sageza

Strategic Snapshot

Processor Defined:
Changing Markets or Changing Definitions?

By Joyce Tompsett Becknell

The Sageza Group, Inc.

May 2006

The Sageza Group, Inc.
32108 Alvarado Blvd #354
Union City, CA 94587
510·675·0700 fax 650·649·2302
London +44 (0) 20·7900·2819
Milan +39 02·9544·1646

sageza.com
info@sageza.com

Processor Defined:
Changing Markets or Changing Definitions?

ABSTRACT

As vendors roll out the many changes happening to processor technology, confusion is arising in the marketplace.Vendors exploit different advances in their chip and processor designs, depending on their products' past architectural directions. For customers, trying to understand the difference between processors and the impact to their applications and infrastructure has become more complicated due to marketing obfuscation by the vendors in order to differentiate their products and architectures. By changing the classical definitions of computer terminology they risk confusing users who think in technical terms and aren't aware of technical words being used differently.Users may find themselves comparing systems which they believe to be alike but are in fact very different because two vendors are using common words differently.

There is no easy way to compare the various technologies, and it's challenging for a buyer to sort out which technology and implementation is best for their workload or environment. This paper seeks to lay out the facts behind multi-core, multi-threading and cache and examine the philosophical approaches. We also touch upon other technology issues affecting processor implementation, and try to peer through the layers of marketing to focus clearly on the differences in architectures and their benefits for various workloads. In the end, we hope to reinforce the definition of a processor and demonstrate that consistency in terms is important, not only for technical understanding, but also for licensing and overall IT costs.

Processor Defined:
Changing Markets or Changing Definitions?

TABLE OF CONTENTS

Introduction

There are many changes happening to processor technology that make computing more efficient and for that we are thankful. At the same time, as vendors roll out these new changes, confusion is arising in the marketplace. As expected, vendors have chosen to exploit different advances in their chip and processor designs, depending on their products' past architectural directions. As the technology has matured, most of these changes are evolutionary, while some are revolutionary. For customers, trying to understand the difference between processors and the impact to their applications and infrastructure has become more complicated. However, we don't believe that the technology evolution itself is the issue. Users are savvy enough to work out the details of the technologies they use. The problem here is in fact marketing obfuscation by the vendors in order to differentiate their products and architectures. However, to do this they are changing the classical definitions of computer terminology, and they risk confusing users who think in technical terms and aren't aware of technical words being used differently. Less technically inclined users may find themselves comparing systems which they believe to be alike but are in fact very different because two vendors are using common words differently.

Processor technology is complex enough, and each vendor has adopted marketing and positioning to differentiate its products most favorably. There is no easy way to compare the various technologies, and it's challenging for a buyer to sort out which technology and implementation is best for their workload or environment. In particular, the evolution of multi-core processors, multi-threading, and cache sizes have contributed to the confusion, as each technology provides advances both individually and when interacting together.

This paper seeks to lay out the facts behind multi-core, multi-threading and cache and examine the philosophical approaches, focusing specifically on Sun and IBM. This paper also touches upon other technology issues affecting processor implementation, including power consumption and application performance as well as market issues such as licensing costs and classifying the number of processors. We also try to peer through the layers of marketing to focus clearly on the differences in architectures and their benefits for various workloads so that IT managers can be better informed when choosing processors for their application needs. In the end, we hope to reinforce the definition of a processor and demonstrate that consistency in terms is important, not only for technical understanding, but also for licensing and overall IT costs.

Processors and Cores and Chips… Oh My

Perhaps it is best to begin with a definition of what is a processor, as that is one of the key confusion points today. According to computer science sources, a processor is a functional unit that interprets and executes instructions. A processor consists of an instruction control unit, as well as one or more arithmetic units for calculations.

The history of computing reveals that processors were once spread across multiple chips, but in the recent past, most processors have been contained on a single chip, in a one-to-one ratio. The stage we're entering now puts multiple processors on a single physical chip. This is an example of the evolutionary progress of technology, which is making microprocessor technology ever smaller. While this is a straightforward evolutionary development, it has not generally been marketed as such. The terms processors and chips have become synonymous and used interchangeably. This is not accurate for today's new generation of processors, and as we'll demonstrate, it is instead becoming a source for much confusion. We would argue that it is important to continue to use the classical definition of a processor rather than to refit the definition for marketing du jour, which is a dangerous precedent to set.

Now that we've defined a processor as a functional unit that interprets and executes instructions, it is also important to remember that processors do not operate in isolation. Optimizing processor performance is a moving target for researchers, as the right choice for any given application requires finding a platform that has been optimized for one or more types of workload. The combination of hardware platform plus operating system creates the environment to run an application, and the processor is one of the key pieces of the hardware platform. This is a somewhat simplified description, as different workloads tax various subsystems differently, and server vendors design systems that function well enough in diverse environments. Mainstream servers are never optimized completely for one application or workload; they are general-purpose systems with enough flexibility to allow optimization by the user for their specific needs. As a result, the processor and server vendors have taken different approaches based on the operating system and workloads most popular among their user base. The result is that the range and type of processors used in general-purpose servers have grown in different directions, and it can be confusing to sort out the hardware for an environment without having to wade through numerous layers of marketing and technical minutia.

The approach to microprocessor design that most of the system processor vendors have followed for several years is called superscalar computing. In superscalar computing, the central processing unit (CPU) manages multiple instruction pipelines to execute several instructions concurrently during a unit of measurement called a clock cycle. Superscalar CPUs therefore can decode multiple instructions in parallel. The results are then put into a buffer and issued to multiple units. There are buffers (in some designs for as many as 200 or more instructions) to keep many instructions moving at the same time, to prevent (or at least avoid) bottlenecks. This approach is good for gaming environments, digital content, and scientific computing which involve lots of calculations, and it can even be beneficial for commercial applications that depend on caches to help keep large volumes of instructions flowing, while data is moving about between disk and memory, but it isn't the only way to solve the processing challenge. Caching was invented to help with the problem of balancing multiple activities happening simultaneously and making sure that resources weren't kept waiting. Instruction execution was frequently faster than data retrieval from memory: if that data could be cached, then performance would be increased, and a cache with faster access time could improve overall performance. While it would seem that the larger the cache, the better, this is not always the optimal way to solve the problem. In fact, in some cases management of the cache is more important than just making the cache larger.

Putting multiple processors on a single chip is the next evolutionary step, and this is where the confusion begins. The industry has begun referring to these processors as cores, while some vendors have continued to refer to chips and processors as synonymous things, when in fact processors and cores are synonymous.

Multi-threading is another advance that puts multiple threads on a processor or core. Threads are a stream of instructions that the application thinks will be executed on a processor. Applications see multiple cores as multiple processors, each capable of running a thread. In essence, a dual-core processor looks like two processors to an application. These multiple-core chips are also sometimes known as massively multi-threaded chips if they have many threads running, and sometimes as Chip Multiprocessing (CMP). So where previously one chip was synonymous with one thread and one processor or core, all has changed.

Each vendor has its own permutations and combinations that buyers must sort through. As an example, IBM began to use multiple cores with the Power4 chip, which had two cores per physical processor. With Power5 IBM introduced multiple threads (two) on multiple cores. Sun also had two cores with the UltraSPARC IV and has recently launched a new chip, the

UltraSPARC T1, which is the first in the new Niagara family of processors using larger numbers of cores, this first having eight cores and four threads per core.

A Little More Depth; A Little More Clarity

While each of the vendors is evolving its own chip architecture, each is doing it in a slightly different way. There is a limited amount of real estate on a chip, and multiple cores, multiple threads, and larger caches all take up more space. Vendors must therefore make choices between how much space they allot to each function, with the tradeoffs dictated in part by the architecture they've chosen. In general, adding threads can be a win-win from a price/performance viewpoint. Adding another thread takes less space than adding another core, but this is not always the best way to boost performance, although the performance boost of another thread is generally greater than the performance boost of adding another core in comparison to the real estate price. Another way to say this is that an increase in performance for multi-threading is greater than the increase in the area required to implement multi-threading. The ratio makes multi-threading a sensible idea. However, the actual benefit of adding another thread depends on the core itself. In general, the more efficient the core is, the less benefit is derived from adding more threads. A less efficient core can benefit more from more threads, so the performance boost a vendor receives from multi-threading is dependent on the overall efficiency of the core. At the same time, each marginal thread adds less overall performance than its predecessor, so the absolute performance increase for the third thread is less than that of the second thread, the fourth is less than the third, and so forth. Therefore the reasonable number of threads for a given design depends on both the marginal area as well as the marginal performance increase.

The other tradeoff a chip designer makes is related to cache. This depends somewhat on the applications that a vendor anticipates will be run on its system. Many users tend to do repetitive things and the repetitive data is thrown into the cache. For applications with many repetitive tasks, a large cache is useful. Some applications, such as scientific applications don't always take advantage of cache because they only do a calculation once and then move on, so the cache serves no useful purpose. Most applications fall somewhere in between, and some workloads have many users and many cache uses, at which point—as indicated previously—having good cache management, an issue not covered in this paper, becomes more important than just having a really big cache. In essence, lots of data is running through the cache, but it is often interdependent, because one transaction may have to finish before another can be started.

IBM and Power5

IBM's current implementation provides two hardware threads per core or processor. IBM's Unix operating system is AIX 5L, and it recognizes these hardware threads as two logical processors. Both SUSE and RedHat Linux distributions also recognize multi-threaded implementations on Power5 chips. According to IBM, a P5 server with eight physical processors (which is four physical chips, each with two processors per chip) can have 1sixteen logical processors (two threads per processor). When IBM speaks of an 8-way system, they are referring to processors and not sockets, which is the traditional way in which the industry has referred to server size, although for many years the number of processors was equal to the number of chips and sockets.

The operating system controls which processor a thread is dispatched on. AIX will only switch a processor from single threading to multi-threading if there are more active threads than physical processors. If SMT is used, two logical processors are assigned to a single physical processor in the same partition. This means that threads are dispatched to each physical processor in pairs, so there are eight pairs on an 8-way machine (one pair per

physical processor). This SMT policy is controlled by the OS, which means it is partition-specific. SMT operates at the physical processor level, and most applications will be able to benefit from this feature transparently, since it is the OS's responsibility to make this happen, not the application's.

Sun and UltraSPARC T1

Sun's new processor, also known as Niagara, has eight cores, with each core capable of running four threads concurrently, for a total of thirty-two concurrently running threads per chip. Sun's approach to threading is based on a different paradigm than IBM's. Whereas IBM's focus is on cores, and multiple threads are used only when needed, Sun's architecture is built purposely to use threads: Sun in fact views them as processors within processors. Sun views threads more from a software point of view than a hardware point of view. According to Sun, each thread can be a process (what a user thinks of as a program), or one process (program) can run all thirty-two threads itself, or theoretically, any variation in between, depending on the software's capabilities (as traditional software programming doesn't work that way.) Sun uses a different form of multi-threading than IBM, and believes that its method minimizes the number of CPU cycles that could be wasted when threads are switched. Sun views the four threads per each of eight cores as thirty-two logical processors (which Sun calls virtual processors) with concurrent task capabilities. However, all thirty-two threads are competing for the same core resources, including the single floating point unit on the physical chip.

From a software viewpoint, with symmetrical multiprocessing (SMP) systems, different software processes run on different processors, but different parts of the same process can run on different processors, which is why multi-threaded applications (not to be confused with threads on a processor, which are different) run better on SMP systems than single-threaded applications do. The Niagara family will only be available with a single chip; more than one eight-core chip will not be possible. Customers who want more chips will have to wait until the future generations of Sun's Niagara chip.

Intel and AMD

Intel has a different implementation than either IBM or Sun. IBM and Sun, once they delivered dual-core systems, went to add multi-threading. Intel was just the opposite, starting with multi-threading on a single core for Xeon, which it referred to as hyper-threading, and then transitioning to multiple cores per chip. While IBM had the first dual-core general-purpose chips, Intel was the first to bring them to desktop PCs. Intel's x86 products (the non-Itanium) which are the majority of its chips, appear to be moving away from multi-threading and embracing multi-cores. Intel has stated that by the end of 2006, it expects 85% of its server products to be shipped with dual- or multi-core processors. For Itanium, Intel plans to ship Montecito, the first dual-core, by mid-2006. Clovertown, its first quad-core processor, is expected to ship in the first part of 2007.

AMD claims it has had a strategy for multi-core computing since the late 1990s and positions itself as a pioneer in the x86 space. AMD also states that in fact its AMD64 architecture was built from the ground up with with multi-core in mind. AMD brought out dual-core chips for servers before desktop PCs with Opteron. AMD has also indicated that it is working on quad-core chips. There has been no indication from AMD that it is going to a multi-threaded implementation any time soon.

Implications to the Market

The various approaches vendors have taken to processor technology have resulting technology and marketing implications. The industry has clearly embraced multiple cores on

chips as a good thing, largely because it is evolutionary and is a natural consequence of increased density of each new semiconductor technology generation. However, with the exception of Sun, it looks like dual core will be the common use for a while, much as dual-processor SMP that is dual chip—with each chip being a single processor or core—was the most popular implementation of SMP for years. Sun has been able to achieve eight cores but at a real estate cost to cache sizes; that is, they sacrificed cache size to permit a larger number of cores on the chip. Time will tell if this bet pays off for them, or if the more traditional core numbers of other vendors remain the dominant approach.

Sun claims that Niagara is currently a network-oriented design and not data-oriented. By this they mean that it is designed for workloads that don't require large amounts of logic and calculations and have smaller data footprints so that application performance is not dependent on the storage subsystem. In other words, the UltraSPARC T1 is designed for large numbers of repetitive tasks but not for database workloads. IBM, on the other hand, continues to target large database workloads with its pSeries Unix systems that use the Power5 architecture. In general, while multi-core is the way of the future, multi-threading still has limited appeal and may or may not become standard.

AMD and Intel continue to derive a large portion of their processor revenue from desktop PCs rather than servers, and multi-threading has less of a performance advantage there as multi-threaded applications are less common in a desktop environment than in a server environment. This may be the reason for their greater interest in multiple cores.

While technology issues may arise for users with specific environmental concerns, marketing issues frequently have wide-ranging impact on a greater percent of the customer base. Perhaps the biggest marketing implication is with benchmarking. The Olympic sport of server vendors is benchmarking, which is a double-edged sword. On one hand, benchmarks are necessary, because having standard ways of measuring performance is the only way to compare various technology implementations. On the other hand, they are mostly unrealistic because they are done in a lab, in a rarified environment, with large amounts of money spent on tuning in comparison to what an average user would or could do. They may not also be germane to the user's actual workload or system use but are sometimes a close enough approximation. With benchmarking, terminology becomes very important so that like systems can be compared. With the advent of multi-core and multi-threading, the benchmark councils and groups have stepped back from traditional definitions and instead let vendors label their systems. As a result, Sun, HP, AMD, and Intel would take the approach of referring to a sixteen-chip system (with two cores each, for a total of thirty-two cores) as a 16-way, whereas IBM has maintained the traditional SMP approach and refers to eight chips with sixteen cores as a 16-way, as each core is a processor and IBM counts physical processors rather than chips and sockets.

In essence, with dual core systems, when Sun says 16-way it means double the amount of sockets or chips that IBM does, which leads to confusion in the market. Sun markets its new UltraSPARC T1 has having thirty-two virtual (logical) processors, but because they are on one chip, it calls this a single processor. It is particularly difficult to follow in benchmarking, where suddenly vendors are using different metrics for counting and holding this up to their competitors without reconciling the naming conventions when they boast results. So when Sun announces a benchmark comparing its 16-way to IBM's 16-way, you have to translate the terminology. In Sun's terminology, it's actually a Sun 16-way and an IBM 8-way. And using IBM's terminology, it's a Sun 32-way and an IBM 16-way. Either way is fine, but they are currently presented as two 16-ways which is simply misleading, as they are comparing a Sun 32-core system to an IBM 16-core system. We would strongly encourage those who publish and certify benchmarks to make a decision for their benchmarks on how they will classify various processors and then stick to that nomenclature. It would greatly assist in clearing this

up before it gets further out of hand, and vendors show no signs of playing together on this any time soon.

Licensing issues are another problem. When it comes to licensing by number of processors, many applications are priced based on the number of CPUs. This is really just one of the earliest issues in the road to virtualization, where a CPU may be virtual or logical and not physical. What do they mean by CPU? Should pricing be by core (processor) or by chip (socket)? Application vendors must rationalize pricing based on what an application believes it is using and not necessarily what it is actually using. Naturally this is much easier to say than to work out as the underlying technology is constantly shifting based on actual usage and total system requirements.

Microsoft was one of the first vendors to clarify its position. Microsoft claims that it has always charged by (Microsoft's definition of) processor; that is, by chip, not by core, and it will continue to do so. Microsoft claims that "a physical processor is a single chip that houses a collection of one or more cores." According to computer science definitions of a processor, this is not true, but at least Microsoft is defining its terms outright for those who are understandably confused by this new use of the word processor. This is a good scheme for a Niagara user who would be charged for one processor (as all Niagara systems are single chip systems) rather than for eight representing the number of cores or processors on the chip, although Niagara systems cannot run the Microsoft OS.

On the other side, Oracle has been a holdout for pricing per core, but has recently amended its policy to recognize the implications of multi-core chips on its model. For Oracle, the required processor license is dependent upon the specific multi-core chip on which Oracle is deployed. Depending on which chip is used from which vendor, Sun, AMD/Intel, or IBM, a different price will be charged as each implementation is viewed as a certain number of processors. IT managers planning on purchasing software licenses by processor should make sure they understand both how the system will be used and how each application vendor's philosophy and approach affects their purchase decision.

Many IT managers are frustrated by the decisions the vendors have taken. They would of course prefer charging by socket as that yields them a lower number than charging by core. We suspect the debate has only just begun, but if vendors amend their plans, it could lead to significant license cost changes and we encourage IT buyers to track this debate closely.

Wrapping It Up

In the end, multi-core systems solve some of the problems encountered as the current growth provided by superscalar computing reaches its limits. This is why the industry has embraced multi-core systems. Multi-core processing is the new wave, but there will be confusion in the industry until there is agreement on the use of terms like cores, processors and CPU's. IT managers need to understand how applications function and how they will be used in their environment in order to make the best choice of operating system, and processor technology.

In addition, the most important consideration for any system is the need to be balanced. Efficient systems architecture balances all resources, including storage I/O, network I/O, memory requirements, CPU, and power consumption. This is done at not only at the chip level, but at the system and data center levels as well.

Benchmarks that focus on processor performance as well as power consumption should be used judiciously as they reveal only one facet of a complex environment. While the measurements of the benchmark may be certified, they run the risk of being irrelevant to a particular application or workload environment. Finally, when reading benchmarks, users should make sure they understand how many processors are actually being deployed so that they are making truly accurate comparisons.