SEMINAR REPORT

HYPERTHREADING TECHNOLOGY

SUBMITTED IN PARTIAL FULFILMENT OF THE DEGREE OF

BACHELOR OF TECHNOLOGY

by

Ajith Prasad B

Under the guidance of

Prof. K . Abdul Nazeer

Department of Computer Engineering
National Institute of Technology, Calicut

National Institute of Technology, Calicut
Department of Computer Engineering

Certified that this Seminar Report entitled

HYPERTHREADING TECHNOLOGY

is a bonafide work carried out by

Ajith Prasad B

in partial fulfilment of their
Bachelor of Technology Degree
under our guidance

Prof. K . Abdul Nazeer
Faculty
Dept. of Computer Engineering

Dr. V.K. Govindan
Professor and Head
Dept. of Computer Engineering

Acknowledgement

I thank Mr.K.Abdul Nazeer,lecturer in Department of Computer Science and Engineering for his guidance and co-operation.I also thank Mr.Madhu Kumar, lecturer in Department of Computer Science and Engineering,for co-ordinating our seminar classes.I also acknowledge the advice and guidance given to me by my friends and classmates .

Ajith Prasad B

Abstract

Hyperthreading technology is a new innovation by Intel, paving the path for thread level parallelism . Traditionally multithreaded systems execute two or more threads by rapidly switching between them.This high-speed switching gives the external world a feeling that more than one process is executing at the same time.Hyperthreading enables a processor to carry out two or more instructions simultaneously as long as there is no conflict of resources.To take full advantage of hyperthreading, software support is also essential.Intel has achieved a performance enhancement of about 30 percent upon using hyperthreading-supported software on hyperthreaded processor

Contents

1 Introduction **1**
1.1 Notion of Thread and Process 1
1.1.1 Process 1
1.1.2 Thread 1
1.2 Parallelism 1

2 Why Hyperthreading **2**
2.1 Superthreading 3
2.2 Multiprocessor systems 4

3 Hyperthreading Explained **4**
3.1 Inside The Processor 4
3.2 From Operating System's perspective 4

4 Software on a Hyperthreaded processor **5**
4.1 Threads from different processes 6
4.2 Programmer's challenge 6

5 Drawbacks **6**
5.1 Cache conflicts 7

6 Conclusion **8**

1 Introduction

Microprocessor designers mainly give emphasis to two main factors - speed and parallelism in their quest towards efficiency.This seminar focusses on hyperthreading, the latest technology introduced in microprocessor design aiming for more parallelism in computing.

1.1 Notion of Thread and Process

These two terms have been used by computer programmers and hardware designers in varying contexts.Various operating systems and architectures give different definitions to them.For the purpose of this seminar I shall stick to the following definitions which are also the most popular.

1.1.1 Process

A program in execution which has a memory associated with it .

1.1.2 Thread

Actual running component of a process.A process can be multi-threaded; i.e, can consist of more than one thread.An operating system supporting multithreading has to do both process scheduling and thread scheduling . From the processor's point of view, it is concerned only with the threads. it cannot distinguish between threads of different processes.

1.2 Parallelism

Initial microprocessors in the 80x86 series supported one instruction at a time.Also, the predominant operating system during those days - MS DOS - supported only one process at a time.Pentium and Pentium - Pro series had increased number of execution cores. Their superscalar architecture and pipelining resulted in a certain degree of parallelism.But no processor had the capability to execute two instructions from different threads simultaneously. Due to rapid context switching between threads enabled by high speed switching logic,users experienced parallelism.

2 Why Hyperthreading

Even with the optimised pipelining architectures and increased number of execution units available with modern microprocessors like Pentium and Athlon, only one-third of the resources were being used at a time.This is because of the inability of the processor to execute more than one thread at a time.Even after applying the intricacies of Out of Order Execution and advanced branch prediction logic,instructions from a single thread are far below the maximum utilisation potential of a processor resulting in wastage of resources.

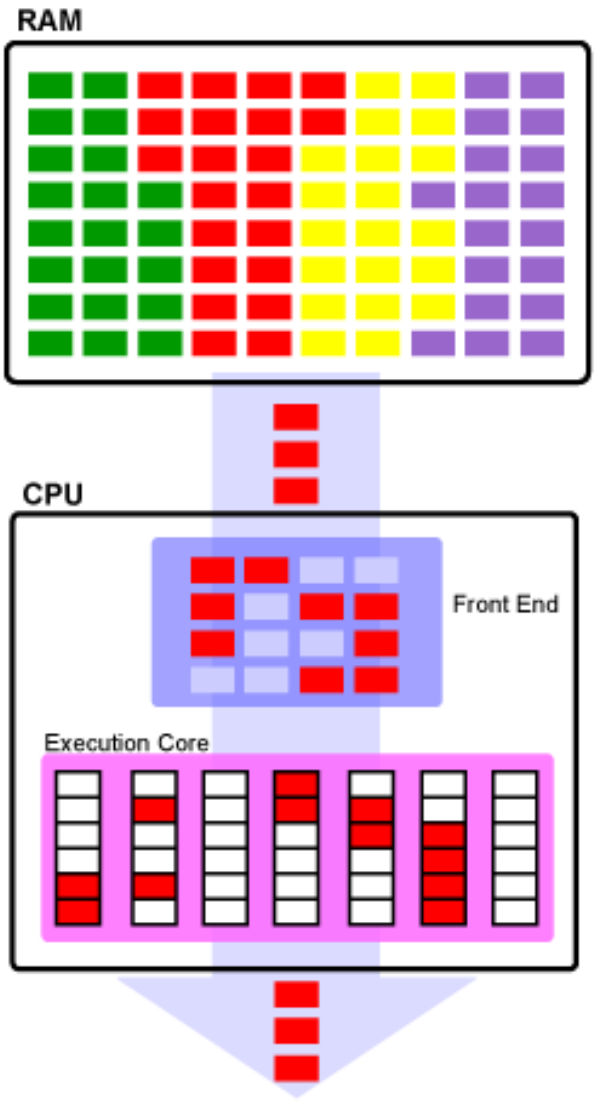

Figure 1: Single Threaded CPU.The processor shown in the figure above has seven execution paths in the execution core .Red colour indicates executing threads. White spaces are blank slots.Processor uses just one-third of its available potential.

2.1 Superthreading

An alternative mechanism called Superthreading, implemented in some microprocessors provided some capability to execute multiple threads at time.But it also had its limitations. All instructions in one pipeline stage of a superthreaded processor have to be from the same thread.Superthreading involved limited amount of context switching too.This technology also didn't eliminate the wastage of resources completely.

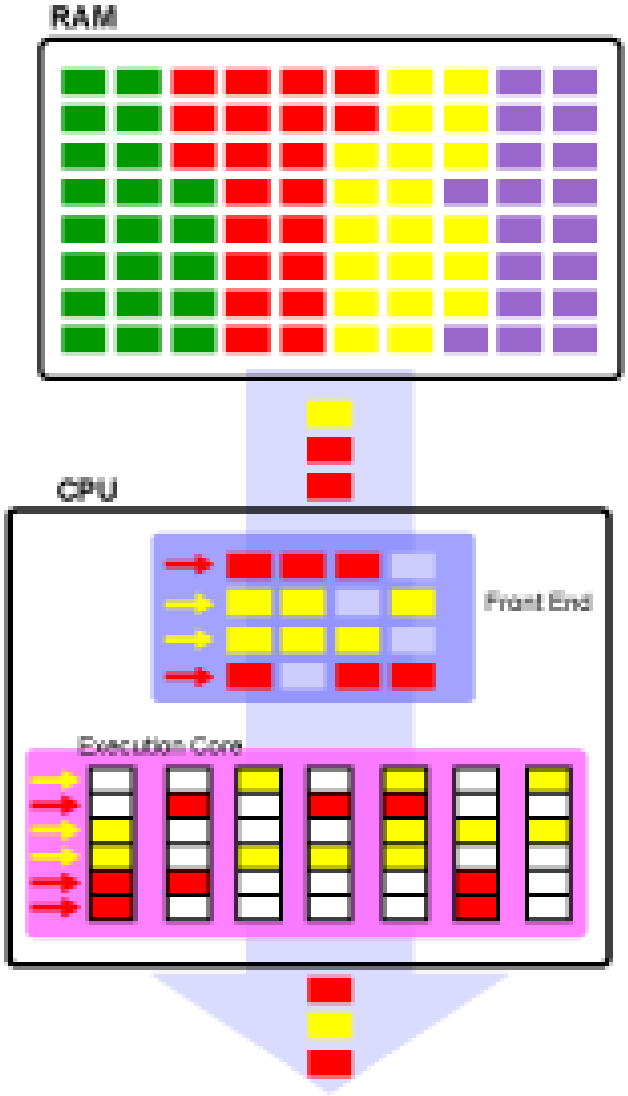

Figure 2: Super Threaded CPU.Red and yellow colours indicate executing threads.Each of the six pipeline stages has just one thread.Resource utilisation is about 50 - 60 percent

2.2 Multiprocessor systems

Using more than one processor in a system directly provided thread level parallelism.But this was a costly solution and required special on board support for multiple processors. These systems also required some type of synchronising mechanism so that both processors function in unison.The overhead for this synchronization and the frequent collision for the need of resources slowed down such implementations considerably.Due to these reasons, multiprocessor systems remain confined to special purpose implementations and dedicated systems.

These circumstances envisaged the need of a technology that could provide thread level parallelismin the processor.Hyperthreading , introduced by Intel provided the solution for this problem, but also provided its own challenges.

3 Hyperthreading Explained

Hyperthreading is superthreading without the restriction that each pipeline stage or clock cycle must contain instructions from the same thread.Vacant execution slots in each pipeline stage is filled by instructions from another thread,enabling thread level parallelism. This can be done as long as there is no collision of resources required by two threads.

3.1 Inside The Processor

Since more than one thread is being run at the same time, certain architectural features are replicated , some are shared and some are partitioned.The replicated resources are instruction pointer, return stack pointer,register renaming logic etc.Caches(L1,L2,L3) and execution units are shared.Partitioned resources include scheduling queues,load-store and reorder buffers.Another subtle feature is that when only one thread is executing, all shared and partitioned resources are combined to give maximum throughput.These additional resources in a hyperthreaded pentium 4 or Xeon processors increased the processor die area by just five percent.Hyperthreading's strength is that it allows the scheuling logic maximum flexibility to fill the execution slots , thereby making more efficient use of available execution resources by keeping the execution core busier.

3.2 From Operating System's perspective

When viewed from OS or user level , the hyperthreaded processor is split up into two or more logical processors and threads can be scheduled to execute on any of the two logical processors as they are free.

Operating system schedules the threads appropriately into the available logical processors.Scheduling is done in such a way that two running threads won't request for a resource simultaneously as far as possible.It is a must that operating system support hyperthreading.Presently the number of logical processors supported by most operating systems and underlying hardware is two.Latest operating systems like Windows XP and Red Hat Linux 9 support hyperthreading.

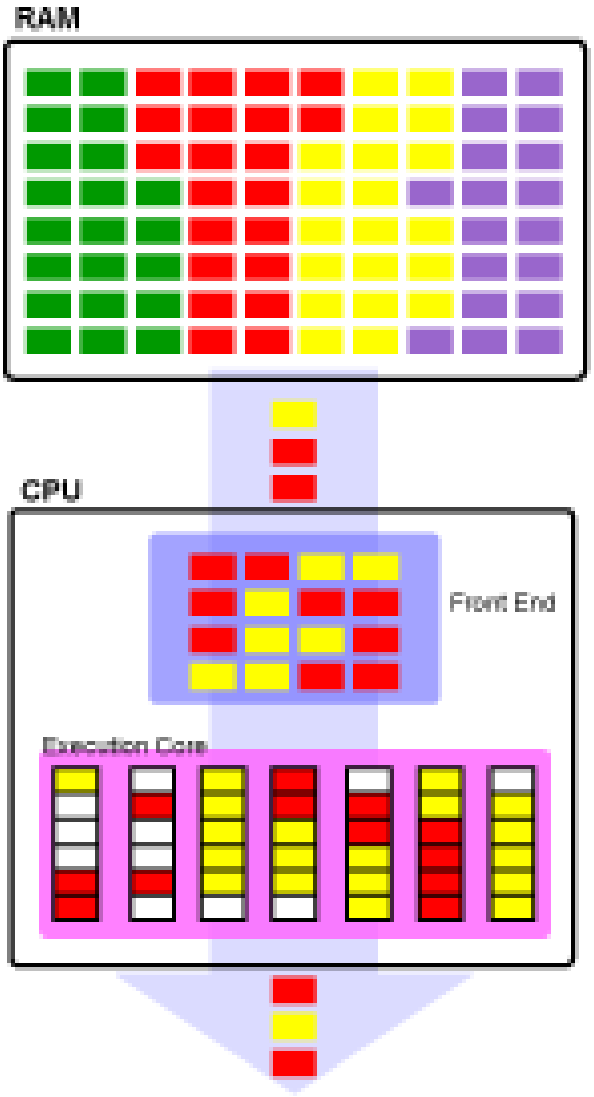

Figure 3: HyperThreaded CPU.Each pipeline stage has more than one thread in execution.Resource utilisation is around 75 - 80 percent.

4 Software on a Hyperthreaded processor

To get full advantage out of a hyperthreaded processor, an application must not only be multithreaded , but the individual threads must be coded in such a way that they should not raise the request for one resource simultaneously. For example, if two threads of an application request disk access at the same time, hyperthreading won't be of much help.Ofcourse no sensible programmer will write a code having such conflicts.But to write optimised code on a hyperthreaded processor,programmer must have a clear knowledge about the shared resources and possible conflicts.

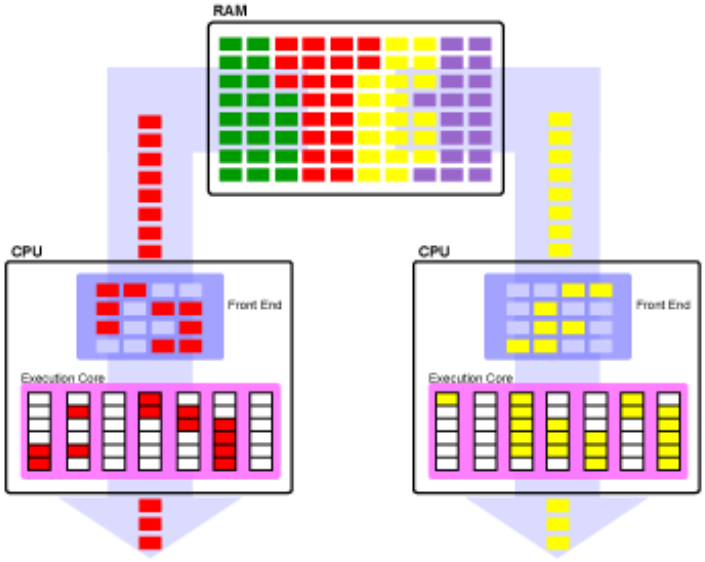

Figure 4: Logical View Of Hyperthreading.The actual processor in the previous figure is split up into two logical processors.

4.1 Threads from different processes

It is not possible to write optimised hyperthreaded code considering the other applications that will be running on the system.Programmer of one application will be oblivious to the internals and resource requirements of another application. And in normal case, it will be difficult to anticipate the applications running in tandem.For example,coder of a text editor will not be aware of the antivirus tool or messenger system running on a computer.So in the of threads from different processes, it is possible only to hope for the best.But some improvements are likely in this scenario.This is because the processor is unaware of the process-thread relation. It does not take care or does not know, which thread is from which process.It tries to execute the threads presented to it as fast as possible, inducing into the hyperthreading mechanism.

4.2 Programmer's challenge

Full benefits of hyperthreading will be available only when some general purpose code library supporting hyperthreading becomes available.Also there will be some security issues with this new methodology.This hyperthreaded programming style is yet to be popular.And it is written in the assembly level now.If some high level language or paradigm supports this new style, coding will be easier and benefits will come along.

5 Drawbacks

Hyperthreading's greatest strength -shared resources- also turns out to be its greatest weakness.Problems arise when one thread monopolizes on crucial resource; like the floating point unit and in doing so, it starves the other thread and cause it to stall. For example, if two floating-point intensive threads are trying to execute a long series of complex, multi-cycle floating-point

instructions on the same physical processor, then depending on the activity of the scheduler and the composition of the scheduling queue one of the threads could potentially tie up the floating-point unit while the other thread stalls until one of its instructions can make it out of the scheduling queue.

5.1 Cache conflicts

Since both logical processors share the same cache, the prospect of cache conflicts increase. This potential increase in cache conflicts has the potential to degrade performance seriously. None of the caches know the difference between one logical processor and another, or between code from one thread or another. This means that one executing thread can monopolize virtually the entire cache if it wants to, and the cache, unlike the processor's scheduling queue, has no way of forcing that thread to cooperate intelligently with the other executing thread. The processor itself will continue trying to run both threads, though, issuing fetches from each one. This means that, in a worst-case scenario where the two running threads have two completely different memory reference patterns (i.e. they're accessing two completely different areas of memory and sharing no data at all) the cache will begin thrashing as data for each thread is alternately swapped in and out and bus and cache bandwidth are maxed out.

In sum, resource contention is definitely one of the major pitfalls of hyperthreading, and it's the reason why only certain types of applications and certain mixes of applications truly benefit from the technique. With the wrong mix of code, hyper-threading decreases performance, just like it can increase performance with the right mix of code.

6 Conclusion

This promising technology has the potential to influence the computing world in a big way.But for that, software support is essential.Also, owing to the pitfalls of this technology, it is quite possible that another innovation providing enhanced parallelism come along and displace this altogether.

References

[1] Intel's HTTP Site, "http://www.intel.com"

[2] PCQuest's HTTP Site, "http://www.pcquest.com"