Are Multiprocessor Systems Irrelevant?

Abstract

The performance growth rate of uniprocessors was at its highest rate since the first transistorized computers in last 20 years. However, it is doubtful that this rate can be maintained for over another 15 years. Computer architecture renovations are needed to sustain the high performance growth rate. Multiprocessor adds a new dimension to the design space of computer architecture—the number of processors. It will definitely have a great role in the future. This paper gives the current technology trends, application requirement trends and the architecture trends to show that multiprocessor systems are necessary to satisfy the computational requirements and are cost-effective based on the current development in architecture. Then the advantages of multiprocessors are given. The conclusion is that we have enough reasons to be optimistic for the future of multiprocessor systems—they are not irrelevant.

1. Introduction

1.1 Technology Trends in Computer Architecture

Computer technology has made incredible progress in the years since the first general-purpose electronic computer was created. In the late 1970s, the emergence of microprocessor led to high performance improvement—about 35% growth per year. Owning to this growth rate and the cost advantage of the mass-produced microprocessor, an increasing fraction of computer business is based on microprocessors. From mid-1980s, the growth rate is 50%-100% per year. At the same time, the number of transistors on chip grows rapidly. Clock rates also experienced tremendous growth rates. Figure 1 shows us such general technology trend. Figure 2 shows the technology trends. We can see that multiprocessors have the fastest performance growth rate. Figure 3 shows us the clock frequency growth rate. We can see that the clock frequency increases 30% per year. Figure 4 shows us the transistor count growth rate, from which we can see that now there are 100 million transistors on chip and the transistor count grows much faster than clock rate—currently 40% per year. Parallel processing is a way to use more transistors.

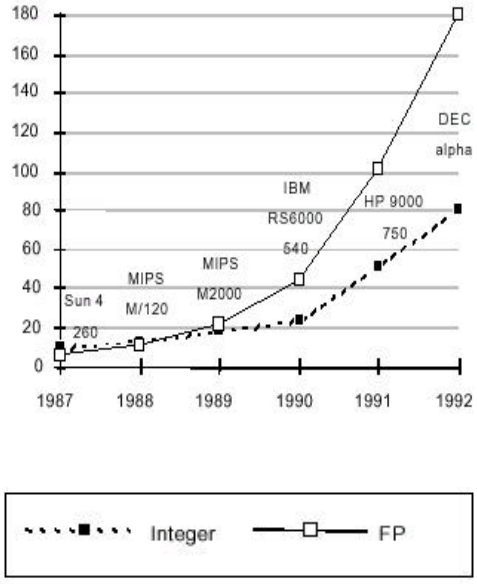

Figure 1 General Technology Trends

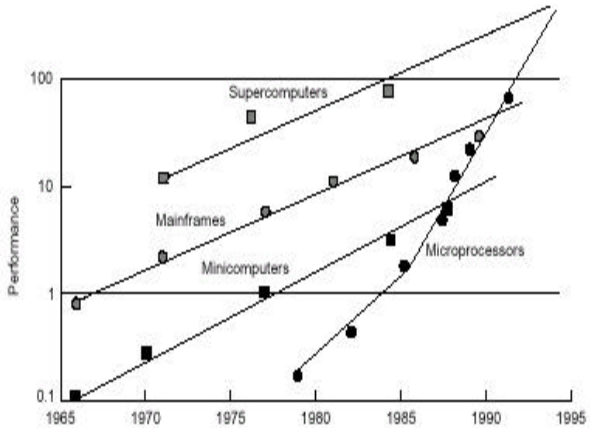

Figure 2 Technology Trends

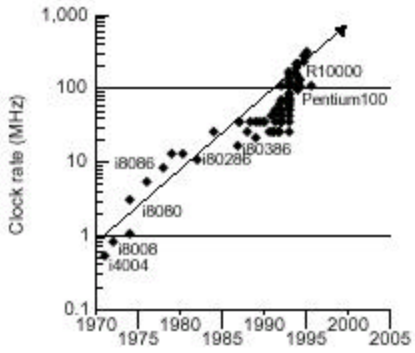

Figure 3 Clock Frequency Growth Rate

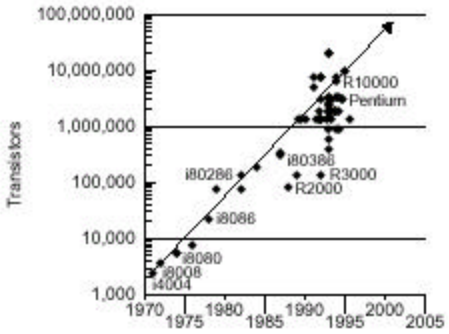

Figure 4 Transistor Count Growth Rate

Besides, DRAM density quadruples in three to four years; disk density quadruples in two years; network technology grows faster than before. Cost decreases at the rate at which density increases.

1.2 Application Demand Trends

Parallel computing is inevitable in the future. Performance is always the objective. Higher computational performance is always welcome in scientific computing. Today's research in human genome, fluid turbulence, vehicle dynamics, ocean circulation, superconductor modeling and some other fields requires 1TFLOPS computational performance requirement and 1TB storage requirement. It is

harder and harder to satisfy such requirement using uniprocessors.

Some general-purpose computing also requires high computing performance. For example, engineering computing (CAD, database, visualization, image processing…) has higher and higher computing performance demands (Figure 5 is the computing demand of speech and image processing before 1995). Commercial computing (transaction processing, data mining…) relies on parallelism for high end because the computational power determines the ability to handle business.

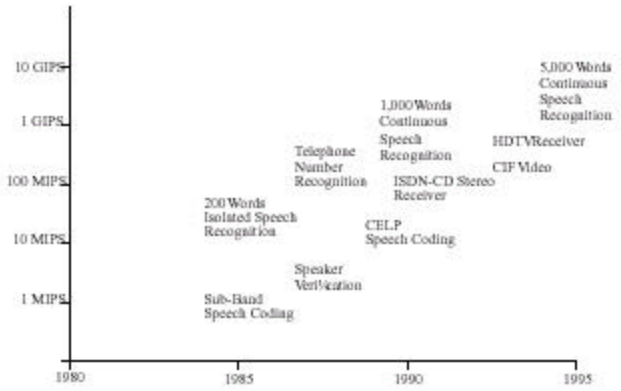

Figure 5 Computing demand of speech and image processing

Some embedded systems, especially some mobile terminals such as handheld devices, require significant computational power. Moreover, low power consumption and low cost are usually strict specification constraints.

Currently, parallel computing has occurred in scientific and engineering computing. In commercial computing, parallel computing is more and more popular. Using large-scale multiprocessors instead of supercomputers is already under way. The demand for improving throughput on sequential workloads maybe is the greatest use of small-scale multiprocessors.

1.3 Architecture Trends

In the last two decades, performance growth of uniprocessors exploiting instruction-level parallelism, driven by the microprocessors, was at its highest rate since the first transistorized computers in the late 1950s and the early 1960s. This rapid rate of performance growth will continue at least for the next 5 years. Instruction-level parallelism is still valuable but limited. Coarser-level parallelism (e.g. thread-level parallelism) is the most viable approach. Bus technology and

processor performance accelerate the advances of each other.

From all of the above trends in technology, application demand and architecture, we can see that parallelism is more and more important and multiprocessor, as a cost-effective parallel architecture, is increasingly attractive, not irrelevant. Today's microprocessor has multiprocessor support. Many servers and workstation are becoming multiprocessors to get higher performance at reasonable cost. We can say that multiprocessor will definitely have a greater role in the future. Tomorrow's microprocessors will be multiprocessors.

2. Counter Arguments and Challenges in Multiprocessors

The rapid performance growth in uniprocessor architecture and the obstacles in wide-spreading multiprocessor architecture support the argument that multiprocessor is irrelevant.

2.1 Rapid Performance Growth in Uniprocessor

As we said in the last section, performance growth of uniprocessors exploiting instruction-level parallelism, driven by the microprocessors, was at its highest rate since the first transistorized computers in the late 1950s and the early 1960s. This rapid rate of performance growth will continue at least for the next 5 years.

However, the view that advances in uniprocessor architecture were nearing end has been widely held at varying times.

In 1972, W.Jack Bouknight et al. " The turning away from the conventional organization came in the middle 1960s, when the law of diminishing returns began to take effect on the effort to increase the operational speed of a computer…. Electronic circuits are ultimately limited in their speed of operation by the speed of light…and many of the circuits were already operating on the nanosecond range."

In 1989, Angel L. DeCedama, "…sequential computers are approaching a fundamental physical limit on their potential computational power. Such a limit is the speed of light."

In 1989, Divid Mitchell, "…today's multiprocessors…are nearing an impasses as technologies approach the speed of light. Even if the components of a sequential processor could be made to work this fast, the best that could be expected is no more than a few million instructions pre second."

From the above views, we can see that the improvement in the performance of uniprocessor architecture is limited. However, people always have limitless thirst for higher performance. Multiprocessor is a logical way to improve performance beyond a uniprocessor by connecting multiple processors together. It adds a new dimension to the design space of computer architecture—the number of processors. The design is driven by the demand for performance at acceptable cost. Since the cost of computer hardware has dropped, multiprocessor architecture is likely to be more cost-effective than designing a custom processor.

Although the performance improvement of the uniprocessor is limited, we cannot predict the death of advances in uniprocessor architecture. Whether the rapid performance growth pace of this architecture can be sustained more than 5 years is also difficult to predict but hard to bet against. There is still tremendous revenue stream for further development. However, the pace of the progress in uniprocessor will eventually slow down, multiprocessor architecture will become increasingly attractive at that time.

2.2 Challenges

To be more cost-effective than the highly developed uniprocessor, multiprocessor has to overcome some obstacles. Two of them are very important:[1]

- The limited parallelism available in programs
- The relatively high cost of communications

2.2.1 Parallelism in programs

The performance gain of multiprocessor is strongly dependent on the characteristic of the program. From Amdahl's law, the speedup of a parallel processing architecture is limited by the fraction of the program that cannot be paralleled executed. That is:

$$\text{Speedup} = 1/(\text{Fraction}_{\text{enhenced}}/\text{speedup}_{\text{enhenced}} + (1 - \text{Fraction}_{\text{enhenced}}))$$

For example, if we want to get a speedup of 80 using 100 processors, only 0.25% of the original computation can be sequential. To get a cost-effective speedup, almost the entire program should be parallel, which is impossible for most of the current applications. However, there are many programs with natural parallelism in some application areas, such as server and embedded applications. We can also write software using some new algorithms to exploit the parallel performance of multiprocessor.

2.2.2 Communication cost

The communication cost between processors is affected by the communication mechanism, interconnection network and scale of the multiprocessor. Long communication delays have substantial effect on the performance of multiprocessor. For example, in existing shared-memory multiprocessors, communication of data between processors may cost anywhere form 100 clock cycles to 1000 clock cycles. Besides the time to require and transmit the data in remote reference, contention caused by many references trying to use the global interconnect can lead to increased delays.

We can reduce the impact of long remote latency by the architecture and by the programmer. Caching shared data and reconstructing the data can be used to reduce the frequency of remote accesses. Communication delay hiding overlaps communication with computation or with other communication.

2.2.3 Other obstacles

The additional obstacles include load balance, synchronization, memory management and data distribution. Data distribution is critical for the performance of some applications executed on multiprocessors. Good data distribution can bring us super-linear performance improvement over uniprocessors. However, even in “sequential” computers, we have to handle some similar issues, such as resource arrangement among functional units, memory sharing, caches exploiting locality and wires providing communication bandwidth.

Although there are various obstacles that make multiprocessors challenging, we can see the slow but steady progress on the obstacles in the last 15 years. New memory sharing techniques and bus techniques decrease the communication cost. New processor scheduling makes better use of the processors such that performance is improved. We can be optimistic about the future of

multiprocessors.

3. Why multiprocessor?

The idea of using multiple processors to increase performance and/or improve availability dates back to the earliest electronic computers. More than 30 years ago, Flynn placed all computers into one of the four categories based on the parallelism in the instructions and data stream called for by the instructions at the most constrained component of the multiprocessor: SISD, SIMD, MISD and MIMD. From 1980s, the smaller size of the microprocessor allowed the memory bus to replace the interconnection network hardware and the portable operating systems eliminated the requirement for a new operating system. Besides, the development of wide, high-speed buses, multiple buses and crossbar interconnects improved the communication delay between processors. Multiprocessor got success in 1990s and it will have more prospective development and use in the future.

Despite the challenges in multiprocessors, we should be optimistic about its future.

3.1 Cost-Efficiency

Building a multiprocessor or a cluster is a more effective way to build a computer that offers more performance than that achieved with a single-chip microprocessor.

In 1967, Amdahl said “For over a decade prophets have voiced the contention that the organization of a single computer has reached its limits and that truly significant advances can be made only by interconnection of a multiplicity of computers in such a manner as to permit cooperative solution…. Demonstration is made of the continued validity of the single processor approach.”[1]

Microprocessor remains the dominant uniprocessor technology. As we say in the first section, uniprocessor architecture got its highest performance growth rate during 1985-2000 owning to the development of microprocessor. But from the views in the first section, we can also see that the limit of the performance improvement of uniprocessor. What is more, lots of the previous architectural innovation is based on increased exploitation of instruction-level parallelism. Improving instruction-level parallelism means to supply the processor with data

and instruction fast enough to keep it busy. In order to satisfy the increasing instruction and data bandwidth requirement, larger and larger caches were placed on chip with the processor and more and more transistors were consumed. The path between the cache and the processor should be very wide to satisfy the requirement of multiple instruction and data accesses per cycle. Cache miss has more significant effect on the performance because when a value is required by an instruction and is loaded from memory, the processor may have to wait for the latency of a cache miss. Even if some instruction processing mechanisms were exploited to reduce such latency, modern multiple-issue processors are more and more complex, and the gained performance through increasing complexity, silicon and power seems to be diminishing. It is doubtful that the uniprocessor performance growth can be maintained for more than 15 years. We need coarser-level parallelism to get more performance.

Multiprocessor is a logical way to improve performance beyond a uniprocessor by connecting multiple microprocessors together. It makes thread-level even process-level parallelism easier to be implemented. The hardware cost of the multiprocessor is the sum of the cost of the microprocessors and the wires connecting the microprocessors. Using multiprocessors, we can complete large and complicated scientific computation efficiently on some low cost microprocessors. To achieve the same performance on a uniprocessor, this uniprocessor must be very complex and it takes architects much time and energy to implement such unoprocessor or even it is impossible to get such high-performance uniprocessor since the performance improvement is limited and the reason from the former paragraph. So, multiprocessor is a more cost-effective way to offer better performance than uniprocessor.

3.2 Wider Use and Proved Efficiency in Commercial Workloads

The use of parallel processing in some domains is beginning to be understood and multiprocessors are highly effective for multiprogrammed and certain intensive commercial workloads

There are many domains that have lots of natural parallelism. One example is the domain of scientific computing. In this domain, more computations are always expected. Figure 6 is the speedup with the number of processors for MICOM.[2] Another example is the large-scale database and transaction-processing systems, in which independent requests can be processed in parallel. Using multiprocessor

architecture in these areas is no doubt a good idea. We have seen the computational performance requirements in the first section. Since the performance improvement of uniprocessor is not unlimited and the growth pace will slow down, using multiprocessors to satisfy the requirement is inevitable.

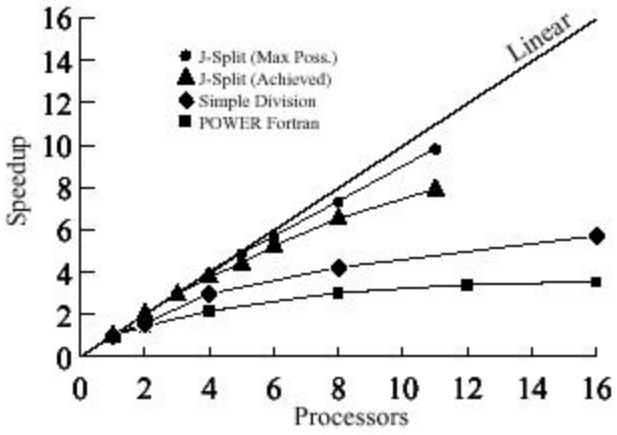

Figure 6 Speedup vs. processors for MICOM

Embedded applications exhibit natural parallelism and multiprocessors are expected to be widespread in this area in the future. In this application area, software is often written from the scratch for some special use, so software compatibility is less relevant than in server and desktop systems. For example, some special-purpose multiprocessor is commonly used in computer graphics, media processing and telecommunication applications. Embedded multiprocessors built from several general-purpose processors also appear in the embedded space, e.g. MXP for Voice over TCP. Some embedded systems, especially some mobile terminals such as handheld devices, require significant computational power. Moreover, low power consumption and low cost are usually strict specification constraints. A possible solution for addressing these conflicting needs is the adoption of a simple multiprocessor on a single chip. This reuses the existing low-power processor cells. What is more, on-chip multiprocessor is more sensitive to die cost so that it can make more efficient use of silicon. We can see that in this application domain, the first main obstacle we have talked about is not a hard problem now. But the communication cost is still a hard one. Maybe on-chip multiprocessor design can solve this problem and simplify the memory management in multiprocessors. [4][5][8]

In the future, workloads using mainstream and large servers (file servers and Web servers) may be a large portion of the market for high-performance multiprocessors. Such workloads have independent batch jobs that can be

processed independently on the processors. Multiprocessors are proved to be highly effective for some intensive commercial workload, such as OLDP, DSS and large-scale, web searching applications.

3.3 Slow but steady progress on major obstacles

From the second section, we know that the major obstacles in wide-spreading use of multiprocessors are the limited parallelism available in programs and the relatively high communication cost. New algorithms that lead to better parallel software can solve the first obstacle. As to the second one, there are many techniques to reduce the impact of inter-processor communication.

Based on the observation that large, multiple level caches can substantially reduce the memory bandwidth demands of a processor, small-scale multiprocessors share a single memory connected by a shared bus and they are extremely cost-effective provided that a sufficient amount of memory bandwidth exists. Caching is used to reduce the memory bandwidth demands. Cache coherence is an inevitable problem here and various protocols (e.g. directory-based protocols and snooping protocols) are exploited to enhance cache coherence. At the same time, user-level software routines constructed on basic hardware primitives are used to solve the potential performance bottleneck in large-scale multiprocessors—synchronization. [3][4][6]

4. Conclusions

Despite the rapid performance growth rate of uniprocessors in the last 20 years, the improvement of uniprocessors is limited. The inefficient use of silicon area, external connections and power leads to diminishing returns that have slowed the rate of performance growth. We have to invent new computer architecture to sustain the rapid growth rate. Multiprocessors are more cost-effective way to get higher performance and have been proved highly effective in certain application areas. The steady progress in attacking the obstacles also makes us optimistic about the future of the multiprocessors. [1]

References

[1] John L. Hennessy, David A. Patterson *Computer Architecture: A Quantitative Approach.* Morgan Kaufmann Publishers, 2002

[2] Aaron Sawdey, Matthew O'Keefe, Rainer Bleck, Robert W. Numrich *The Design, Implementation, and Performance of a Parallel Ocean Circulation Model*

[3] James K Archibald *A Cache Coherence Approach For Large Multiprocessor Systems* ACM 1988

[4] Der-Lin Pean, Cheng Chen *Enchanced Linked-Based Cache Coherence Protocols With A Hardware Mechanism To Reduce The Migratory Sharing Overhead.* International Journal of High Speed Computing, Vol. 11, No. 4 (2000) 223-252

[5] Kunle Olukotun, Basem A. Nayfeh, Lance Hammond, Ken Wilson, and Kunyung Chang *The Case for a Single-Chip Multiprocessor* Proceedings Seventh International Symp. Architectural Support for Programming Languages and Operating Systems (ASPLOS VII) Cambridge, MA, October 1996

[6] Brian N. Bershad, Thomas E. Anderson, Edward D. Lazowska, And Henry M. Levy *User-Level Interprocess Communication for Shared Memory Multiprocessors* ACM Transactions on Computer Systems, Vol 9, No. 2, May 1991, Pages 175-198

[7] Julita Corbalán González *Coordinated Scheduling and Dynamic Performance Analysis in Multiprocessor Systems*. Ph.D. thesis

[8] Alessio Bechini, Cosimo Antonio Prete *Evaluation Of On-Chip Multiprocessor Architectures For An Embedded Cartographic System. Proc*. of IASTED Applied Informatics Conference, Symposium on Software,Innsbruck (Austria), Feb. 2001, pp. 686-691.