IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 11, November 2016

Review Paper on Research Trends in Computing

Sharad Srivastava[1], **Shubham Jain**[2], **Manjula R**[3]

School of Computer Science and Engineering, VIT University, Vellore, India[1, 2]

Associate Professor (Guide), School of Computer Science and Engineering, VIT University, Vellore, India[3]

Abstract: A processor is responsible for several thousand calculations that are required to run a system and this capability, efficiency of a processor to do so is based on its type of architecture, the sole purpose of which is to find a balance between the applications that run on the system and the maximum performance attainable in implementing such an architecture. Additionally there are physical limitations to processor architecture which include how many transistors can be placed on a chip or the number of interconnection one can make on the chip and how much heat a chip can dissipate. All these factors contribute towards the computing capabilities of the processor, In regard to which this paper undertakes a critical review of the current challenges and growth in processor technology and design decisions for future implementation. This paper also compares the various architecture alternatives to traditional processor in form of multi core architecture and parallel processing (as an implementation in processing) and the viability of using GPUs as an alternative to CPU processing.

Keywords: Processor, Multicore Architecture, Parallel processing, CPU, GPU.

1. INTRODUCTION

With reference to [1] we can say that a Processor figuratively is the convergence of a Central Processing Unit on to either single or multiple integrated circuits. In terms of factual units as is stated in [2] a processor is an electronic device with programmable capabilities and can carry out multiple tasks hence is multipurpose. Thus it is a device that accepts binary input that is executed based on a set of instructions to produce an expected output. The modern counterpart to the traditional single core processor which was designed to provide maximum efficiency generally known as multi core processor has been at the helm of modern day computing since its inception over 15 years ago. With the creation of Power4 by IBM which was a result of 4 years of intense effort a new era of processing was born in the field of computer processing. This new multi core processor was a VLSI chip with two 64 bit microprocessors with over 170 million transistors. Since then multicore architecture has become a norm in development of computer processors as single core processors reach their physical potential of complexity and efficiency, however just as a transition became necessary from single core processors that operated at high frequency to multi core processors which operate at low frequency due to the increasing thermal design power (TDP) so as to maintain an optimal performance, similarly the development of multicore processors will reach a barrier of its physical capabilities and will require a new approach to tackle it.

This power barrier could lead to a gradual increase in dark silicon for the future multicore processors which in turn would lead to part of the chip to operate at low frequencies at all-time causing loss of performance. This requires a considerable time be given to not only improve upon the technology but also work on the possible barriers that will eventually occur in the development of future processors.

The paper deals with a possible solution to the computing problem through GPUs, in form of GPU acceleration and its parallel computing abilities as GPUs are developed with multiple cores that can execute multiple instruction sets in parallel thus assisting the main processing unit in its operation.

In this paper we discuss the technology that has led to the current generation of processors and processing architecture alongside the current research trends in computing and future challenges for the same. Furthermore we look at the most promising architecture scheme and trends. Multicore processors implementation in the same architectural class may vary depending upon the targeted application and the provided power budget and a dynamic approach to the problem could provide us with a better solution and mitigate several of the expected problem.

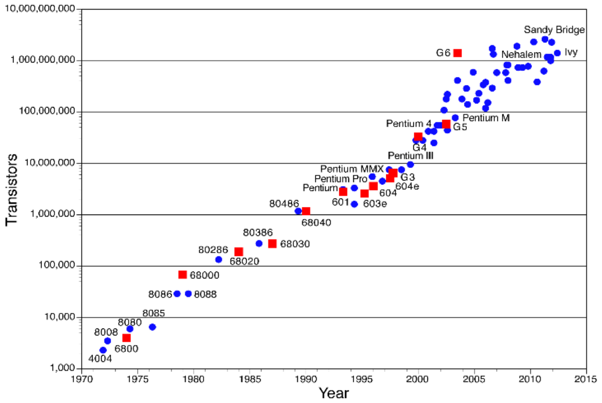
2. LITERATURE SURVEY

The field of processing architecture is constantly under the development phase due to the ever increasing required computing power, and one of the major step forward in this field was the transition from single core processors to multicore processors. Once it became clear that including multiple processors on a board had its limitation, we had to find alternatives to the problem of computing that was not just more efficient but required less real estate.

This led to the development of multicore architecture due to our computing needs, which is aptly defined by the quote “necessity is the mother of invention”. In the era of single core computers the only option to increase the computational capacity of processors was to have higher clock speeds, increased thermal capacity and better power dissipation to list a few. These options although useful had their own limitations.

DOI 10.17148/IJARCCE.2016.51152

In April 1965, Gordon Moore published an article in Electronics magazine titled “Cramming more components onto integrated circuits” [3]. He predicted that the number of transistors on a chip would double every 12 months, which later on was revised to 18 months became the driving force behind long term development goals and integrated circuits. Known to the world as Moore’s law this observation stood the test of time but has not been as prominent of late and predictions are being made that it might end by 2025.



2 (a) Moore’s Law

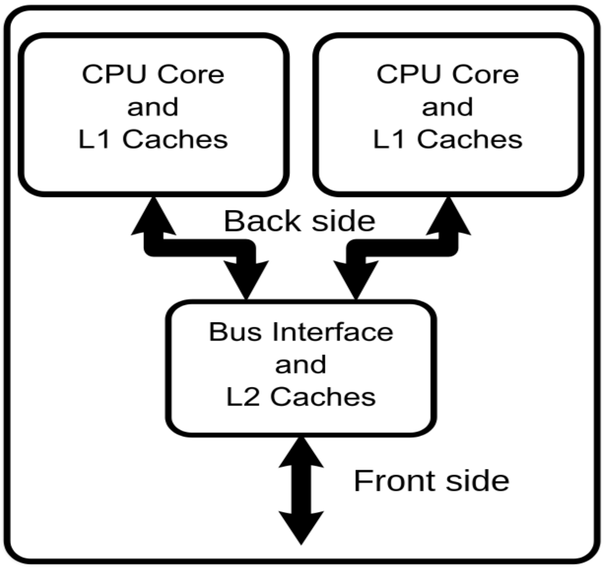
Another attempt to increase computational power was by transistor power reduction based on Dennard scaling and allowed for greater clock frequencies without increasing overall circuit power consumption. Dennard scaling relates to Moore’s law as it states that performance per watt of computing is growing exponentially at almost the same rate as Moore’s law.

Similarly ILP or Instruction Level Parallelism was introduced to run multiple instructions in parallel and in conjunction with Dennard’s scaling and Moore’s law provided exponential growth to computing power. However the fact that parallelism has hit an impenetrable wall of logical complexity that makes any further improvement incremental at best and have actually been the cause of regression in execution speeds. [4].

A major factor that has played into the transition from single core architecture to multicore architecture is the fact that we have hit the power wall in terms of power consumption and the power boundary given to developers to stay within. One way to increase the computing power was to develop aggressive circuits and use deep pipelining in conjunction with other techniques but the available power saving techniques were not able to cope with the increasing demand of power.

Additionally many of these improvements such as increased pipelining and increased cache size do not have high returns when considering the fact that it has a parallel increase in the required area and power consumption. Still this has allowed processors on the lower end of the spectrum to close the overall performance gap while providing an affordable tradeoff in area and power. Considering the limitations that were a part of voltage supply scaling, threshold scaling, and clock frequency scaling, with an addition of increasing design complexity, an alternative had to be developed. This idea is presented in [5] which states that this barrier to performance might have been what led to the development of multi-core which enabled the computer industry to keep up with the increasing requirement of computing power that could not be satisfied with the current hardware. In the next part of our paper we discuss more about multi-core systems and their benefits to the computer industry.

3. OVERVIEW OF MULTICORE ARCHITECTURE



3 (a) A generic dual-core processor

By definition a Multi-Core processor is inclusive of a single processor that includes several cores. The cores in turn are functional units that are made up of computation units and caches. These multiple cores then work in sync to replicate the performance of a single faster and efficient processor. As is presented in [6] it is not the individual cores of a multi-core system that generate the power instead it is the sheer number of such cores that in unison generate the high performance that is found in multicore systems. Based on this idea it can be deduced that the throughput of a single-core in comparison to multi-core is based on how they execute the instructions. In the case of single-core design we have to the face the restriction that all the tasks have to be executed by the same core, which is carried out by allotting a fixed runtime to each instruction. But the disadvantage of such a process is that if one instruction lags all others do too. Whereas in a multicore environment the presence of multiple cores solves this issues as it can run several instructions in parallel and not have to deal with one causing disruption in the execution of others.

As the market for computing is ever increasing and never satisfying, microprocessors have always been designed keeping performance in mind, this is clearly represented by the fact that the first microprocessor had 2300 (Intel 4004) transistors for its computational power in

DOI 10.17148/IJARCCE.2016.51152

comparison to 10,000,000,000 (SPARK M7) by ORACLE so very clearly the count of transistors has increased at a very high rate as predicted by Moore's law however just increasing the amount of functional units in a processor is not useful due to lack of parallelism in typical instruction. The solution to this would be multicore. The purpose of multicore here is to duplicate the entire processor core on the same chip to run two (or more) simultaneous threads of execution on one chip. Parallelism addresses the issue of power while maintaining the performance attaining higher data throughput with lower voltage and frequency.
This was one of the major driving force behind the movement to multi core architecture. The advancement in chip fabrication technology and integrated circuit processing technology has made it possible to integrate over 1 billion transistors on modern day chips, but with the benefit of lower power dissipation and power density.

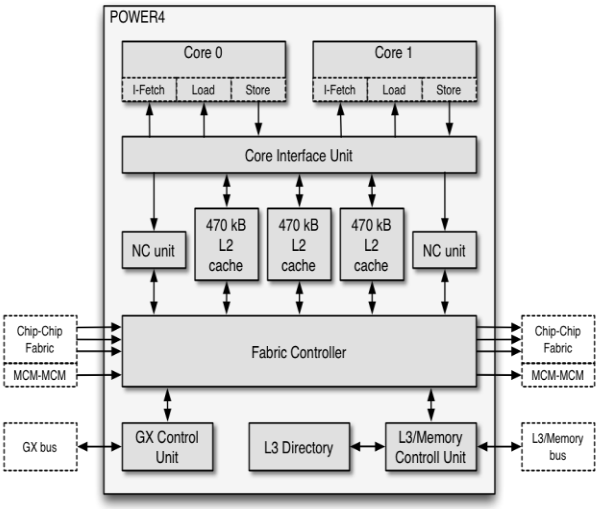
Semiconductor industry that had its roots in performance as the major design objective, has shifted to considering factors like chip fabrication costs, fault tolerance, power efficiency and heat dissipation as their major objective, In turn leading to the development of multi core architecture.
One major benefit of multi-core architecture is that it allows for IP reuse. Multi-core has created an architectural environment where processor core complexity is not required for higher performance. Instead, an alternative approach to achieve higher performance was by adding cores which required only minor changes from previous generation designs. To prevent over complicating and designing complex architecture requires restraint on the part of developers to not optimize an already design compliant core any further. With this development strategy it was finally possible to provide higher performance with lower cost and in a single unit.

3.1 MULTICORE EVOLUTION

There has been a lot of development in terms of the first multicore processor to what we have today. To understand this evolution more clearly we will discuss a few of the most prominent multicore processors/architectures to have been developed.

3.1.1 IBM 100 – POWER 4

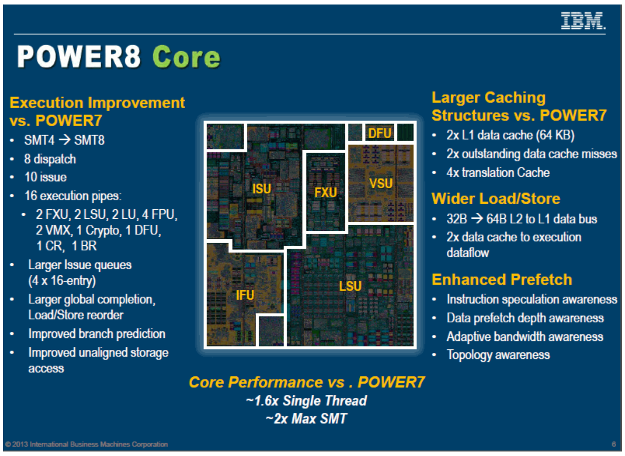
The Power 4 was the first of the multicore processors to have been developed and was a creation of developers at IBM. The Power4 implemented a superscalar micro-architecture through high-frequency speculative out-of-order execution using eight independent execution units. They were: two floating-point units (FP1-2), two load-store units (LD1-2), two fixed-point units (FX1-2), a branch unit (BR), and a conditional-register unit (CR) [8].
It had a Maximum CPU clock speed between 1.1 GHz and 1.9 GHz and a Min Feature size between 180nm to 130nm. It was based on the PowerPC instruction set. The POWER4 was created with a unified L2 cache divided into 3 parts, each with its own L2 controller. It also had a 32 bit off chip L3 cache. The Power4 consisted of 174 million transistors.



3.1.1 (a) Logic schema of POWER4 processor

3.1.2 IBM POWER 8

In September 2013 IBM announced a new 12 core processor named Power8 which was to be manufactured in 22nm and was released in 2014. POWER 8 was created to have a 4 GHz, 12 core processor that is supported by a total of 96 threads that are used for parallel execution. This build is further supported by a 96 MB L3 cache and 128 MB L4 cache with a new extension bus to replace the old one. Power 8 was able to get a 60% performance gain as compared to the Power7+ core.



POWER8 Core

Execution Improvement vs. POWER7
- SMT4 → SMT8
- 8 dispatch
- 10 issue
- 16 execution pipes:
  - 2 FXU, 2 LSU, 2 LU, 4 FPU, 2 VMX, 1 Crypto, 1 DFU, 1 CR, 1 BR
- Larger Issue queues (4 x 16-entry)
- Larger global completion, Load/Store reorder
- Improved branch prediction
- Improved unaligned storage access

Larger Caching Structures vs. POWER7
- 2x L1 data cache (64 KB)
- 2x outstanding data cache misses
- 4x translation Cache

Wider Load/Store
- 32B → 64B L2 to L1 data bus
- 2x data cache to execution dataflow

Enhanced Prefetch
- Instruction speculation awareness
- Data prefetch depth awareness
- Adaptive bandwidth awareness
- Topology awareness

Core Performance vs . POWER7
~1.6x Single Thread
~2x Max SMT

© 2013 International Business Machines Corporation 6

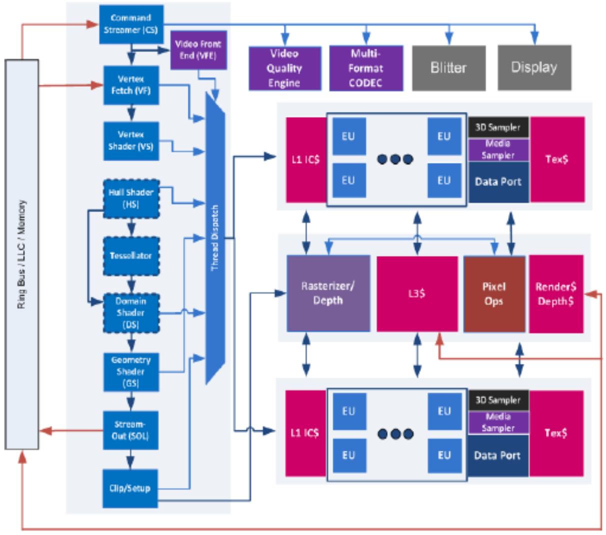
3.1.2 (a) Power8 architecture

3.1.3 INTEL HASWELL AND BROADWELL

• HASWELL

It is manufactured using 22nm process technology and 3D FinFET transistors. The die size of the standard quad core "Core i7" desktop chip including 8 MB of shared L3 cache, and integrates 1.4 billion transistors. Compared to its Sandy Bridge predecessors, the Haswell quad core chips offer a lower TDP (84 W) and an increased maximum turbo frequency of up to 3.9 GHz [9]. It has 64 KB (32 KB Instruction + 32 KB Data) L1 cache and 256 KB L2 cache per core.

DOI 10.17148/IJARCCE.2016.51152



3.1.3(a) Haswell Architecture

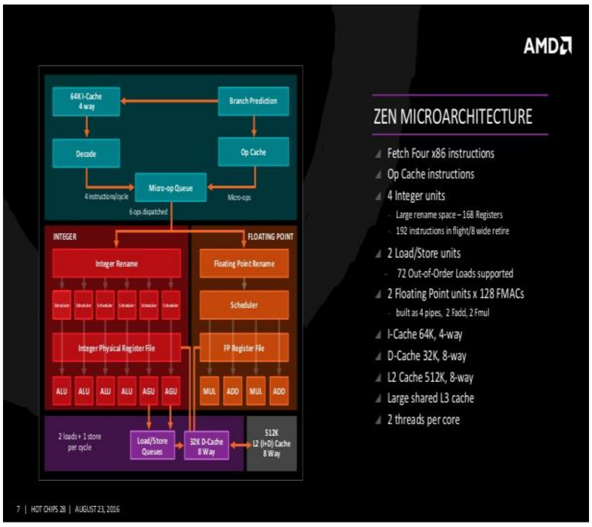
• BROADWELL

Broadwell is the codename for 14 nanometer die shrink of Haswell. As stated it is manufactured using 14 nm process technology and has cores ranging from 2 (on the lower end) to as high as 24(in Xeon). It has a 64 KB per core L1 cache, a 256 KB per core L2 cache, 2-6 MB shared L3 cache and 128 MB of eDRAM L4 cache.

3.1.4 AMD ZEN

The latest in computer processing architecture from AMD codenamed ZEN is poised to release in early 2017 and will using a 14 nm FinFET process, which is supposed to be more energy efficient and have a higher IPC. The introduction of SMT will allow each core to run 2 threads. The cache system has also been redesigned, making the L1 cache write-back. The ZEN processors will be making the use of the AM4 socket, allowing DDR4 support.

Zen is based on AMD64 instruction set and will have anywhere between 2 to 32 cores depending on the required usage.



ZEN MICROARCHITECTURE

- Fetch Four x86 instructions
- Op Cache instructions
- 4 Integer units
  - Large rename space – 168 Registers
  - 192 instructions in flight/8 wide retire
- 2 Load/Store units
  - 72 Out-of-Order Loads supported
- 2 Floating Point units x 128 FMACs
  - built as 4 pipes, 2 Fadd, 2 Fmul
- I-Cache 64K, 4-way
- D-Cache 32K, 8-way
- L2 Cache 512K, 8-way
- Large shared L3 cache
- 2 threads per core

7 | HOT CHIPS 28 | AUGUST 23, 2016

3.1.4(a) ZEN Architecture

3.2 SOFTWARE MULTIPROCESSING

When the computing world hit a wall in terms of capabilities of a single computing unit we found alternates such as multiple CPUs, multiple cores or even multiple chips on a system. But as is explained by the authors in [16] these were all limited to the hardware part of the system which were generally divided into two parts, if the entire implementation was based on the same architecture, it was homogeneous multicore otherwise heterogeneous multicore.

The next step was to increase performance on the software components of a system, this was done with the creation of Asymmetric Multi-Processing (AMP) and Symmetric Multi-Processing (SMP) which are explained in detail below.

3.2.1 Asymmetric Multi-Processing (AMP)

An Asymmetric Multi-Processor system was developed on the concept of multiple cores but was created with the freedom to have different architectural design. Considering that these were separate cores that may have different architectures they had to be developed to have separate address spaces to prevent errors and could run separate instances of OS on each of them. They were also provided a line of communication between themselves. AMP is mostly used when different core architectures are optimal for specific activities – like a DSP and an MCU.

3.2.2 Symmetric Multi-Processing (SMP)

Similar to Asymmetric Multi-Processing, Symmetric Multi-Processing systems were also based on multiple cores but with the major difference that they all had same architectural design amongst them. Such a system was created with a shared memory space as well as the same OS on all of them. The author in [7] explains that these systems have a communication system that is implemented through shared memory but via OS APIs. Generally Symmetric Multi-Processing is used when embedded application require more computation power to manage its workload.

4. ALTERNATIVES TO CPU COMPUTING

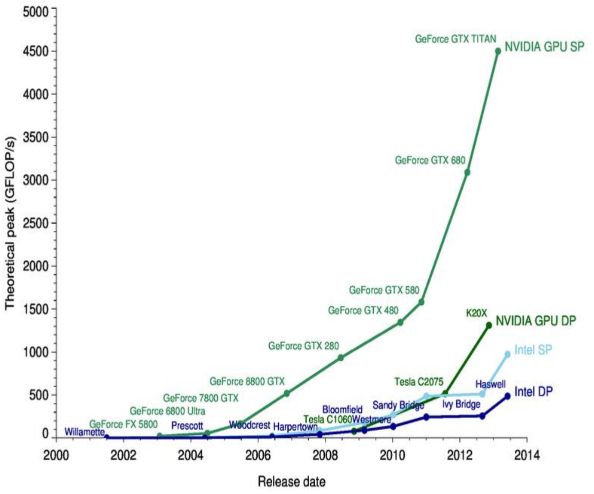
4.1 GPUs

A graphics processing unit (GPU), also referred to as visual processing unit (VPU), is a specialized electronic circuit designed to efficiently and rapidly manipulate and alter memory to accelerate the creation of images in a frame buffer intended for output to a display [10].

As for the use of GPUs in place of CPUs, it provides an interesting option as GPUs are traditionally used to handle computations for computer graphics and to use them to perform computation in applications traditionally handled by the CPU could be highly effective as multiple GPUs can be connected together further parallelizing the already parallel nature of graphical processing.

Additionally, even a single GPU-CPU framework can provide advantages that multiple CPUs cannot provide due to the specialization in each chip Essentially, a GPGPU pipeline works as a parallel processing environment between a GPU and a CPU which works on set of data as if it were a graphical image. While GPUs do operate on a lower frequency level they generally have a higher number of cores in comparison to a traditional CPU thus allowing better and faster computing. Transforming data into graphical form and then using the GPU to "look" at it and analyze it can result in profound speedup. The GPU has always been a processor with ample computational resources. In the recent times however the most important development has been the realization that computation to the programmer to be used in places where the need is of high computational processing and massive parallelism.

4.1.1 CPUs vs GPUs



During the early stages in the life of the GPU the direction of flow of data was from the CPU to the GPU which was primarily a graphics driver but with the growth in the sector of computing, it became useful for a GPU to hold complex data structures which it passed to the CPU. The CPU in turn would analyze this data. Considering the fact that a GPU has access to draw operations it can analyze data in the form of images more efficiently in comparison to a CPU as it has to go through every pixel slowly, the reason for it being that the speed of accessing memory is slower for the CPU which has a larger pool of memory whereas the GPU has a smaller but much more sophisticated and faster memory. The transmission of generated data as an image or a format that the GPU can understand is an effective and fast way to analyze data. When considering a GPGPU we have to consider the advantage that it creates a duplex transfer between the CPU and GPU and considering the rate of data transfer is considerable it produces a multiplier effect on the whole as is presented in [11]. These pipelines are thus very efficient in dealing with large amount of data.

4.1.2 GPU Advantages and Disadvantages

Normally a GPU contains thousands of cores to CPU'S multiple cores. So assuming a pipeline working in CPU (that should process large number of elements). In CPU after a task is completed the data is sent to next stage of pipeline and so it divides the pipeline in time frames however in GPU the resources are divided into various cores, so the output from one core can be fed directly to another core. This is good because of 2 reasons namely:

1) First, the hardware in any given stage could exploit data parallelism within that stage, processing multiple elements at the same time [12]. (Suppose after computing data in let's say core A, that data could now be sent to various other cores directly). This is the reason why GPU can meet large computing needs of graphics processing.

2) Secondly, each stage's hardware could be customized with special-purpose hardware for its given task, allowing substantially greater compute and area efficiency over a general-purpose solution.

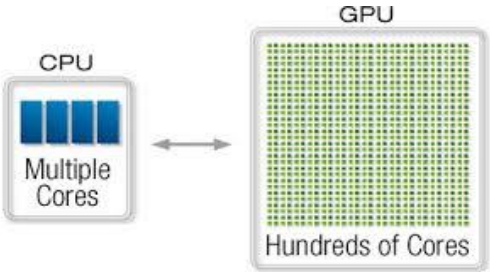
3) NVIDIA GPU has about 30multiprocessor each multiprocessor can launch up to 8 threads.

The major disadvantage of the GPU task-parallel pipeline is load balancing. Like any pipeline, the performance of the GPU pipeline is dependent on its slowest stage [13]. If the input forming (normally in GPU the input is the vertices that must be mapped to screen) is complex and the fragment program (used for producing the color the data is obtained from vertices and relative color is mapped to screen) is simple, the resulting overall throughput depends on the performance of the vertex program.

4.1.3 GPU Applications

1) **Parallelism**

CPU can power up only small or light weighted applications like PowerPoint& basic video games. However, GPUs are used to increase computer efficiencies has limited core whereas GPU has 50X more cores than CPU so it helps in compiling code faster through parallelism. GPU are mainly used in image processing or high end research where a lot of calculations takes place. GPU with its large number of cores makes large calculations faster (high parallelism). Normally CPU is left with computation part (light weight work) whereas GPU is left with heavy duty work such as 3D image processing (heavy duty since a lot of matrices need to evaluated which can only happen in several cores of GPU) General Purpose Graphical Processing Unit (GPGPU) uses gp for computing



4.1.3 (a) Cores on a CPU vs GPU

Copyright to IJARCCE
DOI 10.17148/IJARCCE.2016.51152

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

Normally GPU is used for image related applications, however GPGPU is used for computation (to leverage) large number of GPU cores.

2) GPGPU

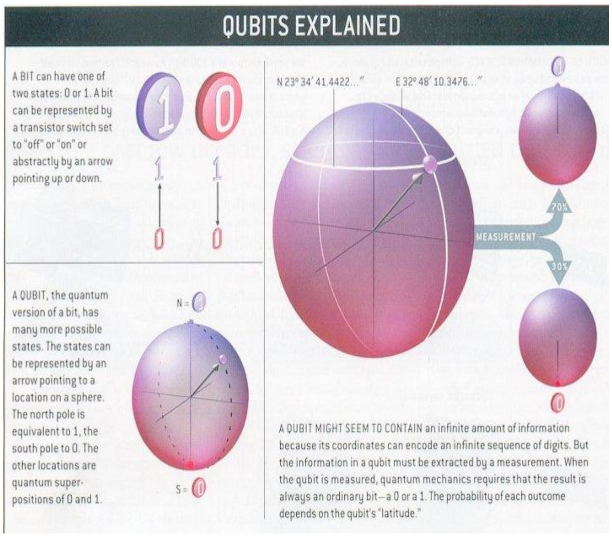
The benefit of this system is presented in [14] where the authors explain how in a GPU-accelerated system there is a performance increase because the heavy computations are handled by the GPU and the remaining is handled by the CPU. The GPU runs faster owing to its several cores. GPGPU are extensively used in applications having: large datasets, high parallelism and high arithmetic intensity

3) Deep Learning Using GPUs

Deep Learning is based on convolution neural networks (CNN). CNN is type of feed forward method inspired by neurons. Convolution Neural Network (CNN) is used for deriving a result from a series of inputs using feed forward algorithm. Normally CNN is used for computer vision. It therefore needs heavy computation capacities which only GPU could offer. Deep learning is largely used in computer vision problems, for example to determine whether shown face is, we find several features in a face. These features are then passed across a neural network and a binary vector is computed to find the features. GPU are being used to train deep leering network to reduce the error obtained in image classification. Another example of CNN is text and semantic analysis, Data from variety of formats and sources come up and a particular analysis could be done using deep learning. It needs GPU working in GPGPU mode (means GPU alongside CPU). Example includes doing sentiment analysis from tweets, analyzing for accurate whether prediction is by analyzing a number of tweets of the people present in the vicinity. Other examples of GPU usage may include better diagnosis using machine leering can be trained to perform analysis on a particular report and based on previous reports it has used as training data (by using machine learning algorithm) it may provide better results for patients.

4.2 Quantum Computing

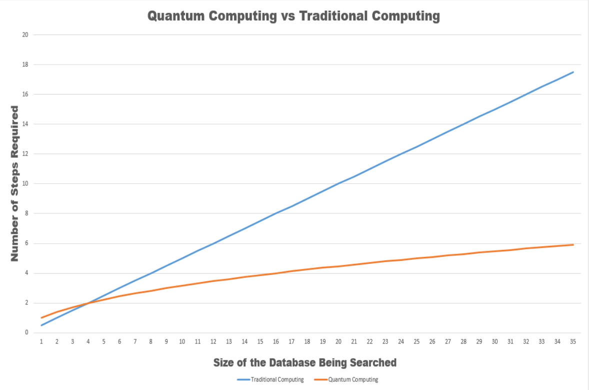
The problem with traditional computing architectures is the fact that they are dependent largely on the number of transistors that can be placed on a chip, and as Moore's law advances the size of the transistors gets smaller. Eventually though we are going to hit the physical barrier as to how small a transistor can be made thus halting the growth of computing architecture based on transistors. This has lead people to consider quantum computing as a possible alternative to traditional computing. As is explained in [15] by the authors, In the world of physics Quantum theory explains and observes the world in terms of atoms and subatomic particles. When implemented to the world of computing this concept of atomic particles translated to bits called Qubits or Quantum bits instead of the regular '0' and '1' that we deal with in the world of computers. These theoretically can exist in any state, which can range from '0', '1', both '0' and '1' and anywhere between them.


QUBITS EXPLAINED
A BIT can have one of two states: 0 or 1. A bit can be represented by a transistor switch set to "off" or "on" or abstractly by an arrow pointing up or down.
1
0
1
0
1
0
A QUBIT, the quantum version of a bit, has many more possible states. The states can be represented by an arrow pointing to a location on a sphere. The north pole is equivalent to 1, the south pole to 0. The other locations are quantum superpositions of 0 and 1.
N = 1
S = 0
N 23° 34′ 41.4422..."
E 32° 48′ 10.3476..."
MEASUREMENT
70%
30%
1
0
A QUBIT MIGHT SEEM TO CONTAIN an infinite amount of information because its coordinates can encode an infinite sequence of digits. But the information in a qubit must be extracted by a measurement. When the qubit is measured, quantum mechanics requires that the result is always an ordinary bit—a 0 or a 1. The probability of each outcome depends on the qubit's "latitude."

4.2 (a) Qubits

In quantum computing each qubit is either a photon or electron. The basic property of a qubit is that it may be able to spin and take a different shape to represent 1 or 0 as is explained in figure 4.2(a) presented by authors of [17]. Quantum computers are also known as probabilistic computers. Qubits can spin upwards or downwards and hence may take value 0 or 1, when passed through a magnetic field. So basically each digit in classical computing model may represent 2 bits in probabilistic model In classical computing model 2 bits can take up 4 values namely 00,11,01,10.In quantum computing each digit(in classical model) can be represented by 2 bits (spin up or spin down).So thereby we have 2^4 =16 digits in quantum computing. Generalizing the above scenario for every n digits in classical model can be represented 2^N digits in quantum computing.

Qubits can take spins only during Hall Effect (when magnetic field is applied) but after computation the elements take up one shape either spin up or spin down. So the increase in computational capacity is only during computation however during final result it ends up in either of 2 shapes. Now the 2 shapes (spin up or spin down) are determined through probabilistic models (for e.g. spin up can have 67% chance of occurring and spin down can have 33% of chance).



4.2 (b) Quantum Computing vs Traditional Computing

Copyright to IJARCCE

DOI 10.17148/IJARCCE.2016.51152

So we basically find that probabilistic computing is beneficial for certain applications however (cryptography for example), for applications such as reading, videos, images it is not useful because the models do not give a clear answer (there is a certain probability that model chosen may be incorrect).

5. SUMMARY

We are entering a period of dramatic change as direct performance-scaling is almost nearing its limit and this creates an exciting time for system architects, to whom it provides a challenge to improve and maintain the trend of the historic advances in system performance. Their desire to not just explore but also create more novel architectures keeps the computing industry on edge thus ensuring the constant development in the field. This is the beginning of the long-awaited phase of computing where we have systems that are capable of solving even the most complex of problems yet there are still problems that are out of reach. We need to take advantage of the reuse in multi-core designs and the emerging trends in computing.

The world of Computer Science in terms of Academics has for a long time has been creating base technologies as well as people trained on relevant problems pulling the computing industry forward along with themselves and has to continue to play a unique role in ensuring the continued success of the computing technology by addressing challenges associated with not just multi-core designs but also look for new alternatives that are better. Designers also must ensure and enable the practice of IP re-use in their design as well as the task of validation to guarantee the success of future design and not just on the test floor but also in the field.

The next decade in computing is going to witness huge changes in how our computing architecture is made and the laws that govern it. The development will be driven by the ever increasing requirement of computational power as we try to find answers to questions larger than us and in this regard high performance computing in the form of Quantum Computing may just be the answer to solve such computational needs.

As we step into the age of concepts like “internet of things”, “augmented reality”, and “Artificial Intelligence” our need for better computing architecture increases and the hardware has to scale up at a similar pace. An example of one such solution is using GPU for computational photography, computer vision problems. That said the future of computing is going to be very different from what we have seen.

REFERENCES

[1] Osborne, Adam (1980). An Introduction to Microcomputers. Volume 1: Basic Concepts (2nd ed.). Berkeley, California: Osborne-McGraw Hill. ISBN 0-931988-34-9.

[2] Krishna Kant Microprocessors And Microcontrollers: Architecture Programming And System Design, PHI Learning Pvt. Ltd., 2007 ISBN 81-203-3191-5, page 61, describing the iAPX 432.

[3] Gordon E. Moore, Cramming More Components onto Integrated Circuits. Electronics, April 19, 1965.

[4] Future of computing - Part 3: The ILP Wall and pipelines, Russell Fish -February 14, 2012.

[5] Jeff Parkhurst, John Darringer, Bill Grundmann : "From Single Core to Multi-Core Preparing for a new exponential ".

[6] D. Geer, “Chip Makers Turn to Multicore Processors,” Computer, vol. 38, pp. 11-13, 05, 2005.

[7] Colin Walls : "Multicore basics: AMP and SMP", http://www.embedded.com/

[8] “Power4”: Wikipedia.

[9] Christian Märtin : "embedded world Conference 2014 Multicore Processors: Challenges, Opportunities, Emerging Trends"

[10] https://en.wikipedia.org/wiki/Graphics_processing_unit

[11] https://en.wikipedia.org/wiki/Generalpurpose_computing_on_graphics_processing_units

[12] John D. Owens, Mike Houston, David Luebke, Simon Green, John E. Stone, James C. Phillips : "GPU Computing"

[13] John D. Owens, Mike Houston, David Luebke, Simon Green, John E. Stone, James C. Phillips : "GPU Computing"

[14] https://www.nvidia.com/object/what-is-gpu-computing.html

[15] http://www.explainthatstuff.com/quantum-computing.html

[16] http://www.embedded.com/design/mcus-processors-and socs/4429496/Multicore-basics

[17] https://universe-review.ca/R13-11-QuantumComputing.html

DOI 10.17148/IJARCCE.2016.51152