로써 품질의 저하 없이 적은 파워로 태스크를 수행시킬 수 있다.

본 논문에서는, ECVH를 적용시킬 수 있는 하드웨어 구조에 대해 제안하고, 제안된 하드웨어에 추가한 저전력 방법에 대해 설명한다. 그리고, 제안된 하드웨어에 ECVH를 적용시켰을 때의 성능 및 파워 소모를 시뮬레이션을 통해 알아보도록 하겠다.

II. 제안하는 하드웨어 구조

ECVH를 움직임 추정에 적용시키기 위해 H. Jong, 이 제안한 움직임 추정기 구조[5]를 검토하였다. 그러나 이 구조는 수행 사이클 수가 일정하여 ECVH를 적용할만한 남은 시간(slack time)이 없고, 3개의 PE(Processing Element)를 갖는 변형에서는 파워를 많이 소모하는 외부 메모리 참조(access)가 많았다. 본 논문에서는 ECVH를 적용시킬 수 있도록 원 구조의 데이터 흐름(data-flow)을 변경하고 R. Richmond, 가 제안한 저전력 방법[6]을 추가한 새로운 구조를 제안하고자 한다.

2.1 여러 알고리즘의 적용 및 연산 순서

(1) 여러 알고리즘의 적용

ECVH에서 사용할 알고리즘으로 3 step search(TSS)[2], 4 step search(FSS)[3]와, 각각에 대한 multi-candidate 변형 알고리즘[7]을 사용한다.

TSS, FSS의 알고리즘을 살펴보면 기본적으로 스텝(step) 당 9개의 옵셋 포인트(offset point)에 대해 SAD를 구하는 형태를 볼 수 있고, FS의 경우에는 옵셋 포인트를 9개씩 묶으면 하나의 스텝으로 볼 수 있다. 이와 같은 유사점을 이용하여 하나의 하드웨어 구조를 통해 여러 가지 알고리즘을 구현할 수 있다.

(2) 연산 순서

그림 1은 움직임 추정기의 연산 순서를 보여주고 있는 그림이다. 연산은 9개의 옵셋 포인트를 동일 행(row)에 위치하는 3개의 옵셋 포인트씩 3개의 행으로 묶고, 한 번에 한 행씩 순서대로 진행한다. 즉 3개의 PE가 먼저 첫 번째 행에 위치하는 3개의 옵셋 포인트에 대해 계산을 끝내고, 다음 행을 계산하게 된다.

2.2 저전력 방법(low power scheme)

(1) 화소 재사용(Pixel reuse)

검색 영역 화소(Search region pixel)들은 보통 여러 번 참조된다. 하지만 화소를 가져오기 위해 외부 메모리를 참조(access)하는 데에는 많은 에너지가 소모되기 때문에 되도록이면 줄이는 것이 좋다. 그림 2를 보면 인접한 매크로 블록(macro block)의 검색 영역(search region)이 서로 겹치는 것을 볼 수 있는데, 이러한 특성을 이용해 화소를 재사용함으로써 외부 메모리 참조를 줄일 수 있다.

본 논문에서는 HSA(Half Search Area buffer)를 이용한 가로로 인접한 macroblock 간의 겹치는 search region pixel을 재사용하는 방법[5]을 사용하였을 뿐만 아니라, 세로로 인접한 macroblock에 대한 연산을 동시에 진행함으로써 세로로 인접한 search region pixel들도 재사용하였다.

(2) 불필요한 계산 제거

한 행에 대한 계산이 끝나고 나면 그 행에 위치하는 3개 옵셋 포인트 중의 최소의 SAD를 갖는 포인트를 알 수 있다. 다음 행을 계산할 때 각 포인트에 대한 SAD 값이 이전 행의 최소 SAD 값을 넘어서면 그 포인트

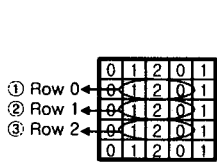


그림 1. 움직임 추정기의 연산 순서

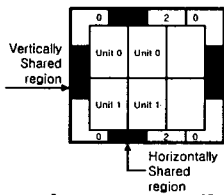


그림 2. Pixel reuse를 통한 메모리 참조 감소

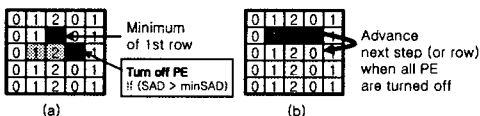


그림 0. 저전력 기법이 적용된 연산 방법

(a) 불필요한 연산 제거 (b) 이른 종료

트에 대한 SAD를 구할 필요가 없다. 또한 TSS의 경우를 예로 들면, 2번째 이후 스텝부터의 중심점(center

point)은 이전 스텝에서 이미 SAD를 구했기 때문에 다시 구할 필요가 없다.

그림 3(a)는 두 번째 행의 오른쪽 포인트가 이전 행의 최소 SAD 값을 넘어서 계산이 정지되는 예를 보여주고 있다.

(3) 이른 종료(Early termination)

하나의 행을 계산할 때 3개의 옵셋 포인트 모두 이전 행 또는 이전 스텝에서 계산한 최소 SAD를 넘어서 모든 PE의 동작이 중지되 되면, 다음 행 또는 다음 스텝으로 진행시키지 수행 사이클(clock cycle) 수를 줄일 수 있다. 이른 종료(Early termination)로 인한 남은 시간(slack time)을 이용하여 ECVH를 적용할 수 있다. 그림 3(b)는 첫 번째 행의 모든 포인트가 이전 최소 SAD 값을 넘어 다음 행으로 넘어가는 예를 보여주고 있다.

2.2 제안하는 하드웨어 구조

(1) 최상위 블록(Top-level block)

그림 4(a)는 전체 하드웨어 구조를 나타내고 있으며 최상위 블록은 연산 유닛들과 전력 관리자, 메인 컨트롤러로 구성된다.

1) 연산 유닛(operation unit)

그림 4(b)는 연산 유닛의 내부 구조를 보여주는 그림이다. 연산유닛은 한 번에 하나의 매크로 블록에 대한 움직임 벡터(motion vector)를 계산하며 높은 처리율을 요구하는 경우에는 연산유닛 여러 개를 사용하여 동시에 여러 매크로 블록을 처리하게 할 수 있다.

2) 메인 컨트롤러(Main controller)

외부 메모리 참조(access) 및 연산 유닛을 제어한다.

3) 전력 관리자(Power manager)

ECVH를 제어하는 block으로 수행 시간 중에 동적으로 움직임 추정에 사용할 알고리즘 및 동작 주파수, 공급 전압을 결정하는 역할을 한다.

(2) 메모리 모듈(Memory module)

본 논문에서는 분산 구성된 메모리 모듈을 사용한다 [5]. 단, 숫자는 9개에서 3개로 통합하고 인터리빙(interleaving) 방법을 변경하였다. 메모리 모듈(M으로 표기)은 PE의 개수와 같이 3개가 존재하며, 하나의 메모리 모듈은 3개의 RAM으로 구성된다. 하나의 HSA는 메모리 모듈 당 하나씩, 3개의 RAM으로 구성된다.

현재 화소 저장(load)용으로 HSAi가 선택됐다고 가정하자. HSAi를 구성하는 검색 영역 화소(search region pixel)를 외부 메모리로부터 가져왔을 때, 그 화소를 Mi부터 시작하여 (i, (i+1)%3, (i+2)%3) 순서대로 인터리브(interleave)한다. 각 메모리 모듈로 인터리브된 화소는 RAMi에 저장된다. Mi부터 인터리브하는 이유는 인터리빙의 불연속을 없애 구하고자 하는 검색 영역 화소가 어느 메모리 모듈에 존재하는 지를 쉽게 알 수 있도록 하기 위함이다.

(3) PE (Processing element)

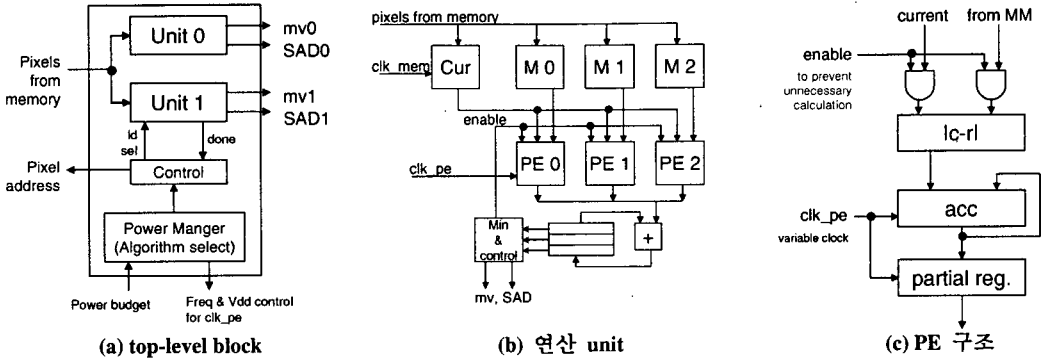


그림 4. 제한한 움직임 추정기 구조

그림 4. (c)는 PE의 구조를 보여주고 있는 그림으로, 기본적인 PE 구조[5]에 인에이블(enable) 입력을 추가한 것을 보여주고 있다[6]. 인에이블 신호를 받아 인에이블 되어 있을 때만 연산을 진행하며, 인에이블 되어 있지 않을 때는 입력이 뿔셈기로 들어가지 못하도록 하여 불필요한 에너지 소모를 막는다.

현재 화소(current pixel)의 비교 순서는 원 구조[5] 구조를 따르지만 인터리브 방법이 다르기 때문에 PE 기능 재분배(PE function redistribution) [5]순서는 달라진다. 각 행의 연산 위치를 왼쪽부터 L0, L1, L2라고 할 때 연산 위치를 1, 중앙점의 화소가 위치한 메모리 모듈의 번호를 k, 연산 위치간의 간격을 S, 각 매크로 블록의 행에서 PE 기능 재분배가 일어난 회수를 dis 라 하면, 매 순간에 각 연산 위치(i)를 계산하는 PE는 다음과 같이 표현된다: $PE[(dis + k + 2 * S + 1 * S) \% 3]$.

III. 실험 결과

본 논문에서는 제한한 구조를 시뮬레이션을 통해 분석하였다. 그림 5는 제작한 시뮬레이터를 보여주고 있다. 기준 클럭(reference clock period) 당 소모하는 에너지를 1로 볼 때, 각각의 저전력 방법을 사용할 때

소모하는 에너지를 표 1에 나타내었다. 괄호 안의 숫자는 각 알고리즘의 최악 에너지 소모(WCET와 같다)에 대한 비율이다. 참조 프레임(Sample frame)으로는 CIF(352x288) 크기를 갖는 foreman을 사용하였다. ECVH를 위한 주파수 레벨(frequency level set)은 $\{1, 0.75, 0.5\}$ 인 주파수를 시뮬레이션을 통해 최적으로 결정하여 사용하였다.

표 1. 각 저전력 방법에 대한 에너지 소모

	TSS	FSS	TSS2	FSS2	ADT***
PSNR	30.87	31.13	31.18	31.23	31.23
WCET	2312	3084	3848	5388	5388
No tech	2312	3084	3848	5388	-
Early*	1897.49 (82%)	1836.78 (60%)	3261.82 (85%)	2890.52 (54%)	-
ECVH**	1547.64 (67%)	1094.85 (36%)	2752.30 (72%)	1523.04 (28%)	1475.33 (27%)

* 이른 종료(Early termination)

** 단일 알고리즘만 사용하여 ECVH를 적용

***TSS, FSS, TSS2, FSS2 알고리즘을 사용하여 ECVH를 적용

표 1의 결과를 보면, 각 알고리즘에 대해 저전력 기법을 적용시켰을 때 줄어드는 에너지량을 알 수 있으며, 이른 종료만을 적용시켰을 때 54~85% 정도의 에너지만을 소모하며, ECVH를 적용시켰을 때 28~72% 정도의 에너지만을 소모하는 것을 알 수 있다. 또한, 본 논문에서 고려한 모든 고속 탐색 알고리즘을 갖고 ECVH를 적용시켰을 때 FSS2와 동일한 PSNR에서 27% 정도에 에너지만을 소모하는 것을 알 수 있다. 이는 TSS만 가지고 ECVH를 적용시켰을 때보다도 적은 것으로, 여러 알고리즘을 갖고 ECVH를 적용시켰을 때, 더 적은 에너지를 소모하면서 더 좋은 성능을 얻을 수 있는 것을 알 수 있다.

그림 6은 전력 예산(power budget)에 따라 PSNR과 소모하는 에너지를 나타내고 있는 그림으로, 에너지는 FSS 2-candidate 알고리즘을 저전력 방법을 사용하지 않았을 때 소모하는 에너지로 정규화(normalize)하였다. 그림 6은 ECVH이 갖는 성능과 에너지 소모간의 절충(trade-off) 관계를 보여주고 있다. ECVH에서는 알고리즘의 성능이 WCET에 비례하여 증가한다고 가정[4]하고 있지만, 움직임 추정의 고속 탐색 알고리즘



그림 5. 제작한 시뮬레이터

의 경우에는 이런 가정이 항상 성립하는 것은 아니었다. 따라서 PSNR 그래프는 전력 예산(power budget)에 따라 단조 증가하지 않는 것을 볼 수 있었다.

그림 7은 전력 예산에 따라 사용되는 알고리즘 분포를 보여주고 있다. 전력의 여유가 없을 때는 WCET가 작은 알고리즘이, 여유가 많을 때는 WCET가 큰 알고리즘을 사용하려는 경향이 있으며 그 사이를 중간 정도의 WCET를 갖는 알고리즘이 매우고 있는 것을 알 수 있었다.

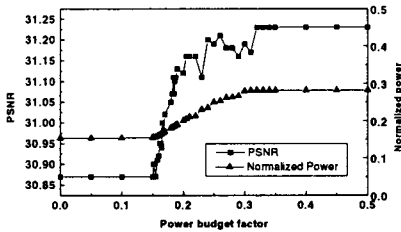


그림 6. Power budget에 따른 PSNR과 Power

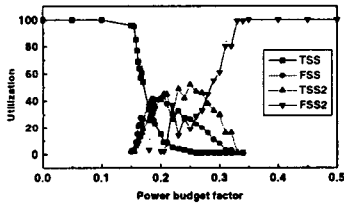


그림 7. 알고리즘 사용 분포

IV. 결론

본 논문에서는 여러 가지 알고리즘을 적용시킬 수 있는 저전력 움직임 추정기 구조를 제안하였다. 이 구조에는 소비 전력을 줄이기 위해 저전력 방법을 적용하였으며, ECVH를 사용할 수 있도록 하였다.

또한 제안한 구조에 저전력 기법을 적용했을 때 성능과 에너지 소모를 제작한 시뮬레이터를 통해 구하였다. 실험 결과, 이른 종료만을 적용시켰을 때 54~85% 정도의 에너지를 소모하며, 단일 알고리즘에 대해 ECVH를 적용시켰을 때 28~72% 정도의 에너지만을 소모하는 것을 알 수 있었다. 또한 여러 가지 알고리즘을 갖고 ECVH를 적용시켰을 때 같은 성능에서 27% 정도의 에너지만을 소모하는 것을 알 수 있었다.

감사의 글

본 연구는 소프트웨어 진흥원 IT-SoC 학생 인력 양성 사업의 지원으로 이루어졌습니다.

References

- [1] Vasily G. Moshnyaga, "A New Architecture for Computationally Adaptive Full-Search Block-Matching Motion Estimation," Proceedings of the 1999 IEEE ISCS, vol.4, pp.219-22, 1999
- [2] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, "Motion compensated interframe coding for video conferencing," National Telecommunications Conference 3, pp. G5.3.1-G5.3.5, 1981
- [3] Lai-Man Po and Wing-Chung Ma, "A Novel Four-Step Search Algorithm for Fast Block Motion Estimation," IEEE TCSVT, vol.6, no.3, pp.313-17, June 1996
- [4] S. Lee, S. Lee, and T. Sakurai, "Journal of Circuits, Systems, and Computers," Vol. 11, No.6, 601-620, 2002
- [5] H. Jong, L. Chen, and T. Chiueh, "Parallel Architectures for 3-Step Hierarchical Search Block-Matching Algorithm," IEEE TCSVT, vol.4, no.4, August 1994
- [6] R. Richmond, D. Ha, "A Low-Power Motion Estimation Block for Low Bit-Rate Wireless Video," ISLPED, pp. 60-63, Huntington Beach, CA, USA, August 2001
- [7] H. Jong, L. Chen, and T. Chiueh, "Accuracy Improvement and Cost Reduction of 3-Step Search Block Matching Algorithm for Video Coding," IEEE TCSVT, vol.4, no.1, February 1994