DOI: 10.3745/KIPSTA.2011.18A.3.099

임베디드 병렬 프로세서를 위한 픽셀 서브워드 병렬처리 명령어 구현

정 용 범† · 김 종 면††

요 약

프로세서 기술은 공정비용의 증가와 전력 소모 때문에 단순 동작 주파수를 높이는 방법이 아닌 다수의 프로세서를 집적하는 병렬 프로세싱 기술 발전이 이루어지고 있다. 본 논문에서는 멀티미디어에 내재한 무수한 데이터를 효과적으로 처리할 수 있는 SIMD(Single Instruction Multiple Data) 기반 병렬 프로세서를 소개하고, 또한 이러한 SIMD 기반 병렬 프로세서 아키텍처에서 이미지/비디오 픽셀을 효율적으로 처리 가능한 픽셀 서브워드 병렬처리 명령어를 제안한다. 제안하는 픽셀 서브워드 병렬처리 명령어는 48비트 데이터패스 아키텍처에서 4개의 12비트로 분할된 레지스터에 4개의 8비트 픽셀을 저장하고 동시에 처리함으로써 기존의 멀티미디어 전용 명령어에서 발생하는 오버플로우 및 이를 해결하기 위해 사용되는 패킹/언팩킹 수행의 상당한 오버헤드를 줄일 수 있다. 동일한 SIMD 기반 병렬 프로세서 아키텍처에서 모의 실험한 결과, 제안한 픽셀 서브워드 병렬처리 명령어는 baseline 프로그램보다 2.3배의 성능 향상을 보인 반면, 인텔사의 대표적인 멀티미디어 전용 명령어인 MMX 타입 명령어는 baseline 프로그램보다 단지 1.4배의 성능 향상을 보였다. 또한, 제안한 명령어는 baseline 프로그램보다 2.5배의 에너지 효율 향상을 보인 반면, MMX 타입 명령어는 baseline 프로그램보다 단지 1.8배의 에너지 효율 향상을 보였다.

키워드 : 멀티미디어 전용 명령어, SIMD 병렬 프로세서, 이미지/비디오 처리

Implementation of Pixel Subword Parallel Processing Instructions for Embedded Parallel Processors

Yong-Bum Jung† · Jong-Myon Kim††

ABSTRACT

Processor technology is currently continued to parallel processing techniques, not by only increasing clock frequency of a single processor due to the high technology cost and power consumption. In this paper, a SIMD (Single Instruction Multiple Data) based parallel processor is introduced that efficiently processes massive data inherent in multimedia. In addition, this paper proposes pixel subword parallel processing instructions for the SIMD parallel processor architecture that efficiently operate on the image and video pixels. The proposed pixel subword parallel processing instructions store and process four 8-bit pixels on the partitioned four 12-bit registers in a 48-bit datapath architecture. This solves the overflow problem inherent in existing multimedia extensions and reduces the use of many packing/unpacking instructions. Experimental results using the same SIMD-based parallel processor architecture indicate that the proposed pixel subword parallel processing instructions achieve a speedup of 2.3x over the baseline SIMD array performance. This is in contrast to MMX-type instructions (a representative Intel multimedia extension), which achieve a speedup of only 1.4x over the same baseline SIMD array performance. In addition, the proposed instructions achieve 2.5x better energy efficiency than the baseline program, while MMX-type instructions achieve only 1.8x better energy efficiency than the baseline program.

Keywords : Multimedia Specific Instructions, SIMD Parallel Processor, Image/Video Processing

1. 서 론

최근 모바일 멀티미디어 기기들의 사용이 증가하고, 이러한 기기에서 처리되는 멀티미디어 정보의 양이 방대해 짐에 따라 정보를 얼마나 효율적으로 처리하는가 하는 문제가 중요한 이슈가 되고 있다. 특히 이미지/비디오와 같은 멀티미디어 데이터는 멀티미디어 기기에 있어서 정보를 전달하는

※ 이 논문은 2011년도 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. 2011-0017941).
† 준 회 원 : 울산대학교 전기공학부 석사과정
†† 정 회 원 : 울산대학교 컴퓨터정보통신공학부 교수(교신저자)
논문접수 : 2011년 1월 12일
수 정 일 : 1차 2011년 3월 10일
심사완료 : 2011년 3월 11일

중요한 수단이지만 상당한 양의 연산과 입·출력 처리를 요구하기 때문에 고성능, 저전력의 멀티미디어 처리에 대한 필요성이 높아지고 있다.

실시간 서비스를 만족시키기 위해 기존의 모바일 멀티미디어 기기에 사용된 범용 마이크로프로세서의 성능 발전은 동작 주파수의 향상으로 발전해 왔지만, 프로세서 공정비용이 기하급수적으로 증가하면서 프로세서 기술 발전 방향이 동작 주파수의 고도화가 아닌 다수의 프로세서를 집적하는 멀티코어 혹은 매니코어 프로세서로 기술 발전이 이루어지고 있다[1]. 멀티미디어 애플리케이션의 효율적인 처리를 위한 고성능 프로세서 모델 중에서 SIMD (Single-Instruction Multiple-Data) 기반 병렬 프로세서 아키텍처가 유망한 대안으로 부각되고 있다[2],[3]. 명령어 레벨 (Instruction-Level) 이나 쓰레드 레벨 (Thread-Level) 프로세서들은 실리콘 면적을 멀티포트 레지스터 파일(multiported register file), 캐쉬(cache), deep pipelined 기능 유닛 등으로 사용하는 반면, SIMD 병렬 프로세서는 수천 개의 저비용 프로세싱 엘리먼트(processing element, PE)들을 이용하여 고성능을 추구하고 동시에 저장장소와 데이터 통신 요구를 최소화하기 위해 PE 와 데이터 입출력을 동일위치에 배치함으로써 저전력을 만족시킨다. 특히, SIMD기반 병렬 프로세서는 지역성(locality)이나 규칙성(regularity)이 있는 2차원 패턴의 이미지나 비디오 픽셀 처리에 있어서 최적의 프로세서 구조이다[4].

본 논문에서는 멀티미디어에 내재한 무수한 데이터 레벨 병렬성(Data-Level Parallelism, DLP)을 효과적으로 처리할 수 있는 SIMD 기반 병렬 프로세서 아키텍처를 소개할 뿐만 아니라, 픽셀 서브워드 병렬처리 명령어를 제안하여 멀티미디어 처리에서 DLP뿐만 아니라 서브워드 레벨 병렬성(Subword-Level Parallelism, SLP)을 추구함으로써 고성능을 얻고자 한다. 제안하는 픽셀 서브워드 병렬처리 명령어는 48비트 데이터패스 아키텍처에서 하나의 레지스터를 4개의 12비트 공간으로 분할하고 각 12비트 공간에 8비트 픽셀 데이터를 저장하고 동시에 처리함으로써 기존의 멀티미디어 전용 명령어 (e.g., Intel MMX [5]와 SSE [6], Hewlett Packard MAX-2 [7], Sun VIS [8], Alpha MVI [9], Motorola ALTIVEC [10])에서 발생하는 오버플로우 (48비트 레지스터를 6개의 8비트 공간으로 분할하고 각 8비트 공간에 8비트 픽셀을 저장하고 처리함으로써 발생) 및 이를 해결하기 위해 사용되는 상당한 패킹/언패킹 명령어에 대한 오버헤드 문제를 해결할 수 있다.

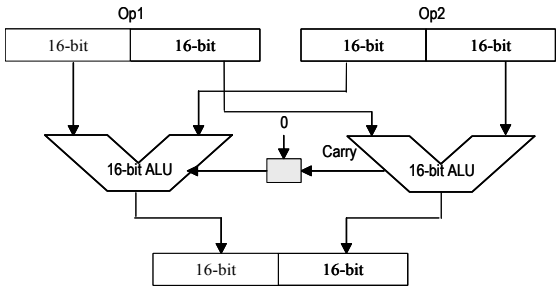
본 논문에서는 SIMD 기반 병렬 프로세서 아키텍처를 이용하여 제안하는 픽셀 서브워드 병렬처리 명령어가 이미지 처리 애플리케이션의 성능 (performance) 및 효율 (efficiency)에서 어떤 영향을 미치는지 평가하고, 또한 기존의 멀티미디어 전용 명령어와의 성능 비교를 통하여 그 우수성을 입증하고자 한다. 동일한 SIMD기반 병렬프로세서에서 모의실험한 결과, 제안한 픽셀 서브워드 병렬처리 명령어 기반 프로그램은 기존의 baseline 프로그램보다 2.3배 성능 및 2.5배 에너지 효율 향상을 보인 반면, 인텔사의 대표적인 멀티미디어 전용 명령어인 MMX 타입 명령어는 baseline 프로그램보다 단지 1.4배 성능 및 1.8배 에너지 효율 향상을 보였다. 이러한 결과들은 제안한 픽셀 서브워드 병렬처리 명령어가 멀티미디어용 프로세서에 적용될 경우, 상당한 성능 및 에너지 효율 향상이 이루어 질 것으로 기대된다.

본 논문은 다음과 같이 구성되어 있다. 2장에서는 관련 연구를 소개하고, 3장에서는 본 논문에서 제안하는 픽셀 서브워드 병렬처리 명령어를 기술한다. 4장에서는 시뮬레이션 환경에 대하여 설명하고, 5장에서는 제안하는 픽셀 서브워드 병렬처리 명령어의 성능 분석 및 인텔사의 대표적인 멀티미디어 전용 명령어인 MMX 타입 명령어와의 성능을 비교한다. 끝으로 6장에서는 이 논문의 결론을 맺는다.

2. 관련 연구

2.1 범용 마이크로프로세서용 멀티미디어 전용 명령어

범용 마이크로프로세서 제조회사는 멀티미디어 애플리케이션의 성능을 향상시키기 위해 멀티미디어 전용 명령어를 그들의 instruction set architecture (ISA)에 추가하였다. <표 1>은 기존의 마이크로프로세서 제조회사에서 발표한 멀티미디어 전용 명령어의 리스트를 보여준다[11]. 이러한 멀티미디어 명령어의 주요 장점은 하나의 넓은 레지스터 (예를 들어 64 비트 혹은 128비트)에 여러 개의 작은 데이터를 저장하고 동시에 처리함으로써 성능을 향상시킨다(그림 1 참조).



(그림 1) 분할된 ALU 기능 유닛

제조회사의 타겟 애플리케이션에 따라 이러한 멀티미디어 전용 명령어는 다양하다. Motorola AltiVec은 가장 많은 수의 멀티미디어 전용 명령어 (162개)를 가지고 있는 반면, HP MAX-1은 단지 8개의 멀티미디어 전용 명령어를 가지고 있다. 대부분의 멀티미디어 명령어 (AMD 3DNow!, DEC MVI, Intel MMX, Sun VIS)는 64 비트 레지스터를 사용하는 반면, Motorola AltiVec과 Intel SSE는 128 비트 레지스터를 이용한다. 한가지 주목할 만한 예외는 MIPS MDMX인데 이 명령어는 여러 번의 연산에 의한 결과를 축적하기 위해 하나의 넓은 누산기를 가지고 있다. 이러한 명령어의 유사성에도 불구하고 각각의 명령어는 독특한 특징을 가지

〈표 1〉 멀티미디어 전용 명령어 예

Multimedia Extension	MDMX	MMX	VIS	MAX2	MVI
Company	MIPS(SGI)	INTEL	SUN	HP	DEC
No. of registers	32	8	32	31	31
Register type	FP	FP	FP	Integer	Integer
Subword	8X8-bit 4X16-bits	8X8-bit 4X16-bits	8X8-bit 2X32-bits	4X16-bit	MIN/MAX only
Unsaturation	No	Yes	Yes	Yes	N/A
Saturation	Yes	Yes	No	Yes	N/A
3 operand	Yes	No(2)	Yes	Yes	Yes
Parallel multiply	4 or 8	4	4	None	None
Multiply-and-add	8X8-bit 8X16-bits	16X16=32	8X16=16	Shift and Add	None
Vector-to-scalar	Yes	No	No	No	No
Address manipulation	No	No	Yes	No	No
Parallel shift	Yes	No	No	No	No
Parallel average	No	No	No	Yes	No
Parallel compare	Yes	Yes	Yes	No	No
Pack/Unpack	Yes	Yes	Yes	No	No
Interleave	Yes	Yes	Yes	Yes	No
Permute	No	No	No	Yes	No
Motion estimation	No	No	Yes	No	Yes
Block load/store	No	No	Yes	No	No
Parallel FP	Yes	No	No	No	No

고 있다. 예를 들어, MAX-2는 실행 유닛과 정수 레지스터를 재사용하여 추가적인 하드웨어를 필요로 하지 않는 반면, AltiVec은 전적으로 새로운 실행 유닛을 요구한다.

2.2 SIMD기반 프로세서 관련 연구

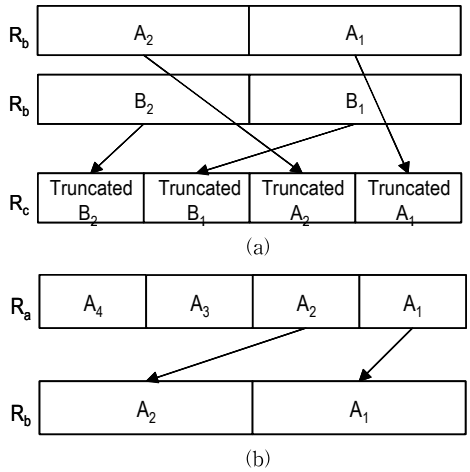
멀티미디어 애플리케이션에 대한 데이터 레벨 병렬성(Data-Level Parallelism, DLP)에 관한 연구는 크게 두 개의 연구 그룹으로 나누어 진다. (1) 현재의 SIMD 명령어를 이용하여 성능을 향상시키는 그룹 [12],[13],[14]과 (2) 병렬 프로세서를 이용하여 성능을 향상시키는 그룹 [2],[3],[15]. 많은 연구 그룹 혹은 개인들이 범용 마이크로프로세서에서 멀티미디어 애플리케이션에 대한 SIMD 명령어의 효율성에 대하여 분석하였다. [12]에서는 UltraSPARC 프로세서에서 이미지와 비디오 처리에 대한 VIS 명령어의 효율성을 기술하였다. 4-way out-of-order 프로세서는 single in-order 프로세서보다 2.3배~4.2배의 성능을 향상시켰고 더불어 VIS 명령어는 1.1배~4.2배의 성능을 더 향상시켰다. [13]에서는 DSP와 멀티미디어 애플리케이션에 대한 MMX 명령어의 성능 평가를 기술하였다. MMX 명령어는 81%의 다이내믹 명령어를 감소시켜 평균 5.5배의 성능 향상을 보였다. 이러한 결과에서 보는 바와 같이 SIMD 명령어는 적당한 수준의 성능을 향상시킨다. 하지만 멀티미디어 애플리케이션에 내재한 완전한 데이터 병렬성을 얻지 못하기 때문에 다양한 형태의 멀티미디어에서 요구되는 상당한 양의 성능 요구를 만족시키지 못한다.

SIMD기반 병렬 프로세서는 공간적 병렬성(spatial parallelism)을 실현하기 위해 여러 개의 동기화된 프로세싱 유닛(processing unit, PE)들을 사용한다. 이 유닛들은 하나의 제어 유닛으로부터 동시에 전송되는 동일한 연산 명령을 서로 다른 데이터에 대하여 수행함으로써 성능을 향상시킨다. SIMD기반 병렬 프로세서들은 거의 30년 동안 이미지와 같은 2차원 데이터 처리에서 효과적으로 사용되어 왔지만, 초기의 SIMD 병렬 프로세서 (TMC Connection Machine 1 [16])는 I/O 테크놀로지에 의해 제한되었다. 이후의 SIMD 병렬 프로세서인 TMC CM-2 [17]와 MasPar MP-2 [18]는 큰 버퍼 병렬 디스크 어레이의 사용을 통해 이러한 제한을 극복하였지만 비용과 이동성 (portability)을 희생하였다. Fine-grained 병렬 프로세서인 MGAP [19]와 ABACUS [20]는 이러한 이동성 이슈를 해결하였지만, 그들의 성능은 I/O 대역폭 (bandwidth)과 지연 (latency)에 의해 제한되었다.

이러한 SIMD기반 병렬 프로세서와 다르게, 본 논문에서 모의실험을 위해 사용한 SIMD Pixel Processor[3]는 프로세서와 센서의 직접적 연결을 통해 I/O 대역의 문제를 해결하고, 또한 짧은 와이어의 사용으로 높은 면적 효율과 에너지 효율을 보인다. 본 논문에서는 이 병렬프로세서 아키텍처를 이용하여 제안하는 픽셀 서브워드 병렬처리 명령어에 대한 성능 및 에너지 효율을 분석하여 그 우수성을 입증하고자 한다.

3. 이미지/비디오 전용 픽셀 서브워드 병렬처리 명령어

이미지/비디오 처리 애플리케이션에서 이미지의 각 픽셀은 8비트 단위로 구성되어 있다. 따라서 48비트 데이터패스 아키텍처에서 8비트 크기의 픽셀을 하나를 48비트 레지스터에 저장하고 처리하는 것은 나머지 40비트의 유휴비트가 생김으로써 공간적 낭비가 발생한다. 이러한 문제를 해결하기 위해 기존의 범용 마이크로프로세서 제조회사들은 멀티미디어 전용 명령어 (서브워드 병렬처리 기법)를 추가하여 8비트의 픽셀들과 같은 여러 개의 작은 데이터를 넓은 레지스터에 패킹하고 동시에 처리함으로써 멀티미디어의 성능을 향상시켰다. 하지만 48비트 워드 단위의 레지스터에 여섯 개의 8비트 서브워드를 임시 적재하고 여섯 개의 8비트 산술 논리 유닛들을 통해 상기 서브워드들을 병렬로 처리함으로써 오버플로우가 발생하게 되고, 또한 이를 해결하기 위해 상당한 패킹/언패킹 명령어 (그림 2 참조)를 사용함으로써 성능저하의 원인이 된다.

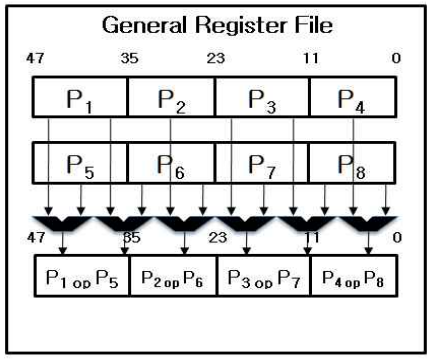


(그림 2) (a) 패킹 (packing) 명령어,
(b) 언패킹 (unpacking) 명령어

이러한 문제를 해결하기 위해 본 논문에서는 8비트 분할 공간을 12비트로 확장하여 픽셀 서브워드들을 병렬로 처리 가능한 명령어를 제안한다. 이러한 픽셀 서브워드 병렬처리 명령어는 48비트 데이터패스 아키텍처에서 48비트 레지스터를 4개의 12비트 공간으로 분할하고 4개의 8비트 픽셀 데이터를 적재 및 동시에 처리함으로써 기존의 멀티미디어 전용 명령어에서 발생하는 오버플로우 및 패킹/언패킹 명령어의 수를 상당히 줄일 수 있다. 제안하는 픽셀 서브워드 병렬처리 명령어는 다음과 같은 2개의 그룹 [(1) 병렬 산술·논리 (ALU) 명령어, (2) 특정목적 (Special-purpose) 명령어]으로 구성된다.

3.1 병렬 산술·논리(ALU) 명령어

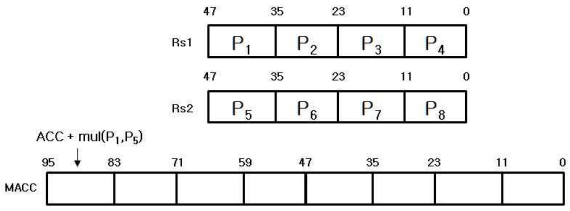
(그림 3)은 분할된 12비트 병렬 산술·논리 명령어의 예를 보여준다. 병렬 산술·논리 명령어에는 가산, 감산, 그리고 시프트 명령어가 있다. 가산/감산 명령어에는 오버플로우가 발생 할 때 주어진 데이터 형에서 가장 큰 값이나 작은 값을 반환하는 포화 연산자를 포함하고 있다. 이러한 포화 산술 연산자는 오버플로우가 발생하면 검정색에서 흰색으로의 변환을 막는 픽셀 관련 연산에 유용하다. 시프트 명령어는 각 레지스터에 저장된 각 서브워드를 동시에 시프트 처리한다.



(그림 3) 분할된 12비트 병렬 산술 명령어

3.2 특정 목적 명령어

특정 목적 (special-purpose) 명령어에는 ADACC (absolute-differences accumulate), MACC (multiply-accumulate), RAC (read accumulate), ZACC (zero accumulate) 등이 있으며, 성능 면에서 가장 뛰어난 명령어들이다. 예를 들어, MACC은 디지털 필터링 (digital filtering) 또는 컨볼루션 (convolution)과 같은 vector dot-product 연산을 수행하는 DSP 알고리즘에 유용하다. (그림 4)에서 보는바와 같이 두 개의 소스 레지스터에 있는 4개의 서브워드는 곱셈 연산이 수행되고 그 결과 값은 분할된 누산기 (accumulator)에 저장된다. ADACC는 다양한 움직임 추정과 같은 알고리즘에 사용된다. 마지막 두 명령어 (RAC, ZACC)는 누산기를 관리하는 명령어이다.

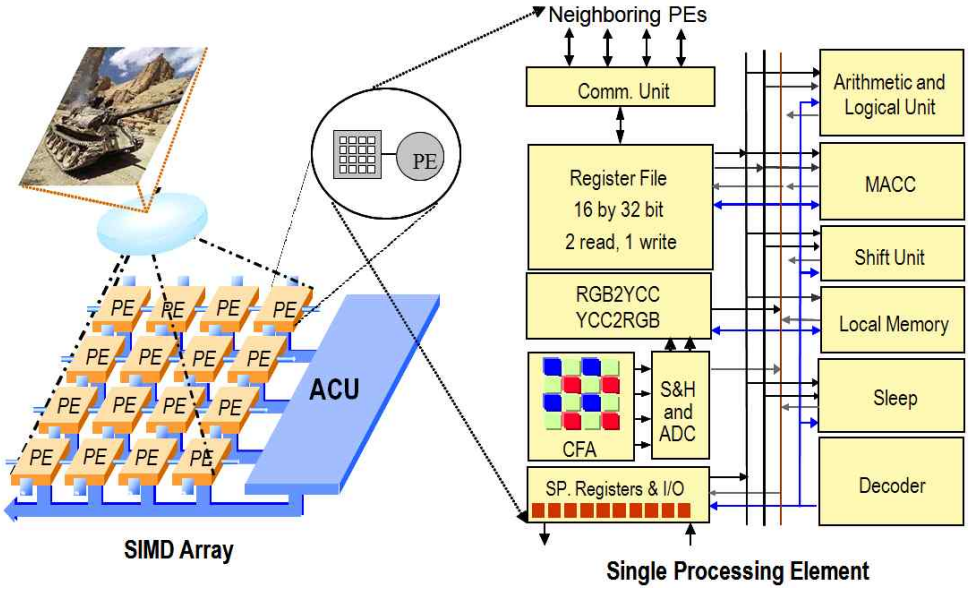


(그림 4) MACC 병렬 명령어 예

4. 시뮬레이션 환경

4.1 SIMD기반 병렬 프로세서 구조

본 논문에서 사용한 SIMD기반 병렬 프로세서는 (그림 1)과 같이 2차원 격자(mesh)구조로 구성된 PE어레이, 각 데이터의 입출력을 위한 로컬 메모리 및 각 PE와 입출력 유닛을 제어하는 ACU(Array Control Unit)으로 구성되어 있다.



(그림 5) SIMD 기반 병렬 프로세서 구조

각 PE의 이미지 센서는 이미지 프레임으로부터 정해진 픽셀 데이터를 추출하여 특정 레지스터 파일에 저장함으로써 I/O 대역의 문제를 해결한다. 또한, 짧은 와이어의 사용으로 높은 시스템 면적 효율과 에너지 효율을 보인다.

<표 2>는 사용된 병렬 프로세서 아키텍처 모델의 파라미터를 보여준다.

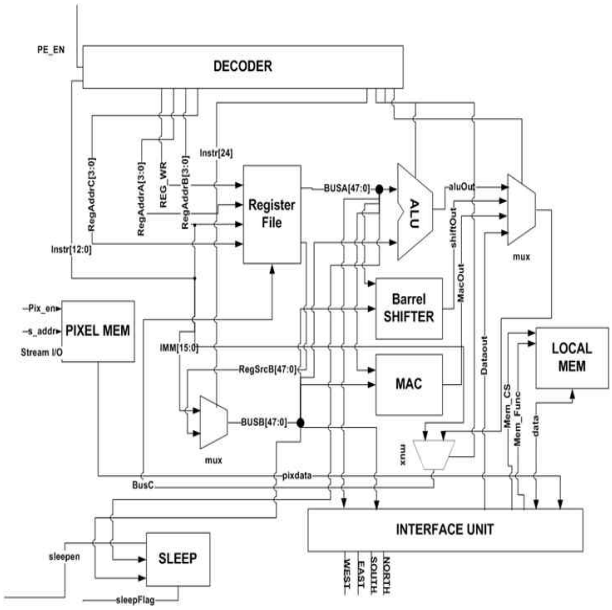
〈표 2〉 SIMD기반 병렬 프로세서 아키텍처 파라미터

Parameter	Value
System Size (# of PEs)	4096
Image Pixel per PE	16 (4x4)
VLSI Technology	130 nm
Clock Frequency	50 MHz
Interconnection Network	Mesh
intALU/intMUL/Barrel Shifter/intMACC/Comm	1 / 1 / 1 / 1 / 1 / 1
Local Memory Size	256 word

4.2 프로세싱 엘리먼트 구조

(그림 6)과 같이 각 프로세싱 엘리먼트는 다음과 같은 특징을 가진 RISC (Reduced Instruction Set Computer) 아키텍처이다.

- 256개 48비트 워드로 구성된 로컬 메모리
- 16개 48비트 3-포트 범용 레지스터
- 기본적인 산술/논리 연산을 수행하는 ALU
- 멀티 비트 산술/논리 시프트 연산을 수행하는 배럴 시프트 (Barrel Shifter)
- 곱셈 및 누산기 (multiply-accumulator, MACC)
- 지역 정보를 이용해 각 PE들을 활성 및 비활성 시키는 Sleep 유닛



(그림 6) 프로세싱 엘리먼트 구조

- 이웃하는 PE들과 데이터 통신을 위한 NEWS (north-east-west-south) 네트워크 및 serial I/O 유닛

4.3 이미지 애플리케이션

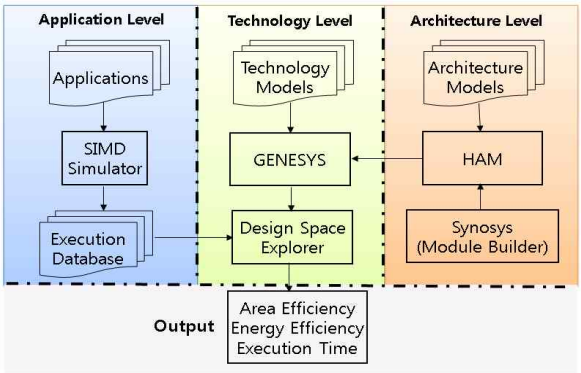
본 논문에서는 제안하는 픽셀 서브워드 병렬처리 명령어의 성능 및 효율을 분석하기 위해 <표 3>와 같이 네 가지의 대표적인 이미지 처리 알고리즘을 채택하여 구현하였다. 선택된 애플리케이션은 Blurring, Convolution, Median Filter, Moving Average Method이다. 모의실험에서 이러한 애플리케이션들은 256x256 해상도(resolution)의 이미지를 사용하여 구현되었다.

〈표 3〉 모의실험을 위해 선택된 이미지 처리 애플리케이션 요약

애플리케이션	설명
Blurring	영상에서 선명하지 않거나 흐린 부분을 제거하는 방법
Convolution	두 신호의 결합을 나타내는 한 방법으로 입력신호와 시스템특성으로부터 출력신호를 구할 때, 두 시스템의 연결특성을 구하고자 할 때 사용하는 방법
Median Filter	3x3 블록의 이미지 픽셀 값 중에서 중간 값을 중앙 값으로 대치하여 출력하는 방법
Moving Average Method	이동 평균법을 이용하여 출력하는 방법

4.4 실험 방법론

(그림 7)은 세 가지 레벨(애플리케이션, 아키텍처, 테크놀로지)로 구성되어 있는 SIMD 기반 병렬프로세서의 실험 방법론 구조를 보여준다. 애플리케이션 레벨에서는 SIMD기반 병렬 프로세서용 정밀 사이클(cycle-accurate) 시뮬레이터를 이용하여 사이클 개수, 동적 명령어 빈도, 시스템 이용률(system utilization) 등의 실행 데이터를 추출하였다. 아키텍처 레벨에서는 모델링된 아키텍처의 디자인 변수들을 계산하기 위해 Chai가 제안한 SIMD 기반 병렬 프로세서용 이종 아키텍처 모델링 툴을 사용하였다[21]. 테크놀로지 레벨에서는 각 아키텍처 모델들의 테크놀로지 변수(latency, power, clock frequency)를 계산하기 위해 Generic System Simulator(GENESYS)를 사용하였다[22]. 마지막으로 세 레벨에서 구해진 데이터베이스를 조합하여 각 경우에 대한 실행시간, 처리량, 에너지 효율을 결정하였다.



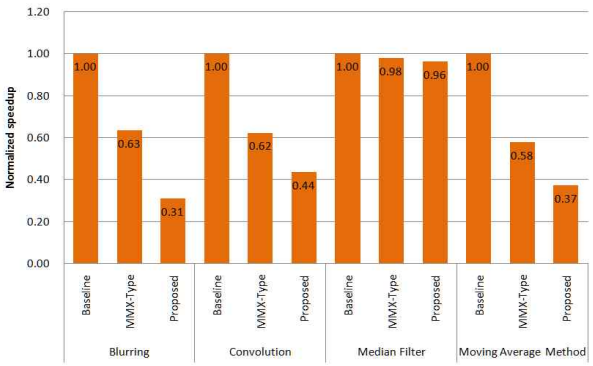
(그림 7) SIMD기반 병렬 프로세서를 위한 실험 방법론 구조

5. 모의실험 및 결과 분석

각 아키텍처 모델링을 이용하여 구현된 애플리케이션 태스크의 성능 및 효율을 결정하기 위해 본 논문은 cycle-accurate 시뮬레이션과 테크놀로지 모델링을 사용하였다. 각 애플리케이션은 병렬프로세서용 어셈블리 언어로 구현하였다. 공정한 평가를 위해 제안한 픽셀 서브워드 병렬처리 명령어 기반 프로그램, MMX 타입 명령어 기반 프로그램 및 baseline 명령어 기반 프로그램은 같은 변수, 데이터 셋, 콜링 시퀀스를 가진다. <표 4>는 각 경우의 성능 비교를 위한 3가지 지표(execution time, sustained throughput, energy efficiency)를 보여 준다.

5.1 성능평가 결과

(그림 8)은 SIMD기반 병렬 프로세서에서 수행된 baseline 프로그램 성능 대비 MMX 및 제안한 픽셀 서브워드 병렬처리 명령어 기반 프로그램의 성능을 보여준다. 제안한 명령어는 baseline 프로그램보다 약 2.3배 성능 향상을 보인 반면, MMX 기반 프로그램은 baseline 프로그램보다 단지 1.4배의 성능 향상을 보였다.



(그림 8) Baseline 프로그램 대비 MMX 및 제안한 명령어 기반 프로그램의 성능 비교

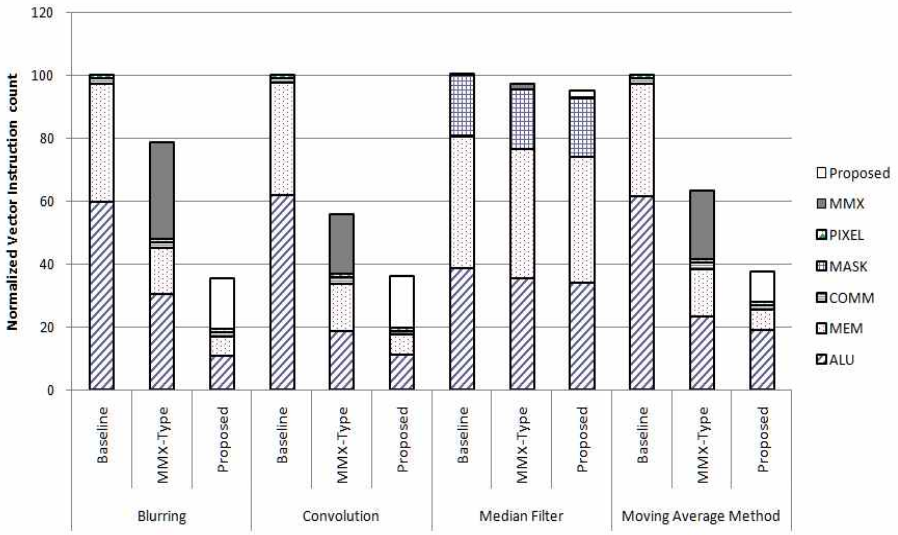
또한 MMX 기반 프로그램은 평균 362.9 Gops/sec의 처리량을 보인 반면, 제안한 명령어기반 프로그램은 평균 376.47 Gops/sec의 높은 처리량을 보였다. <표 5>는 4,096 PE 어레이 시스템에서 수행된 baseline, MMX 및 제안한 명령어 기반 프로그램들의 전체 성능을 보여준다.

〈표 4〉 평가 지표 요약

실행시간 (execution time)	처리량 (sustained throughput)	에너지 효율 (energy efficiency)
$t_{exec} = \frac{C}{f_{ck}}$	$Th_{sust} = \frac{O_{exec} \cdot U \cdot N_{PE}}{t_{exec}}$	$\eta_E = \frac{O_{exec} \cdot U \cdot N_{PE}}{Energy} [\frac{Gops}{Joule}]$
C : 사이클 개수, f_{ck} : 클럭 주파수, O_{exec} : 수행된 연산 개수 U : 프로세싱 엘리먼트 이용률, N_{PE} : 프로세싱 엘리먼트의 개수		

〈표 5〉 Baseline, MMX 및 제안한 명령어 기반 프로그램의 성능 비교

Application	ISA	Vector Instruction	Scalar Instruction	System Utilization [%]	Sustained Throughput [Gops/sec]
Blurring	Baseline	1,459	372	100	204.80
	MMX-Type	1,148	12	100	444.57
	Proposed	514	52	100	482.12
Convolution	Baseline	1,233	65	100	204.80
	MMX-Type	796	12	100	414.75
	Proposed	514	52	100	482.12
Median Filter	Baseline	10,759	2,772	83.32	169.98
	MMX-Type	10,656	2,604	82.05	177.77
	Proposed	10,392	2,644	82.89	179.18
Moving Average Method	Baseline	1,219	340	100	204.80
	MMX-Type	890	12	100	414.66
	Proposed	530	52	100	362.46



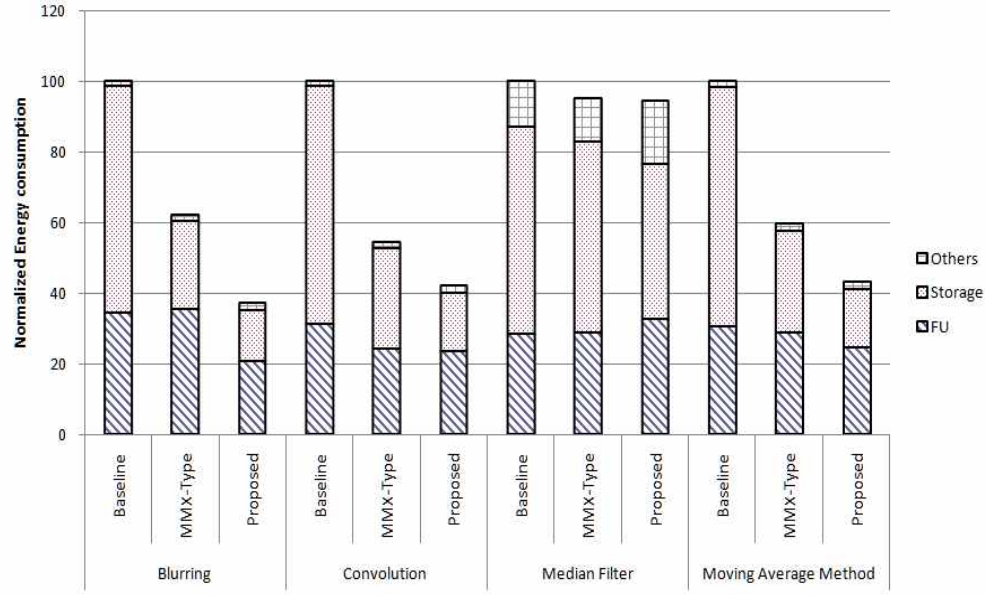
(그림 9) Baseline 프로그램 대비 MMX 및 제안한 명령어 기반 프로그램의 벡터 명령어

5.2 제안한 픽셀 서브워드 병렬처리 명령어의 이점

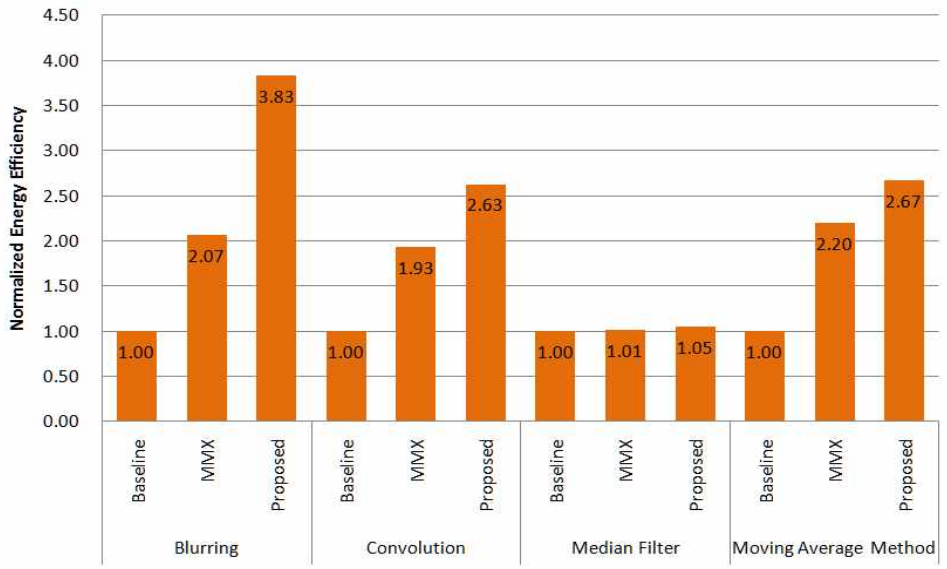
(그림 9)는 병렬 프로세서에서 수행된 baseline 프로그램 대비 MMX 및 제안한 픽셀 병렬처리 명령어 기반 프로그램의 벡터 명령어 분포도를 보여준다. 각 바 (bar)는 논리/연산 유닛 (ALU), 메모리 (MEM), 커뮤니케이션 유닛 (COMM), PE 동작 컨트롤 유닛 (MASK), 이미지 픽셀 로딩 유닛 (PIXEL), MMX, Proposed 명령어로 세분화 된다. 그림에서 보는 바와 같이, MMX 기반 프로그램은 baseline 프로그램에 비해 단지 1.34배의 명령어 개수를 감소시킨 반면, 제안하는 명령어는 baseline 프로그램 보다 평균 2.85배의 명령어 개수를 감소시켰다. 예상한 대로 제안한 명령어를 사용함으로써 ALU와 메모리에서 많은 명령어 개수를 줄였다.

5.3 에너지 효율 비교 결과

(그림 10)은 병렬 프로세서에서 baseline 프로그램 대비 MMX 및 제안하는 픽셀 서브워드 병렬처리 명령어 기반 프로그램의 에너지 소비를 보여준다. 각 바 (bar)는 기능 유닛 (FU: ALU, Barrel Shifter, MACC), storage (Register file, Memory), others (Communication, Sleep, Serial I/O, Decoder)하드웨어의 에너지 분포를 나타낸다. 동일한 클럭 주파수(50MHz), 공정 (130nm) 및 프로세서 파라미터에서 프로그램의 수행시간은 에너지 소비에 비례한다[23]. 따라서, 제안한 명령어를 이용하여 많은 양의 명령어 및 사이클을 줄였으므로 상당한 양의 소비 에너지를 감소시킬 수 있다. 제안한 명령어는 baseline 프로그램 대비 평균 2.44배의 소비 에너지를 감소시킨 반면, MMX는 baseline 프로그램 대비 단지 1.57배의 소비 에너지를 감소시켰다. 예상한 대로 제안한 명령어를 사용함으로써 상당한 수의 ALU와 memory 접근 수를 줄였기 때문에 적은 양의 에너지가 ALU와 storage 유닛에서 소모된다.



(그림 10) Baseline 프로그램 대비 MMX 및 제안하는 명령어 기반 프로그램의 소비 에너지



(그림 11) Baseline 프로그램 대비 MMX 및 제안한 명령어 기반 프로그램의 에너지 효율

(그림 11)은 baseline 프로그램 대비 MMX 및 제안한 픽셀 서브워드 병렬처리 명령어 기반 프로그램의 에너지 효율을 보여준다. MMX 기반 프로그램은 baseline 프로그램보다 평균 1.8배의 에너지 효율을 증가시킨 반면, 제안한 명령어는 baseline 프로그램보다 평균 2.54배의 에너지 효율을 증가 시켰다. 이러한 결과는 제안한 명령어가 적은 시스템 전력을 증가시키는 반면, 높은 처리량을 제공하기 때문이다. 에너지 효율의 증가는 시스템의 배터리 수명을 증가시키는 결과를 가져온다.

6. 결 론

본 논문에서는 멀티미디어에 내재한 무수한 데이터 병렬성을 효과적으로 처리할 수 있는 SIMD 기반 병렬 프로세서를 소개하였고, 또한 이미지 및 비디오 처리에서 효율적인 픽셀 서브워드 병렬처리 명령어를 제안하였다. 제안한 픽셀 서브워드 병렬처리 명령어는 48비트 데이터패스 아키텍처에서 분할된 4개의 12비트 레지스터 공간에 4개의 8비트 픽셀을 저장하고 동시에 처리함으로써 기존의 멀티미디어 전용 명령어에서 발생하는 오버플로우 및 이를 해결하

기 위해 사용되는 패킹/언패킹 명령어의 상당한 수를 줄일 수 있다. 동일한 병렬프로세서 아키텍처에서 모의 실험한 결과, 제안한 픽셀 서브워드 병렬처리 명령어는 baseline 프로그램보다 평균 2.3배의 성능 및 2.5배의 에너지 효율 향상을 보인 반면, 기존의 MMX 타입 명령어는 동일한 baseline 프로그램 보다 단지 1.4배의 성능 및 1.8배의 에너지 효율 향상을 보였다.

참 고 문 헌

[1] 정무경, 박성모, 엄낙웅, "병렬 프로세서 기술 및 동향", 전자통신동향분석 제24권, 제6호, 86-93쪽, 2009년 12월.

[2] A.D. Blas et. al., "The UCSC Kestrel Parallel Processor," IEEE Trans. on Parallel and Distributed Systems, vol.16, No.1, pp. 80-92, Jan., 2005.

[3] A. gentile and D. S. Wills, "Portable Video Supercomputing," IEEE Trans. on Computers, Vol.53, No.8, pp.960-973, Aug., 2004.

[4] Luong Van Huynh, 김철홍, 김종면, "퍼지 백터 양자화를 위한 대규모 병렬 알고리즘 ", 한국정보처리학회논문지 A, 제16-A권, 제6호, 411-418쪽, 2009년 12월.

[5] A. Peleg and U. Weiser, "MMX Technology Extension to the Intel Architecture," IEEE Micro, Vol.16, No.4, pp.42-50, Aug., 1996.

[6] S. K. Raman, V. Pentkovski, and J.Keshava,"Implementing Streaming SIMD Extensions on the Pentium III Processor," IEEE Micro, Vol.20, No.4, pp.28-39, 2000.

[7] R. B. Lee, "Subword Parallelism with MAX-2," IEEE Micro, vol. 16, no. 4, pp. 51-59, Aug. 1996.

[8] M. Tremblay, J. M. O'Connor, V. Narayanan, and L. He, "VISSpeedsNewMediaProcessing,"IEEE Micro, Vol.16, No.4, pp.10-20, Aug., 1996.

[9] R. Sites, Ed., Alpha Reference Manual, Burlington, MA:Digital,1992.

[10] H. Nguyen and L. John, "Exploiting SIMD Parallelism in DSP and Multimedia Algorithms using the AltiVec Technology," in Proc. Intl. Conf. on Supercomputer, pp.11-20, June, 1999.

[11] 박경, "멀티미디어 확장 명령어 세트의 조사", 정보통신산업진흥원, [IITA] 정보통신연구진흥원 학술정보 주간기술 853호, http://kidbs.itfind.or.kr/WZIN/jugidong/853/85302.html

[12] P. Ranganathan, S. Adve, and N. P. Jouppi, "Performance of image and video processing with general-purpose processors and media ISA extensions," in Proc. of the 26th Intl. Sym. on Computer Architecture, pp.124-135, May, 1999.

[13] R. Bhargava, L. John, B. Evans, and R. Radhakrishnan, "Evaluating MMX technology using DSP and multimedia applications," in Proc. of IEEE/ACM Sym. on Microarchitecture, pp.37-46, 1998.

[14] N. Slingerland, and A. J. Smith, "Measuring the performance of multimedia instruction sets," IEEE Trans. on Computers, Vol51, No.11, pp.1317-1332, Nov., 2002.

[15] A. Krikelis, I. P. Jalowiecki, D. Bean, R. Bishop, M. Facey, D. Boughton, S. Murphy, and M. Whitaker, "A programmable processor with 4096 processing units for media applications," in Proc. of the IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing, Vol.2, pp.937-940, May, 2001.

[16] L. W. Tucker, and G. G. Robertson, "Architecture and applications of the connection machine," IEEE Computer, Vol.21, No.8, pp.26-38, 1988.

[17] "Connection machine model CM-2 technical summary," Thinking Machines Corp., version 51, May, 1989.

[18] MarPar (MP-2) System Data Sheet. MarPar Corporation, 1993.

[19] M. J. Irwin, R. M. Owens, "A Two-Dimensional, Distributed Logic Processor," IEEE Trans. on Computers, Vol.40, No.10, pp.1094-1101, 1991.

[20] M. Bolotski, R. Armithrajah, W. Chen, "ABACUS: A High Performance Architecture for Vision," in Proceedings of the International Conference on Pattern Recognition, 1994.

[21] S. M. Chai, T. M. Taha, D. S. Wills, and J. D. Meindl, "Heterogeneous architecture models for interconnect-motivated system design," IEEE Trans. VLSI Systems, special issue on system level interconnect prediction, Vol.8, No.6, pp.660-670, Dec., 2000.

[22] J. C. Eble, V. K. De, D. S. Wills, and J. D. Meindl, "A generic system simulator (GENESYS) for ASIC technology and architecture beyond 2001," In Proc. of the Ninth Ann. IEEE Intl. ASIC Conf., pp.193-196, Sept., 1996.

[23] V. Tiwari, S. Malik, and A. Wolfe, "Compilation Techniques for Low Energy: An Overview," in Proc. of the IEEE Intl. Symp. on Low Power Electron., pp.38-39, Oct., 1994.

정 용 범

e-mail : smartnow@nate.com

2009년 울산대학교 컴퓨터정보통신공학부 (학사)

2009년~현재 울산대학교 석사과정

관심분야 : 병렬처리, 임베디드시스템, 컴퓨터구조, 임베디드 소프트웨어

김 종 면

e-mail : jmkim07@ulsan.ac.kr
1995년 명지대학교 전기공학과(학사)
2000년 Electrical & Computer Engineering, University of Florida, USA(공학석사)
2005년 Electrical & Computer Engineering, Georgia Institute of Technology, USA(공학박사)
2005년~2007년 삼성종합기술원 전문연구원
2007년~현 재 울산대학교 컴퓨터정보통신공학부 교수
관심분야 : 임베디드시스템, 시스템-온-칩, 컴퓨터구조, 병렬처리, 신호처리 등