논문 2014-09-03

래스터화 알고리즘을 위한 최적의 매니코어 프로세서 구조 탐색

(Architecture Exploration of Optimal Many-Core Processors for a Vector-based Rasterization Algorithm)

손 동 구, 김 철 홍, 김 종 면*
(Dong-Koo Son, Cheol-Hong Kim, Jong-Myon Kim)

Abstract : In this paper, we implement and evaluate the performance of a vector-based rasterization algorithm for 3D graphics by using a SIMD (single instruction multiple data) many-core processor architecture. In addition, we evaluate the impact of a data-per-processing elements (DPE) ratio that is defined as the amount of data directly mapped to each processing element (PE) within many-core in terms of performance, energy efficiency, and area efficiency. For the experiment, we utilize seven different PE configurations by varying the DPE ratio (or the number PEs), which are implemented in the same 130 *nm* CMOS technology with a 500 *MHz* clock frequency. Experimental results indicate that the optimal PE configuration is achieved as the DPE ratio is in the range from 16,384 to 256 (or the number of PEs is in the range from 16 and 1,024), which meets the requirements of mobile devices in terms of the optimal performance and efficiency.

Keywords : 3D graphics, Graphic processing unit (GPU), Many-core processors, Vector-based rasterization algorithm

Ⅰ. 서론

최근 영상 처리 및 통신 기술의 발달로 휴대용 기기에서도 다양한 멀티미디어 기능이 가능하게 되었다. 더불어 휴대용 기기의 성능이 크게 증가하면서 3차원 그래픽스 처리에 대한 요구도 증가되고 있다. 하지만 3차원 그래픽을 표현하기 위해서는 많은 연산량을 처리할 수 있어야 하며 동시에 모바일 디바이스에서의 제한된 소비 전력을 만족시켜야 한다. 이와 같은 이유로 휴대기기용 3차원 그래픽스에 대한 연구가 활발하게 진행되고 있다[1].

*Corresponding Author (jmkim07@ulsan.ac.kr)
Received: 2 July 2013, Revised: 31 July 2013,
Accepted: 1 Oct. 2013.
D.K. Son, J.M. Kim: Ulsan University
C.H. Kim: Chonnam National University
※ 이 논문은 2013년도 정부(미래창조과학부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. NRF-2013R1A2A2A05004566).

ISSN : 1975-5066
http://dx.doi.org/10.14372/IEMEK.2014.9.1.17

이러한 휴대용 기기에서 요구되는 고성능 및 저전력을 만족시키기 위한 대안 가운데 하나로 SIMD(Single Instruction Multiple Data)기반 매니코어 프로세서가 유망하다[2, 3]. 명령어 레벨(instruction-level)이나 스레드 레벨(thread-level)로 동작하는 다중 슈퍼스칼라 프로세서들은 실리콘 면적을 멀티포트 레지스터 파일(multiported register file), 캐쉬(cache), 파이프라인(deep pipelined) 기능 유닛들로 사용한다. 반면에 SIMD 기반 매니코어(또는 graphics processing unit, GPU)는 동일한 수백, 수천 개의 저비용 프로세싱 엘리먼트(processing element, PE)를 이용하여 고성능뿐만 아니라 저전력도 만족시킨다[4]. 특히 SIMD 기반 매니코어는 지역성(locality)이나 규칙성(regularity)이 있는 2차원 패턴 이미지나 3D 그래픽스 처리에 더욱 효과적인 구조이다[5].

본 논문에서는 백터 기반의 래스터화 알고리즘을 SIMD 기반의 매니코어 프로세서에서 수행할 수 있도록 구현하였다. 또한 휴대기기용 고성능 래스터화

표 1. 매니코어 시스템 파라미터
Table 1. Many-core system parameters

Parameter	Value						
# of PEs	4	16	64	256	1,024	4,096	16,384
DPE	65,538	16,384	4,096	1,024	256	64	16
Memory / PE [*words*]	131,110	32,820	8,250	2,100	570	170	70
Clock Frequency	500 *MHz*						
Interconnection Network	Mesh						

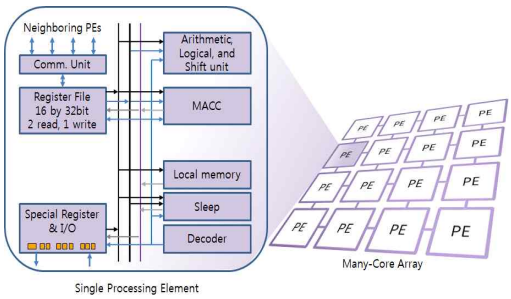

그림 1. SIMD 기반 매니코어 프로세서 아키텍처
Fig. 1 SIMD-based many-core processor architecture

알고리즘을 위한 최적의 매니코어 구조를 탐색하기 위해 PE당 처리하는 데이터(data-per-processing element, DPE) 수를 변화시키는 실험을 통해 각 PE 구조 당 실행시간, 에너지 효율 및 시스템 면적 효율을 측정하였다. 실험은 표 1과 같이 동일한 130 *nm* 테크놀로지와 500 *MHz*의 동작 주파수를 사용하여 일곱 가지의 PE 구조(PEs=4, 16, 64, 256, 1,024, 4,096, 16,384)에 대해 수행하였다.

II. 실험 환경

1. SIMD 기반 매니코어 프로세서 모델

그림 1은 SIMD기반 매니코어 프로세서 아키텍처의 블록도를 보여준다[6, 7]. 매니코어 프로세서는 여러 개의 PE와 이를 제어하는 어레이 제어 유닛(array control unit, ACU)으로 구성되어 있고, 데이터가 각각의 PE에 균등하게 분배되면 PE들은 메쉬 배열 구조에서 명령어들을 동시에 수행하며, 각 PE는 다음과 같은 특징을 가진다.

□ 32비트 폭의 256개 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 시프터(barrel shifter)
□ 지역 정보를 이용해 각 PE들을 활성 및 비활성시키는 SLEEP 유닛
□ 이웃하는 PE들과 데이터 통신을 위한 NEWS (north-east-west-south) 네트워크 및 직렬 I/O유닛

2. 벡터 기반 래스터화 알고리즘

래스터화 알고리즘 중의 하나인 스캔라인(scan-line) 알고리즘은 순차연산을 기반으로 하는 알고리즘이다. 스캔라인 알고리즘에서 어떤 픽셀을 계산하기 위해서는 이전 픽셀의 계산이 선행되고, 선행된 계산의 결과를 이용해야 한다. 이러한 픽셀 간의 연계성 때문에 스캔라인 알고리즘은 병렬프로세서 상에서 효율이 많이 떨어진다. 반면 본 논문에서 사용한 벡터 기반 래스터화 알고리즘을 이용하여 폴리곤 내부의 픽셀 값을 계산할 경우 데이터 간 의존성이 존재하지 않아 폴리곤 내부의 어떤 픽셀이라도 동시에 처리가 가능하므로 벡터 연산(vector operation)을 수행하는 매니코어 프로세서에 적합하다.

벡터 기반 래스터화 알고리즘에서 사용하는 원시 폴리곤은 삼각형이며, 삼각형의 세 정점의 좌표를 $V_0(x_0, y_0)$, $V_1(x_1, y_1)$, $V_2(x_2, y_2)$라 하자. 삼각형 내부에서 처리할 픽셀 $V_p(x_p, y_p)$는 세 정점에 대한 벡터로 표현된다. 정점 V_0과 정점 V_1을 이은 벡터를 Vt_0이라 하고, 정점 V_0과 정점 V_2를 이은 벡터를 Vt_1이라 하였다. 이때 V_p는 Vt_0과 Vt_1을 이용하여 표현할 수 있다. 이때 적절한 a와 b값을 선택한

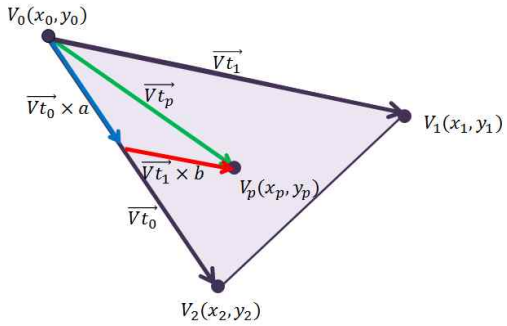

그림 2. 폴리곤 내부의 픽셀
Fig. 2 Pixels inside a polygon

다면 벡터 Vt_0을 a배 축소하고, 벡터 Vt_1을 b배 축소한 벡터를 더한 합성벡터를 이용하여 삼각형 내부의 어떤 픽셀도 표현 가능하다. 그림 2는 삼각형에 대한 정점의 좌표 및 벡터 기반 연산을 통한 삼각형 내부 임의의 좌표 V_p(x$_p$, y$_p$)의 계산 과정을 보여준다.

그림 2에서 벡터간의 관계를 수식으로 표현하면 식 (1)과 같다.

$$Vt_p = a \times Vt_0 + b \times Vt_1. \quad (1)$$

식 (1)을 정점으로 다시 표현하면 식 (2)와 같다.

$$V_p = V_0 + a \times (V_1 - V_0) + b \times (V_2 - V_0). \quad (2)$$

식 (2)에서 a와 b는 식 (3)과 같이 계산된다.

$$a = \frac{(x_2 - x_0)(y_p - y_0) - (y_2 - x_0)(x_p - x_0)}{(x_2 - x_0)(y_1 - y_0) - (y_1 - x_0)(x_2 - x_0)}$$
$$b = \frac{(x_1 - x_0)(y_0 - y_p) - (y_1 - x_0)(x_0 - x_p)}{(x_2 - x_0)(y_1 - y_0) - (y_1 - x_0)(x_2 - x_0)}. \quad (3)$$

마지막으로 좌표 V_p(x$_p$, y$_p$)에 보간 할 픽셀 정보는 식 (4)와 같다.

$$R_p = R_0 + a \times (R_1 - R_0) + b \times (R_2 - R_0)$$
$$G_p = G_0 + a \times (G_1 - G_0) + b \times (G_2 - G_0). \quad (4)$$
$$B_p = B_0 + a \times (B_1 - B_0) + b \times (B_2 - B_0)$$

3. SIMD기반 매니코어 프로세서에서의 구현

표 2는 벡터 기반 래스터화 알고리즘 수행 중에 필요한 데이터를 각 PE구조에 따라 단일 PE의 지역 메모리(local memory)에 할당한 내용을 보여준다. 각 PE의 지역 메모리에는 각 정점의 좌표와 픽셀 값, 최종 출력 결과가 저장될 영역으로 나뉜다. PE 개수가 많아질수록 단일 PE가 필요로 하는 지역 메모리의 양은 적어지지만 데이터가 저장되는 양은 동일하다. 알고리즘을 수행하기 전 각 PE의 지역 메모리에 삼각형 정점의 좌표와 색상 값을 적재해야하며, 각 PE는 DPE만큼의 픽셀 계산을 담당한다. 본 논문에서는 폴리곤 외부의 픽셀을 담당하는 PE들은 식(4)의 연산 과정을 수행할 필요가 없으며, 식 (5)를 이용하여 PE의 활성화/비활성화를 결정한다.

$$P_a = \begin{cases} 1 & \text{if}(a_{num} < 0) \parallel (a_{den} < 0) = true \\ 0 & otherwise \end{cases}$$
$$P_b = \begin{cases} 1 & \text{if}(b_{num} < 0) \parallel (b_{den} < 0) = true \\ 0 & otherwise \end{cases}$$
$$P_{ab} = \begin{cases} 1 & \text{if}(a > 1) \parallel (b > 1) = true \\ 0 & otherwise \end{cases} \quad (5)$$
$$P = \begin{cases} Activate & P_a \times P_b \times P_{ab} = 1 \\ Deactivate & otherwise \end{cases}$$

여기서 a_{num}, a_{den}, b_{num}, b_{den}은 식 (3)에서 변수 a와 b를 계산할 때 사용하는 분자와 분모를 각각 나타낸다.

표 2. 단일 PE의 지역 메모리 사용
Table 2. Local memory usage of a single PE

용도	로컬 메모리 영역						
PE 개수	4	16	64	256	1,024	4,096	16,384
V0의 좌표 및 컬러 값	0~4						
V1의 좌표 및 컬러 값	5~9						
V2의 좌표 및 컬러 값	10~14						
Variable	15~31	15~31	15~31	15~31	15~31	15~31	15~31
Color Channel Result	32~65567	32~16415	32~4127	32~1055	32~287	32~95	32~47
Reserved	65568~65569	16416~16419	4,128~4,129	1,056~1,059	287~299	96~99	48~49
BITMAP_ADDR	65570~131105	16420~32803	4,130~8,225	1,060~2,083	300~555	100~163	50~65
Stack	131106~131110	32803~32820	8,225~8,250	2,083~2,100	555~570	164~170	66~70

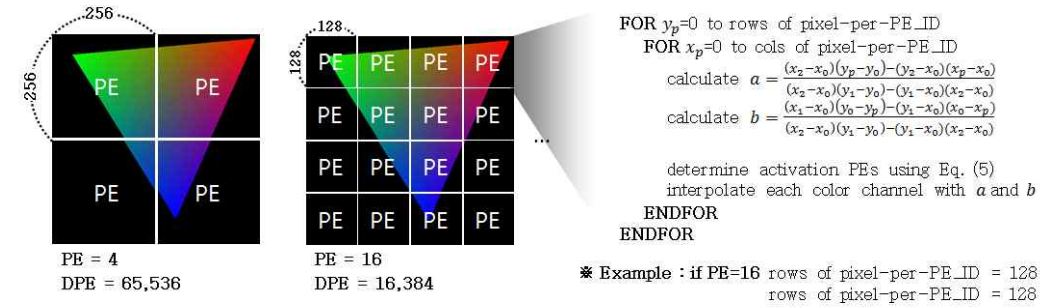

그림 3. 매니코어 프로세서에 구현한 벡터기반의 래스터화 알고리즘
Fig 3 Vector-based rasterization algorithm on many-core processor

그림 3은 PE 구조에 따른 데이터 처리량 및 단일 PE에서의 벡터 기반 래스터화 알고리즘의 처리 과정을 나타낸다. 그림 3에서 볼 수 있듯이 각 PE는 DPE만큼 반복문을 수행하면서 폴리곤 외부 및 내부에 해당하는 픽셀 값을 계산한다. 앞서 언급하였듯이 식 (5)를 이용하여 각 PE는 폴리곤 내부의 픽셀 값을 계산해야하는지 그렇지 않은지를 결정하며, 본 논문에서 폴리곤 외부에 해당하는 픽셀 값은 0을 출력하도록 하였다. 따라서 동일한 PE 구조에서는 폴리곤을 포함한 영상의 크기가 바뀌지 않는 이상 폴리곤의 형상이나 면적에 따라 실행 시간의 변화는 발생하지 않는다. 하지만 폴리곤 내·외부에 해당하는 픽셀을 처리하는 과정에 있어서 폴리곤 내부의 픽셀 수(혹은 폴리곤의 면적)에 따라 PE 활성화/비활성화 횟수가 결정되는데, 이는 각 PE의 처리량(throughput)에 영향을 미친다. 따라서 면적이 동일하다면 폴리곤의 형상은 실행 시간 및 PE의 처리량에 아무런 영향을 미치지 않는다.

Ⅲ. 실험 방법론 및 성능 평가 지표

1. 실험 방법론

그림 4는 SIMD 기반 매니코어 프로세서의 성능, 시스템 면적 효율 및 에너지 효율을 평가하기 위한 실험 방법론을 보여준다. 실험 방법론은 애플리케이션, 아키텍처 및 테크놀로지 레벨로 구성되어 있다. 애플리케이션 레벨에서는 사이클(cycle) 단위의 정밀도를 가진 매니코어 프로세서용 시뮬레이터를 이용하여 벡터 기반 래스터화 알고리즘을 병렬 구현하여 실행 사이클 개수, 동적 명령어 빈도, 시스템 이용률 등의 실행 결과를 얻는다. 아키텍처 레벨에서는 HAM(heterogeneous architecture

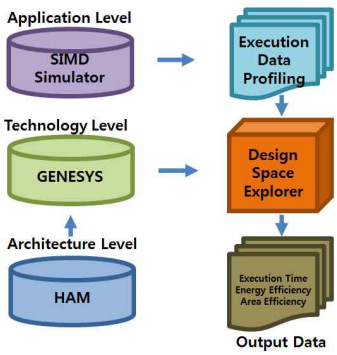

그림 4. 매니코어 프로세서를 위한 실험 방법론
Fig. 4. Simulation methodology for the reference many-core processor

models) 툴을 사용하여 모델링된 아키텍처의 디자인 변수(functional unit type/size, transistor number, critical path)들을 계산한다[8]. 테크놀로지 레벨에서는 아키텍처 레벨에서 구해진 디자인 변수들을 Generic System Simulator(GENESYS)의 입력으로 사용하여 아키텍처 모델들의 사이클 시간(cycle time), 와이어 지연(wire latency), 전력(power), 클록 주파수(clock frequency)를 계산한다[9]. GENESYS는 매니코어 배열 및 멀티 클러스터 프로세서 등의 다양한 시스템 구조를 모델링하기 위한 테크놀로지 모델링 툴로써 각 코어들은 동기 ASIC 칩으로 표현되는 계층적 모델로 구성된다. 이러한 계층적 모델은 기본요소, 재료, 디바이스, 회로, 시스템 등 5-레벨로 이루어져 있다. 처음 세 가지 레벨은 재료특성과 스위칭 디바이스 특성의 물리적 효과를 모델링하며, 회로 레벨은 신호 지연, 동적 및 정적 에너지와 같은 게이트의 특

표 3. 성능평가 지표 요약
Table 3. Summary of performance evaluation metrics

실행 시간 (execution time)	$t_{exec} = C/f_{clk}$
처리량 (throughput)	$Th_{sust} = \frac{O_{exec} \bullet U \bullet N_{PE}}{t_{exec}} \left[\frac{Gops}{s}\right]$
에너지 효율 (energy efficiency)	$\eta_E = \frac{O_{exec} \bullet U \bullet N_{PE}}{Energy} \left[\frac{Gops}{Joules}\right]$
면적 효율 (area efficiency)	$\eta_A = \frac{Th_{sust}}{Area} \left[\frac{Gops}{s \times mm^2}\right]$

징을 모델링한다. 마지막으로 시스템 레벨은 싱글 ASIC 칩을 묘사하기 위한 아키텍처, 연결구조, 패키징 상세 정보를 포함한다. 마지막으로 디자인 공간 탐색기(design space explorer)를 이용하여 위의 세 가지 레벨(애플리케이션, 아키텍처, 테크놀로지)에서 구해진 파라미터들을 조합하고, PE 구조별 실행시간, 시스템 면적 효율 및 에너지 효율을 측정한다.

2. 성능 분석 지표

표 3은 매니코어 프로세서의 성능 평가를 위한 세 가지 지표를 보여준다. 실행 시간(execution time)은 래스터화 알고리즘이 수행된 시간을, 에너지 효율(energy efficiency)은 단위 에너지당 소비된 명령어 개수(giga-operations/Joule)를 나타내고, 시스템 면적 효율(area efficiency)은 단위 시간에 단위 시스템 면적당 소비된 명령어 개수를 나타낸다. 여기서 C는 사이클 수, f_{clk}는 클록 주파수, $Energy$는 130 nm공정에서 소비된 에너지($Joule$)를, $Area$는 시스템의 면적(mm^2)을 나타낸다. 처리량에서의 O_{exec}는 발행된 명령어 수를, U는 시스템 이용률을, N_{PE}는 PE의 개수를 나타낸다.

Ⅳ. 결과 및 성능 분석

표 4는 실험에 사용된 정점의 좌표와 초기 픽셀 값을 나타낸다. 본 논문의 2장 3절에서 언급하였듯이 PE 구조 별 폴리곤의 면적에 따라 각 PE의 처리량이 달라지며 이는 궁극적으로 표 3에서 볼 수

표 4. 실험에 사용된 정점 데이터
Table 4. Vertex data used in the experiment

폴리곤	정점	X좌표	Y좌표	R	G	B
P_{max}	V0	0	0	0	250	0
	V1	511	0	0	0	255
	V2	511	511	255	0	0
P_{half}	V0	0	255	0	250	0
	V1	511	0	0	0	255
	V2	511	511	255	0	0
$P_{quarter}$	V0	0	384	0	250	0
	V1	511	0	0	0	255
	V2	511	511	255	0	0

있듯이 에너지 효율과 시스템 면적효율에 영향을 미친다. 이를 확인하기 위해 본 논문에서는 512×512 크기의 영상에서 표현할 수 있는 최대 넓이에 해당하는 폴리곤(P_{max})과, 최대 넓이의 $\frac{1}{2}$ (P_{half})과 $\frac{1}{4}$(P_{quarter})인 폴리곤 3개를 대상으로 PE 구조 별 실행 시간, 에너지 효율, 시스템 면적 효율을 측정한다.

1. 실행 시간

그림 5는 PE 구조 별 실행 시간으로, 폴리곤의 면적이 다르더라도 PE의 개수가 4배 증가할수록(혹은 DPE가 4배 감소할수록) 실행 시간은 거의 4배씩 동일한 비율로 감소하는 것을 확인할 수 있었다. 이는 2장 3절에서 언급한데로 벡터 기반 래스터화 알고리즘이 데이터 병렬성(data parallel)의 특징을 보이기 때문이라고 할 수 있다. 하지만 그림 5에서 PE를 16,384개 사용하였을 경우에는 실행 시간의 감소폭이 다른 PE 구조에 비하여 다소 줄어드는 것을 확인할 수 있는데, 이는 하나의 PE가 담당하는 픽셀(혹은 데이터)의 개수가 줄어들어 PE가 폴리곤을 처리하는데 사용되는 연산량 보다 반복문을 처리하는데 사용된 연산량의 비율이 높아졌기 때문이다.

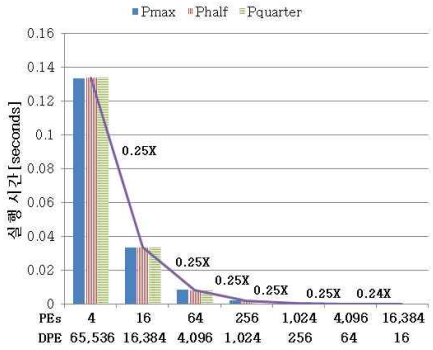

그림 5. PE 구조별 실행 시간
Fig. 5 Execution time for variable PE configurations

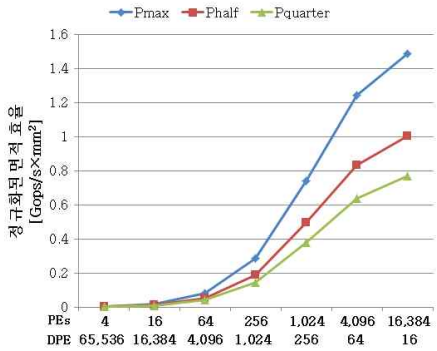

그림 6. PE 구조별 면적 효율
Fig. 6 Area efficiency for variable PE configurations

2. 면적 효율

면적 효율은 시스템의 단위 시간에 단위 시스템 면적 당 수행할 수 있는 작업량으로 정의되며, 그림 6은 PE 구조 별 시스템 면적 효율을 보여준다. 표 4는 PE 구조 별 시스템 처리량과 시스템 면적을 보여준다. 표 5에서 확인 할 수 있듯이 PE 구조별 시스템 처리량은 거의 4배씩 일정하게 증가하는데 반해, 시스템 면적 증가율은 0.98배에서 3.34배로 증가폭이 처리량의 증가폭보다 작으므로 시스템 면적 효율은 계속적으로 증가하는 추세를 보인다. 또한 폴리곤의 면적이 감소할수록 동일 PE 구조에서의 시스템 면적 효율 또한 감소하는 것을 확인할 수 있는데, 이는 폴리곤 내부의 픽셀을 처리하는데 소요되는 총 명령어 수가 감소하여 시스템 전체 처리량의 감소를 가져왔기 때문이다.

표 5. PE 구조 별 시스템 면적과 처리량
Table 5. System area and sustained throughput for variable PE configurations

삼각형	PE 개수 / 항목	4	16	64	256	1,024	4,096	16,384
P_{max}	처리량($Gops/s$)	0.8430	3.3720	13.488	53.947	215.75	863.10	3452.3
	시스템 면적(mm^2)	167.02	164.53	167.08	189.15	292.03	694.87	2319.2
P_{half}	처리량($Gops/s$)	0.5665	2.2663	9.0651	36.261	145.06	580.81	2332.0
	시스템 면적(mm^2)	167.02	164.53	167.08	189.15	292.03	694.87	2319.2
$P_{quarter}$	처리량($Gops/s$)	0.4307	1.7229	6.8917	27.570	110.34	442.09	1781.4
	시스템 면적(mm^2)	167.02	164.53	167.08	189.15	292.03	694.87	2319.2

표 6. PE 구조 별 에너지 소비와 처리량
Table 6. Energy consumption and sustained throughput with varying numbers of PE

삼각형	PE 개수 / 항목	4	16	64	256	1,024	4,096	16,384
P_{max}	처리량($Gops/s$)	0.8430	3.3720	13.488	53.947	215.75	863.10	3452.3
	에너지 소비($Joules$)	0.0640	0.0159	0.0044	0.0017	0.0010	0.0009	0.0008
P_{half}	처리량($Gops/s$)	0.5665	2.2663	9.0651	36.261	145.06	580.81	2332.0
	에너지 소비($Joules$)	0.0346	0.0067	0.0025	0.0010	0.0007	0.0006	0.0005
$P_{quarter}$	처리량($Gops/s$)	0.4307	1.7229	6.8917	27.570	110.34	442.09	1781.4
	에너지 소비($Joules$)	0.0198	0.0050	0.0015	0.0007	0.0005	0.0005	0.0004

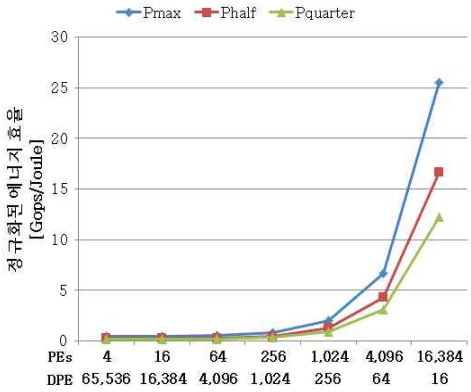

그림 7. PE 구조별 에너지 효율
Fig. 7 Energy efficiency for variable PE configurations

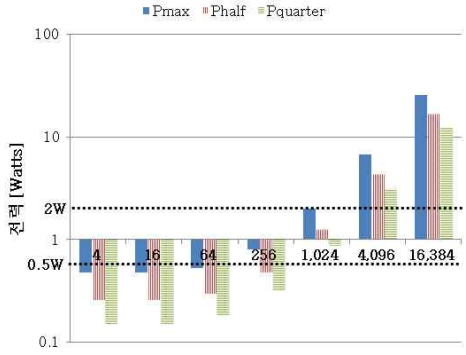

그림 8. PE 구조별 전력소비
Fig. 8 Power consumption for variable PE configurations

3. 에너지 효율

표 3에서와 같이 에너지 효율은 단위 에너지 당 수행할 수 있는 작업량으로 정의할 수 있으며, 그림 7은 PE 구조 별 에너지 효율을 보여준다. 그림 7에서 볼 수 있듯이 PE 개수가 많아질수록 에너지 효율이 계속적으로 증가하는 추세를 보이는데, 이 또한 시스템 면적 효율과 유사하게 PE 구조 별 시스템 처리량 증가폭 대비 벡터 기반 래스터화 알고리즘을 처리하는데 소모되는 에너지의 증가폭이 그에 미치지 못하기 때문이다(표 6 참고).

본 논문에서는 휴대기기용 고성능 래스터화 알고리즘을 위한 최적의 매니코어 구조를 탐색하고자 한다. 하지만 휴대기기의 경우 기기 및 전력 공급 장치의 소형화에 따른 전력 소모 제한이 있다. International Technology Roadmap for Semiconductor(ITRS)에 따르면 일반 휴대기기에서의 전력은 0.5 *Watts*, 최근 출시되는 휴대용 태블릿(tablet)의 경우에는 전력이 2 *Watts*로 제한된다[10]. 따라서 그림 8에서와 같이 일곱 가지 PE 구조에서 0.5 *Watts*의 전력 제한을 만족시키는 구조는 16개의 PE를 사용하였을 때이며, 2 *Watts*의 전력 제한을 만족시키는 구조는 1,024개의 PE를 사용하였을 때이다. 실행 시간, 시스템 면적 효율 및 에너지 효율의 경우는 실험결과 PE 개수를 많이 사용할수록 증가하는 추세에 있으므로 휴대기기에서 벡터 기반 래스터화 알고리즘을 위한 최적의 PE 개수는 16개에서 1,024개 사이로 볼 수 있다.

Ⅴ. 결론

본 논문에서는 휴대기기용 고성능, 저전력 벡터 기반 래스터화 알고리즘을 SIMD 기반 매니코어 프로세서를 이용하여 구현하였다. 또한 벡터 기반 래스터화 알고리즘을 위한 최적의 매니코어 프로세서 구조를 탐색하기 위해 일곱 가지 PE 구조(PEs=4, 16, 64, 256, 1,024, 4,096, 16,384)에 대해 512×512 출력 영상에 포함되는 면적이 다른 폴리곤을 각각 출력되도록 실험하였다. 또한 실험은 동일한 130 *nm* 테크놀로지와 500 *MHz*의 동작 주파수를 사용하였다. 본 논문에서는 각 PE 구조 별 실행 시간, 시스템 면적 효율, 그리고 에너지 효율을 측정하였다. 실행 시간은 PE 개수가 증가할수록 감소하는 경향을, 시스템 면적 효율과 에너지 효율을 PE 개수가 증가할수록 증가하는 경향을 나타내었다. ITRS에 따르면 휴대기기에서의 전력 소비는 0.5 *Watts* ~ 2 *Watts*로 제한되어 있으며, PE 구조 별 실행 시간, 시스템 면적 효율, 에너지 효율 및 전력 소비의 실험결과를 바탕으로 본 논문에서는 16개에서 1,024개 PE를 사용하였을 때 휴대기기용 고성능 벡터 기반 래스터화 알고리즘을 위한 최적의 PE 구조임을 확인할 수 있었다.

References

[1] W. Yoo, S. Shi, W.J. Jeon, K. Nahrstedt, R,H. Campbell, “Real-Time Parallel Remote Rendering for Mobile Devices using Graphics Processing Units,” Proceedings of the IEEE

International Conference on Multimedia and Expo, pp.902-907, 2010.
[2] N. Singhal, J.W. Yoo, H.Y. Choi, I.K. Park, "Implementation and Optimization of Image Processing Algorithms on Embedded GPU," IEICE Trans. Inf. & Syst., Vol. E95-D, No. 5, pp.1475-1484, 2012.
[3] I.K. Park, N. Singhal, M.H. Lee, S. Cho, C.W. Kim, "Design and Performance Evaluation of Image Processing Algorithm on GPUs," IEEE Trans. Parallel Distrib. Syst., Vol. 22, No. 1, pp.91-104, 2011.
[4] Y.H Ahn, Y.S. Hwang, K.S. Chung, "Kernel Level Power Management Solution for Multi-Core," Journal of IEMEK, Vol. 4, No. 2, pp.50-54, 2009 (in Korean).
[5] D.K. Shon, J.M. Kim, "Implementation and Performance Evaluation of Vector based Rasterization Algorithm using a Many-Core Processor," Journal of IEMEK, Vol. 8, No. 2, pp.87-93, 2013 (in Korean).
[6] B.K. Choi, C.H. Kim, J.M. Kim, "Implementation of SIMD-based Many-Core Processor for Efficient Image Data Processing," Journal of the KSCI, Vol. 16, No. 1, pp.1-9, 2011 (in Korean).
[7] Y.M. Kim, J.M. Kim, "Design and Verification of High-Performance Parallel Processor Hardware for JPEG Encoder," Journal of IEMEK, Vol. 6, No. 2, pp.100-107, 2011 (in Korean).
[8] S.M. Chai, T.M. Taha, D.S. Wills, 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, 2000.
[9] S.P. Nugent, "A Second Generation GENEric SYstems Simulator (GENESYS) for a Gigascale System-on-a-Chip (SoC)," PhD dissertation, Georgia Institute of Technology, USA, 2005.
[10] International Technology Roadmap for Semiconductors 2011 Edition, http://www.itrs.net/Links/2011ITRS/2011Chapters/2011SysDrivers.pdf

저 자 소 개

손 동 구

현재, 울산대 전기공학부 석사과정.
관심분야: 병렬프로세서 구조, GPGPU 프로그래밍, 임베디드시스템

Email: dongkoo88@gmail.com

김 철 홍

1998년 서울대 컴퓨터공학사.
2000년 서울대 컴퓨터공학부 석사.
2006년 서울대 전기컴퓨터공학부 박사.
현재, 전남대 전자컴퓨터공학부 교수.
관심분야: 임베디드시스템, 컴퓨터구조, SoC설계, 저전력 설계.

Email: cheolhong@gmail.com

김 종 면

1995년 명지대 전기공학과 학사.
2000년 University of Florida 전기컴퓨터공학과 석사.
2005년 Georgia Tech. ECE 박사.
현재, 울산대 전기공학부 교수.
관심분야: 임베디드 SoC, 컴퓨터구조, 병렬처리.

Email: jmkim07@ulsan.ac.kr