The Journal of The Institute of Internet, Broadcasting and Communication (IIBC)
Vol. 18, No. 6, pp.201-206, Dec. 31, 2018. pISSN 2289-0238, eISSN 2289-0246

https://doi.org/10.7236/JIIBC.2018.18.6.201
JIIBC 2018-6-27

범용 그래픽 처리장치 (GPGPU)의 성능에 대한 연구

A Study of The GPGPU Performance

이종복*

Jongbok Lee*

요 약 최근에 이르러 인공지능과 빅데이터 기술이 발달함에 따라, 범용 그래픽 처리장치인 GPGPU에 대한 중요성이 강조되고 있다. 또한, 블럭체인의 응용기술인 비트코인을 얻기 위한 채굴기에 대한 수요가 급증하여 GPGPU의 가격이 급상승하는 등 품귀현상이 일어나고 있다. 만일 범용 그래픽 처리장치를 정밀하게 모의실행할 수 있다면, 고가의 범용 그래픽 처리장치를 구매하지 않고도 다양한 범용 그래픽 처리장치 유형에 대한 실험을 수행하여 그 성능을 분석할 수가 있다. 본 논문에서는 GPGPU-Sim을 이용하여 범용 그래픽 처리장치 모의실험기의 구성을 고찰하고, 다양한 벤치마크 프로그램에 대한 성능을 측정하였다.

Abstract As the artificial intelligence and big data technology has been developed recently, the importance of GPGPU, which is a general purpose graphics processing unit, is emphasized. In addition, by the demand for mining equipment to obtain bit coins, which is a block chain application technology, the price of GPGPU has increased sharply with scarcity. If a GPGPU can be precisely simulated, it is possible to conduct experiments on various GPGPU types and analyze performance without purchasing expensive ones. In this paper, we investigate the configuration of a GPGPU simulator and measure the performance of various benchmark programs using GPGPU-Sim.

Key Words : GPGPU, GPGPU-Sim, performance

Ⅰ. 서 론

그래픽 처리장치(Graphic Processing Unit, GPU)는 프레임버퍼 (frame buffer) 내부의 영상을 재현하기 위하여 메모리를 고속으로 업데이트하여 컴퓨터 모니터와 같은 디스플레이 장치에 영상을 출력할 수 있도록 한다. 오늘날 그래픽처리장치의 복잡도가 증가하여 그 자체를 컴퓨터라고 할 수 있을 정도이다.

초기의 그래픽 처리장치는 그래픽 카드로 불리웠는데, 그래픽 카드는 PCB 형태로 제조되어 확장슬롯에 삽입된다. 그래픽 카드는 위와 같이 확장슬롯에 삽입하는 전용 그래픽 장치와 달리, 마더보드에 포함된 집적 그래픽 처리장치의 형태로도 이용된다. 집적 그래프장치의 장점은 단순, 간결하고 전력소모가 작다는 점이며, 단점은 CPU와 시스템 자원을 공유하기 때문에 CPU의 성능에 영향을 끼친다는 점이다. 오늘날 대부분의 현대적 비디오 카드들은 양대 제조사인 AMD나 Nvidia에서 생산된 그래픽 칩들로 구성된다. 전용 그래픽장치는 자체의 RAM, 냉방장치, 전력조정기 및 영상데이터 전용처리 장치들로 구성된다. 집적 그래픽장치에서 전용 그래픽장치로 구조

*정회원, 한성대학교 전자정보공학과
접수일자 : 2018년 11월 1일, 수정완료 : 2018년 11월 30일
게재확정일자 : 2018년 12월 7일

Received: 1 November, 2018 / Revised: 30 November, 2018 / Accepted: 7 December, 2018
*Corresponding Author: jblee@hansung.ac.kr
Dept of Electronics and Information Eng., Hansung University, Korea

를 개선시키면 CPU와 RAM의 부담이 경감되므로 그래픽처리만 빨라지는게 아니라, 컴퓨터의 전체적인 성능이 향상된다.

그래픽처리장치의 연산기능이 증가함에 따라, 전력 수요 역시 급증한다. 최근 들어 CPU와 전력공급기 제조업체들이 효율을 중시하고 있는 반면에, GPU에 대한 전력 수요는 계속 증가하여 현재 컴퓨터에서 가장 많은 전력을 소비하는 장치가 되었다. 기존의 PCI-Express 연결로는 75 W 밖에 공급할 수가 없기 때문에 최근에는 75 W의 6핀과 150 W의 8핀을 이용하여 전력공급기에 직접 연결한다. 여러 장의 그래픽카드를 장착한 컴퓨터에서는 1000 W 이상의 전력이 필요하며, 냉각장치가 주요 이슈로 대두되었다.

오늘날 그래픽처리장치는 단순히 디스플레이 장치에 출력하는 것 뿐만이 아니라, 추가의 연산처리를 통하여 컴퓨터의 CPU의 연산업무를 경감시킨다. 2010년도 하반기부터 그래픽 프로세서의 연산 기능으로 그래픽 처리 외에 그래픽이 아닌 일반 연산을 수행하기 시작했다.

하나의 컴퓨터에서 여러 장의 비디오 카드나 다수의 그래픽 칩을 활용하면 그래픽 처리를 더욱 병렬화할 수 있다. 심지어는 여러 장의 비디오카드를 쓰지 않고 오직 한 개의 GPU와 CPU로 구성을 하더라도 각 칩의 전문화로 인해 멀티코어 CPU보다 높은 성능을 제공한다.

기본적으로 GPGPU 파이프 라인은 여러 개의 GPU와 CPU로 구성된 병렬 처리기이다. GPU는 CPU보다 더 낮은 주파수에서 작동하지만 CPU보다 코어의 수가 매우 많다. 따라서 GPU는 기존 CPU보다 초당 훨씬 많은 영상과 그래픽 데이터를 처리 할 수 있다. 나아가, 그래픽이 아닌 데이터조차 그래픽 형식으로 변형시켜 GPU를 사용하여 스캔하고 분석하면 더욱 큰 속도의 향상을 얻을 수 있다. GPGPU 파이프 라인은 그래픽 처리를 위해 21 세기 초에 개발되었으며, 과학적 컴퓨팅 요구 사항에 부합하는 방향으로 발전되었다.

본 논문에서는 GPGPU-Sim을 이용하여 GTX580 범용 그래픽 처리장치의 성능을 측정하였다. 측정에 사용된 벤치마크는 대규모 그래프 알고리즘 가속기, 입체구조망에 대한 라플라스 이산화, 신경망, 분산기억시스템, 기상예보 등 9 개의 프로그램으로 구성된다.

본 논문은 다음과 같이 구성된다. 2 장에서 GPGPU 모델과 GPGPU-Sim 모의실험기에 대하여 살펴보고, 3 장에서 GPGPU-Sim 모의실험 환경에 대하여 고찰하고 모의실험 결과를 보이며, 4 장에서 결론을 맺는다.

II. GPGPU 모델 및 모의실험기

GPGPU-Sim은 현재 시장에서 판매중인 GPU에 대한 마이크로 아키텍처 타이밍모델을 반영한 것으로서, CUDA (Compute Unified Device Architecture) 및 OpenCL 프로그램을 실행할 수가 있다. CUDA는 엔비디아(nVIDIA)에서 개발되었으며, 그래픽 처리장치에서 수행하는 병렬처리 알고리즘을 C 프로그램 언어를 비롯한 산업 표준 언어를 사용하여 작성할 수 있도록 하는 기술이다. OpenCL (Open Computing Language)은 애플(Apple)이 최초로 개발하였으며, 병렬컴퓨팅을 위한 개방형 프레임워크이다.

그림 1에 나타낸 것과 같이 GPGPU는 CPU의 보조 프로세서로서의 역할을 수행하는데, 그 구조는 여러 개의 코어가 내부 연결망에 의하여 메모리제어기 및 DRAM과 연결된다.

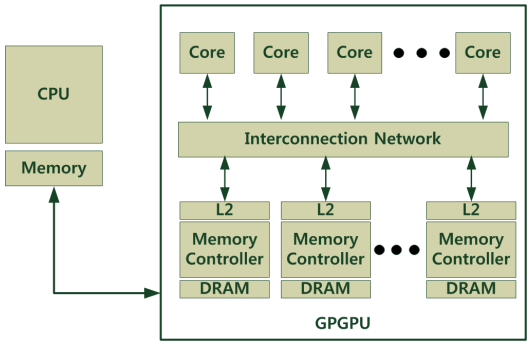

그림 1. GPGPU 모델
Fig. 1. The GPGPU Model

GPGPU 내부의 각 코어들은 쉐이더 코어 (Shader core)라고 부르며, 세밀한 멀티쓰레딩 (Fine-grained multi-threading)을 수행한다. 원래 쉐이딩(shading) 또는 그림자처리란 영상을 묘사할 때 빛의 어두운 정도를 달리해서 거리감을 표현하는 방법으로, 쉐이더 코어(shader core)는 그래픽처리장치에서 쉐이딩을 처리하는 전용 프로세서 코어라는 의미에서 출발했다. 그러나, 현대의 쉐이더 코어는 위에서 설명한 쉐이딩 처리 기능 이외의 영상처리를 수행하거나 심지어는 그래픽처리가 아닌 분야로까지 그 기능이 확대되었다.

그림 2에 도시하였듯이 쉐이더 코어의 집합이 워프(warp)이며, 워프는 스케줄러에 의하여 스케줄링되는 여러 개의 쓰레드 워프(thread warp)로 구성된다. GPU는 쓰레드 워프를 병렬실행 시키면서 지연시간이 오래 걸리는 경우, 문맥교환(context switching)을 통하여 성능이 저하되는 것을 경감시킨다. 스케줄러에 의하여 투입된 쓰레드 워프는 쉐이더 코어 파이프라인에서 인출, 해독, 실행, 예비 메모리, 메모리, 되쓰기의 SIMD 파이프라인 단계를 거친다. 쓰레드의 실행은 지연시간을 없애기 위하여 인터리브 (interleaved) 방식으로 진행된다. 이것은 쓰레드 1번이 실행되다가 분기 명령어 등을 만나서 더 이상 실행이 불가하면, 쓰레드 2번이 실행되어 지연시간을 없애는 방식이다. 인출 단계에서 한 개의 워프를 스케줄링하여 파이프라인 내에서 실행을 시작할 수 있도록 한다.

분기에 의하여 제기되는 불규칙 흐름은 비활성화된 쓰레드를 활성화시키는 방법으로 처리한다. 본 GPU의 쓰레드 스케줄링 정책의 기본으로 PDOM (Post-dominator reconvergence)과 DWF (Dynamic warp formation)를 이용한다.

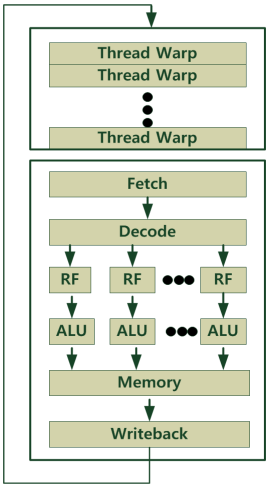

그림 2. 쉐이더 코어의 구조
Fig. 2. The Shader Core

PDOM은 워프를 구성하는 쓰레드가 활성화되어있을 동안에는 바뀌지 않는 정적인 워프 형성에 기반을 둔다. 그리고 다음 워프를 파이프라인에 삽입하기 위하여 라운드 로빈 (round-robin) 방법을 이용한다. GPGPU에서 여러 개의 쓰레드가 병렬처리될 때, 분기가 발산하는 경향이 있다. 분기가 발산한다는 것은 분기와 관련된 여러 개의 쓰레드들이 서로 다른 실행 경로를 따라 수행되고, 결국에는 그 중에서 오직 분기에 의하여 선택된 쓰레드들만 이용되고 나머지 쓰레드를 폐기하는 불필요한 연산처리에 의하여 전체 성능이 저하되는 것을 의미한다. PDOM에서는 분기가 발산하는 것을 막기 위하여 마스킹 기술을 쓴다. 마스킹 기술이란 워프 당 여러 개의 프로그램 카운터를 저장하고, 각 프로그램 카운터와 연관된 쓰레드의 프로그램 카운터들을 모아서 마스크 벡터로 나타내는 방법이다. 이러한 마스크 벡터의 정보를 이용하여, 특정한 프로그램 카운터와 연관되지 않은 쓰레드들의 실행을 일시적으로 비활성화시킨다. PDOM의 장점으로, SIMD에 대한 최대한의 활용을 통하여 최고수준의 기본 블럭에 대한 실행이 보장된다는 것이다. 그러나, PDOM의 단점으로는 발산경로에 많은 수의 명령어가 있을 때는 성능이 저하되고, 만일에 그대로 수행되었다면 인터리브 방식에 의하여 메모리에 대한 접근 지연시간을 제거할 수 있는 쓰레드임에도 불구하고, 마스킹 기술로 굳이 비활성화시킬 수가 있다는 점이다.

DWF의 기본 개념은 SIMD하에서의 워프에 대한 높은 활용율을 견지하기 위하여 동일한 발산경로를 갖는 워프들을 병합하는 것이다. 만일 동일한 발산경로를 갖는 워프의 개수가 충분하지가 않다면 애초의 DWF의 목표를 달성할 수가 없다. DWF의 장점으로는 PDOM과는 달리 어느 쓰레드도 마스킹하지 않으므로 모든 쓰레드들이 활성화되어있다는 점이고, 수렴을 고려하지 않으므로, 임의의 기본블럭에 대하여 SIMD의 최대활용을 통한 실행이 보장되지 않는다는 점이다.

각 쓰레드의 레지스터 값들은 레지스터 화일에 저장되며, 각 순간마다 각 쓰레드 당 한 개의 명령어가 파이프라인 내에 존재한다. GPGPU-Sim은 PTX(Parallel Thread Execution)에 대하여 기능적 모의실행을, GPU의 연산부에서는 타이밍 모의실행을 수행한다. GPGPU-Sim은 구동이 시작된 각 CUDA 커널을 실행시키는 GPU의 타이밍 모델을 모의실행하면서, 각 커널을 실행시키는데 소용된 싸이클 수를 기록하며, GPU가 동작중일 때는 CPU는 휴지기 상태에 있다.

텍스쳐 캐쉬는 현대 GPU의 필수적인 성분으로서 실제영상을 구현할 때 실시간 성능을 얻기 위하여 중요한 역할을 수행한다. 텍스쳐 캐쉬는 읽기 전용 캐쉬이며, 영상을 삼각형의 형태로 배열시키는 텍스쳐 매핑 처리를 위하여 영상 데이터를 저장하는 역할을 한다[1-6].

Ⅲ. 모의실험 환경 및 결과

본 논문의 모의실험은 운영체제 Ubuntu-14.04에서 3.1GHz로 동작하는 Intel Core i7에서 시행하였으며, GPGPU-Sim을 설치하기 위하여 CUDA 4.0 Toolkit을 이용하였다[7]. 표 1은 모의실험에 이용된 9 개의 벤치마크 프로그램으로서, 그래프 탐색, 쿨롱 전위 분포, 몬테카를로 시뮬레이션, 라플라스 이산화, 신경망의 계산, 체스판 놀이, 선추적, 분산저장시스템, 수치해석을 이용한 기상예보를 수행하는 프로그램들이다[8-10].

표 1. GPGPU-Sim의 입력 벤치마크 프로그램
Table 1. Benchmark Programs for GPGPU-Sim

벤치마크	기술
BFS	그래프의 폭 위주 탐색
CP	쿨롱 전위 분포
LIB	몬테카를로 시뮬레이션
LPS	입체구조망에 대한 라플라스 이산화
NN	신경망 계산
NQU	체스 게임
RAY	선의 추적
STO	분산 저장 시스템
WP	기상 예보 수치 해석

표 2에 본 논문에서 사용하는 GPGPU 아키텍처의 하드웨어 사양을 나타냈다. 본 GPGPU는 1,536 개의 쓰레드를 가지며, 쓰레드의 집합인 워프는 32 개로 구성된다. 코어 당 레지스터의 수는 32,768 개이고, 이러한 레지스터로 구성되는 레지스터 화일은 4 개이다.

GPU의 소프트웨어는 여러 개의 병렬성을 나타내는 기본 작업 모듈인 커널들로 이루어지고, 이 커널들은 다시 쓰레드의 집합인 쓰레드 블럭 또는 병렬 쓰레드 배열(Concurrent Thread Arrays, CTA)로 불린다. 쉐이더 내의 CTA 의 수는 8 개이다.

CTA는 GPU의 SM(Streaming Multiprocessor)로 할당되는 기본 작업 단위인데, CTA 내부의 쓰레드들은 다시 워프들로 구성된다. 워프는 동일한 프로그램 카운터를 공유하는 최소의 실행 단위이다. 클러스터는 각 노드를 GPU로 연결하여 하나의 시스템으로 만든 집합을 의미하며 15 개이다.

표 2. GPGPU 아키텍쳐 하드웨어의 사양
Table 2. The architecture specification of GPGPU

항목	값
쓰레드의 수	1536
워프의 수	32
레지스터 화일 그룹의 수	4
코어 당 레지스터의 수	32,768
쉐이더 내 CTA 수	8
CTA당 배리어 수	16
클러스터의 수	15
클러스트 당 SIMD 수	1
코어 당 공유메모리의 수	16384
레지스터 뱅크의 수	16
코어 당 워프 스케줄러	2

그림 3에 9 개의 벤치마크에 대하여 모의실험을 수행하여 그 성능을 IPC로 나타냈다. GPGPU가 수퍼스칼라 또는 멀티코어 프로세서로서는 도달할 수 없는 월등한 성능을 가져다 주는 것을 알 수 있다. *CP*는 521.5 IPC의 최고 성능을 나타냈으며 *LPS*와 *RAY*가 각각 445.2 IPC와 436.4 IPC로 그 뒤를 이었다. 반면에, *BFS*, *LIB*, *NQU*는 각각 22.0, 26.0, 33.5 IPC에 그쳤다. 9 개 벤치마크 프로그램의 성능에 대한 기하평균을 구하면 131.7 IPC를 얻을 수 있다.

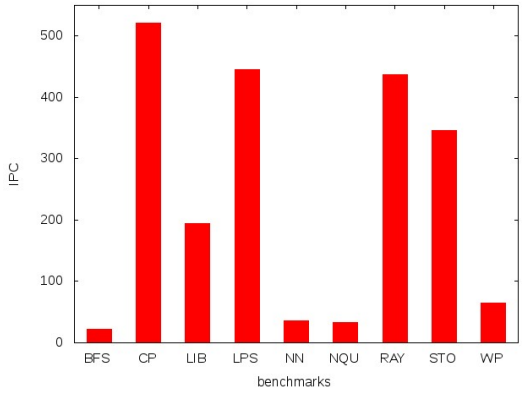

그림 3. GPGPU 모의실험 성능 결과
Fig. 3. GPGPU-Sim simulated performance results

그림 4에 9 개의 벤치마크에 대하여 모의실험을 수행하여 평균 전력 소비량의 결과를 보였다. 각 벤치마크의 전력 소비량은 성능이 높을 수록 증가하는 것으로 나타났다. 최저의 성능을 보인 BFS는 평균 전력 소비량 역시 하위권인 39.0 W에 불과한 반면, 최고의 성능을 보인 *CP*는 평균 전력 소비량이 101.9 W를 기록했다. *LPS*는 106.8 W의 가장 많은 전력을 소비했으며 그에 대응하여

성능도 *CP* 다음으로 높다. 모의실험 결과에서 알 수 있듯이, 각 벤치마크 프로그램 별로 최저 33.6 W에서 최고 106.8 W 범위의 평균 소비 전력을 기록하였다.

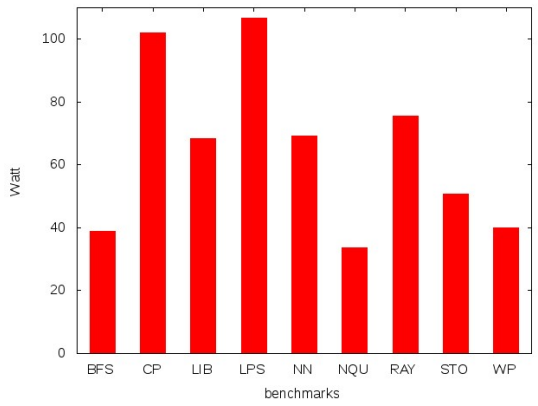

그림 4. GPGPU 모의실험 전력 소모량
Fig. 4. GPGPU-Sim simulated power dissipation

IV. 결 론

본 논문에서는 최근에 인공지능이나 비트코인 등의 용도로 뜨거운 관심을 받고 있는 범용 그래픽 장치인 GPGPU에 대하여 고찰하고, GPGPU-Sim을 이용하여 GTX580의 성능과 전력을 측정하였다. 그래프 탐색, 쿨롱 전위 분포, 몬테카를로 시뮬레이션, 입체구조망, 신경망, 체스 게임, 선의 추적, 분산 저장 시스템, 기상예보 9개에 대한 성능을 모의실험을 통하여 성능과 소비전력을 측정한 결과, 각각 평균 131.7 IPC와 65.0 W를 얻었다.

추후로, 현존하는 다양한 범용그래픽처리장치에 대하여 GPU 아키텍처의 파라미터를 다변화하고 확대된 벤치마크 프로그램을 대상으로 모의실험을 수행하여 더욱 심층적인 분석을 수행할 예정이다.

References

[1] A. Bakhoda, G. L. Yuan, W. W. L. Fung, H. Wong, and T. M. Aamodt, "Analyzing CUDA Workloads Using a Detailed GPU Simulator," 2009 International Symposium on Performance Analysis of Systems and Software, pp.163-174, May. 2009.

[2] A. Lshagar, A Baniasadi, "Performance in GPU Architectures : Potentials and Distances," 9th Annual Workshop on Duplicating, 2001.

[3] W. W. L. Fung, I. Sham, G. Yuan, and T. M. Aamodt. Dynamic warp formation and scheduling for efficient GPU control flow. In Proc. 40th IEEE/ACM Int'l Symp. on Microarchitecture, 2007.

[4] S. Ryoo, C. I. Rodrigues, S. S. Baghsorkhi, S. S. Stone, D. B. Kirk, and W. W. Hwu. Optimization principles and application performance evaluation of a multithreaded GPU using CUDA. In Proc. 13th ACM SIGPLAN Symp. on Principles and Practice of Parallel Programming, pages 73 - 82, 2008.

[5] Z. S. Hakura and A. Gupta. The design and analysis of a cache architecture for texture mapping. In Proc. 24th Int'l Symp. on Computer Architecture, pages 108 - 120, 1997.

[6] W. W. L. Fung, I. Sham, G. Yuan, and T. M. Aamodt. Dynamic warp formation and scheduling for efficient GPU control flow. In Proc. 40th IEEE/ACM Int'l Symp. on Microarchitecture, 2007.

[7] NVIDIA Corporation. NVIDIA CUDA Programming Guide, 1.1 edition, 2007.

[8] P. Harish and P. J. Narayanan. Accelerating Large Graph Algorithms on the GPU Using CUDA. In HiPC, pages 197 - 208, 2007.

[9] M. Giles. Jacobi iteration for a Laplace discretisation on a 3D structured grid. http://people.maths.ox.ac.uk/~gilesm/hpc/NVIDIA/laplace3d.pdf.

[10] J. Michalakes and M. Vachharajani. GPU acceleration of numerical weather prediction. IPDPS 2008: IEEE Int'l Symp. on Parallel and Distributed Processing, pages 1 - 7, April 2008.

저자 소개

이 종 복(정회원)

- 1964년 8월 20일생.
- 1988년 서울대 컴퓨터공학과 졸업.
- 1998년 동 대학 전기공학부 졸업(공박).
- 1998년~2000년 LG반도체 선임연구원.
- 2000년~현재 한성대 전자정보공학과 교수
- Tel : 02-760-449
- Fax : 02-760-4435
- E-mail : jblee@hansung.ac.kr

<관심분야 : 마이크로 프로세서, 멀티코어 프로세서, 텐서 프로세서 유닛>

※ 본 연구는 한성대학교 교내학술연구비 지원과제임.