Volume 4, Issue 4, April 2014 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

Parallel Processing Unit with MIMD Architecture

Anil Rajput*
CSA PG Nodel College Sehore, India

Bhushan Ishwarkar, Sarita Kadhao
Unique College Bhopal, India

Abstract*— Face is significant method to identifying person. Face Recognition is process which is base on graphic & image processing. GPU is Graphic processing unit. GPU has become an integral part of today's mainstream computing systems, becoming more widely used in demanding consumer applications and high-performance computing. This article describes the rapid evolution of GPU architectures with Flynn's Taxonomy how it can exploit for Face Recognize. This article also describe the graphics processors to massively parallel many-core multi threading multiprocessor multiprocessors, recent developments in GPU computing architectures, and how the enthusiastic adoption of CPU + GPU co-processing is accelerating parallel applications.

General Terms

Face Recognize, Uniform Memory Access.

***Keywords*— GPU, Flynn Taxonomy, UMA, PPU, SISD, SIMD, MISD & MIMD**

I. INTRODUCTION

Face Recognition is part of biometrics system. The graphics processing unit (GPU) has become an essential part of today's conventional computing systems. In few years, there has been a marked raise in the performance and capabilities of GPUs. Graphics Processing Units is powerful, programmable, and highly parallel computing, which are increasingly targeting for general-purpose computing applications. The modern GPU is not only a powerful graphics engine but also a highly parallel programmable processor featuring peak arithmetic and memory bandwidth that substantial as CPU.

The GPU's has capability computing broad range of computation, complex problems solve & especially for high range of graphical data. This effort in general purpose computing on the GPU, also known as GPU computing, which proposed GPU as an alternative to traditional microprocessors in high-performance computer systems of the future. This article describe the background, hardware, and programming model for GPU computing, summarize the state of the art in tools and techniques, and how it is suitable for the face Recognize, Flynn's Taxonomy with GPU.

II. APPLICATION OF FACE RECOGNITION

Face recognition is application of digital image processing & analysis, has recently received significant attention. As the technologies develop, it demands more robust technique or synchronization of many to achieve peak level. Even though current machine recognition systems have reached a certain level of maturity, their success is limited by the conditions imposed by many real applications. Some of the applications are listed in Table 1. In other words, current systems are still far away from the capability of the human perception system.

TABLE I: SOME APPLICATION OF FACE RECOGNITION[10]

Areas	Specific applications
Entertainment	Video game, virtual reality, training programs, Human-robot-interaction, human-computer-interaction.
Smart cards	Drivers' licenses, entitlement programs, Immigration, national ID, passports, voter registration, Welfare fraud.
Information security	TV Parental control, personal device logon, desktop logon, Application security, database security, file encryption, Intranet security, internet access, medical records.
Law enforcement and surveillance	Advanced video surveillance, CCTV control, Portal control, postevent analysis, Secure trading terminals.

III. WHY GPU?

Today's GPUs rapidly solve large problems having substantial inherent parallelism by using hundreds of parallel processor cores executing tens of thousands of parallel threads. They're now the most pervasive massively parallel processing platform ever available, as well as the most cost effective. GPU is a multi-core multithreaded multiprocessor that excels at both graphics and computing applications.

Exposing high-definition graphics scenes is a problem with fabulous inherent parallelism. A graphics programmer writes a single-thread program that draws one pixel. The GPU runs multiple instances of this thread in parallel are known as multi-thread parallel computing. It draws multiple pixels in parallel at a time. As Software scalability rapidly increase GPU parallelism, simultaneously in hardware increasing transistor density with performance.

A. GPU Evaluation

The first GPU is Video graphics array (VGA) controllers generated 2D graphics displays for PCs to accelerate graphical user interfaces. In the early 1990s, there were no GPUs. NVIDIA produce GPU with various technologies, which perform the parallel computation. The evaluation of GPU Produced by NIVIDIA is list in table 2. Recently NVIDIA produce acceleration base GPU Tesla K40 GPU Accelerator with configuration Listed in table 3. The NVIDIA GeForce 8800 GTX (top) features 16 streaming multiprocessors of 8 thread (stream) processors in Fig 2.

Day by day hundred of application of GPU is growing some are list below that are published by NVIDIA cooperation in Mar 2014.[7]

1. *Research: higher education and supercomputing*
   - Computational chemistry and biology
   - Numerical analytics
   - Physics
2. *Defense and intelligence*
3. *Computational finance*
4. *Manufacturing: CAD and CAE*
   - Computational fluid dynamics
   - Computational structural mechanics
   - Computer aided design
   - Electronic design automation
5. *Media and entertainment*
   - Animation, modeling and rendering
   - Color correction and grain management
   - Compositing, finishing and effects
   - Editing
   - Encoding and digital distribution
   - On-air graphics
   - On-set, review and stereo tools
   - Simulation
   - Weather and climate forecasting
6. *Oil and gas*

TABLE II: GPU EVALUATION REPORT [3]

NVIDIA GPU technology development.[4]				
Date	**Product**	**Transistors**	**CUDA cores**	**Technology**
1997	RIVA 128	3 million	—	3D graphics accelerator
1999	GeForce 256	25 million	—	First GPU, programmed with DX7 and OpenGL
2001	GeForce 3	60 million	—	First programmable shader GPU, programmed with DX8 and OpenGL
2002	GeForce FX	125 million	—	32-bit floating-point (FP) programmable GPU with Cg programs, DX9, and OpenGL
2004	GeForce 6800	222 million	—	32-bit FP programmable scalable GPU, GPGPU Cg programs, DX9, and OpenGL
2006	GeForce 8800	681 million	128	First unified graphics and computing GPU, programmed in C with CUDA
2007	Tesla T8, C870	681 million	128	First GPU computing system programmed in C with CUDA
2008	GeForce GTX 280	1.4 billion	240	Unified graphics and computing GPU, IEEE FP, CUDA C, OpenCL, and DirectCompute

2008	Tesla T10, S1070	1.4 billion	240	GPU computing clusters, 64-bit IEEE FP, 4-Gbyte memory, CUDA C, and OpenCL
2009	Fermi	3.0 billion	512	GPU computing architecture, IEEE 754-2008 FP, 64-bit unified addressing, caching, ECC memory, CUDA C, C++, OpenCL, and DirectCompute

TABLE III: LATEST NEW PRODUCTS OF COMPARE WITH PREVIOUS VERSION TESLA SERIES[9]

Features	**Tesla K40**	**Tesla K20X**	**Tesla K20**	**Tesla K10**
Number and Type of GPU	1 Kepler GK110B	1 Kepler GK110		2 Kepler GK104s
Peak double precision floating	1.43 Tflops	1.31 Tflops	1.17 Tflops	0.19 Tflops
Peak single precision floating point performance	4.29 Tflops	3.95 Tflops	3.52 Tflops	4.58 Tflops
Memory bandwidth (ECC off)	288 GB/sec	250 GB/sec	208 GB/sec	320 GB/sec
Memory size (GDDR5)	12 GB	6 GB	5 GB	8 GB
CUDA cores	2880	2688	2496	2 x 1536

B. Architecture of GPU

GPU is best amazing solution with multiprocessing processors operate on multithread operation on multicore system for graphical processing. Since 2003, researchers have proposed GPU initially designed for computing 3D functions, which includes lighting effects, object transformations, and 3D motion [1].

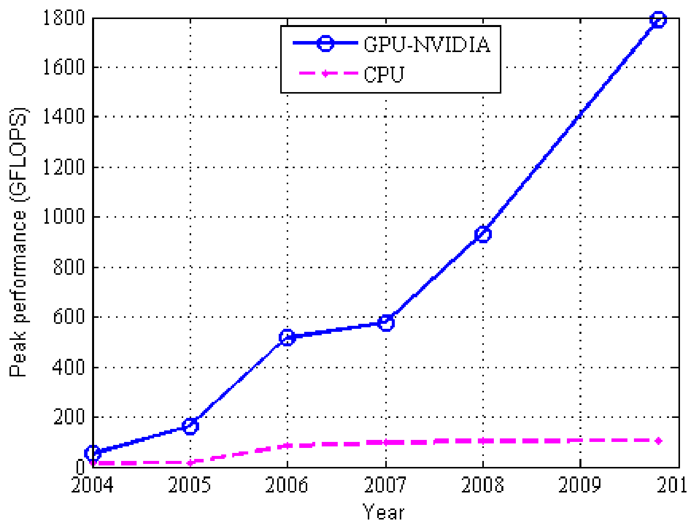

Fig 1: CPU versus GPU evolution

GPU is used for intensive computing that has multiple cores with a software layer that allows parallel computing. Contrary to CPU, the state of the art of GPU shows that the performances in term of execution time are in constant evolution. This study extended the study of [4] to estimate the performances of GPU and CPU in 2010. If observed the ratio between the GPU and CPU, GPU has maximum 15 more operating frequencies than CPU. This means that image processing through GPU is faster than on CPU. The computing speed is measured by GFLOPS (Giga FLoating point Per Second) which is equal to 109 floating-point arithmetic operations per second. The performance evolution graph of CPU versus GPU shown in fig. 1.

GPU is stress on throughput rather than latency computing in parallel so it has large demands than the CPU. Accordingly, the architecture of the GPU has progressed in a different direction than that of the CPU. Computing more than one element at the same time is data parallelism. To execute such a pipeline, a CPU would take a single element (or group of elements) and process the first stage in the pipeline, then the next stage, and so on. The CPU divides the pipeline in time, applying all resources in the processor to each stage in turn. The GPU divides the resources of the processor among the different stages, such that the pipeline is divided in space, not time. The part of the processor working on one stage feeds its output directly into a different part that works on the next stage. GPU may take thousands of cycles from start to finish for a graphics operation. Task and data parallelism across stages delivered high throughput. The latency is long for any given operation.

Load balancing is the major disadvantage of the GPU task-parallel pipeline. The performance of the GPU pipeline is dependent on its slowest stage. Consider the situation in which the vertex program is complex and the fragment program is simple, overall throughput is dependent on the performance of the vertex program. In the Latest programmable stages, the instruction set of the fragment and vertex programs were quite different, so these stages were separate. However, as both the vertex and fragment programs became more fully featured, and as the instruction sets converged, GPU architects reconsidered a strict task-parallel pipeline in favour of a unified shader architecture, in which all programmable units in the pipeline share a single programmable hardware unit. The programmable units now divide their time among vertex work, fragment work, and geometry work (with the new DirectX 10 geometry shaders). Much of the pipeline is still task-parallel. These units can exploit both task and data parallelism. As the programmable parts of the pipeline are responsible for more and more computation within the graphics pipeline, the architecture of the GPU is migrating from a strict pipelined task-parallel architecture to one that is increasingly built around a single unified data-parallel programmable unit.

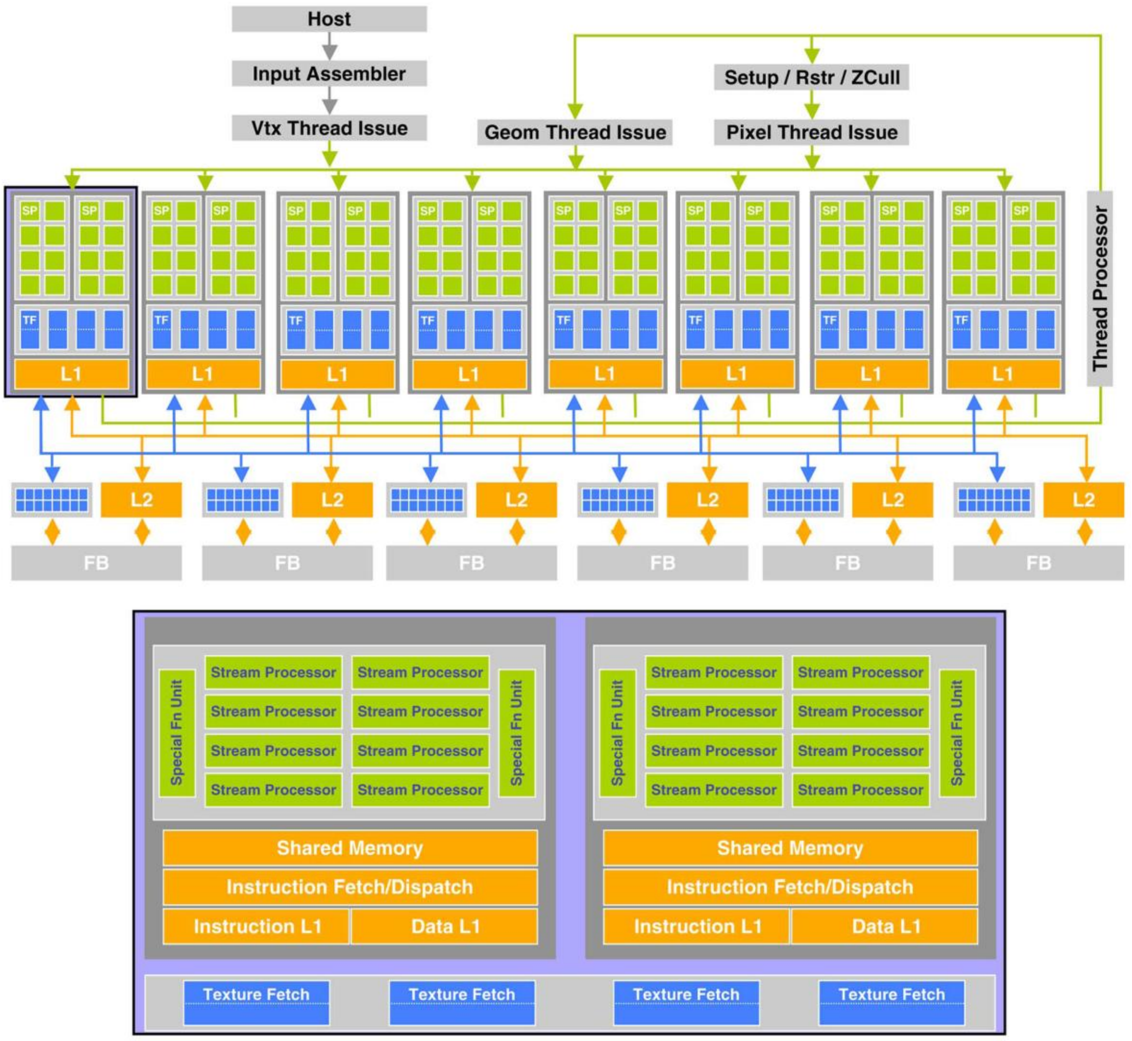

Fig. 2 NVIDIA build architectures with unified, massively parallel programmable units at their cores. The NVIDIA GeForce 8800 GTX (top) features 16 streaming multiprocessors of 8 thread (stream) processors each. One pair of streaming multiprocessorsis shown below; each contains shared instruction and data caches, control logic, a 16 kB shared memory, eight stream processors, and two special function units. (Diagram courtesy of NVIDIA.)[5]

C. FLYNN'S TAXONOMY

The requirement of present & future is need of high Computational processor that can operate on high resolution of graphics throughput with the less effort of time. Single chip CPU Architecture is unable to fill this requirement.

Flynn's taxonomy in use since 1966, who classify the different way of parallel computers in four way on the basis of instruction & data with two state single & multiple, as SISD(Single Instruction Single Data), SIMD(Single Instruction Multiple Data), MISD(Multiple Instruction Single Data) & MIMD(Multiple Instruction Multiple Data).

SISD is present single chip CPU, which is serial processor, while SIMD, MISD & MIMD are pure parallel processor which work simultaneous multiple process in a cycle. In SISD, one instruction stream can be operated on single stream data in one clock cycle. It has deterministic execution. In SIMD, one instruction stream can be operated on multiple stream data in one clock cycle. It has synchronous & deterministic execution. It is best suited for specialized problem characterized by a high degree of regularity, such as graphics & image processing. It has two varieties i.e. processor arrays & vector pipelines. GPU is designed on the base of SIMD architecture. GPU computing, demanding applications with substantial parallelism increasingly use the massively parallel computing capabilities of GPUs to achieve superior performance and efficiency. So, GPU with its feature can applicable & better for face recognition than CPU.

IV. OWN CONCEPT

GPU is Parallel computing graphical processing unit, which perform various graphical operations in parallel. Graphical data is suitable for parallel computation because the same type of operation perform on every pixel to form graphical output. So most GPU hardware are implement by using a very restrictive multi-threaded SIMD-based execution model [8].

If parallel computing multi processors are designed with the MIMD architecture, in which each processor contain multiple cores that are executing multiple independent threads in parallel. Each processor has some amount of local memory for calculation & storing purpose. There is share memory that can share (UMA) by every processor which is used to hold the code of various process & data (Just like RAM in CPU). Share Memory has segments for each processor so that processor can take same time to access any memory word in the system. This can be called as PPU(Parallel Processing Unit)which can perform parallel operation on general data as well as graphical data means there is no need of CPU+GPU(GPGPU produce by NVIDIA Tesla K40 which contain GPU for intensive work & for general CPU). This architecture can run the program & application that are run on CPU; there is no special need of changing the software environment. If observe one processing unit of PPU MIMD then it’s just like stand alone CPU but it contain thousand of cores like GPU. In other word it can say its hybrid version of GPU & CPU. The question is that how it do so?

Operating system manages all things, suppose there are n processing unit in PPU MIMD and m number of processes. consider the cases if m=n then allocate the each process to each processor, each process are divide into the thread each thread are executing on cores in parallel in processor. If some process complete its task then that processing unit can be jump to the other process that are newly inserted or that processor behave like help worker of the processing unit(processor) which are still in processing. The process complete its task can run the some threads of process that are still running that are independent. Same way it can handle other situation related to number of process & number of processing unit.

V. CONCLUSIONS

The GPU can be used as general purpose programming which perform massive parallelism the term GPU is not suitable according to working & architecture, GPU can be call as PPU. According to the algorithm given in section 5th, for massively parallel on any type data PPU MIMD architecture is best. Various numbers of algorithms can run in parallel on huge amount of data take decision according to the priority of algorithm & average of calculating matching percentage to make the perfect face detection. This type of architecture is useful for face recognize.

VI. FUTURE WORK

Design the architecture of PPU with MIMD, Design algorithm of memory management, scheduling ….. etc

REFERENCES

[1] K. Moreland and E. Angel, The FFT on a GPU, Proc. of Graphics Hardware, pp. 112-136, 2003.

[2] John D. Owens, Mike Houston, David Luebke, Simon Green, John E. Stone, and James C. Phillips, “GPU Computing” Vol. 96, 0018-9219 ©2008 IEEE No. 5, May 2008 | Proceedings of the IEEE.

[3] John Nickolls, William J. Dally NVIDIA, “THE GPU COMPUTING ERA”, Published by the IEEE Computer Society 0272-1732/10 © 2010

[4] E Stephanie and K. Ronan, Etat de l’art des processeurs du marche dans le contexte du codec H.264, Technical report Projet TransMedi@, Institut Telecom/Telecom-Bretagne, Departement Informatique, Ver. 1.0, May 2009.

[5] A white paper by Peter N. Glaskowsky, “NVIDIA’s Fermi: The First Complete GPU Computing Architecture”, Prepared under contract with NVIDIA Corporation Copyright ©2009.

[6] NVIDIA, “Nvidia compute unified device architecture”, http://www.nvidia.com/object/cuda.html, 2008.

[7] “GPU-ACCELERATED APPLICATIONS”, © 2014 NVIDIA Corporation. All rights reserved.

[8] Henry G. Dietz and B. Dalton Young, “MIMD Interpretation on a GPU”, University of Kentucky, Electrical and Computer Engineering.

[9] *www.nvidia.com/object/tesla-servers.html*

[10] W. Zhao R. Chellappa, P. J. Phillips, A. Rosenfield, “Face Recognition: A Literature Survey”, © 2003 ACM 0360-0300/03/1200-0399

[11] *en.wikipedia.org/wiki/MIMD*