The CORE Problem

El Problema CORE

Andrew Bailey
Cambridge University, England
Systems Analyst, Cray Research UK 1989-1993
andy@hazlorealidad.com

Recibido para revisión 15 de Noviembre de 2007, Aceptado 19 de Mayo de 2008, Versión final 30 de Mayo de 2008

Abstract—In the following paper an analysis of the current trends of the computer industry related to multi core processors is presented with its implications on software architecture and design. We also discuss a Space Based Architecture currently being developed by Hazlorealidad.com.

Keywords—Moore’s Law, Amdahl’s Law, Multi Core CPU, Concurrency, Service Oriented Architecture, Space Based Architecture.

Resumen— En la publicación se realiza un analisis de las tendencias actuales en la industria de la informatica relacionado con procesadores multi-core y sus implicaciones para la arquitectura y diseño de software. Además discutimos una arquitectura basada en espacios (Space Based Architecture), desarrollada por Hazlorealidad.com.

Palabras claves—Ley de Moore, Ley de Amdahl, CPU Multiple Core, Concurrencia, Arquitectura Orientada a Servicios (SOA), Arquitectura Orientada a Espacios (SBA)

I. LAPTOP SUPERCOMPUTERS

The Cray-2 computer system was introduced by Cray Research in 1983. The Cray documentation states “The CRAY-2 Computer System sets the standard for the next generation of supercomputers. It is characterized by a large Common Memory (256 million 64-bit words), four Background Processors, a clock cycle of 4.1 nanoseconds (4.1 billionths of a second) and liquid immersion cooling.”[I] Therefore the Cray-2 had a quad core, 250MHz clock and 2GB RAM, characteristics that you can now find on todays laptops[II] , apart from the liquid immersion cooling!

II. MOORE’S LAW

In 1965, Intel co-founder Gordon Moore predicted that the number of transistors on a chip doubles about every two years, a statement now popularly known as Moore’s Law.

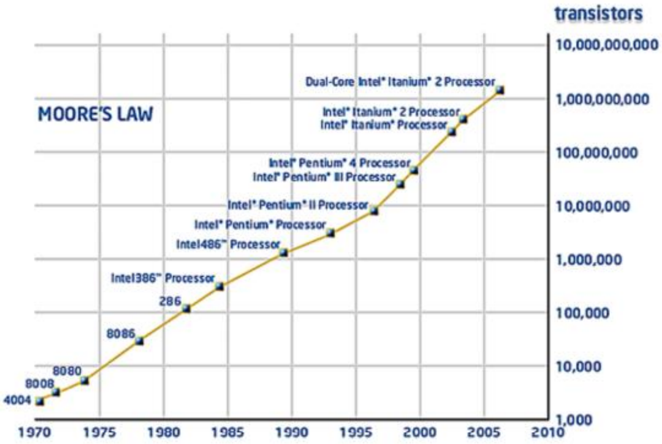

Figure 1. Moore’s law h

This exponential growth of processor potential explains how it is possible that a supercomputer of 25 years ago has been outpaced by a laptop.

[I] Introducing the CRAY-2 Computer System http://archive.computerhistory.org/resources/text/Cray/Cray.Cray2.1985.102646185.pdf

[II] First quad-core laptop hits U.S. August 17, 2007 http://www.news.com/8301-10784_3-9761814-7.html

[III] Graph courtesy Intel Corporation

Chip makers have been doubling the transistor density every two years and for the next few years the trend is set to continue, Intel started production of 65nm in 2005, 45nm in 2007 and are on target to produce 32nm in 2009. For reference the diameter of a Silicon atom is 0.24nm, so the transistors width is now of the order of a 100 atoms.

> "Continuing to deliver innovation to make the predictions of Moore's Law a reality means shrinking the nominal size of the devices that populate the silicon. Skeptics in the industry have believed that going down that path of decreasing transistor sizes would be more and more difficult since, as transistors shrink in size, they consume less power (they scale in voltage), but their leakage current (the continued flow of current even when transistors are "off") increases. The more transistors there are on a chip, the more power is wasted. Also, as transistor density and speed increase, the chip as a whole consumes more power and generates more heat. Thus, the efficiency of cooling techniques must also increase to dissipate the heat from the increases in device density and current leakage."[IV]

III. QUANTUM TUNNELING

In part the leakage current is due to a Quantum Mechanical effect known as Quantum tunneling, in which a particle, in this case an electron, passes through a barrier that according to classical mechanics does not have sufficient energy to do so. Perhaps the easiest explanation is that Heisenbergs uncertainty principal implies that if you measure one quantity you affect another, you cannot measure an objects velocity without altering its position and vice versa. A lesser known implication of the Heisenberg's uncertainty principal is that there is also an Energy-time uncertainty principal. The product of the uncertainties in Energy and time is of the order of 10-35 Joule-seconds, which although for macroscopic object negligible is an important factor on the atomic and sub atomic particles. Basically it means that the energy of a particle varies and although its average energy is not sufficient to cross the barrier, for a short period of time it can have sufficient energy to do so. Hence the electrons in todays semiconductor chips, leak across the part of the transistor called the gate. To counteract this manufacturers are using materials which are able to decrease the probability of electrons tunneling through the gate without adversely affecting the transistors performance using high-k gate dielectrics.[V]

Also other options are being investigated such as tri-gate transistors, where they implement 3 gates instead of one in order to reduce even further the leakage current. The problem is that the transistors would not be developed layer by layer on the silicon wafer as they are today, instead they would need to be created in "3d" however it is a technical problem, not a physical one. So it appears that Moore's law will still hold at least for the next 10 years.

IV. PROCESSOR SPEED

The doubling of transistor density on processors had lead to a doubling of speed, however the first 2GHz processor was released in August 2001, if this trend had continued processors would have been 4GHz in 2003, 8GHz in 2005 and 16GHz in 2007. Clearly this is not the case.

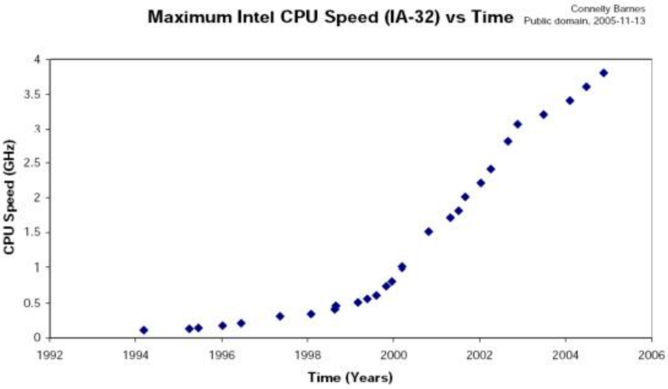

Figure 2. In the graph three trends can be observed: the exponential increase in speed up to 2000, an almost linear region from 2000 to 2003 and a sharp change in Cpu Speed against time at around 3GHz.[VI] One of the reasons for this is that "designers are now coming up against the physical, atomic limitations of today's materials science. Advances in power technology are now lagging behind advances in transistor technology, making power/thermal issues an increasingly critical design (and performance) constraint."[VII]

In order to see the problem facing chip manufacturers we can see from data from an 80 core Intel research chip that the performance can be increased by 80% but only at the cost of using 300% more power.

Table 1. Data from the Intel Teraflops 80 core Research Chip[VIII]

Frequency	Voltage	Power	Aggregate Bandwidth	Performance
3.16 GHz	0.95 V	62W	1.62 Terabits/s	1.01 Teraflops
5.1 GHz	1.2 V	175W	2.61 Terabits/s	1.63 Teraflops
5.7 GHz	1.35 V	265W	2.92 Terabits/s	1.81 Teraflops

V. MULTICORE CPU (CHIP LEVEL MULTIPROCESSOR - CMP)

In the near future it will not be the processor speed that doubles every two years it will be the number of cores[1] on a chip. This effectively means that the throughput of the computer will continue to grow exponentially but the raw speed will not. A program that has a single thread of execution will see little performance gain in the coming years.

This has serious implications for the software industry, and hence for systems engineers, universities and companies. Up

IV Moores Law, Intel Corporation http://www.intel.com/technology/magazine/silicon/moores-law-0405.pdf

V High K Gate Dielectrics http://www.intel.com/technology/silicon/high-k.htm

VI http://oregonstate.edu/~barnesc/documents/cpu_speed.pdf

VII http://www.intel.com/technology/magazine/research/EPI-throttling-1005.htm

VIII Intel Teraflops Research Chip http://techresearch.intel.com/articles/Tera-Scale/1449.htm till recently if a process needed to be executed faster you could solve the problem by getting a faster processor, however now that the processor speed has leveled out this is no longer the case. Now programs have to be written to take advantage of the parallel processing capabilities of modern CPUs.

To a certain extent compilers can take advantage of parallel architectures, however there is only so much a compiler can do, it is quite possible to write a program that turns what could have been a parallel algorithm into a serial one, destroying any chance of speedup on a multi core machine.

Most software on the market today is not written for multi core processors and there are good reasons for this, subtle errors in the software can lead to programs that work perfectly 99.9% of the time and then under certain conditions fail. This introduces non deterministic effects in software programs, not too far removed from Heisenberg's uncertainty principal.

Traditionally Software design has been taught using flow diagrams where there is a single thread of execution, and the vast majority of engineers approach problems this way.

What is now needed is to change the mindset of software engineers, it is no longer possible to think of problems essentially as a serial process, the simple flowchart will change to have around 100 simultaneous threads of execution in 5 years or so.

This raises some important questions, are Universities teaching students these core competences (pun intended) of concurrent programing?

Few systems engineering students today learn to program using mutexes, cyclic barriers, latches and semaphores, but this is precisely what they will need to do after graduating in a few years.

Are companies taking into account the changing environment of computing in their Requests for Proposals? After all it is very possible that the software purchased today will be running on a computer with 100 core within 10 years, if it was written as a single threaded program it will only be using 1% of the computers potential.

VI. AMDAHL'S LAW

This law relates to the parallelism of the algorithm to the speed of execution running on multiple processors. There will always be a percentage of code that has to execute in series, on a single processor, and a percentage that can operate in parallel, on multiple processors. The parallel part with an infinite number of processors will take zero time and in the serial part of the algorithm all of the processors except one will be idle waiting for the single processor to complete the task. If the mix is 10% in series and 90% in parallel, the maximum speed up we can see is 10 times the original speed. If the program is 25% in series and 75% in parallel then the maximum speedup is of 4.

The graph below shows the decrease of throughput per processor against number of processors for varying percentages of serial code in an algorithm.

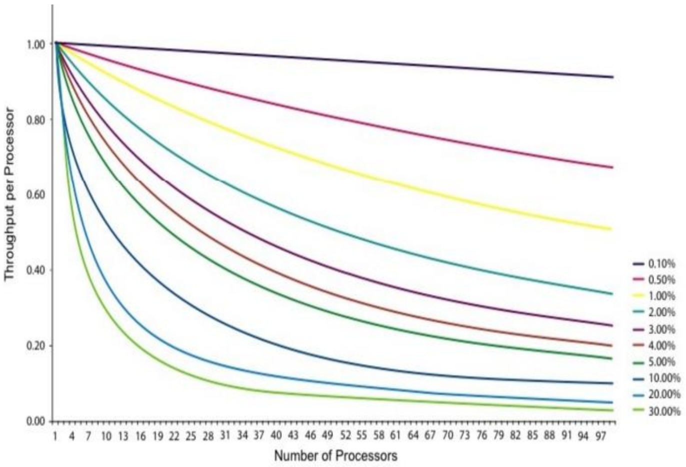

Figure 3. MultiCore processor

The diagram shows that although in a multi core processor the potential speedup is linear, this only applies in the case that the problem is completely parallel, ie each task is unrelated. This means that although the number of cores is set to double every two years, the increase in throughput will tail off, depending on how parallel the code is.

VII. WEB APPLICATIONS

There are also certain tasks that are inherently parallel, for example web applications for a long time have had to deal with concurrency issues and in a traditional web application, there is little information flow between the various users, this avoids one of the main problems of concurrent programs the coordination and sharing of information between different threads of execution. However, the database is likely to be the main bottleneck of a web application.

Enterprise Application Servers

Many enterprise applications, run on application servers with architectures designed for parallelism, such as Java Enterprise Edition, however, it is often the case that in todays computers that the processor is not being occupied 100% of the time. There are delays due to I/O, thread synchronization and locking, also in many architectures there are certain operations that need to wait for a database transaction to be committed, or that need to share state with other computers in a cluster.

Also in many systems all of the application state is stored persistently in a database, so that in the event of a system failure the process can be continued by another computer. However, in a typical enterprise architecture the database is the part that is the most difficult to scale. Also due to the possibility of failure of the database, normally the database is replicated to at least another machine, however, there is a high overhead in the synchronization of data to more than a few databases. And also the storage of the whole of the application state to the database introduces a bottleneck, if memory is considered slow by todays cpu speeds then disk speed is still in a prehistoric era, it can take 15 million cpu cycles just to access the disk.

Relative access frequencies of current hardware:

[1] A core essentially is one of multiple cpus on a single chip.

Table 2. Frecuencies

Hardware	Frequency	Period	CPU Cycles (approximate)
CPU	3 Ghz	0.3ns	1
Memory (DDR3)	800 Mhz	1.25ns	4
LAN Latency	500 Hz	2ms (estimate)	6.000.000
Disk Seek	200 Hz	5ms	15.000,000

We can see that the most likely bottleneck in an application is the disk, followed by network access.

VIII SERVICE ORIENTED ARCHITECTURE (SOA)

Many companies have separate systems to realize their activities, however, in the majority of cases they are from different vendors and do not interact resulting in the manual reprocessing of information taking data from one system and entering it into another. In some companies they have taken steps to automate this process, creating specialized software to interface one system with another. One of the potential problems lies with the number of interfaces created, if there are N different systems that all interact then N2 interfaces need to be written. SOA is a way of designing systems taking into account the services that each one provides and also the services that each one requires. The goal is for the services to be connected together with a minimum of effort. For example an accounting service offers the service of realizing financial transactions and report generation, the CRM offers the service of consulting the customers details, instead of duplicating the data within the accounting package.

The ultimate goal is to obtain the seamless flow of information through every enterprise process, having access to the data at any instant for analyze and decision making.

SOA is an enterprise architecture that is commonly implemented using web services, messaging systems and databases. However, each of these has its drawbacks, web services are based on the transfer of xml files, essentially text files, which need to be created, sent over a network and then interpreted. Message queues normally are separate processes that also communicate using the network, and in most enterprises are backed by persistent stores in order to guarantee message delivery. Also most applications use databases to save the state after every operation in order to recover from system failure. Evey step in the process taken consumes extra cpu time and adds latency.

IX. SPACE BASED ARCHITECTURE

"Space-Based Architecture (SBA) is a software architecture pattern for achieving linear scalability of stateful, high-performance applications using the tuple space paradigm" . It is similar to the blackboard design pattern used in artificial intelligence systems. In essence the architecture resembles a mix between Service Oriented Architecture and Event Driven Architecture, however the services and events are all contained within a single operating system process with multiple threads. This does not impose any restriction on the overall architecture of the system, it could participate as a service in an enterprise SOA, or each space could be replicated in a cluster of commodity hardware.

The basic principal is to have a shared space, common to multiple processors, where messages can be passed. This implements a form of high speed memory to memory message queue enabling peer to peer communications, which in turn allows loose coupling between the software components, and does not have the overhead of traditional message queues.

One of the key benefits of the architecture is that as each separate task is carried out by a self contained module, and each module interacts using a shared thread safe space, the implications are that the software engineer can develop each module as if it were a serial process ignoring the fact that many modules may be executing concurrently. The developer only needs to deal with concurrency if it is implemented within the module.

Although as in any enterprise application there is a database, read access to the database is minimized by using a distributed thread safe data cache and write access is minimized by persisting temporary state in the tuple space, with only the final results being persisted to the database. Also wherever possible idempotent operations are implemented so that the state does not need to be persisted, if an idempotent operation f Moores Law, Intel Corporation http://www.intel.com/technology/magazine/silicon/moores-law-0405.pdf ails it can be retried without the risk of corrupting data.

The goal is to turn enterprise applications into multiple unrelated tasks. In this way then linear scalability can be achieved with number of processors.

X. CONCLUSIONS

The next few years will hold many challenges not only for the semiconductor industry, as the limits of quantum mechanics are reached, and creative ways are found to extend the timespan of the exponential Moore's Law. It will also force a major transformation in the software industry, requiring a Quantum Leap in the mindset of software engineers, in order to design software solutions that are able to make use of hundreds of cores on a single chip.

Many universities have taught concurrency at the level of the operating system, but have not addressed the problem in detail at the level of software design. A major shift in emphasis needs to occur in their curriculum for their graduate students to have the competences that they will need in the coming years.

Software Engineers will need to understand issues surrounding concurrency including: race conditions, deadlock, livelock, starvation and priority inversion, and the software tools to manage concurrency such as: semaphores, mutexes, barriers and latches.

The challenges for businesses and institutions is that many enterprise applications will not scale effectively on multi-core hardware without re-writing for multi-threading. Today we are facing a similar situation to the problem Y2K, which was caused by the change of the millennium, programs that did not handle the change of century had to be modified, now the problem is that with the introduction of multi core cpus many programs will have to be modified or even redesigned completely.

In the light of recent developments in chip manufacture a reexamination of enterprise architecture is needed in order to make use of the hardware efficiently. Frameworks such as that being investigated and developed at Hazlorealidad.com can allow organizations to implement highly-scalable business applications well into the future, while maintaining critical transaction and integrity requirements.

Today it is almost acceptable that a program only uses 50% of a dual core processor, however will it be acceptable for a program to use 1% of a 100 core machine?

Universidad Nacional de Colombia Sede Medellín
Facultad de Minas

120 años
TRABAJO Y RECTITUD

Escuela de Ingeniería de Sistemas

Misión

La misión de la Escuela de Ingeniería de Sistemas es fomentar y apoyar la generación o la apropiación de conocimiento, la innovación y el desarrollo tecnológico en el área de ingeniería de sistemas e informática sobre una base científica, tecnológica, ética y humanística.

Visión

La formación integral de profesionales desde el punto de vista científico, tecnológico y social que les permita adoptar, aplicar e innovar conocimiento en el campo de los sistemas e informática en sus diferentes aspectos, aportando con su organización, estructuración, gestión, planeación, modelamiento, desarrollo, procesamiento, validación, transferencia y comunicación; para lograr un desempeño profesional, investigativo y académico que contribuya al desarrollo social, económico, científico y tecnológico del país.

Escuela de Ingeniería de Sistemas
Dirección Postal:
Carrera 80 No. 65 - 223 Bloque M8A
Facultad de Minas. Medellín - Colombia
Tel: (574) 4255350 Fax: (574) 4255365
Email: esistema@unalmed.edu.co
http://pisis.unalmed.edu.co/