韓國컴퓨터情報學會 論文誌
第16卷 第1號, 2011. 1.

2011-16-1-1-1

효율적인 영상데이터 처리를 위한 SIMD기반 매니코어 프로세서 구현

최 병 국*, 김 철 홍**, 김 종 면***

Implementation of SIMD-based Many-Core Processor for Efficient Image Data Processing

Byong-Kook Choi*, Cheol-Hong Kim**, Jong-Myon Kim***

요 약

최근 모바일 멀티미디어 기기들의 사용이 증가하면서 고성능, 저전력 멀티미디어 프로세서에 대한 필요성이 높아지고 있는 추세이다. 주문형반도체 (ASIC)는 모바일 멀티미디어에서 요구되는 고성능을 만족시키지만 다양한 형태의 멀티미디어 애플리케이션에서 요구되는 범용성을 만족시키지 못한다. 반면 DSP기반의 시스템은 범용성에 기인하여 다양한 형태의 애플리케이션에서 사용될 수 있으나, 주문형반도체 보다 높은 가격, 전력소모 및 낮은 성능을 가진다. 이러한 문제점을 해결하기 위해 본 논문에서는 범용성을 유지하면서 고성능, 저전력으로 영상데이터 처리가 가능한 단일 명령어 다중 데이터(Single Instruction Multiple Data, SIMD)처리 방식의 매니코어 프로세서를 제안한다. 제안한 SIMD기반 매니코어 프로세서는 16개의 프로세싱 엘리먼트(processing element, PE)로 구성되어 영상데이터 처리에 내재한 무수한 데이터 레벨 병렬성을 높인다. 모의 실험한 결과, 제안한 SIMD기반 매니코어 프로세서는 현재 상용 고성능 프로세서보다 평균 22배의 성능, 7배의 에너지 효율 및 3배의 시스템 면적 효율을 보였다.

▸ Keyword : 매니코어 프로세서, 이미지/비디오 처리, 데이터 레벨 병렬성

Abstract

Recently, as mobile multimedia devices are used more and more, the needs for high-performance and low-energy multimedia processors are increasing. Application-specific integrated circuits (ASIC) can meet the needed high performance for mobile multimedia, but they provide limited, if any, generality needed for various application requirements. DSP based systems can used for various types of applications due to their generality, but they require higher cost and energy consumption as well as less performance than ASICs. To solve this

• 제1저자 : 최병국 교신저자 : 김종면
• 투고일 : 2010. 08. 24, 심사일 : 2010. 09. 16, 게재확정일 : 2010. 10. 29.
* 울산대학교 전기공학부 (School of Electrical Engineering, University of Ulsan) 석사과정
** 전남대학교 전자컴퓨터공학과 (Chonnam National University,) 교수
*** 울산대학교 전기공학부 (School of Electronics and Computer Engineering, University of Ulsan) 교수
※ 이 논문은 2010년도 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. 2010-0010863).

problem, this paper proposes a single instruction multiple data (SIMD) based many-core processor which supports high-performance and low-power image data processing while keeping generality. The proposed SIMD based many-core processor composed of 16 processing elements (PEs) exploits large data parallelism inherent in image data processing. Experimental results indicate that the proposed SIMD-based many-core processor higher performance (22 times better), energy efficiency (7 times better), and area efficiency (3 times better) than conversional commercial high-performance processors.

▸ Keyword : Many-core processor, image/video processing, data level parallelism

Ⅰ. 서 론

최근 모바일 멀티미디어 기기들의 사용이 증가함에 따라 멀티미디어의 방대한 데이터를 얼마나 낮은 전력과 고성능으로 처리하는가 하는 문제가 크게 대두 되고 있다[1].

기존의 ASIC(Application-Specific Integrated Circuit)은 이러한 모바일 멀티미디어에서 요구되는 고성능, 저전력을 만족 시킬 수 있지만 다양한 형태의 멀티미디어 애플리케이션에서 요구되는 범용성을 만족시키지 못한다[2][3][4].

반면에 범용 마이크로프로세서 (General-Purpose Processor, GPP)나 DSP (Digital Signal Processor)들은 다양한 애플리케이션에 대해 충분한 범용성을 제공한다. 하지만, 멀티미디어 애플리케이션에서 요구되는 높은 레벨의 성능을 만족시키지 못한다. 왜냐하면 GPP나 DSP는 프로세서 구조의 특성상 멀티미디어에 내재한 고도 병렬성 (massive parallelism)을 활용하지 못하기 때문이다.

고성능 멀티미디어 처리를 위한 대안 중에 하나로 SIMD (Single Instruction Multiple Data)기반 병렬 프로세서 아키텍처가 유망하다[5][6]. 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 여러 개의 저비용 프로세싱 엘리먼트 (processing element, PE)들을 이용하여 고성능을 추구하고 동시에 저장장소와 데이터 통신 요구를 최소화하기 위해 프로세싱 엘리먼트와 데이터 입출력을 동일위치에 배치함으로써 저전력을 만족시킨다[7]. 특히, SIMD기반 병렬 프로세서는 지역성(locality)이나 규칙성(regularity)이 있는 2차원 패턴의 이미지나 비디오 픽셀 처리에 있어서 최적의 프로세서 구조이다.

본 논문에서는 모바일 영상데이터 처리를 위한 저전력, 고성능 SIMD기반 매니코어 프로세서를 제안한다. 제안한 SIMD기반 매니코어 프로세서는 16개의 프로세싱 엘리먼트로 구성되어 있으며, 각각의 프로세싱 엘리먼트는 자신에게 맵핑된 영상의 지역데이터를 처리함으로써 데이터 레벨 병렬성을 높인다. (예를 들어, 사이즈가 256x256 픽셀인 입력 이미지는 16 PE 아키텍처의 각 PE 메모리에 64x64 픽셀씩 균등하게 할당되고 하나의 명령어에 의해 각 PE에 있는 데이터가 동시에 처리됨) 현재 상용화되고 있는 고성능 프로세서 (C6416[8], ARM926EJ-S[21], ARM1020E[22])와 비교하여 평균 22배의 성능, 7배의 에너지 효율 및 3배의 시스템 면적 효율을 보였다.

본 논문의 구성은 다음과 같다. 2장에서는 제안한 SIMD기반 매니코어 프로세서의 관련 연구에 대해 소개하고, 3장에서는 성능 평가를 위해 선택된 영상처리 애플리케이션을 소개한다. 4장에서는 제안한 매니코어 프로세서 모델 및 실험 방법론을 소개하고, 5장에서는 시뮬레이션 결과와 성능 분석에 대해 설명한다. 끝으로 6장에서는 이 논문의 결론을 맺는다.

Ⅱ. 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 [9],[10],[11]과 (2) SIMD기반 병렬 프로세서를 이용하여 성능을 향상시키는 그룹 [6],[12]. 많은 연구 그룹 혹은 개인들이 범용 마이크로프로세서에서 멀티미디어 애플리케이션에 대한 SIMD 명령어의 효율성에 대하여 분석하였다. [9]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. [10]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 못할 것이다.

SIMD기반 병렬 프로세서는 공간적 병렬성(spatial parallelism)을 실현하기 위해 여러 개의 동기화된 프로세싱 유닛(processing unit)들을 사용한다. 이 유닛들은 하나의 제어 유닛으로부터 동시에 전송되는 동일한 연산 명령을 서로 다른 데이터에 대하여 수행한다. 따라서 데이터 병렬 모델을 이용하여 성능을 향상시킨다. 고도 데이터 병렬 어레이 (massively data parallel array)들은 거의 30년 동안 이미지 처리에 사용되어 왔지만, 초기의 SIMD기반 병렬 프로세서 (TMC Connection Machine 1[13])는 I/O 테크놀로지에 의해 제한되었다. 이후의 SIMD 병렬 프로세서인 TMC CM-2[14]와 MasPar MP-2[15]는 버퍼 이미지의 큰 병렬 디스크 어레이의 사용을 통해 이러한 제한을 극복하였지만 큰 비용과 휴대성에서 문제가 있다. Fine-grained 병렬 프로세서인 MGAP[16]와 ABACUS[17]는 이러한 휴대성 이슈를 해결하였지만, 그들의 성능은 I/O bandwidth와 latency에 의해 제한되었다.

이러한 기존의 병렬 프로세서와 다르게, 본 논문에서 모의실험을 위해 사용한 SIMD기반 매니코어 프로세서는 프로세서와 센서의 직접적 연결을 통해 I/O 대역의 문제를 해결하고, 또한 짧은 와이어의 사용으로 높은 면적과 에너지 효율을 보이는 동시에 많은 데이터에 동일한 명령어를 수행하여 고성능을 추구한다.

III. 영상처리 알고리즘

제안한 SIMD기반 매니코어 프로세서 아키텍처의 성능을 분석하기위해 다섯 가지의 영상처리 알고리즘을 선택하고 구현하였다. 영상의 기하학적 변환을 표현하기 위한 Translation Transform, 영상의 그레이 레벨 연산을 위한 Subtraction과 Mask Scaling, 영상의 분할을 표현하기 위한 Histogram Segmentation, 마지막으로 영상의 모폴로지 표현을 하기 위한 Edge Detection을 매니코어 프로세서용 시뮬레이터를 이용하여 구현하였다.

Translation Transform은 영상을 특정 크기만큼 가로 또는 세로 방향으로 이동시키는 변환을 의미한다. 입력 영상의 특정 좌표(x, y)를 가로로 a만큼, 세로로 b만큼 이동시키는 이동 변환을 식 (1)과 같이 표현할 수 있다.

$$\begin{vmatrix} x' \\ y' \end{vmatrix} = \begin{vmatrix} x \\ y \end{vmatrix} + \begin{vmatrix} a \\ b \end{vmatrix} \quad \cdots\cdots (1)$$

Subtraction은 하나의 영상에서 다른 영상을 빼는 연산을 의미하며, 식 (2)와 같이 표현할 수 있다.

$$h(x,y) = f(x,y) - g(x,y) \quad \cdots\cdots (2)$$

Mask Scaling은 현재 영상과 Mask 영상을 이용하여 그레이 스케일 뺄셈 연산을 수행 후 적용된 해당 영역의 영상 크기를 변환 시키는 작업을 수행한다. Mask 영상의 성질에 따라 영상의 적용되는 영역이 선택되며, 적용된 영역의 크기가 확대 또는 축소되는 변환이다. 영상의 크기를 가로 방향 s_x, 세로 방향 s_y 배로 변환 시키는 방법은 식 (3)과 같이 표현할 수 있다.

$$\begin{vmatrix} x' \\ y' \end{vmatrix} = \begin{vmatrix} s_x & 0 \\ 0 & s_y \end{vmatrix} \begin{vmatrix} x \\ y \end{vmatrix} \quad \cdots\cdots (3)$$

Histogram Segmentation은 Histogram 표현과 Otsu 알고리즘을 이용하여 특정 점을 기준으로 영상의 분할을 표현하였다. 입력 영상의 Histogram을 구한 후 Otsu 알고리즘을 적용하여 선택된 임의의 특정점 다수를 기준으로 영상의 각 영역을 분할하는 방법이다. Otsu 알고리즘은 임계값을 설정하는데 있어서 비용함수를 설정하고 그 비용함수의 최소 값을 주어 임계값을 표현하는 방법으로 Classification에 이용되고 있다.

Edge Detection은 영상의 경계선 정보를 찾아내는 방법을 의미한다. 영상의 경계선에서는 그레이 스케일 값이 급격하게 변화하기 때문에 영상의 미분 함수를 구하여 그 값이 크게 나타나는 위치를 찾으면 경계선 위치를 검출 할 수 있다.

IV. 매니코어 프로세서 모델 및 실험방법론

4.1 SIMD기반 매니코어 프로세서 모델

그림 1은 SIMD기반 매니코어 프로세서 아키텍처의 블록 다이어그램을 보여준다. 제안한 매니코어 프로세서는 16개의 프로세싱 엘리먼트와 이를 제어하는 Array Control Unit (ACU)으로 구성되어 있고, 데이터가 각각의 프로세싱 엘리먼트에 일정하게 분배되면 프로세싱 엘리먼트들은 메쉬 배열 구조에서 명령어들을 수행한다. 각 프로세싱 엘리먼트는 다음과 같은 특징을 가진다.

- 32비트 폭의 4096개 워드로 구성된 로컬 메모리
- 32비트 폭의 16개 3포트 범용 레지스터
- 기본적인 산술/논리 연산을 수행하는 ALU
- 64비트 곱셈 및 누산기 (multiply accumulator)
- 멀티 비트 산술/논리 시프트 연산을 수행하는 배럴 시프트 (Barrel Shifter)
- 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/O유닛

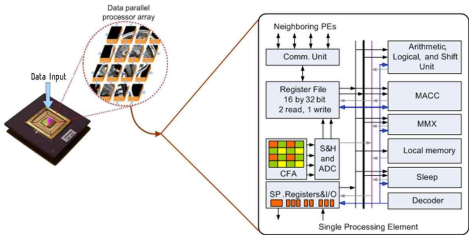


그림 1. SIMD기반 매니코어 프로세서 아키텍처와 싱글 프로세싱 엘리먼트
Fig. 1. A block diagram of SIMD based many-core processor Processor architecture and single PE

4.2 매니코어 프로세서의 파이프라이닝

그림 2와 같이 SIMD기반 매니코어 프로세서는 패치(Fetch), 디코더(Decode), 실행(Execution)의 3단계 파이프라인 구조로 설계되었다. 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.

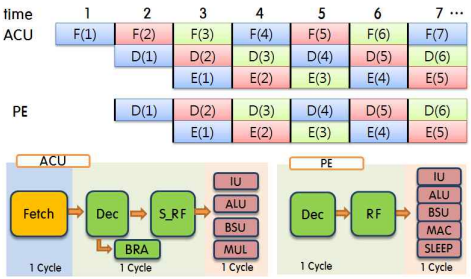


그림 2. ACU와 PE의 파이프라인 단계
Fig. 2. Pipeline stage of ACU and PE

4.3 매니코어 프로세서 명령어 종류

제안하는 SIMD기반 매니코어 프로세서의 명령어 종류에는 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 3은 SIMD기반 매니코어 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.

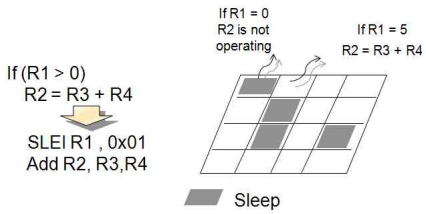


그림 3. Sleep 명령어를 사용한 PE 활성화
Fig. 3. Activation of PE using a Sleep instruction

4.4 실험 방법론 구조

그림 4는 세 가지 레벨 (애플리케이션, 아키텍처, 테크놀로지)로 구성되어 있는 SIMD기반 매니코어 프로세서의 실험 방법론이다. 애플리케이션 레벨에서는 명령어 레벨의 SIMD 병렬 프로세서용 정밀 사이클 시뮬레이터를 이용하여, 영상처리 알고리즘에 사용되는 사이클 개수, 동적 명령어 빈도, 프로세싱 엘리먼트 이용률 (utilization) 등의 실행 데이터를 제공한다. 아키텍처 레벨에서는 모델링된 아키텍처의 디자인 변수들을 계산하기 위해 Chai가 제안한 SIMD 병렬 프로세서용 이종 아키텍처 모델링 툴을 사용하였다[18]. 테크놀로지 레벨에서는 각 아키텍처 모델들의 테크놀로지 변수 (latency, power, clock frequency)를 계산하기 위해 Generic System Simulator (GENESYS)를 사용하였다[19]. 마지막으로 세 레벨에서 구해진 데이터베이스를 조합하여 각 경우에 대한 실행 시간, 처리량, 에너지 효율을 결정하였다.

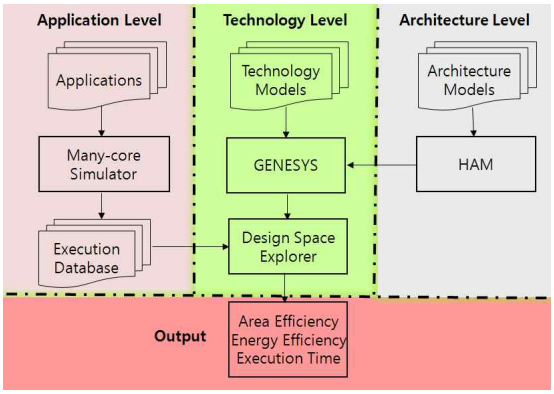


그림 4. 매니코어 프로세서 시뮬레이션을 위한 실험 방법론
Fig. 4. Experiment methodology for many-core processor simulation

V. 모의실험 및 성능 분석

5.1 영상처리 알고리즘 결과

그림 5는 입력으로 사용한 MRI 영상이고, 그림 6은 선택한 다섯 가지 영상처리 알고리즘을 SIMD기반 매니코어 프로세서를 이용해 구현한 결과 영상들을 보여준다.

Translation의 경우 x축, y축으로 각각 20픽셀씩 이동한 경우이고, Subtraction의 경우 현재의 영상과 이전의 영상을 비교하여 그 차를 출력한 결과이다. Mask Scaling은 마스크를 사용하여 특정 부분만 추출 후 가로 축 그림 크기를 2배로 늘린 출력 결과이고, Histogram Segmentation은 2가지의 임계값을 기준으로 분할한 출력 결과이며, Edge Detection은 소벨(Sobel)마스크를 이용한 출력 결과이다.

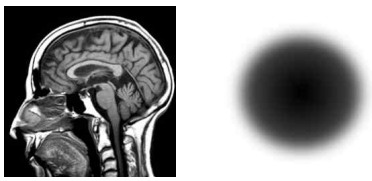
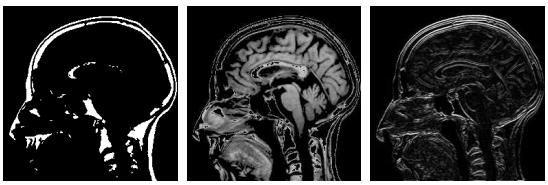


그림 5. 입력 MRI 영상과 마스크 영상
Fig. 5. Input MRI image and mask image

1 2 3



4 5 6

그림 6. 출력 영상(1.Translation, 2.Subtraction, 3.Mask Scaling, 4.Histogram1, 5.Histogram2, 6.Edge Detection)
Fig. 6. Output images(1.Translation, 2.Subtraction, 3.Mask Scaling, 4.Histogram1, 5.Histogram2, 6.Edge Detection)

5.2 매니코어 프로세서의 성능 평가 지표

표 1은 구현된 매니코어 프로세서의 파라미터를 보여주며, 성능분석을 위해 SIMD기반 매니코어 프로세서용 정밀 사이클 (cycle-accurate) 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 4096개의 메모리를 가지고 있으며, 130nm 테크놀로지와 720MHz 클럭 주파수를 사용하여 시뮬레이션 하였다.

표 1. 구현된 매니코어 프로세서의 파라미터
Table 1. Parameters for the implemented many-core processor

Parameter	Value
Number of PEs	16
Pixels/PE	4096
Memory/PE [Word]	4096 [32-bit word]
VLSI Technology	130nm
Clock Frequency	720MHz
Interconnection Network	Mesh
IntALU/intMUL/Barrel Shift/intMACC/Comm	1/1/1/1/1

표 2는 제안한 SIMD기반 매니코어 프로세서의 성능을 평가하기 위해 사용된 4가지 지표를 보여준다. 실행 시간 (execution time)은 각각의 영상처리 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 나다낸다[20].

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

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

5.3 성능 평가 결과 및 분석

본 논문에서는 기존의 고성능 프로세서인 TI C6416, ARM926EJ-S[21], ARM1020E[22]와의 성능 비교를 통해 제안하는 매니코어 프로세서의 잠재 가능성을 보여주고자 한다. 따라서 공정한 성능 평가를 위해 제안한 매니코어 프로세서와 고성능 프로세서들을 동일한 130nm 테크놀로지로 실험하였다. 제안한 매니코어 프로세서는 16개의 프로세싱 엘리먼트(PE)를 사용하여 데이터 레벨 병렬성 (data-level parallelism)을 추구하는 반면, TI C6416은 8-way VLIW 아키텍처로서 8개의 명령어를 동시에 처리할 수 있는 명령어 레벨 병렬성 (instruction-level parallelism)을 추구한다.

표 3은 선택된 다섯 가지 영상처리 알고리즘을 매니코어 프로세서로 이용하여 수행한 결과를 보여주며, 표 4는 각 영상처리 알고리즘에 대해 매니코어 프로세서와 상용 TI C6416, ARM926EJ-S 및 ARM1020E의 성능 비교를 보여준다. 그림 7, 8, 9는 매니코어와 TI C6416, ARM926EJ-S, ARM1020E 의 실행시간, 에너지효율 및 시스템 면적 효율을 그래프로 비교한 그림이다.

예를 들어, 제안한 매니코어 프로세서는 Edge Detection 알고리즘에 대해서 상용 프로세서보다 실행 시간 면에서는 4~39배 이상의 향상을 보여 주고, 에너지 효율 면에서는 5.5~8.5배 이상의 향상을 보여 주며, 시스템 면적 효율 면에서는 1.9~3.8배 이상의 향상된 결과를 보여 준다. 이러한 결과는 제안한 매니코어 프로세서가 TI DSP나 ARM프로세서와 비교하여 시스템 면적과 에너지가 적을 뿐만 아니라 높은 처리량을 보이기 때문이다. 실행 시간이 향상됨으로 인하여 실시간(>30ms) 영상 처리가 가능하며, 동시에 에너지 효율의 증가로 인해 시스템의 배터리 수명을 증가시키는 결과를 가져온다.

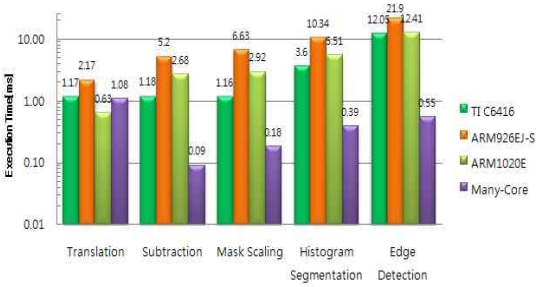


그림 7. 실행시간 비교
Fig. 7. Execution time comparison

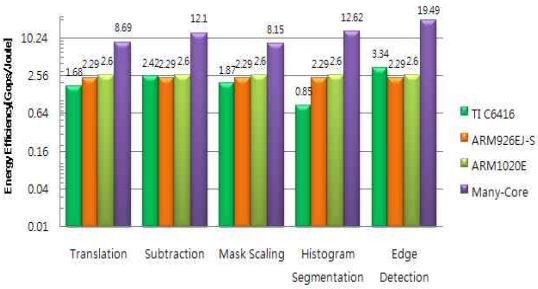


그림 8. 에너지 효율 비교
Fig. 8. Energy efficiency comparison

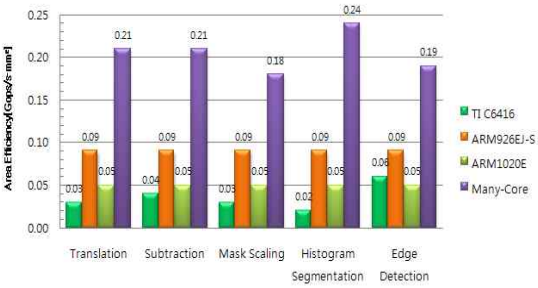


그림 9. 면적 효율 비교
Fig. 9. Area efficiency comparison

표 3. 매니코어 프로세서를 이용한 각 영상처리 알고리즘의 성능 결과
Table 3. Performance of each image processing algorithm using many-core processor

Algorithm	Total Cycle [cycles]	Vector Instruction [cycles]	Scalar Instruction [cycles]	system utilization [%]	sustained throughput [Gops/sec]	execution time [ms]
Translation	776,365	515,504	260,861	98.75	7.65	1.08
Subtraction	61,460	45,064	16,396	92.80	7.84	0.09
Mask Scaling	132,454	98,879	33,575	74.81	8.43	0.18
Histogram Segmentation	277,570	127,356	150,214	90.90	8.68	0.39
Edge Detection	397,616	250,767	146,849	95.94	6.97	0.55

표 4. 매니코어 프로세서와 TI DSP C6416, ARM926EJ-S, ARM1020E와의 성능 비교
Table 4. Performance comparison of many-core processor, TI DSP C6416, ARM926EJ-S, and ARM1020E

Algorithm		Translation				Subtraction				Mask Scaling			
parameter	unit	Many-core	TI C6416	ARM9 26EJ-S	ARM 1020E	Many-core	TI C6416	ARM9 26EJ-S	ARM 1020E	Many-core	TI C6416	ARM9 26EJ-S	ARM 1020E
Technology	[nm]	130	130	130	130	130	130	130	130	130	130	130	130
Clock Frequency	[Mhz]	720	720	250	400	720	720	250	400	720	720	250	400
Average Power	[mW]	1,841.28	950	120	200	1,226.83	950	120	200	1,469.35	950	120	200
Average Throughput	[MIPS]	7,649.24	1,595.85	275	520	7,838.98	2,296.79	275	520	6,433.69	1,773.85	275	520
Execution Time	[ms]	1.08	1.19	2.18	0.63	0.09	1.18	5.20	2.68	0.18	1.16	6.63	2.93
Energy	[μJoule]	1,985.42	1,127.14	261.47	126.43	104 .72	1,121 .81	623.71	535.46	270.31	1,105.07	795.74	585.90
Energy Efficiency	[Gops/ Joule]	8.69	1.68	2.29	2.60	12.10	2.42	2.29	2.60	8.15	1.87	2.29	2.60
Area Efficiency	[Gops/ (s·mm²)]	0.21	0.03	0.10	0.05	0.21	0.04	0.10	0.05	0.18	0.03	0.10	0.05

Algorithm		Histogram Segmentation				Edge Detection			
parameter	unit	Many-core	TI C6416	ARM9 26EJ-S	ARM 1020E	Many-core	TI C6416	ARM9 26EJ-S	ARM 1020E
Technology	[nm]	130	130	130	130	130	130	130	130
Clock Frequency	[Mhz]	720	720	250	400	720	720	250	400
Average Power	[mW]	1,152.24	950	120	200	787.79	950	120	200
Average Throughput	[MIPS]	8,577.49	806.97	275	520	6,970.46	3,358.40	275	520
Execution Time	[ms]	0.39	3.59	10.34	5.51	0.55	12.05	21.90	12.41
Energy	[μJoule]	444.21	3,413.90	1,241.05	1,101.77	435 .05	2,136 .39	2,627.78	2,482.50
Energy Efficiency	[Gops/ Joule]	12.62	0.85	2.29	2.60	19.49	3.54	2.29	2.60
Area Efficiency	[Gops/ (s·mm²)]	0.24	0.02	0.10	0.05	0.19	0.06	0.10	0.05

5.4 합성 및 실험 결과

제안된 SIMD기반 멀티코어 프로세서 구조를 검증하기 위하여 RTL레벨로 설계하고, Xilinx사의 Vertex-4 XC4VLX60 FPGA[23]를 이용하여 합성하고 테스트하였다. 그림 10은 16개의 PE를 내장한 매니코어 프로세서의 스키매틱을 보여주며, 합성한 결과는 표 5와 같다. 각 PE는 1095개의 LUT와 195개의 register가 사용되었으며, ACU는 1147개의 LUT와 124개의 register가 사용되었다. 16 PE로 구성된 매니코어 프로세서는 18,667개의 LUT와 3,244개의 레지스터가 사용되고 전체 메모리 비트는 4,202,496bit이다.

그림 10. 매니코어 프로세서의 하드웨어 스키매틱
Fig. 10. Hardware schematic for the many-core processor

표 5. 매니코어 프로세서 구현의 합성 결과
Table 5. Synthesis result of the many-core processor

합성 결과 리포트		
Array Control Unit	LUTs	1,147
	Register	124
Processing Element	LUTs	1,095
	Register	195
Total Block Memory bits		4,202,496

VI. 결론

본 논문에서는 영상처리 알고리즘을 저전력, 고성능으로 처리하기 위해 SIMD 기반 매니코어 프로세서를 제안하였다. 제안한 매니코어 프로세서는 16개의 프로세싱 엘리먼트를 메쉬 배열 구조로 구성하였으며, 각각의 프로세싱 엘리먼트는 자신에게 맵핑된 영상의 지역데이터를 효율적으로 병렬처리한다. 동일한 공정 (130 nm Technology)과 클럭 주파수(720MHz)를 사용하여 제안한 매니코어 프로세서를 고성능 TI C6416 DSP와 비교한 결과, 실행 시간에서 평균 22배, 에너지 효율에서 평균 7배, 시스템 면적 효율에서 평균 3배의 성능 향상을 보였다. 이러한 결과는 제안한 매니코어 프로세서가 영상처리 애플리케이션 처리에 있어서 무한한 잠재 가능성을 보여주며, 모바일 시스템에 적용할 경우 상당한 성능 향상 및 에너지 소비 감소가 기대된다.

참고문헌

[1] S.-H. Kim, S.-Y. Nam, and H.-J. Lim, "An improved area edge detection for real-time image processing," Journal of the Korea Society of Computer and Information, vol. 14, no. 1, pp. 99-106, Jan. 2009.

[2] X.-G. Jiang, J.-Y. Zhou, J.-H. Shi, H.-H. Chen "FPGA Implementation of Image Rotation Using Modified Compensated CORDIC," in Proc. of 6th Intl. Conf. on ASIC, vol. 2, pp. 752 - 756, 2005.

[3] E. B. Bourennane, S. Bouchoux, J. Miteran, M. Paindavoine, S. Bouillant, "Cost comparison of image rotation implementations on static and dynamic reconfigurable FPGAs," in Proc. of IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing (ICASSP '02), vol. 3, pp. III-3176-3179, 2002.

[4] S.-H. Lee, "The design and implementation of prallel processing system using the Nios(R) II embedded processor," Journal of the Korea Society of Computer and Information, vol. 14, no. 11, pp. 97-103, Nov. 2009.

[5] 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.

[6] A. Gentile and D. S. Wills, "Portable Video Supercomputing," IEEE Trans. on Computers, vol. 53, no. 8, pp. 960-973, Aug. 2004.

[7] L. V. Huynh, C.-H. Kim, and J.-M. Kim, "A massively parallel algorithm for fuzzy vector quantization," The KIPS Transactions: PartA, vol. 16-A, no. 6, pp. 411-418, Dec. 2009.

[8] TMS320C64x families, *http://www.bdti.com/procsum/tic64xx.htm.*

[9] 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.

[10] 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.

[11] 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.

[12] 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.

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

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

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

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

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

[18] S. M. Chai, T. Taha, D. S. Wills, J. D. Meindl, "Heterogeneous Architecture Models for Interconnect-Motivated System Design," IEEE Trans. on VLSI Systems, vol. 8, no. 6, pp. 660-670, 2000.

[19] V. Tiwari, S. Malik, and A. Wolfe, "Compilation techniques for Low Energy: An Overview," in Proc. IEEE Intl. Symp. on Low Power Electrin., pp. 38-39, 1994.

[20] 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.

[21] ARM 926EJ-S data sheet, *http://www.arm.com/products/processors/classic/arm9/arm926.php.*

[22] ARM 1020E data sheet, *http://www.hotchips.org/archives/hc13/2_Mon/02arm.pdf*

[23] Xilinx Vertex-4 FPGA XC4VLX60 data sheet, *http://www.alldatasheet.net/datasheet-pdf/pdf/152986/XILINX/XC4VLX60.html*

저 자 소 개

최 병 국

2009 : 울산대학교 컴퓨터공학사.
2009 : 울산대학교 컴퓨터정보통신공학부 석사과정 입학.
관심분야 : 임베디드 SoC, 컴퓨터 구조, 의료영상처리, 병렬처리
Email: dowonbest@naver.com

김 철 홍

1998 : 서울대학교 컴퓨터공학사.
2000 : 서울대학교 컴퓨터공학부 석사.
2006 : 서울대학교 전기컴퓨터공학부 박사
2005 - 2007년 : 삼성전자 반도체총괄 책임연구원
2007 - 현재 : 전남대학교 전자컴퓨터 공학부 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