韓國컴퓨터情報學會 論文誌
第16卷 第10號, 2011. 10.

2011-16-10-1-2

임베디드 병렬 프로세서 상에서 MMX타입 명령어의 성능평가 및 검증

정용범*, 김용민**, 김철홍***, 김종면****

Performance Evaluation and Verification of MMX-type Instructions on an Embedded Parallel Processor

Yong-Bum Jung*, Yong-Min Kim**, Cheol-Hong Kim***, Jong-Myon Kim****

요 약

본 논문에서는 멀티미디어에 내재한 무수한 데이터를 효율적으로 처리할 수 있는 SIMD(Single Instruction Multiple Data) 기반 병렬 프로세서를 소개한다. 또한, 인텔사의 대표적인 멀티미디어 전용 명령어인 MMX (MultiMedia eXtension)타입 명령어를 병렬 프로세서에 구현하여 성능을 평가하고 결과를 분석한다. 16개의 32-비트 프로세서로 구성된 병렬프로세서를 이용하여 1280x1024픽셀 이미지의 JPEG 압축 애플리케이션을 구현하고 모의 실험한 결과, 동일한 병렬프로세서 기반에서 MMX타입 명령어는 베이스라인 명령어보다 약 50%의 성능 향상을 보였다. 또한, MMX타입 명령어는 베이스라인 명령어보다 에너지 효율에서 100%, 시스템 면적 효율에서 51%의 향상을 보였다. 이러한 결과는 MMX를 포함한 멀티미디어 전용 명령어들이 현재 널리 사용되고 있는 매니코어 GPU(Graphics Processing Unit) 및 다양한 형태의 병렬프로세서에서도 잠재 가능성이 있음을 보여준다.

▸ Keyword : 멀티미디어 전용 명령어, SIMD기반 병렬프로세서, JPEG 압축 알고리즘, 매니코어 GPU

Abstract

This paper introduces an SIMD(Single Instruction Multiple Data) based parallel processor that efficiently processes massive data inherent in multimedia. In addition, this paper implements MMX(MultiMedia eXtension)-type instructions on the data parallel processor and evaluates and analyzes the performance of the MMX-type instructions. The reference data parallel processor consists of 16 processors each of which has

• 제1저자 : 정용범 • 교신저자 : 김종면
• 투고일 : 2011. 03. 20, 심사일 : 2011. 04. 04, 게재확정일 : 2011. 04. 13.
* 울산대학교 전기공학부(School of Electrical Engineering, University of Ulsan)
** 울산대학교 전기공학부(School of Electrical Engineering, University of Ulsan)
*** 전남대학교 전자컴퓨터공학부(Electronics and Computer Engineering, Chonnam National University)
**** 울산대학교 전기공학부(School of Electrical Engineering, University of Ulsan)
※ 이 논문은 2011년도 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. 2011-0017941). 또한 이 논문은 2011년 울산대학교의 연구비에 의하여 연구되었음.

a 32-bit datapath. Experimental results for a JPEG compression application with a 1280x1024 pixel image indicate that MMX-type instructions achieves a 50% performance improvement over the baseline instructions on the same data parallel architecture. In addition, MMX-type instructions achieves 100% and 51% improvements over the baseline instructions in energy efficiency and area efficiency, respectively. These results demonstrate that multimedia specific instructions including MMX-type have potentials for widely used many-core GPU(Graphics Processing Unit) and any types of parallel processors.

▸ Keyword : Multimedia-specific instructions, SIMD based parallel processor, JPEG compression algorithm, many-core GPU

I. 서 론

최근 모바일 멀티미디어 제품들의 사용이 증가하고, 이러한 기기에서 처리되는 다양한 멀티미디어 정보의 양이 방대해짐에 따라 멀티미디어 데이터를 얼마나 효율적으로 처리하는가 하는 문제가 크게 대두하고 있다.

이런 문제점을 해결하기 위해 프로세서 성능 발전은 동작 주파수의 향상으로 발전해왔다. 하지만, 공정 기술의 발전이 물리적인 이유 때문에 예전과 같은 속도의 발전이 어려울 뿐만 아니라, 공정비용이 기하급수적으로 증가하면서 프로세서 성능 발전이 단순 동작 주파수의 향상이 아니라 병렬 프로세서 기술로 전환되고 있다[1].

표 1. 멀티미디어 전용 명령어 예
Table 1. Examples of multimedia instructions

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

프로세서 기술 발전 방향이 병렬 프로세싱으로 전환되는 이유 중 하나는 프로세서가 수행하는 다양한 멀티미디어 애플리케이션들에 있다. 이러한 멀티미디어 애플리케이션들은 상당한 양의 데이터 및 입출력을 요구한다. 또한 모바일 환경에서는 고성능과 더불어 저전력에 대한 요구도 만족되어야 한다.

멀티미디어 애플리케이션의 효율적인 처리를 위한 고성능, 저전력 프로세서 모델 중에서 SIMD (Single-Instruction Multiple-Data) 기반 병렬 프로세서 아키텍처가 유망한 대안으로 부각되고 있다[2],[3]. 명령어 레벨 (Instruction-Level) 이나 스레드 레벨 (Thread-Level) 프로세서들은 실리콘 면적을 멀티포트 레지스터 파일(multiported register file), 캐쉬 (cache), 파이프라인 (deep pipelined) 기능 유닛 등으로 사용하는 반면, SIMD기반 병렬 프로세서는 수천 개의 저비용 프로세싱 엘리먼트(processing element, PE)들을 이용하여 고성능을 추구하고 동시에 저장장소와 데이터 통신 요구를 최소

화하기 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 SIMD기반 병렬프로세서 아키텍처를 소개할 뿐만 아니라 기존의 멀티미디어 전용 명령어를 병렬프로세서 아키텍처에 추가하여 멀티미디어 처리에서 DLP 뿐만 아니라 서브워드 레벨 병렬성 (Subword-Level Parallelism, SLP)을 추구함으로써 더욱더 향상된 성능을 얻고자 한다. 동일한 SIMD기반 병렬프로세서에서 JPEG 애플리케이션을 구현하는 모의실험 결과, 인텔사의 대표적인 멀티미디어 전용 명령어인 MMX[5] 타입 명령어는 베이스라인 명령어보다 성능에서 50%, 에너지 효율에서 100%, 시스템 면적 효율에서 51%의 향상을 보였다.

본 논문의 구성은 다음과 같다. 2장에서는 관련 연구들을 소개하고, 3장에서는 MMX타입 명령어로 구현한 JPEG 압축 알고리즘의 평가방법에 대해 소개한다. 4장에서는 동일한 병렬프로세서 아키텍처에서 베이스라인 명령어와 MMX타입 명령어와의 성능을 비교 및 분석한다. 끝으로 5장에서는 이 논문의 결론을 맺는다.

II. 관련 연구

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 1은 기존의 마이크로프로세서 제조회사에서 발표한 멀티미디어 전용 명령어의 리스트를 보여준다. 이러한 멀티미디어 전용 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).

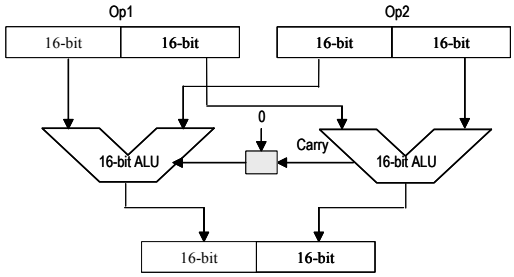

그림 1. 분할된 ALU 기능 유닛
Fig. 1. Partitioned ALU functional unit

제조회사의 타겟 애플리케이션에 따라 이러한 멀티미디어 전용 명령어는 다양하다. Motorola AltiVec[6]은 가장 많은 수의 멀티미디어 전용 명령어 (162개)를 가지고 있는 반면, HP MAX-1[7]은 단지 8개의 멀티미디어 전용 명령어를 가지고 있다. 대부분의 멀티미디어 명령어 (AMD 3DNow![8], Intel MMX, Sun VIS[9])는 64 비트 레지스터를 사용하는 반면, Motorola AltiVec[10]과 Intel SSE[11]는 128 비트 레지스터를 이용한다. 한가지 주목할 만한 예외는 MIPS MDMX[12]인데 이 명령어는 여러 번의 연산에 의한 결과를 축적하기 위해 하나의 넓은 누산기를 가지고 있다. 이러한 명령어의 유사성에도 불구하고 각각의 명령어는 독특한 특징을 가지고 있다. 예를 들어, MAX-2는 실행 유닛과 정수 레지스터를 재사용하여 추가적인 하드웨어를 필요로 하지 않는 반면, AltiVec은 전적으로 새로운 실행 유닛을 요구한다.

2. SIMD기반 프로세서 연구

멀티미디어 애플리케이션에 대한 데이터 레벨 병렬성에 관한 연구는 크게 두 개의 연구 그룹으로 나누어진다. (1) 현재의 SIMD 명령어를 이용하여 성능을 향상시키는 그룹 [13],[14],[15]과 (2) 병렬 프로세서를 이용하여 성능을 향상시키는 그룹 [2],[3],[16]. 많은 연구 그룹 혹은 개인들이 범용 마이크로프로세서에서 멀티미디어 애플리케이션에 대한 SIMD 명령어의 효율성에 대하여 분석하였다. [13]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. [14]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 [17])는 I/O 테크놀로지에 의해 제한되었다. 이후의 SIMD 병렬 프로세서인 TMC CM-2 [18]와 MasPar MP-2 [19]는 큰 버퍼 병렬 디스크 어레이의 사용을 통해 이러한 제한을 극복하였지만 비용과 이동성 (portability)을 희생하였다. Fine-grained 병렬 프로세서인 MGAP [20]와 ABACUS [21]는 이러한 이동성 이슈를 해결하였지만, 그들의 성능은 I/O 대역폭 (bandwidth)과 지연 (latency)에 의해 제한되었다.

이러한 병렬 프로세서와 다르게, 본 논문에서 모의실험을 위해 사용한 SIMD기반 병렬프로세서는[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 아키텍처에 멀티미디어 전용 명령어를 추가하여 성능 및 에너지 효율을 분석한다.

III. 평가 방법론

1. SIMD 기반 병렬 프로세서 구조

SIMD 기반 병렬 프로세서는 그림 2와 같이 2차원 격자(mesh)구조로 구성된 PE 어레이, 각 데이터의 입출력을 위한 로컬 메모리 및 각 PE와 입출력 유닛을 제어 하는 ACU(Array Control Unit)로 구성되어 있다. ACU는 프로그램 메모리에서 명령어를 패치 (fetch)하여 전체 PE에 동시에 전송 (broadcasting)하며, 또한 PE를 활성화 (active) 및 비활성화 (sleep) 시키는 역할을 한다. DEI(data exchange interface)는 외부 I/O와 PE 로컬 레지스터 간의 데이터를 전송하는 역할을 한다.

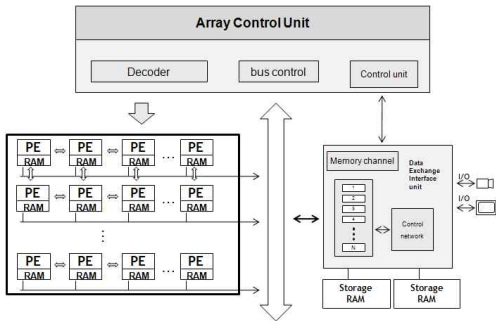

그림 2. SIMD 기반 병렬 프로세서 구조
Fig. 2. SIMD based parallel processor architecture

2. 프로세싱 엘리먼트 구조

그림 3과 같이 각 프로세싱 엘리먼트는 다음과 같은 특징을 가진 RISC(Reduced Instruction Set Computer) 아키텍처이다.

256개 32비트 워드로 구성된 로컬 메모리
16개 32비트 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
이웃하는 PE들과 데이터 통신을 위한 NEWS (North-East
-West-South) 네트워크 및 serial I/O 유닛

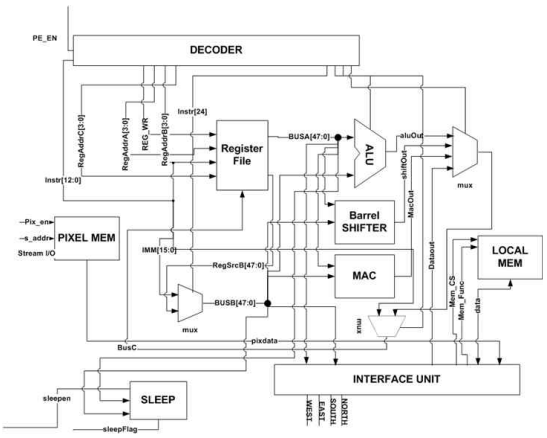

그림 3. 단일 PE의 구조
Fig. 3. Single PE architecture

3. SIMD 기반 병렬 프로세서의 파이프라이닝

그림 4과 같이 SIMD 기반 병렬 프로세서는 패치(Fetch), 디코더(Decode), 실행(Execution)의 3단계 파이프라인 구조로 설계되었다.

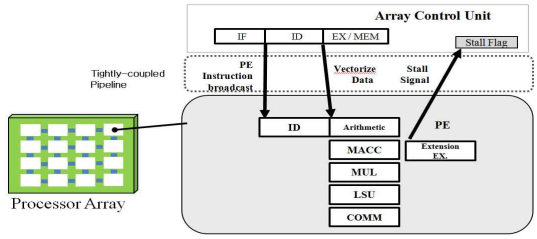

그림 4. 병렬프로세서의 파이프라이닝 단계
Fig. 4. Pipelining stage of the parallel processor

1단계에서는 ACU가 명령어 메모리로부터 명령어(instruction)을 가져온다. 2단계에서는 ACU의 디코더 유닛이 ACU에서 수행되는 스칼라(scalar)명령어인지 PE에서 수행되는 벡터(vector)명령어인지를 구분하여 BusA, BusB, BusC의 각 포트에 해당되는 레지스터 주소 및 immediate값을 할당한다. 마지막 3단계에서는 명령어가 각 유닛들의 컨트롤 시그널에 의해 실행된다.

4. SIMD 기반 병렬 프로세서의 명령어 종류

제안하는 SIMD 기반 병렬 프로세서의 명령어에는 9가지 형태의 명령어가 존재하는데 산술, 논리, 쉬프트(shift), 곱셈, 메모리 명령어, 데이터 지역성의 조건에 따라 PE를 활성화시키는 sleep 명령어, 인접 PE와 외부I/O와 통신하는 NEWS (North, East, West, South)명령어, 프로그램을 분기하는 분기 명령어, ACU의 연산을 담당하는 스칼라 명령어가 있다.

그림 5는 SIMD 기반 병렬 프로세서의 각 PE가 데이터 지역성의 정보 조건에 따라서 실행하는 모습을 보여준다. 두 사이클이 소요되는 branch와 macc(multiply accumulator) 명령어를 제외한 모든 명령어들은 하나의 사이클로 동작한다. Branch 명령어의 경우, 분기 예측이 디코더 단계에서 수행되기 때문에 2 사이클이 소요된다.

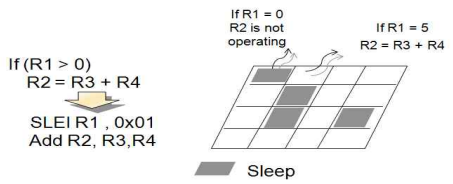

그림 5. Sleep 명령어를 사용한 PE 활성화
Fig. 5. Activation of PEs using a Sleep instruction

5. 실험 방법론 구조

그림 6은 세 가지 레벨(애플리케이션, 아키텍처, 테크놀로지)로 구성되어 있는 SIMD 기반 병렬 프로세서의 실험 방법론을 보여준다. 애플리케이션 레벨에서는 SIMD 기반 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 아키텍처 모델링 툴을 사용하였다[22]. 테크놀로지 레벨에서는 각 아키텍처 모델들의 테크놀로지 변수(latency, power, clock frequency)를 계산하기 위해 Generic System Simulator (GENESYS)를 사용하였다[23]. 마지막으로 세 레벨에서 구해진 데이터베이스를 조합하여 각 경우에 대한 실행시간, 처리량, 에너지 효율을 결정하였다.

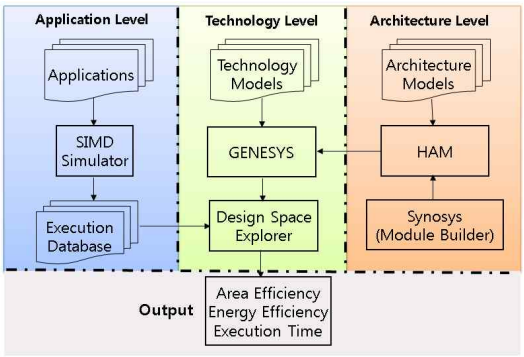

그림 6. SIMD 기반 병렬 프로세서를 위한 실험 방법론
Fig. 6. Methodology infrastructure for the SIMD parallel processor simulation

IV. 실험 및 성능 분석 결과

1. SIMD 기반 병렬 프로세서 성능 평가 지표

MMX타입 및 베이스라인 병렬 프로세서 아키텍처를 이용하여 JPEG 알고리즘을 구현하고 성능, 에너지 효율 및 면적 효율을 결정하기 위해 정밀 사이클 (cycle-accurate) 시뮬레이션과 테크놀로지 모델링을 사용하였다. JPEG 압축 알고리즘은 병렬 프로세서용 어셈블리 언어를 사용하여 구현하였으며, 공정한 평가를 위해 MMX 기반 프로그램과 베이스라인 프로그램은 같은 변수, 데이터 셋 및 콜링 시퀀스로 구현되었다. 표 2는 각 경우의 성능 비교를 위한 4가지 지표(execution time, sustained throughput, energy efficiency, area efficiency)를 보여 준다.

표 2. 성능 평가 지표 요약
Table 2. Summary of performance evaluation metrics

Execution time	$t_{exec} = \frac{C}{f_{ck}}$
Sustained throughput	$Th_{sust} = \frac{O_{exec} \cdot U \cdot N_{PE}}{t_{exec}}$
Energy efficiency	$\eta_E = \frac{1}{t_{exec} \cdot Energy}[\frac{1}{Joule}]$
Area efficiency	$\eta_A = \frac{1}{t_{exec} \cdot Area}[\frac{1}{s \cdot mm^2}]$
C : 사이클 개수, f_{ck} : 클럭 주파수, O_{exe} : 수행된 연산 개수, U: 프로세싱 엘리먼트 이용률, N_{PE} : 프로세싱 엘리먼트의 개수	

실행 시간 (execution time)은 JPEG 압축 알고리즘이 수행된 시간을, 처리량 (sustained throughput)은 단위 시간당 처리되는 명령어 개수 (Giga-operations/second)를, 에너지 효율 (energy efficiency)은 단위 에너지당 소비된 명령어 개수 (Giga-operations/Joule)를 나타내고, 시스템 면적 효율 (area efficiency)은 단위 시스템 면적당 소비된 명령어 개수를 나타낸다.

표 3은 본 논문에서 사용한 병렬 프로세서 아키텍처 모델의 파라미터를 보여준다. 성능분석을 위해 기 개발한 SIMD 기반 병렬 프로세서용 정밀 사이클 시뮬레이터를 사용하였다. 효율적인 영상처리를 위해 16개의 프로세싱 엘리먼트를 메쉬 구조로 연결하였으며, 각각의 프로세싱 엘리먼트는 자신에게 맵핑된 영상의 지역데이터를 처리한다. 각 프로세싱 엘리먼트는 32비트 워드 단위의 31950개의 메모리를 가지고 있으며, 130nm 테크놀로지와 100MHz 클럭 주파수를 사용하여 시뮬레이션 하였다.

표 3. SIMD 기반 병렬 프로세서 파라미터
Table 3. SIMD based parallel processor parameters

Parameter	Value
System Size (# of PEs)	16
Image Pixels per PE	81,920 (1280x1024)
VLSI Technology	130 nm
Clock Frequency	100 MHz
Interconnection Network	Mesh
intALU/intMUL/Barrel Shifter/intMACC/Comm	1 / 1 / 1 / 1 / 1
Memory/PE[words]	31,950
System Memory[words]	511,181

2. MMX를 이용한 JPEG 압축 알고리즘 구현

SIMD 기반 병렬 프로세서에서 MMX타입 명령어의 성능을 평가하기 위해 정지영상압축의 대표적 표준 방식인 JPEG[24] 압축 알고리즘을 구현하였다. 본 장에서는 JPEG 알고리즘의 간략한 소개와 MMX타입 명령어를 사용한 JPEG 알고리즘의 구현에 대해 설명한다. 그림 7은 JPEG 알고리즘의 전반적인 흐름도를 보여준다. 이미지 입력 데이터를 8x8의 매크로 블록으로 분할하는 전처리 과정, RGB 영역에서 YCbCr 칼라 영역으로의 변환 과정, 이산 코사인 변환 과정, 양자화 과정 그리고 마지막으로 엔트로피 부호화 과정을 거쳐 압축된 영상 데이터를 얻는다. 압축된 영상 데이터의 복원은 압축과정의 역 수행으로 가능하다.

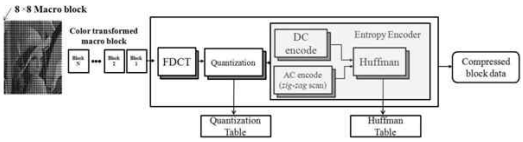

그림 7. JPEG 알고리즘의 블록 다이어그램
Fig. 7. A block diagram of the JPEG algorithm

2.1 이산 코사인 변환

입력 영상은 8x8 블록으로 분할되고, 이산 코사인 변환 (discrete cosine transform, DCT)을 통해 주파수 영역으로 변환된다. 본 논문에서는 일반적인 이산 코사인 변환 대신, 보다 빠른 처리를 위해 널리 사용되고 있는 Chen이 제안한 이산 코사인 변환[25]을 이용하였다. 그림 8은 베이스라인 명령어와 MMX타입 명령어를 사용하여 1차원 열방식의 이산 코사인 변환 계수를 구하는 과정을 보여준다. 기존 베이스라인 명령어는 매크로 블록 연산에서 하나의 데이터를 레지스터에 저장하고 처리하는 반면, MMX타입은 PACKSSDW 패킹 명령어를 사용하여 2개의 데이터를 하나의 레지스터에 패킹하고 동시에 매크로 블록 연산을 수행함으로써 성능을 향상시킨다.

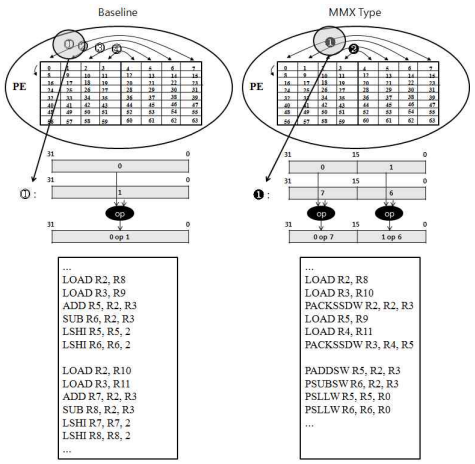

그림 8. MMX를 이용한 이산 코사인 변환의 예
Fig. 8. An example of DCT using MMX

2.2 양자화

양자화 (quantization) 과정은 JPEG 알고리즘에서 매우 중요한 커널 중 하나이다. 주파수로 변환된 값은 매우 다양한 범위를 가지므로 양자화 과정을 거쳐 부호화 하는 것이 효율적이다. 양자화 과정은 미리 정해진 휘도(Y)성분과 색차신호(Cb,Cr)의 양자화 테이블을 이용하여 구현되었다[26]. 그림 9는 베이스라인 명령어와 MMX타입 명령어를 사용하여 매크로 블록 데이터의 양자화 과정을 구현한 예를 보여준다. 이산 코사인 변환과 마찬가지로, MMX는 PACKSSDW 패킹 명령어를 사용하여 휘도 양자화 테이블 데이터 2개와 맵핑되는 매크로 블록 데이터 2개를 각각의 레지스터에 패킹하여 양자화 연산을 병렬 수행함으로써 성능을 향상시킨다.

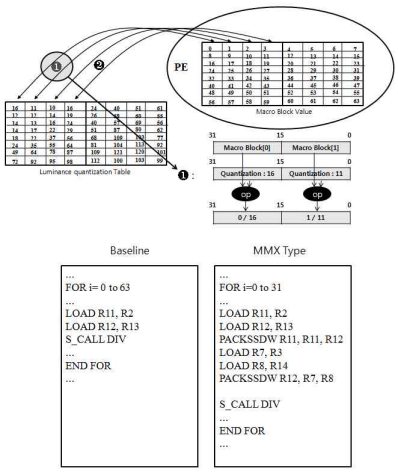

그림 9. MMX를 이용한 양자화의 예
Fig. 9. An example of quantization using MMX

2.3 엔트로피 부호화

JPEG 부호화의 마지막 단계인 엔트로피 부호화(entropy encoding)는 무손실 데이터의 압축 과정으로서 데이터의 실질적인 압축이 일어난다. 엔트로피 부호화는 크게 2가지 과정으로 구성된다. 주파수로 변환된 값을 재조정하는 zig-zag 스캔 과정과 동적인 길이의 압축 과정으로 나뉜다. zig-zag스캔을 거친 1차원 선형구조의 배열 값을 허프만 코딩방식을 이용하여 배열의 첫 번째 값인 DC계수의 차분 부호화와 AC계수 부호화로 나누어 진행한다. 부호화 방식은 테이블을 사용하여 필요한 비트수와 값을 정의한다. 그림 10은 DC 차분 부호화를 병렬 프로세서를 이용하여 구현한 예를 보여준다. 매크로 블록들이 각 PE에 할당 되고, DC값의 차분을 위해 이전 매크로 블록의 DC값이 참조 된다. 따라서 DC 값을 오른쪽에 이웃된 PE에 전송되는 것을 그림 10(a)에서 보여준다. 그림 10(b)는 맨 가장자리에 위치한 PE의 값들을 다음 행의 PE에 전달하기위해 첫 번째 행들만 아래쪽 PE와 통신 하는 예를 보여준다. 그림 10(c)에서는 첫 번째 PE의 값이 최초의 매크로 블록 값이므로 비활성화 시킨 다음 나머지 PE들은 이웃되는 DC계수를 저장하는 예를 보여준다. 그림 11은 베이스라인 명령어와 MMX타입 명령어를 사용하여 매크로 블록 데이터의 엔트로피 부호화 과정을 보여준다. MMX는 PACKSSDW와 PACKSSWB 패킹 명령어를 사용하여 매크로 블록 데이터 4개를 각각의 레지스터에 패킹하여 엔트로피 부호화 연산을 병렬 수행함으로써 성능을 향상시킨다.

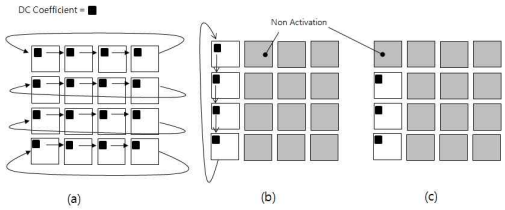

그림 10. 이웃 PE들 간의 DC계수 참조
Fig. 10. DC coefficients reference among PEs

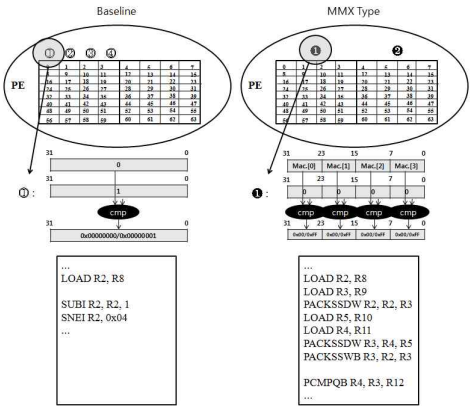

그림 11. MMX를 이용한 엔트로피 부호화의 예
Fig. 11. An example of entropy encoding using MMX

3. 성능 평가 결과 및 분석

본 장에서는 SIMD 기반 병렬 프로세서에서 MMX타입 명령어를 이용하여 JPEG 압축 알고리즘을 구현하고 성능 분석한 결과를 보여준다. MMX 타입 명령어는 병렬 프로세서의

표 4. Baseline 프로그램과 MMX기반 프로그램의 성능 비교
Table 4. Performance comparison between baseline and MMX based programs

		Total Cycle [cycles]	Vector Instruction [cycles]	Scalar Instruction [cycles]	System Utilization [%]	Sustained Throughput [Gops/sec]	Execution Time [ms]
FDCT	Baseline	5,658,117	4,638,996	1,019,121	0.9590	1.2581	56.58
	MMX-Type	5,345,500	4,304,048	1,041,452	0.9669	1.2455	53.46
Quantization	Baseline	14,840,904	11,427,104	3,413,800	0.7305	0.8999	148.41
	MMX-Type	7,534,696	5,289,024	2,245,672	0.7706	0.8655	75.35
Entropy Encoding	Baseline	6,722,144	4,617,080	2,105,064	0.5917	0.6502	67.22
	MMX-Type	5,139,603	3,185,579	1,954,024	0.7872	0.7806	51.40
JPEG	Baseline	27,221,165	20,683,180	6,537,985	0.7508	0.9127	272.21
	MMX-Type	18,019,799	12,778,651	5,241,148	0.8409	0.9541	180.20

정수 레지스터와 실행 유닛을 재사용함으로써 하드웨어 비용의 증가 없이 성능을 개선시킨다.

3.1 성능 평가 결과

그림 12는 SIMD 기반 병렬 프로세서에서 베이스라인 명령어와 MMX타입 명령어를 사용하여 JPEG 및 3가지 커널을 구현하고 성능을 비교한 결과를 보여준다. MMX타입 명령어는 베이스라인 명령어보다 약 50%의 성능 향상을 보였다. 또한 MMX기반 프로그램은 평균 0.95 Gops/sec의 처리량을 보였다. 표 4는 16 PE 어레이 시스템에서 베이스라인과 MMX기반 프로그램의 전체 성능을 보여준다.

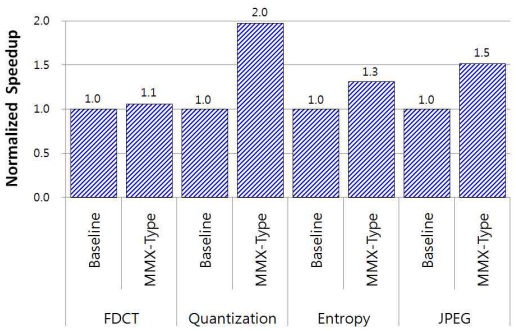

그림 12. 베이스라인 프로그램과 MMX기반 프로그램의 성능 비교
Fig. 12. Performance comparison between baseline and MMX based programs

3.2 MMX타입 명령어의 이점

그림 13은 병렬 프로세서에서 수행된 베이스라인 프로그램 대비 MMX기반 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), MMX타입 명령어로 세분화 된다. 그림에서 보는 바와 같이, MMX기반 프로그램은 베이스라인 프로그램 보다 약 32%의 명령어 개수를 감소시켰다.

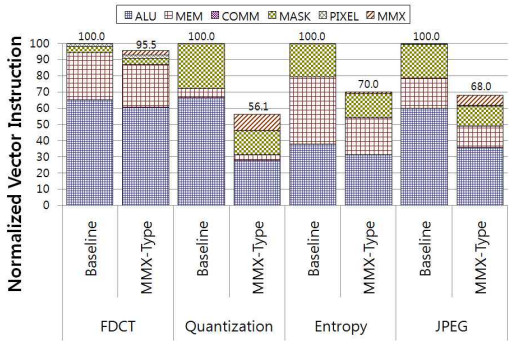

그림 13. 베이스라인 프로그램과 MMX기반 프로그램의 벡터 명령어 비교
Fig. 13. Vector instruction comparison between baseline and MMX based programs

3.3 시스템 에너지 효율 비교 결과

그림 14는 병렬 프로세서에서 베이스라인 프로그램 대비 MMX기반 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 (100MHz), 공정 (130nm) 및 프로세서 파라미터에서 프로그램의 수행시간은 에너지 소비

에 비례한다[27]. 따라서, MMX타입 명령어를 이용하여 많은 양의 명령어 및 사이클을 줄였으므로 상당한 양의 소비 에너지를 감소시킬 수 있다. MMX기반 프로그램은 베이스라인 프로그램보다 약 13%의 소비 에너지를 감소시켰다.

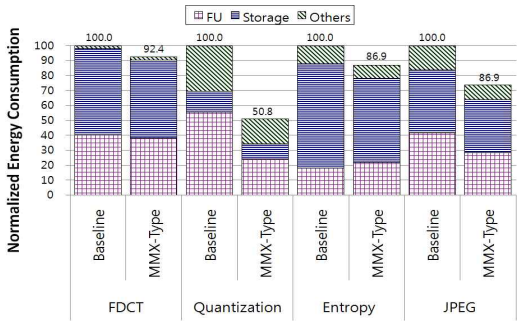

그림 14. 베이스라인 프로그램과 MMX기반 프로그램의 소비 에너지 비교

Fig. 14. Energy consumption comparison between baseline and MMX based programs

그림 15는 베이스라인 프로그램 대비 MMX기반 프로그램의 에너지 효율을 보여준다. MMX기반 프로그램은 베이스라인 프로그램보다 약 100%의 에너지 효율을 증가 시켰다. 이러한 결과는 MMX타입 명령어가 적은 시스템 전력을 증가시키는 반면, 높은 처리량을 제공하기 때문이다. 에너지 효율의 증가는 시스템 배터리 수명을 증가시키는 결과를 가져온다.

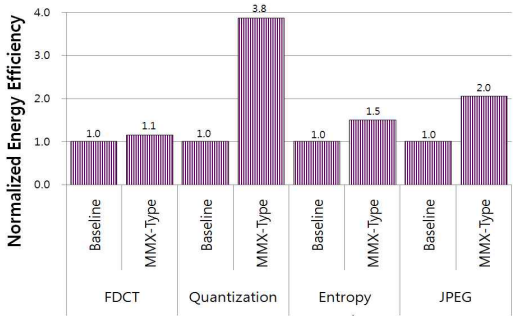

그림 15. 베이스라인 프로그램과 MMX기반 프로그램의 에너지 효율 비교

Fig. 15. Energy efficiency comparison between baseline and MMX based programs

3.4 시스템 면적 효율 비교 결과

그림 16은 베이스라인 프로그램 대비 MMX기반 프로그램의 면적 효율을 보여준다. MMX기반 프로그램은 베이스라인 프로그램보다 약 51%의 면적 효율을 증가 시켰다. 면적 효율의 증가는 주어진 시스템에서 구성요소의 효율을 증가시키는 결과를 가져온다.

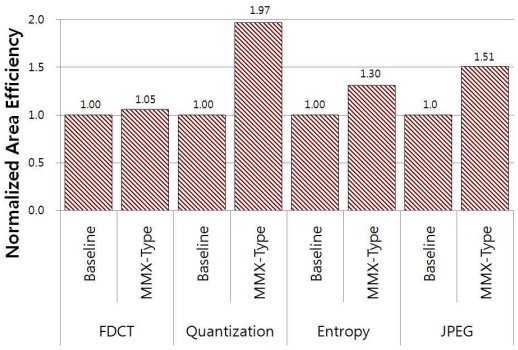

그림 16. 베이스라인 프로그램과 MMX기반 프로그램의 면적 효율 비교

Fig. 16. Area efficiency comparison between baseline and MMX based programs

V. 결 론

본 논문에서는 멀티미디어에 내재한 무수한 데이터 병렬성을 효과적으로 처리할 수 있는 SIMD 기반 병렬 프로세서를 소개하였고, 인텔사의 대표적인 멀티미디어 전용 명령어인 MMX 타입 명령어를 추가하여 멀티미디어 성능 향상을 꾀하였다. 동일한 병렬프로세서를 사용하여 모의 실험한 결과, MMX타입 명령어를 사용하여 구현한 JPEG 애플리케이션은 베이스라인 명령어를 사용하여 구현한 동일한 애플리케이션보다 실행 시간에서 약 50%, 에너지 효율에서 100%, 시스템 면적 효율에서 51%의 성능 향상을 보였다. 이러한 결과는 멀티미디어 전용 명령어들이 현재 널리 사용되고 있는 매니코어 GPU (Graphics Processing Unit) 및 다양한 형태의 병렬프로세서에서도 잠재 가능성이 있음을 보여준다.

참고문헌

[1] M. K. Chung, S. M. Park, N. W. Eum, "Technology and trend of parallel processor," Electronics and Telecommunications Research Institute Trend Analysis, vol. 24, no. 6, Dec. 2009.

[2] A.D. Blas et. al., "The UCSC Kestrel Parallel Process or," 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 Supercomp uting,” IEEE Trans. on Computers, vol. 53, no. 8, pp. 960-973, Aug. 2004.

[4] L. V. Huynh, C.-H. Kim, J.-M. Kim, "A massively parallel algorithm for fuzzy vector quantization," Journal of Korea Information Processing Society, Vol. 16-A, No. 6, pp. 411-418, Dec. 2009.

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

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

[8] S. Oberman, G. Favor, F. Weber, "AMD 3DNow! technology: architecture and implementations," IEE E Micro, vol. 19, no. 2, pp. 37-48, Mar/Apr. 1999.

[9] M. Tremblay, J. M. O'Connor, V. Narayanan, and L. He, “VIS Speeds New Media Processing,”IEEE Micro, vol. 16, no. 4, pp. 10-20, Aug. 1996.

[10] J. Tyler, J. Lent, A. Mather, N. Huy, "AltiVec: bri ng vector technology to the PowerPC processor family," in IEEE International Performance, Com puting, and Communications Conference, p. 437, Feb. 1999.

[11] S. K. Raman, V. Pentkovski, and J.Keshava, “Imple menting streaming SIMD extensions on the pentium III processor,” IEEE Micro, vol. 20, no. 4, pp.28-39, 2000.

[12] MIPS extension for digital media with 3D. Technical Report: http://www.mips.com, MIPS technologies, Inc., 1997.

[13] P. Ranganathan, S. Adve, and N. P. Jouppi, “Perfor mance of image and video processing with genera l-purpose processors and media ISA extensions,” in Proc. of the 26th Intl. Sym. on Computer Architec ture, pp. 124-135, May 1999.

[14] R. Bhargava, L. John, B. Evans, and R. Radhakr ishnan, “Evaluating MMX technology using DSP and multimedia applications,” in Proc. of IEEE/AC M Sym. on Microarchitecture, pp. 37-46, 1998.

[15] N. Slingerland, and A. J. Smith, “Measuring the performance of multimedia instruction sets,” IEEE Trans. on Computers, vol. 51, no. 11, pp. 1317-1332 , Nov. 2002.

[16] 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.

[17] L. W. Tucker, and G. G. Robertson, “Architecture and applications of the connection machine,” IEEE Computer, vol. 21, no. 8, pp. 26-38, 1988.

[18] “Connection machine model CM-2 technical summ ary,” Thinking Machines Corp., version 51, May 1989.

[19] MarPar (MP-2) System Data Sheet. MarPar Corp oration, 1993.

[20] M. J. Irwin, R. M. Owens, “A Two-Dimensional, Distributed Logic Processor,” IEEE Trans. on Computers, vol. 40, no. 10, pp. 1094-1101, 1991.

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

[22] 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.

[23] 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.

[24] Wallace, G.K., “The JPEG still picture compression standard,” IEEE Transactions on Consumer Electronics, vol 38. no 1, pp. 18 - 33 , Feb 1992.

[25] W. H. Chen, C. Smith, S. Fralick, A fast comput

ational algorithm for the discrete cosine transform, IEEE Trans. Commun. 25 (9) (2002), pp. 1004-1009.

[26] Long-Wen Chang, Ching-Yang Wang, Shiuh-Ming Lee, "Designing JPEG quantization tables based on human visual system," ICIP 99, vol. 2, pp. 376-380, 1999

[27] 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.

저 자 소 개

정 용 범

2009 : 울산대학교 정보통신공학사.
2009 : 울산대학교 컴퓨터정보통신공학부 석사과정 입학.
관심분야 : 임베디드 SoC, 컴퓨터 구조, 병렬처리
Email: smartnow@nate.com

김 용 민

2009 : 울산대학교 컴퓨터공학사.
2009 : 울산대학교 컴퓨터정보통신공학부 석사과정 입학.
관심분야 : 임베디드시스템, 컴퓨터구조, 병렬처리
Email: jafstar@nate.com

김 철 홍

1998 : 서울대학교 컴퓨터공학사.
2000 : 서울대학교 컴퓨터공학부 석사.
2006 : 서울대학교 전기컴퓨터공학부 박사
2005 - 2007년 :삼성전자 반도체총괄 책임연구원
2007 - 현재 : 전남대학교 전자컴퓨터공학부 교수
관심분야 : 임베디드시스템, 컴퓨터구조, SoC설계, 저전력 설계
Email: cheolhong@gmail.com

김 종 면

1995 : 명지대학교 전기공학사
2000 : University of Florida ECE 석사
2005 : Georgia Institute of Technology ECE 박사
2005 - 2007 : 삼성종합기술원 전문연구원
2007 - 현재 : 울산대학교 컴퓨터정보통신공학부 교수
관심분야 : 프로세서 설계, 임베디드 SoC, 컴퓨터구조, 병렬처리
Email: jongmyon.kim@gmail.com