GENERAL PURPOSE COMPUTING ON GRAPHICALPROCESSING UNIT: EXTENDING PARALLEL PROCESSING

Anupam Mahajan*

MadhurChanana*

Dishant Sharma*

Kunal Sachdeva*

Abstract: T*he GPU's advanced capabilities were originally used primarily for 3D game and graphics rendering. But now those capabilities are being harnessed more broadly to accelerate computational workloads in areas such as financial modelling, cutting-edge scientific research, high computations and oil and gas exploration. However, now these GPUs are being used in accelerating many high end computation workloads for general computer applications. In this paper we have presented an overview of the concept of general purpose computing on GPUs, which combines the decision making and managing ability of CPUs and the high parallel process power of GPUs. In the end we have also listed the various applications which are being implemented on this new technology along with its future aspects.*

Keywords: *ALU, CPU, CUDA, OpenGl, GPU, GPGPU, GLSL*

*CSE Department, Dronacharya College of Engineering, Gurgaon, Haryana, India

1. INTRODUCTION:

A graphics processing unit (GPU) is a specialized electronic circuit designed to quickly manipulate and alter memory to accelerate the creation of images in a frame buffer intended for output to a display. GPUs are used in embedded systems, mobile phones, pc, workstations, and game consoles. Modern GPUs are very efficient at manipulating computer graphics, and their highly parallel structure makes them more effective than general-purpose CPUs for algorithms where processing of large blocks of data is done in parallel. A central processing unit (CPU) is the hardware within a computer that carries out the instructions of a computer program by performing the basic arithmetical, logical, and input/output operations.

Traditionally for many years, it is just used to accelerate some stages of the graphics rendering pipeline, which helps display the triangles onto screen. However, after the programmability available on this chip, GPU opens a door to developers to take advantage of its ALUs besides graphics processing. This gave birth to a combined technology known as General Purpose computation on GPU (GPGPU) which uses the high parallel processing power of GPU to perform CPU related tasks.In this paper, we will overview the concept of GPGPU and its related applications.

2. HISTORY OF GPU:

The first 3D graphics processing chip was made during the middle of 90's of last century. The first true 3D graphics started with early display controllers, known as video shifters and video address generators.In 1999 NVIDIA began to sell its GeForce 256 processor known as NV10, which for the first time supported hardware-acceleration which includes Transform and Lighting. This formed the creation of GPU. [2] With the programmability of GPU emerging in GeForce 3 by NVIDIA from 2001, GPU entered into a new high speed era. At first people can write vertex shaders which run on the hardware, and then pixel shaders were supported and Read/write memory, floating point precision, and conditional execution were supported consecutively.

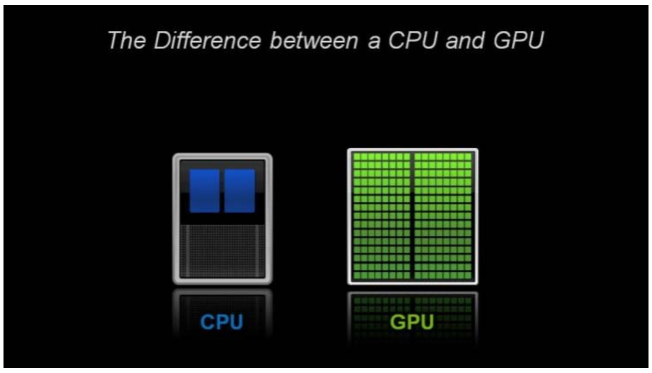
3. PARALLEL ARCHITECTURE OF GPU:

Modern GPUs have hundreds of processing cores that can be used for general-purpose computing beyond graphics rendering. Both NVIDIA and AMD provide convenient programming libraries to use their GPUs for computation or memory-intensive applications.

A GPU executes code in the SIMD fashion that shares the same code path working on multiple data at the same time. For this reason, a GPU is ideal for parallel applications requiring high memory bandwidth to access different sets of data. To make full use of massive cores in a GPU, many threads are launched and run concurrently to execute the kernel code. This means more parallelism generally produces better utilization of GPU resources. GPU kernel execution takes the following four steps:
(i) the DMA controller transfers input data from host memory to GPU memory.
(ii) A host program instructs the GPU to launch the kernel.
(iii) The GPU executes threads in parallel.
(iv) The DMA controller transfers the result data back to host memory from device memory.

4. GPU VS. CPU:

Architecturally, the CPU is contains only only few cores with lots of cache memory that can handle a few software threads at a time. In contrast, a GPU made up of hundreds of cores that can handle thousands of threads simultaneously.The ability of a GPU with hundred and more cores to process thousands of threads can accelerate some software by 100 times over a CPU alone. GPU can achieve this acceleration while being more power and cost efficient than a CPU. Following figure shows basic architecture of cpu and gpu[3].



The following independent study conducted by Keon Jang and Sangjin Han , University of Washington compares the performance of CPU and GPU with respect to three parameters i.e. Latency , Throughput, Peak. It compares a parallel RSA implementation to a serial CPU implementation. CPU used is a single core 2.6 GHz processor and GPU is NVIDIA GeForce GTX580. [5]

	Processor	512	1024	2048	4096
Latency (ms)	CPU core	0.07	0.3	2.3	14.9
	GTX580, MP	1.1	3.8	13.83	52.46
Throughput (ops/s)	CPU core	13,924	3,301	438	67
	GTX580, MP	906	263	72	19
Peak (ops/s)	CPU core	13,924	3,301	438	67
	GTX580, MP	322,167	74,732	12,044	1,661

Table 1: RSA performance with various key sizes

This study concludes that GPU outperforms CPU only in certain conditions, mostly when high parallel processing is required.

Hence , CPU still offers many advantages over GPU, CPU has a greater memory capability and is more efficient at managing and orchestrating the different jobs being performed by the computer, and making decisions on how best to handle a situation. Instead, GPUs act as highly efficient processing units , where data is offloaded from the CPU to perform repetitive calculations at a very fast rate. The processed data would then return to the CPU, which would use the results to form a solution to the problem.

Concluding that not all the functions of a CPU can be carried out by GPU, forms the basic requirement of a combined unit of CPU and GPU .i.e. GPGPU to accelerate general computer based applications instead limiting it to graphical processing.

5. GPGPU MODELS:

1. Early stage GPGPU model:

In early days, some researchers realized that GPU could be used to solve some non-graphics problems. At that time, GPU supported depth buffer and color blending operations which got some hardware acceleration. [8]So, on such kind of chips, these researchers tried to map their problems to a depth buffer operation or color blending operation.

2. GPGPU Shaders model:

After GPU chips open a programmable interface, people began to write shader codes to run on the hardware. Some researchers use vertex programs to solve some general-purpose problems, but most researchers use pixel programs to do so. Data streams are represented as textures, OpenGL Shading Language (GLSL), or High Level Shading Language (HLSL) is used to write the kernel programs to operate the data on the graphics hardware. During this stage, people still need to use graphics API such as OpenGL, Direct3D but there was still need to map the problems to a graphics rendering procedure. This made working on

GPGPUs limited to only a few capable users who knew how to use graphics APIs (OpenGL,Direct3D,GLSL).

3. Latest GPGPU model:

If people want to harness the computation power of GPU, they have to learn the graphics pipeline. To overcome these problems, NVIDIA unveiled the CompCUDA [6] .It is a parallel computing platform and programming model invented by NVIDIA. It enables dramatic increases in computing performance by harnessing the power of the graphics processing unit (GPU).

With millions of CUDA-enabled GPUs sold to date, software developers, scientists and researchers are finding broad-ranging uses for GPU computing with CUDA i.e. compute Unified Device Architecture (CUDA)[6].

PGPU was practically off-limits to those who hadn't memorized the latest graphics APIs until a group of Stanford University researchers set out to reimagine the GPU as a "streaming processor. NVIDIA knew that blazingly fast hardware had to be coupled with intuitive software and hardware tools, and invited Ian Buck to join the company and start evolving a solution to seamlessly run C on the GPU. Putting the software and hardware together, NVIDIA unveiled CUDA in 2006, the world's first solution for general-computing on GPUs. You're faced with imperatives: Improve performance. Solve a problem more quickly. Parallel processing would be faster, but the learning curve is steep .But with CUDA, you can send C, C++ and Fortran code straight to GPU, no assembly language required. Similarly, AMD provides Close To Metal (CTM) SDK and its successor AMD Stream SDK interface to help researchers and developers to use their GPU platform to accelerate the computation.

6. APPLICATIONS OF GPGPU:

Along with the development of GPGPU, we can find variety of GPGPU applications[2] coming up, including those in the fields of computational geoscience, finance, physics, which are all involved with computation-intensive, highly parallel processing tasks.Right now, the majority of GPGPU applications have been limited to scientific computing and video decoding and transcoding

A. High Performance Computing(HPC) /KEPLER GK110 :

The most frequently used application of GPGPU is in high end computing or high performance computing , mostly part of scientific experiments.HPC is very important to some large scale simulations, such as earth climate prediction, nuclear explosion simulation etc. In previous days, researchers have to turn to mainframe computer for help. However such kind of hardware is very expensive, highly unsuitable for most researchers and institutes. With the evolving of the architecture of GPU and its according software platform, GPU begins to enter HPC market NVIDIA introduced concepts like KEPLER GK110, Fermi, Tesla etc. in this domain. As the demand for high performance parallel computing increases across many areas of science, medicine, engineering, and finance, NVIDIA introduced KEPLER GK110 to meet these demands. Kepler GK110 simplifies creation of parallel programs and will further revolutionize high performance computing. Comprising 7.1 billion transistors, Kepler GK110 is not only the fastest, but also the most architecturally complex microprocessor ever built. Adding many new innovative features focused on compute performance, GK110 was designed to be a parallel processing powerhouse for Tesla® and the HPC market.[9]. Kepler GK110 was built first and foremost for Tesla, and its goal was to be the highest performing parallel computing microprocessor in the world. GK110 not only greatly exceeds the raw compute horsepower delivered by previous models, but it does so efficiently, consuming significantly less power and generating much less heat output [9].Each of the Kepler GK110 SMX units feature 192 single precision CUDA cores, and each core has fully pipelined floatingpoint and integer arithmetic logic units. It also introduced the concept of Dynamic Parallelism. With Dynamic Parallelism, the grid resolution can be determined dynamically at runtime in a data-dependent manner.

B. Computational Finance:

There is on-going work in options pricing, risk analysis, and algorithmic trading using CUDA. Major example of this is the Monte Carlo simulation on CUDA .Monte Carlo methodsare a broad class of computational algorithms that rely on repeated random sampling to obtain numerical results; i.e., by running simulations many times over in order to calculate those same probabilities heuristically just like actually playing and recording your results in a real casino situation.[12]Computational finance relies heavily on the use of Monte Carlo simulation techniques. However, Monte Carlo simulation is computationally very demanding.By implementing this method on CUDA it resulted in a speedup of 74× relative to an 8 core multiprocessor system.

C. Basic Linear Algebra Operation:

Basic linear algebra operations are the most common requirement for the general computation. With textures as the data streams, shader programs as the kernels, Bolz&Krüger provided solutions to represent sparse or densematrix and vector by textures, and then execute the computation on GPU with shaders.

D. Medical Imaging:

Medical imaging is one of the earliest applications to take advantage of GPU computing to get acceleration. The use of GPUs in this field has matured to the point that there are several medical modalities shipping with NVIDIA's Tesla GPUs now.

E. Computational Fluid Dynamics:

Several ongoing projects on Navier-Stokes models and Lattice Boltzman methods have shown very large speedups using CUDA-enabled GPUs. Generally, to simulate the fluid motion physically, we need to solve the Navier-Stokes Equations

$\nabla \cdot u = 0 \qquad \partial u \, \partial t = -(u \cdot \nabla) u + v \nabla^2 u - \nabla p + f$

By using the GPU, the performance on computation speed up achieved more than one degree of magnitude. There is also work on weather modelling and ocean modelling using GPUs.

F. Digital Signal Processing:

Researchers from the Chinese University of Hong Kong presented a SIMD algorithm that performs the convolutionbased Discrete wavelet transform (DWT) completely on a GPU, which brings significant performance gain on a normal PC without extra cost.

G. Database operation:

Researchers from UNC used GPU for database operations including relational query, conjunctive selection, and aggregation operations. For a million-scale database, GPU has the advantages over CPU in most situations.

H. Molecular Dynamics:

Christopher et. Al used GPU to accelerate the calculation of cut-off pair potentials, one of the most important computations required by many different molecular modelling applications, and the result shows 12 to 20 times faster than optimized CPU-only code.

I. Others:

Application domain for GPGPUs is being expanded rapidly. This includes many other applications like

- Seismic Exploration
- Defense
- Geographical Information System
- Bioinformatics
- CAD,CAM and CAE.[10]

7. CONCLUSION AND FUTURE WORK:

In this paper, we started by looking upon the origin of the GPU and describing its architecture and working. Then we compared CPU and GPU architecture and their performance to realize the need of the of GPGPUs not of them alone sufficed our requirements. We gave a periodic brief summary of the various GPGPU models developed earlier and the ones in use .We also viewed many applications of the GPGPUs in the current scenario and in the coming one. However, the applications of GPGPU are still restricted to scientific computations for researchers . It still needs to enter the main consumer market. GPGPU also seems to the answer to the main problem faced by many electronics developer and researchers worldwide i.e. providing high processing power at low price , Hence a cost effective production of GPGPUs is require.

REFERENCES:

[1] GPGPU. Available: http://www.gpgpu.org

[2] GPU Available: http://en.wikipedia.org/wiki/Graphics_processing_unit

[3] NVIDIA Blog: WHAT'S THE DIFFERENCE BETWEEN A CPU AND A GPU? http://blogs.nvidia.com/blog/2009/12/16/whats-the-difference-between-a-cpu-and-a-gpu/

[4]Exploring GPGPU Workloads: Characterization Methodology, Analysis and Micro architecture Evaluation Implications Nilanjan Goswami, Ramkumar Shankar,

Madhura Joshi, and Tao Li Intelligent Design of Efficient Architecture Lab (IDEAL) University of Florida, Gainesville, Florida, USA

[5] SSLShader: Cheap SSL Acceleration with Commodity Processors Keon Jang, Sangjin Han, Seungyeop Han, Sue Moon and KyoungSoo Park University of Washington.

[6] NVIDIA's Next Generation CUDATMCompute Architecture: FermiTM. http://www.nvidia.com/content/PDF/fermi_white_papers/NVIDIA_Fermi_Compute _Architecture_Whitepaper.pdf

[7] On the Energy Efficiency of Graphics Processing Units for Scientific Computing S. Huang, S. Xiao, W. Feng, Department of Computer Science, Virginia Tech {huangs, shucai, wfeng}@vt.edu (Proceedings of SIGGRAPH2006), 25(3), pp.724-734, 2006.

[8] Emerging Technology about GPGPU EnhuaWu, Youquan Liu, Faculty of Science and Technology University of Macau, Macao, China ehwu@umac.mo

[9] NVidia KEPLER GK110. Available: http://www.nvidia.com/content /PDF/kepler/NV_DS_Tesla_KCompute_Arch_May_2012_LR.pdf

[10] INDUSTRY CASE STUDIES ,Nvidia , http://www.nvidia.com/object/tesla-case-studies.html

[11] From CPU to GPU, David Robson on the processing burden being shared by GPUs http://www.scientific-computing.com/hpcforscience/feature-gpu.html

[12] OpenCL. Available: http://www.khronos.org/news/press/releases/khronos_ launches_heterogeneous_computing_initiative/

[13] E.H. Wu, Y.Q. Liu, "General-purpose computation on GPU", Journal of Computer Aided Design and Computer Graphics, 16(5), May, 2004. Pp.601-612

[14] L. Seiler, D. Carmean, et.al., "Larrabee: A many-core X86 architure for visual computing," ACM Transactions on Graphics (Proceedings of SIGGRAPH2008), 27(3), 2008.

[15] CUDA 5 Available :http://www.nvidia.com/object/cuda_home_new.html

[16] J. Krüger and R. Westermann, "Linear algebra operators for GPU implementation of numerical algorithms", ACM Transactions on Graphics (Proceedings of SIGGRAPH2003), 22(3), pp. 908-916, 2003.

[17] NVIDIA Compute PTX: Parallel Thread Execution ISA, Version 1.4, 2009.

[18] J. Nickolls, I. Buck, M. Garland, and K. Skadron, Scalable Parallel Programming with CUDA, Queue 6, 2 (Mar. 2008), 40-53.

[19] M. Schatz, C. Trapnell, A. Delcher, and A. Varshney, High-throughput Sequence Alignment using Graphics Processing Units, BMC Bioinformatics,8(1): 474, 2007.

[20] A. Bakhoda, G. Yuan, W. Fung, H. Wong, and T. Aamodt, Analyzing CUDA Workloads using a Detailed GPU Simulator, ISPASS, 2009.