ISSN : 2347 - 8446 (Online)
ISSN : 2347 - 9817 (Print)
Vol. 2, Issue 3 (July - Sept. 2014)
International Journal of Advanced Research in Computer Science & Technology (IJARCST 2014)

Multicore Processor, Parallelism and Their Performance Analysis

[I]Rakhee Chhibber, [II]Dr. R.B.Garg
[I]Research Scholar, MEWAR University, Chittorgarh
[II]Former Professor, Delhi School of Professional Studies & Research (Affiliated to GGSIP Uni., Delhi)

Abstract
Multicore Central Processing Units (CPU) are becoming the standard for the current era of processors through the significant level of performance that CPUs offer. This includes multiple multicore architectures, different level of parallelism, different levels of performance, and with the variety of architectures, it becomes necessary to compare multicore architectures to make sure that the performance aligns itself with the expected specifications. This paper provides an overview to multi-core processors, multi-core processor parallelism and performance measurement for multi-core Central Processing Units (CPUs).

Keywords
Multi-core processors, multi-core CPUs, performance measurement, parallelism

Introduction

Understanding the behavior and architecture of a multi-core processor is necessary to proficiently analyze its performance [Chandramowlishwaran10]. This section defines a single core processor, multi-core processors, the need of multi core processor. The next section introduces different types of parallelism that impact multi-core processor performance. The subsequent sections apply these concepts in a performance analysis context, introducing techniques for performance measurement and analytical modeling. This paper assumes that the analyst is examining the performance of a given application on a multi-core processor, or different types of multi-core processors, seeking to quantify and understand why certain performance characteristics are observed. The First Section of this paper will give a brief introduction about the different types of CPUs depending upon the number of cores present in it.

Types of CPU Based on Number of Cores

Single Core CPU The single-core CPU utilizes one core inside the processor. This was the very first type of CPU, and today, is not used in many machines. The Figure 1 will illustrate the single core CPU chip.

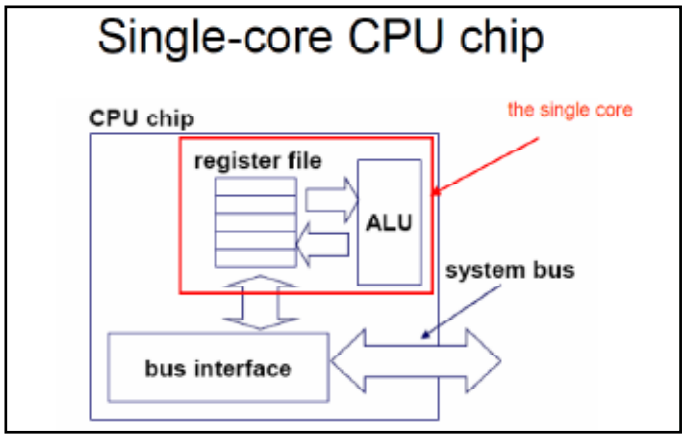

Fig.1 : Single Core CPU Chip

These types of CPUs have distinct advantages and disadvantages.

Advantages

- Uses less power: It takes less power to run a single core CPU. Quad-core and dual core use up a lot of energy, which is not a problem with machines that are plugged into the wall, but it can drain laptop batteries quickly.
- Runs cooler: Using less power means there's less heat generated by the core.

Disadvantages

- Runs slower: In cases where the computer specs are comparable, the single-core processors run slower that multi-core processors. They simply do not have as much computing capacity as the multi-core systems.
- Freezing: Many of today's software programs use up a lot of computing power. Often, a user will use many programs, which overloads the CPU and the computer simply freezes to stop the user from opening any more programs.

The Multi-Core CPU

Multi-core processors, as the name implies, contain two or more distinct cores in the same physical package. In this design, each core has its own execution pipeline and each core has the resources required to run without blocking resources needed by the other software threads. The multi-core design enables two or more cores to run at somewhat slower speeds and at much lower temperatures. Multi-core processors are MIMD because different cores execute different threads (Multiple Instructions), operating on different parts of memory (Multiple Data) and they are also a shared memory multiprocessor in which all cores share the same memory.

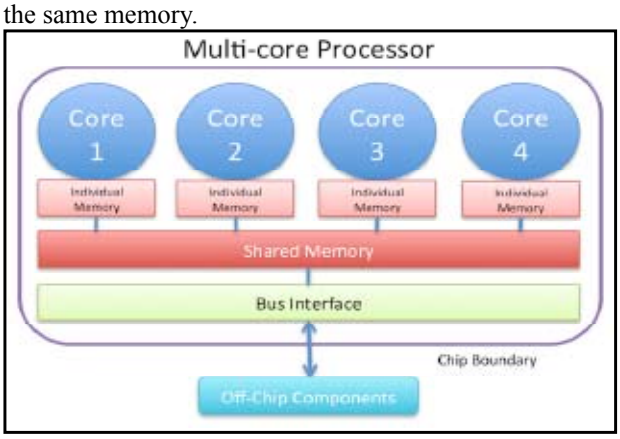

Fig. 2 : Multi Core CPU Chip

ISSN : 2347 - 8446 (Online)
ISSN : 2347 - 9817 (Print)

Multi-core is a design in which a single physical processor contains the core logic of more than one processor for example – when an Intel Xeon processor were opened up and inside were packaged all the circuitry and logic for two (or more) Intel Xeon processors. The multi-core design puts several such processor "cores" and packages them as a single physical processor. The goal of this design is to enable a system to run more tasks simultaneously and thereby achieve greater overall system performance. The following are the examples of multi core processors.

Dual Core CPU

The dual-core CPU is a processor with two execution cores in a single integrated circuit. These cores act as a single unit, but they will each have their own controller and cache, allowing them to perform faster than single-core processors

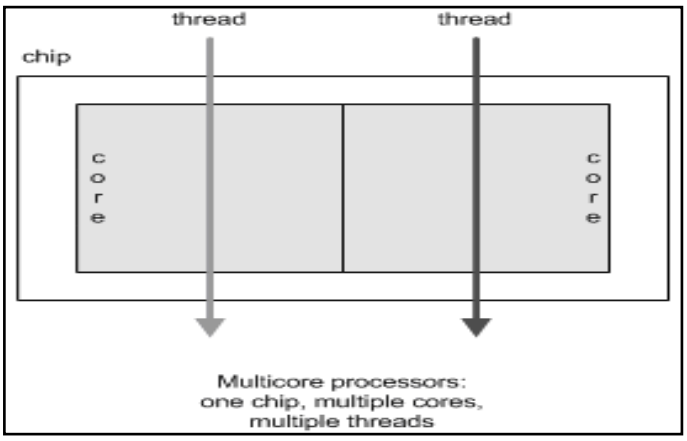

Fig. 3 : Dual Core CPU Chip – One Chip having two core using different threads

Advantages

1. Performs tasks faster: Dual-core CPUs run faster than single-core ones, especially in instances where there are multiple processes at one time. When a single-core processor has to do two different things, it must stop what it's doing and then switch to the next task. This switching is what creates lags, and in dual-core processors, this switching between tasks is reduced because there are two processors that can do the task at once.
2. Reduced costs: Even before there were dual-core CPUs, users were able to build dual-processor units that had the computing power of two computers. So without purchasing two computer we can fit dual core processor into a single motherboard and computer.

Disadvantages

1. Wasted computer power: While dual-core CPUs certainly can blaze through numerous applications, most regular users don't need that much power. Checking email, surfing through text-based sites, and typing documents don't actually use up a lot of power. Unless the person uses graphics or video programs, the dual-core almost seems excessive.
2. Compatibility with software: Dual-core CPUs will run any software, but the software itself has to be programmed for the dual-core CPU. Programmers have to tell the software that when one CPU is overloading, it needs to switch some of the tasks over to the second CPU.

The Quad-Core CPU

Quad-core CPUs contain four processors. Depending on the manufacturer, this can mean that four cores are on the same integrated circuit or the same chip package. Since these are a little more complicated that dual-core CPUs, there are some variations on this design. For example, some chips may or may not share resources like caches. It's also possible to have a quad-core processor with the same types of chips (homogeneous multi-core systems) or different chips (heterogeneous multi-core systems).

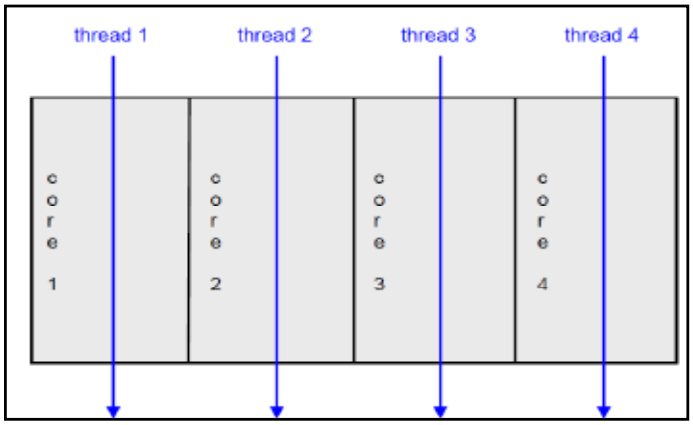

Fig.4 : Quad Core CPU Chip having 4 cores on one single chip

Advantages

1. Multitasking: The quad-core system is one of the best systems for multi-tasking. With the help of many cores, it can do many processes at once and also maintain the integrity of the system.
2. Run intensive applications: Applications that use up a lot of resources, such as graphics programs, video editors, and anti-virus programs, can run smoothly at the same time.
3. Less heat and power consumption: Most of the newer quad-core chips are so small and efficient; they can actually use less power and generate less heat than single-core systems.
4. Use for long term: The problem with Moore's Law is that it practically guaranteed that your computer would be obsolete in about 24 months. Since few software programs are programmed to run on dual-core, much less quad-core, these processors are actually way ahead of software development.

Disadvantages

1. Lowers battery life: Depending on the type of applications, quad-core systems can drain batteries faster.
2. Available software: Software needs to be programmed to take full advantage of quad-core CPUs, so not all programs can utilize the four processors.
3. Hardware compatibility: Multi-core processors are compatible with certain motherboards, so it's not as simple as swapping out the old CPU with a brand new one. Purchase of a new motherboard may also necessitate the purchase of other components that are compatible with the motherboard.

The different companies like Intel® Pentium® 4 and Intel Xeon processors today already use Hyper-Threading Technology (HT Technology) to execute multiple programs simultaneously. But HT Technology and multi-core designs differ significantly and deliver different performance characteristics. The key differentiator is basically - how a program's instructions are executed and which technology has been used like multithreading, Hyper-threading

or Muti-core.

Technology Used to execute multiple programs (Multithreading, Hyper-Threading, or Multi-Core?)

All the programs are made up of execution threads. These threads are sequences of related instructions. In the early days of the PC, most programs consisted of a single thread. The operating systems in those days were capable of running only one such program at a time. The result was-as some of us painfully recall-that your PC would freeze while it printed a document or a spreadsheet. The system was incapable of doing two things simultaneously.

Innovations in the operating system introduced multitasking in which one program could be briefly suspended and another one run. By quickly swapping programs in and out in this manner, the system gave the appearance of running the programs simultaneously. However, the underlying processor was, in fact, at all times running just a single thread.

By the beginning of this decade, processor design had gained additional execution resources (such as logic dedicated to floating-point and integer math) to support executing multiple instructions in parallel. The company reasoned it could make better use of these resources by employing them to execute two separate threads simultaneously on the same processor core. Intel named this simultaneous processing Hyper-Threading Technology and released it on the Intel Xeon processors in 2003.

According to Intel benchmarks, applications that were written using multiple threads could see improvements of up to 30% by running on processors with HT Technology. Two programs could now run simultaneously on a processor without having to be swapped in and out. To induce the operating system to recognize one processor as two possible execution pipelines, the new chips were made to appear as two logical processors to the operating system.

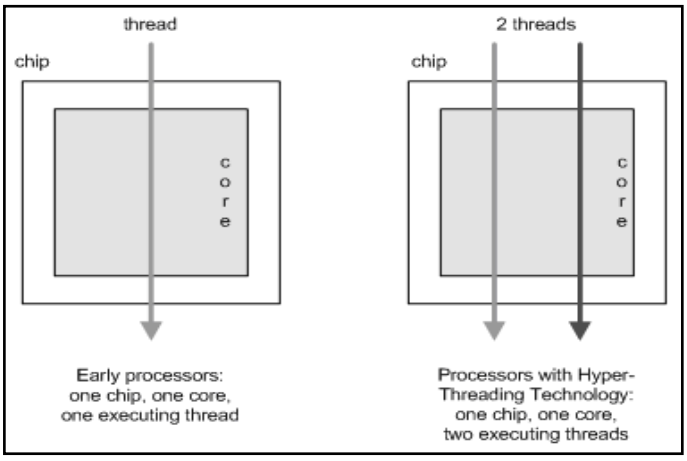

Fig. 5 : Technology in different time processor chip

HT Technology enables two threads to execute simultaneously on a single processor core

The performance boost of HT Technology was limited by the availability of shared resources to the two executing threads. As a result, HT Technology cannot approach the processing throughput of two distinct processors because of the contention for these shared resources. To achieve greater performance gains on a single chip, a processor would require two separate cores, such that each thread would have its own complete set of execution resources.

The Need For Multicore CPU

The high performance speed achieved by multi-processors (multiple CPUs on different chips attached to the same motherboard), produce undesirably high power consumption, and as a result, alternative research trends encouraged the production of multicore CPUs in order to reduce power consumption, while simultaneously increasing the processing speed. The architecture of multicore CPUs provided the hungry applications and devices, speed and performance with lower power consumption.

Parallelism and Performance In Multi-Core CPUS

Because multi-core CPUs exploit parallelism to enhance performance, an understanding of the key types of parallelism is important to analyzing performance. Basic three types of Parallelism is discussed here. The Instruction-level parallelism, thread-level parallelism, and data-level parallelism are all employed by various multi-core CPU architectures, and have different impacts on performance that must be understood to conduct thorough performance analysis.

Instruction-Level Parallelism

The instruction-level parallelism (or ILP) involves executing certain instructions of a program simultaneously which would otherwise be executed sequentially [Goossens10], which may positively impact performance depending on the instruction mix in the application. Most modern CPUs utilize instruction-level parallelization techniques such as pipelining, superscalar execution, prediction, out-of-order execution, dynamic branch prediction or address speculation [Goossens10]. However, only certain portions of a given program's instruction set may be suitable for instruction-level parallelization, as a simple example illustrates below in Figure

Sequential Execution	Instruction-Level Parallelism
1. a = 10 + 5 2. b = 12 + 7 3. c = a + b	1.A. a = 10 + 5 1.B. b = 12 + 7 2. c = a + b
Instructions: 3 Cycles: 3	Instructions: 3 Cycles: 2 (-33%)

Fig. 6 : Instruction Level Parallelism

A simple example of instruction-level parallelism

Because steps 1 and 2 of the sequential operation are independent of each other, a processor employing instruction-level parallelism can run instructions 1.A. and 1.B. simultaneously and thereby reduce the operation cycles to complete the operation by 33%. The last step must be executed sequentially in either case, however, as it is dependent on the two prior steps.

Thread-Level Parallelism

The thread-level parallelism (or TLP), involves executing individual task threads delegated to the CPU simultaneously [Blake10][Ahn07]. Thread-level parallelism will substantially impact multi-threaded application performance through various factors, ranging from hardware-specific, thread-implementation specific, to application- specific, and consequently a basic understanding is important for the analyst. Each thread maintains its own memory stack and instructions, such that it may be thought of as an independent task, even if in reality the thread might not

ISSN : 2347 - 8446 (Online)
ISSN : 2347 - 9817 (Print)

really be independent in the program or operating system. Thread-level parallelism is used by programs and operating systems that have a multi-threaded design. Conceptually, it is straightforward to see why thread-level parallelism would increase performance. If the threads are truly independent, then spreading out a set of threads among available cores on a processor would reduce the elapsed execution time to the maximum execution time of any of the threads, compared to a single threaded version which would require additive execution time of all of the threads. Ideally, the work would also be evenly divided among threads, and the overhead of allocating and scheduling threads is minimal.

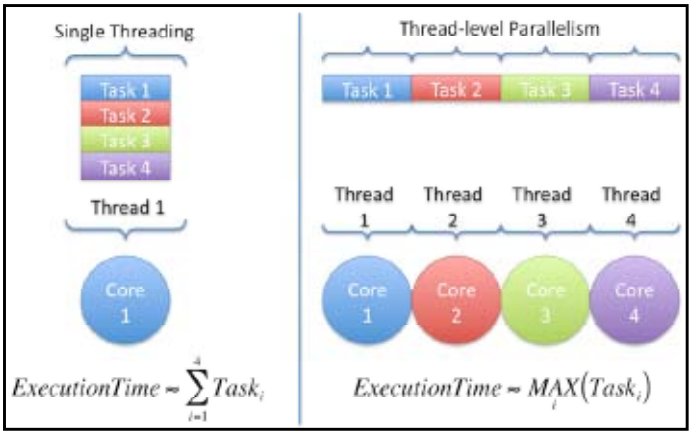

Fig. 7 : Thread-Level Parallelism

A conceptual visualization of thread-level parallelism

The above Figure 7, illustrates these conceptual differences between single threading and thread-level parallelism, assuming independence and no additional per-thread overhead. Performance-impacting factors include the load balance, level of execution independence, thread-locking mechanisms, scheduling methods, and thread memory required. Further, data-level parallelism among the distributed threads may impact performance, as the subsequent section discusses. The thread implementation library in both the operating system and the specific application will also impact performance [Blake10] [Moseley07].

Data-Level Parallelism

The data-level parallelism (or DLP), involves sharing common data among executing processes through memory coherence, improving performance by reducing the time required to load and access memory [Ahn07]. For the analyst, identifying application areas utilizing data-level parallelism will assist in understanding performance characteristics on multi-core processors. In the context of a multi-core CPU, data-level parallelism in the cache memory shared by cores can have a substantial impact on performance [Chandramowlishwaran10][Ahn07]. Here, the executing processes running on multiple cores will be called threads. Performance gains are expected when the threads read from the same data in the shared memory. This scenario allows one copy of the data to be used by the multiple threads, reducing the number of copy operations and thus execution time. When the threads have no data in common, each thread must maintain a copy of its data and no gains are available. However, if the multiple requests to this memory exceed its bandwidth, increasing threads may produce negative performance impacts. Further performance impacts may also occur during write operations. Multiple threads attempting to write to the same memory location at the same time must wait to resolve conflicts. Schemes to handle such situations are well known in computer science, such as spin-locks. The performance impact will depend upon the penalties involved with the scheme employed and how often such conflicts occur. Generally speaking, having threads write to different areas of shared memory would be preferable in lessening the likelihood of incurring these penalties. Non-Uniform Memory Architecture (NUMA) may assist, as it places data used by one particular core physically closer to that core in memory [Chandramowlishwaran10]. Bandwidth may also be an important factor as the number of threads increases on a multi-core processor. Limited cache size (cache misses), limited bandwidth, off-cache latency, and other aspects will have impacts on performance, though data-level parallelism can improve performance in certain situations. Further, the interaction between instruction-level and data-level parallelism affects performance; Flynn's taxonomy is a useful framework to analyze these interactions [Flynn72]. In summary, observations about data-level parallelism in a particular application are crucial to analyzing its performance on multi-core CPUs because memory is frequently the limiting factor. This section introduced three key types of parallelism employed by multi-core to enhance performance. Instruction-level parallelism, thread-level parallelism, and data-level parallelism have varying performance impacts, and consequently understanding, identifying, and studying these factors will lead to improved performance analysis. The next section introduces tools for empirically measuring and benchmarking multi-core CPUs, then provides an example that uses these tools and the concepts in prior sections, using empirical measurements to make performance observations

The Need For Performance Analysis

Performance analysis is a criterion that defines the performance of a system, and is required at every stage of the computer system life-cycle, to ensure high performance at a given cost. The demand for performance analysis was derived by radical changes in a number of elements including,

- The present day computer user who is more demanding than computer users 20 years ago.
- The popularity of computer technology resulted in an inundation in the computer market of different computer manufactures, each differing in performance. Such changes require performance analysis that meet user's demands and help to select the best alternative which provides higher performance at given cost implementing trade-offs between what each technique provides and the required criteria in mind.
- In this paper i will represent the different ways used to evaluate multicore CPUs performance.

Evaluating multicore CPU performance

This section explores the different methods used in evaluating multicore CPUs performance, with the metrics, analysis, and factors, varying performance based on requirements.

Evaluation Techniques

The first step in performance evaluation is to select the proper evaluation technique. The main techniques are: analytical modeling, simulation and measurement. In evaluating multicore CPUs performance, the techniques used are depending on different considerations. However, we cannot trust the result of one technique unless we validate that result with other techniques. For example, we cannot trust the analytical modeling technique without validating the result with simulation or measurement.

That is, we require at the use of two techniques to get an accurate result. The considerations for selecting the appropriate technique listed in table below [Jain91].

Table 1 : Evaluation Techniques Criteria

Criterion	Analytical Modeling	Simulation	Measureme
Stage	Any	Any	Post Prototy
Time	Small	Medium	Varies
Tools	Analyst	Computer Language	Instrumenta
Accuracy	Low	Moderate	Varies
Trade-Off Evaluation	Easy	Moderate	Difficult
Cost	Low	Medium	High
Saleability	Low	Medium	High

The above table shows the consideration in order of importance and the result may be wrong or misleading in all cases. For example, analytical modeling can be done at any stage of the system life-cycle. Although it takes smaller amounts of time than simulation and measurement because they vary with time, analytical modeling needs no tools for analysis; unfortunately it may give less accurate.

Evaluation Metrics

Performance metrics are the measurements of the system performance or activity [Metrics], and the metrics selection depends upon the services provided by the system because metrics quantify the required output of the system. Metrics can be classified in three main classes:

- HB (Higher is Better): include metrics like throughput, higher values for this metric are preferred.
- LB (Lower is Better): include metrics like execution time, where lower values for this metrics preferred.
- NB (Nominal is Best): include metrics like power, with the median values being optimal in this case.

The list of metrics that are used in evaluating multicore CPUs performance for applications is given below.

- Throughput: Average rate of successful processes [Sharma09].
- Response time: The time that the user finishes the request and the system starts a response.
- Execution time: The time needed to complete program execution [Monchiero06].
- Energy: The power needed to run a program [Monchiero06].
- Memory bandwidth: the rate of data sustained from the CPU core to the RAM (Random Access Memory) [Kayi07].
- Memory latency: the time delay between the memory controller signaled the memory module to access data from the RAM and the time the data become available for the memory module, also known as CAS (Column Address Strobe) latency [Monchiero06, CAS].
- Percent of memory contention: the percentage among the cores trying to access the RAM at the same time [Monchiero06].

This above list is an example of metrics used in multicore CPUs performance evaluation. Generally the metrics are related to three criteria (1) time (2) rate and (3) resources, which can be measured and used to determine the system performance. Performance metrics varies upon the services provided by the system, in multicore CPU systems metrics can be chosen based on the purpose of the performance analysis and which type of performance requirements are required for the program to run efficiently by taking advantage of multicore CPU architectures.

Factors Affects The Performance

Factors are the performance parameters that we want to study to see their effects on the system. Factors also depend upon the required performance needed to utilize the CPU and get the expected outcome from it.

- Memory: Memory architecture used and memory speed can affect the performance of the multicore CPU [Sharma09].
- Scalability: Affects performance based on the rate of increase of the workloads (i.e. tasks) [Sharma09, Carpenter07].
- I/O bandwidth: Can affect the performance by utilizing the CPU cores which leads to more resources consumption without performance increasing [Sharma09].
- Inter-core communication: The interaction between cores in multicore CPU's can be implemented by various mechanisms, affecting overall CPU performance due to shared workloads between cores [Sharma09].
- Operating system (OS): OS is the manager for the CPU and it assigned tasks to cores based on a scheduling mechanism, affecting the multicore CPU performance [Sharma09, Pase05].
- CPU clock speed: Clock speed has an impact on processor performance with slow clock speed reducing throughput [Pase05].
- Numbers of cores: Affect the CPU performance as multicore architecture workload is divided between the cores [Pase05].
- Cache coherent: Multicore architectures use different caching mechanisms as the cache is shared among the cores, causing cache coherent to affect CPU performance [Kayi07, Kumar05, Chang06, Zheng04, Yeh83].

These are some examples of the factors affecting multicore CPU performance, and for the analysis of each factor under study we will define ways to optimize the performance by analyzing the effects of the factors and interpret the result to get the optimal expected performance.

Example of Multicore CPU Performance Analysis

There is an example of performance analysis of processes for multicore CPUs that will assist in selecting the proper CPU for a machine specification.

Server virtualization of Multicore CPU

Intel IT (Information Technology) team evaluated server performances based on three Intel multicore CPU servers (A Four-socket server based on Quad-Core Intel Xeon CPU X7350 with 16 cores, a dual-socket server based on Quad-Core Intel Xeon CPU X5355 with 8 cores and a dual-socket server based on Intel Dual-Core Xeon CPU 5160 with four cores) [Carpenter07]. In comparing the performance of the multicore CPUs, the Intel IT team targeted the speed of the CPUs and the power efficiency. Due to CPU clock speed, runtime used to measure the performance on each CPU. The data was normalized. Furthermore, the normalized workload consists of VMs (Virtual Machines) and a copy of a synthetic CPU intensive DB application in each VM.
W-M/Job (Watt-minute per job) metrics was utilized to measure

ISSN : 2347 - 8446 (Online)
ISSN : 2347 - 9817 (Print)

CPU power efficiency with an increasing workload to test the scalability factors of the CPUs [Carpenter07]. The results from [Carpenter07] show that the three servers different levels of scalability in terms of power consumption. As the VMs number increased the run time remains constant until the workload equals the number of cores. After the number of VMs exceeds the number of cores, the run time begins to increase, the below figure shows the result of the servers based on Intel multicore CPU run times.

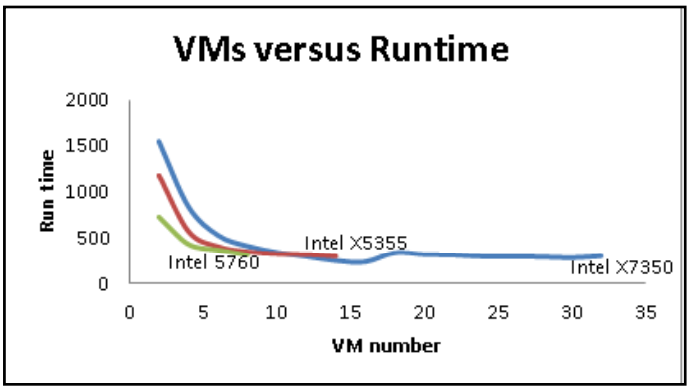

Fig. 8 : Run Time

We can observe that the Intel X7350 CPU run time is almost constant, until the number of VM reach 16, then it starts to increase. Alternatively, in the case of the Intel X5355, the run time starts to increase around 6 VMs, and around 4 VMs were running the Intel 5760. Another approach that is useful in performance evaluation of the CPUs is to measure the power consumption of the different CPUs based upon increasing workload to test the scalability [Carpenter07]. The result of the test shows that Quad-core Intel Xeon CPU X7350 based servers consumed more power than its alternatives due to the larger number of cores. Consuming at an average of 495 W (Watts) on 2 VMs running, 478 W for Quad-core Intel Xeon CPU X5355, and average of 330 W for Dual-core Intel Xeon CPU 5160. As the number of VMs increased, the servers became more power efficient which can be observed from figure 2. The Quad-core Intel Xeon CPU X7350 based servers with the maximum workload showed the power consumption per job decrease from the start and that is due to scalability of the CPU. This Figure shows the increasing of power consumption due to the number of VMs.

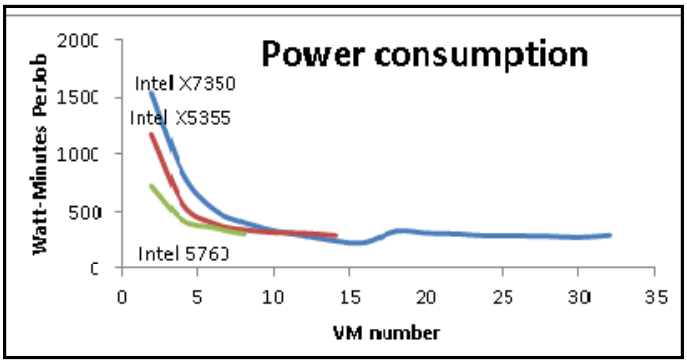

Fig. 9

Power Consumption

This section introduced examples of how performance analysis works. By comparing the multicore CPUs, we can use the result to help us make proper decisions in terms of selecting the appropriate CPU for a required performance level.

Summary and Conclusion

Now a days, CPU performance is increasing rapidly. The number of cores on the chip increases at each release of a new generation of a CPU. With multicore CPU becoming not only faster but more power efficient depends on the required demand. To maintain relevant profiling of these systems, we need to evaluate the CPU depending on the workload we expect to process. We have addressed the techniques used to evaluate multicore CPU performance, metrics, factors, benchmark tools used to measure multicore CPU performance, and we provided an example of performance evaluation for multicore CPUs. Different approaches used in multicore CPU performance analysis depending on the purpose of the study. We are expecting to see new approaches in multicore CPU performance analysis as multicore CPU production increases to new levels.

List of Acronyms

- CPU - Central Processing Unit
- RAM - Random Access Memory
- HB - Higher is Better
- LB - Lower is Better
- NB - Nominal is Best
- CAS - Column Address Strobe
- OS - Operating System
- IT - Information Technology

References

[1] *[Ahn07] J. H. Ahn, M. Erez, and W. J. Dally, "Tradeoff between Data-, Instruction-, and Thread-Level Parallelism in Stream Processors," Proceedings of ICS '07, 18-20 June 2007, 1-12, http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.95.5616. Analyzes the instruction-level, data-level, and thread-level parallelism of the Stream Processor architecture.*

[2] *[Blake10] G. Blake, R. G. Dreslinski, T. Mudge, and K. Flautner, "Evolution of Thread-Level Parallelism in Desktop Applications," ISCA 10, June 2010, 302-313, http://www.eecs.umich.edu/~blakeg/docs/Desktop_TLP_Study_ISCA2010.pdf. Analyzes the thread-level parallelism of a wide range of applications on different operating systems running on multi-core CPU and GPU hardware.*

[3] *[Carpenter07] Carpenter R. E., "Comparing Multi-Core Processors for Server Virtualization, Intel Corporation, August 2007,*

[4] *[Chandramowlishwaran10] A. Chandramowlishwaran, K. Madduri, and R. Vuduc, "Diagnosis, Tuning, and Redesign for Multicore Performance: A Case Study of the Fast Multipole Method," SC '10 Proceedings of the 2010 ACM/IEEE International Conference for High Performance Computing, Networking, Storage, and Analysis, November 2010, 1-12, http://portal.acm.org/citation.cfm?id=1884654. Describes a step-by-step approach to modeling, analyzing, and tuning a program on a multi-core processor system.*

[5] *[Chang06] Chang J., Sohi G. S., â€oeCooperative Caching for Chip Multiprocessorsâ€, ISCA Proceeding 33th annual international symposium on Computer Architecture, Pages 264 - 276, IEEE Computer Society, Washington, DC, 2006, ISBN: 0-7695-2608-X.*

[6] *[Flynn72] M. J. Flynn, "Some Computer Organizations and Their Effectiveness," IEEE Transactions on Computers, Vol.*

C-21, No. 9, September 1972, 948-960, http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=5009071. Develops a hierarchical model of computer organizations for instructions and data streams.

[7] [Goossens10] B. Goossens, P. Langlois, D. Parello, and E. Petit, "Performance Evaluation of Core Numerical Algorithms: A Tool to Measure Instruction Level Parallelism," hal-00477541, version 1, 29 April 2010, 1-4, http://hal.archives-ouvertes.fr/docs/00/47/75/41/PDF/para10.pdf. Measures and analyzes the instruction-levelparallelism of core numerical algorithms in terms of running time and proposes a tool to assist in analysis.

[8] [Jain91] Jain R., The Art of Computer System Performance Analysis , Wiley-Interscience, New York, NY, April 1991, ISBN: 0471503361.

[9] [Kayi07] Kayi A. , Yao W. , EL-Ghazawi T. , Newby G. , Experimental Evaluation of Emerging Multi-core Architecturesâ€ , Parallel and Distributed Processing Symposium, IEEE International, March 2007, http://cecs.uci.edu/~papers/ipdps07/pdfs/PMEO-PDS-21-paper-1.pdf.

[10] [Kumar05] Kumar R., Zyuban V., Tullsen D. M., Interconnections in Multi-core Architectures: Understanding Mechanism, Overheads and Scaling , ISCA 05 Proceeding 32th annual international symposium on Computer Architecture, Pages 408 - 419, IEEE Computer Society, Washington, DC, 2005, ISBN: 0-7695-2270-X. [Metrics] "Performance Metrics", Wikipedia the free encyclopedia, http://en.wikipedia.org/wiki/Performance_metric.

[11] [Monchiero06] Monchiero M. , Canal R. , Gonzalez A. , â€oeDesign Space Exploration for Multicore Architectures: A Power/Performance/Thermal View , Proceedings of the 20th annual international conference on Supercomputers, Pages 177- 186, ACM New York, NY,2006, ISBN:1-59593-282-8.

[12] [Moseley07] T. Moseley, A. Shye, V. J. Reddi, D. Grunwald, and R. Peri, "Shadow Profiling: Hiding Instrumentation Costs with Parallelism," CGO 07, 2007, 1-11, http://portal.acm.org/citation.cfm?id=1252541&dl=ACM&coll=DL. Evaluates the performance and accuracy of a shadow profiling technique for inter-procedural path profiling and value profiling with minimal overhead.

[13] [Pase05] Pase D. M., Eckl M. A., A Comparison of Single-Core and Dual-Core Opteron Processor Performance for HPC , Technical report, IBM Developer Works, IBM Corporation, 2005, ftp://ftp.support.lotus.com/eserver/benchmarks/wp_Dual_Core_072505.pdf

[14] [Sharma09] Sharma A. , Kumble M. , Moktali P. R. , Siri H, â€oePerformance analysis of Multicore Systems , Intel, March 2009,http://software.intel.com/en-us/articles/performanceanalysis-of-multicore-systems-4

[15] [Yeh83] Yeh P. C. C., Patel J. H., Davison E. S., â€oeShared Cache for Multiple-Stream Computer Systems , IEEE Transactions on Computers, Vol. 32, Issue 1, pages 38 - 47, IEEE Computer Society, Washington, DC, 1983.

[16] [Zheng04] Zheng Y., Davis B. T., Jordan M., Performance Evaluation of Exclusive Cache Hierarchies , ISPASS '04 Proceedings of the 2004 IEEE International Symposium on Performance Analysis of Systems and Software, Pages 89 - 96, IEEE Computer Society, Washington, DC, 2004, ISBN: 0-7803-8385-0.

Rakhee Chhibber is working as an Assistant Professor, in a college named RDIAS(Rukmini Devi Institute of Advanced Studies – affiliated with GGSIPU). I am doing my Phd. In computer Science from Mewar University. I am having total 22 years of experience, one year industry and 21 years academics in various colleges and institute of high repute.

Dr. R.B.Garg is a retired Professor from Delhi University. He had also worked with different colleges affiliated from different universities. He has a number of publications to his credit in various National and International Journals. He has written a book 'Contributions to Hardware and Software Reliability' published by World Scientific. He has worked in various capacities in Delhi University, GJU, GGSIPU and various other institutions of high repute.