INTERNATIONAL JOURNAL OF SCIENTIFIC PROGRESS AND RESEARCH (IJSPR) ISSN: 2349-4689
Volume-24, Number - 01, 2016

A Survey on GPU Computing

Parag Pachouri, Prof. Manaswini Panigrahi

Department of Computer Science and Engineering, IES Group of Institute, Bhopal, India

Abstract - In this paper we study about graphics processing unit (GPU) along with its computational power and their application areas. Although these units are manually designed for the graphics application but we can try there computation power for non- graphics application also. GPU has highly parallel processing power, low cost of computation and less time utilization, thus it gives better result of performance in energy efficiency. These GPU expansion properties for excessive computation of similar set of instructions are played a vital role in reducing CPU overhead. GPU architecture has many advantages over CPU architecture as it provides high parallelism, intensive computation and higher throughput. It consists of number of hardware threads that execute the programs in a SIMD (Single Instruction Multiple Data streams) manner hence its high performance the GPU can be an alternate to the CPU in parallel computing environment.

Keyword - GPU, GUP Computing.

I. INTRODUCTION

Inventions and discovery in the field of Technology and Science has always endue human ease with reduce human efforts. These goals have always stimulated the researchers to enhance their various dimensions in technology and science. Recently the computer technology plays a valuable role when it comes to intense computation to solve a special or complex problem. GPUs have been used over a large area as a component of complex graphics application. Now a day’s by the use of these graphics units many ways are opened in the field of cluster computing environment as its high performance computing units and due to its computation power efficiency [1]. Before when CPU was only the single unit for computation many of the task had to wait for their completion, slowly the idea of clustering came into market which not only increased the computation performance but also provide ease for the complex computing. Clustering of processor have been proven beneficial for complex computation but along with its benefits there are few undesirable features like high amount of investment, costly for usage when there is less complex computation.GPU invention proved that it is boon in the field of graphics application but also for the other intense computational task. Over few years GPU has developed from a special purpose processor in the field of into a parallel programming with additional functionality [2].

The rest of this paper is organized as follows: Section II presents some related work in the field of GPU evolution. Section III illustrates the comparison between CPU and GPU. Section IV discusses some limitations in GPU computing. Section V gives a description about GPU Computing including CUDA and OpenCL. In Section VI we discussed about advantages of GPU and Section VII concludes this paper.

II. EVOLUTION OF GPU’S

In the early era of 1980's the term “GPU” was unified frame buffers. These are the TTL logic chips depend on the CPU and could only draw shapes to raster display. The IBM Professional Graphics Controller was the first 2D/3D video cards for the Personal Computers. The PGA used an on-board Intel 8088 microprocessor to take processing over the all video related tasks such as drawing and colouring filled in polygons shapes, and freeing up the CPU for video processing. The PGA's processor are marked a crucial step in the GPU evolution [3]. In 1987, more additional features were being added in the previous GPUs, such as Vertex lighting, Shaded Solids, etc for filling a colour in the polygons figures. There was much support for sharing of computation with the CPU. In the late 1980's, Silicon Graphics Inc. (SGI) emerged as a high performance computer graphics hardware and software company. In 1989, SGI introduce OpenGL, which are platform independent and supported 2D/3D application programming interfaces. OpenGL support has also become a complex part of the design of modern graphics hardware.

A. Generation I:

In 1993, SGI introduce graphics pipeline for graphic processing. There were distinct boards and logic chip for various stages of pipeline but firstly relied on the CPU. In each stage the flow of data is fixed. In the mid of 1990's,SGI cards was available for workstations while 3D graphics hardware such as NVIDIA TNT, Voodoo, ATI and so on are providing for the consumers. [3]. A combination of cheap hardware with games such as Quake and Doom really drove the gaming industry which adopt GPU. Even with deep pipelinesin the early GPUs which only output one pixel per clock cycle, this meaning CPUs could still send more triangles to the GPU than it could handle. This lead to adding more pipelines in parallel to the GPU, so multiple pixels could be processed in parallel each clock cycle.

B. Generation II:

In 1996, the 3DFX Voodoo was introduced, it was considered as one of the first 3D game cards. It offers only 3D acceleration. It operand over Peripheral Component Interconnect (PCI) which consists of millions of transistors. PCI is a local computer bus used for attaching hardware components to a computer. The PCI bus supports the functions found on a processor bus but in the standard format it is independent of any particular processor's native bus. It is a parallel bus, synchronous to a single bus clock. It needs 4 MB of 64-bits DRAM for 50 MHz clock cycle. Voodoo provides text mapping, buffering and so on while CPU provides vertex transformations.

C. Generation III:

In 1999, first Pipeline card was released. With the introduction of NVIDIA GeForce256 (also known as NV10) and ATI Redeon 7500, was the first GPU which are available at the consumer level. Before 1999, the term GPU didn't exist really but the launch of NVIDIA GeForce256, it exists. The GeForce256 consist of up to 23 million transistors, 32 MB of 128 bits DRAM and it operates on 120 MHz clock cycle. This generation of graphics card used Accelerated Graphic Port (AGP) instead of using PCI buses. They offer features in hardware such as multi texturing, light maps, geometry transformations and many more. The first pipeline in hardware of this generation are known as Fixed function pipeline, because in these pipeline once a programmer sent a graphic data, this graphic data could not be modified. With these graphic cards, the combinations of GPU hardware with computer are really started to takeoff in the market. It is much faster but the fixed function pipeline problems are arises in it. To overcome these problems OpenGL and DirectX Application programming Interfaces were implemented in hardware as a new features in graphics APL's[5].

D. Generation IV:

In this generation the programmable pipeline was introduce in GPU architecture. In 2001, NVIDIA released the GeForce 3, which consists of 57 million transistor, 64 MB of 128 bit DRAM and it operate on the 120 MHz's. GeForce 3 provides the ability to the programmers to program a parts in non-programmable pipeline previously instead of sending all the graphics description data to the GPU and it simply flow through the fixed pipeline, the programmer can now send this data along with vertex programs, these vertex programs is known as shaders. These shadder are operating on the data while in the pipeline. These shader programs were small kernels, which are written in shader languages. In this generation there are many other popular graphics card are ATI Radeon 8500, and Microsoft's X-Box.

E. Generation V:

In the year 2002, the first fully programmable graphics cards are released in the markets which are NVIDIA GeForce FX, ATI Radeon 9700. The NVIDIA GeForce FX which consists of 80 million transistors, 128 MB of 128-bit DRAM, and it operated at 400 MHz clock cycle. These cards allowed for per-pixel operations with the programmable shaders, and it also allows for the limited user to specified the mapping of input-output data operations. Separate, dedicated hardware were allocated for pixel shader and vertex shader processing. In 2003, the first GPU computing started with the introduction of DirectX 9, by taking advantage of the programmability in the GPU hardware, but for non graphics also. These GPU's support fully floating point and provides advanced texture processing.

F. Generation VI:

The GPU hardware technology accelerated in this generation. In 2004, the GeForce 6 and Radeon X800 were released with its additional features. These cards are used first time in PCI sexpress buses. The GeForce 6 was introducing in this generation by NVIDIA. NIVDIA GeForce 6 which consists of 146 million transistor, 256 MB of 256 bit of DRAM and it operated at 500 MHz clock cycle. This graphic cards increased the GPU memory and provide texture accessing. [10]. When viewed as a graphics pipeline, the GPU contains a programmable vertex engine, a programmable fragment engine, a texture engine, and a data write engine. When we view the alternative as a processor for non-graphics applications, a GPU can be seen as a large amount of programmable floating-point power and memory bandwidth that can be exploited for compute-intensive applications completely unrelated to computer graphics. A trend towards GPU programmability was starting to form.

G. Generation VII:

In 2006, the introduction of NVIDIA's GeForce 8 series in GPU evolution which exposing the GPU as massively parallel processor. The NVIDIA GeForce 8800[6] architecture was the first unified, programmable shaders. GeForce 8800 is also known as G80 .It is fully programmable unified processors which are called a Streaming Multiprocessor that handled vertex, pixel, and geometry computation. It also introduced a new geometry shader by adding more programmability when it combined with the vertex shader and pixel shaders. The GeForce 8 consist of 681 million transistors, 768 MB of 384-bit DRAM, and the core clock operated at 600MHz.After this generation GPU power was the new programming language CUDA by NVIDIA for NVIDIA only, ATI Stream for ATI cards and DirectX 10 for Microsoft Windows only were introduced.

H. Generation VIII:

In 2010, NVIDIA introduced first Fermi GPU for GPGPU computing with additional features such as high cache memory, concurrent kernel for execution, unified memory address space, high performance. The GTX480 Fermi had a total of 480 CUDA [4]. The first Fermi cards which consist of 3 billion transistors, 1.5 GB of 384- bit DRAM, and it operated at 700MHz .After the year 2010, NVIDIA again released new version of Fermi-based gaming card which are called as GTX580,it offering a higher memory bandwidth than in the previous version. The GPU hardware evolution thus far has gone from an extremely specific, single core, fixed function hardware pipeline implementation just for graphics rendering, to a set of highly parallel and programmable cores for multiple general purpose computation. Now, the architecture of many-core GPUs are starting to look like multi-core, general purpose CPUs. In 2000, AMD announced their Fusion of CPU and GPUs [9].

III. COMPARISON BETWEEN CPU AND GPU ARCHITECTURE

Modern CPUs have evolved towards parallel processing, implementing the Multiple Instruction Multiple Data stream architecture. Most of their part is reserved for the control units and cache and leaving a small area for the numerical computations. This is the reason a CPU performs such different tasks that having advanced cache and control mechanisms are the only way to achieve an overall its performance, while the GPU has a Single Instruction Multiple Data stream architecture that can be well represented by the PRAM. The main goal of GPU architecture is to achieve high performance through massive parallelism. Contrary to the CPU, the die surface of the GPU is mostly occupied by ALUs and a minimum region is reserved for the control and cache. Efficient algorithms are designed for GPUs which speedup over CPU implementations [8].

This difference in architecture which discussed in figure 1 has a direct consequence, the GPU is much more restrictive than the CPU but it is much more powerful. Latest GPU architectures such as NVIDIA’s Fermi has added a significant degree of flexibility by incorporating a cache for handling irregular memory accesses and by improving the performance of the operations.

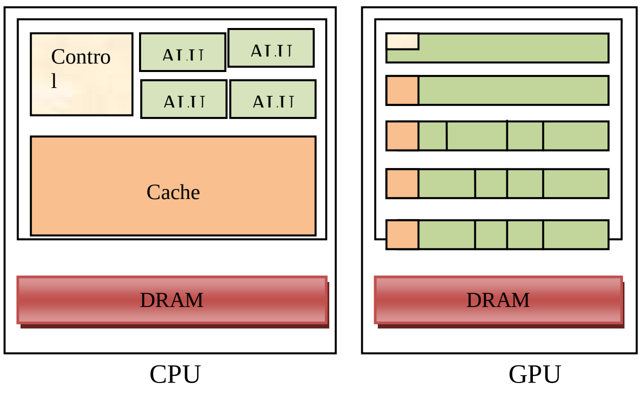

Figure 1. Architecture of CPU and GPU

However, this flexibility is still far from the one found in CPUs. Indeed there is a trade-off between flexibility and computing power. Actual CPUs maintain a balance between computational power and general purpose functionality while GPUs aim at massive parallel in arithmetic computations, with introducing many restrictions. Some of these restrictions are overcome at the implementation phase while some others must be treated when the problem is being parallelized. It is always a good idea to follow a strategy for designing a parallel algorithm. Some other comparisons are discussed in below Table 1

Table 1. Comparison between CPU and GPU

CPU	GPU
CPU use the task parallelism	GPU use the data parallelism
Multiple task mapped into multiple threads	GPU works on the Single instruction and multiple data stream
Task runs different instruction	Same instruction on the different data
CPU optimised for high performance on the sequential codes	GPU optimised for higher arithmetic intensity for parallel manner
It has fine branching granularity	It access to onboard memory faster.

IV. GPU LIMITATION

Along with the GPU’s advantages there is some limitation in the GPU which should be studied while designing GPU application. There are many limitations [10] which can be directly or indirectly affect on the performance or quality of the application are as follows.

1. The memory bandwidth between the CPU and GPU is limited.

2. Read back from the GPU memory to main memory is costly.

3. It has Small instruction streams and it mainly designed for graphics application.

4. No bitwise operation coding are available in GPU.

5. GPU doesn’t support big number operations.

6. GPU doesn’t optimize for serial operations.

V. GPU COMPUTING

GPU computation has provided a huge edge over the CPU with respect of their computation speed. Hence it is one of the most interesting areas for the research in the field of technology and science. GPU is a graphical processing unit which enables to run with high definitions graphics on the

PC, which is very demanded in modern computing. Like the Central Processing Unit, it is a single-chip processor. However, as shown in Fig. 1, the GPU consist of hundreds of cores as compared to the CPUs. The primary goal of the GPU is to compute 3D functions. Because these types of calculations are very heavy on the CPU, thus the GPU can help the computer to run more efficiently. Though, GPU came into existence for graphical purpose, now it evolved into computing, precision and performance. The evolution of GPU has been towards a better floating point performance. In 2006, NVIDIA introduced its massively parallel architecture for GPU programming called CUDA [7]. The CUDA architecture has a number of processor cores that work together to bunch of the data set given in the application. The model for GPU computing is to used both a CPU and GPU together in a different co-processing computing model. The sequential part of the application can runs on the CPU and the computational part is running by the GPU. From the user's point of view, the application is become faster due to the use of the better performance of the GPU to improve its own performance. GPUs are processors that can operate in parallel by running in a single kernel on many records in the stream at once. Stream is defined as a simply a set of records that require similar computation. Streams provide the data parallelism. Multiple inputs and output at the same time, but we cannot have memory types which is not only readable but also writeable. This enables the GPU to have data parallel processing. This results in more computational power for the GPU. But this computational capability was not used in the previous evolution stages of the GPU.

There are multiple Software Development Kits and APIs are available for the programming in GPUs for general purpose computation. For example NVIDIA CUDA, ATI Stream SDK, OpenCL, HMPP, and PGI Accelerator [11].

VI. ADVANTAGES OF GPU

Modern APIs enable efficient and portable heterogeneous computing. This includes enabling the use of the best processor for the task, ability to balance workload across system resources and to off load heavy parallel computation to the GUP. GPU compute tangible advantage in the real world application including to reduced cost, improved performance and user experience, and improved energy efficiency and so on[8].

A Reduce power Consumption:

The architectural characteristics of series of GPUs enable computation of many parallel workloads much more efficient than alternative processors.GPU Compute accelerated applications can therefore benefit by consuming less energy, which translate into longer battery life.

B. Improved performance and user experience:

Where raw performance in the target , the computation of heavy parallel workloads can also be efficiently accelerated through the use of GPU .This translate in increase frame rate or the ability to carry out more works in the same time .

C. Portability, programmability, flexibility:

Heterogeneous compute APIs such as OpenCL is design for concurrency. They allow the developer to migrate some of the load from the CPU to the GPU, or to distribute it between processor in order to enable better load balancing across system resources.

D. Reduction of cost:

System designers may be influenced by various cost, portability, reliability concern when considering migration functionality from dedicated hardware accelerators to software which leverage the CPU or GPU subsystem. This approach is made viable and compelling due to additional computational power provided by the GPU.

VII. CONCLUSION

From the above study we conclude that GPU is the better alternative for intensive computational task. Employing GPU no doubt to increase the speed of execution and also frees the CPU from the load which perform serial executable tasks. The Combination of CPU and GPU in many applications can provide high performance having low cost as compared to using of cluster of CPUs. To program GPU there many API's are used many programming language .These languages are very efficient and easily understandable programming language. The most popular programming language for the programme is the extension of C language. CUDA programming help in the designing of heterogeneous computational code, this code is the combination of serial and parallel execution task performed by the both CPU and GPU unit respectively. Many applications have gained more benefits from the use of GPU in their computation. There are many more applications under study where researchers are trying to deploy GPU units to gain the better results.

REFERENCES

[1] Sung Ye Kim,Jeremy Bottleson,et al, "Power Efficient MapReduce Workload Accerlation using Intergated-GPU", IEEE First International Conference on Big Data Computing Service and Application,2015,pp 162-169.

[2] John D. Owens, Mike Houston, David Lucbke, et al., " GPU Computing Proceedings", IEEE, 2008, 96(5): 879-899.

[3] Chris McClanahan, "History and Evolution of GPU Architecture",IEEE,2010

[4] Jayshree Ghorpade, Jitendra Parande,et al. “GPGPU PROCESSING IN CUDA ARCHITECTURE”, Advanced Computing: An International Journal (ACIJ), January 2012, vol.3, No.1.

[5] Vincent Boyer , Didier El Baz , “Recent Advances on GPU Computing in Operations Research”, IEEE 27th International Symposium on Parallel & Distributed Processing Workshops and PhD Forum,2013.

[6] John D. Owens, Mike Houston, et al, “GPU Computing”, IEEE, May 2008,vol. 96,No. 5.

[7] Jianbin Fang, Ana Lucia Varbanescu and Henk Sips, “A Comprehensive Performance Comparison of CUDA and OpenCL”,IEEE,2009.

[8] Stephen W. Keckler , William J. Dally,et al, “GPUS and the Future of Parallel Computing”, IEEE Computer Society,2011

[9] Paramjeet kaur et al, “A Survey on CUDA”, (IJCSIT) International Journal of Computer Science and Information Technologies, 2014, Vol. 5 (2) ,pp 2210-2214.

[10] R. Vuduc, A. Chandramowlishwaran, J. W. Choi, M. E. Guney, and A. Shringarpure, “On the Limits of GPU Acceleration,” in Proc. USENIX Wkshp. Hot Topics in Parallelism (HotPar), Berkeley, CA, USA, June2010.

[11] Volodymyr V. Kindratenko, Jeremy J. Enos, et al, “GPU Clusters for High Performance Computing”, National Center for Supercomputing Apllication,University of Illinious at Urbana-Champaign,USA,