Survey on Microprocessor Architecture and Development Trends

YaoYingbiao, Zhang Jianwu
College of Telecommunication Engineering,
Hangzhou Dianzi University,
Hangzhou, China

Zhao Danying
Department of Communication Service
PetroChina Tarim Oilfield Company
Xinjiang, China

Abstract—**To improve the performance of microprocessor, many kinds of novel architecture, such as multi-thread processor, CMP, PIM, and reconfigurable computing processor, have been proposed. These new processors improve performance mainly dependant on making use of all kinds of parallelism of workloads, solving the speed mismatch between processor and external memory, reconfigurable computing, and etc. This paper summarizes characteristic of these kinds of architecture, and predicts the development trends of microprocessor in the future.**

Keywords - Microprocessor; Architecture; Parallelism; Performance

I. INTRODUCTION

At present, the microprocessor architecture is facing new challenges and new opportunities. On the one hand, integrated circuits will continue to be sustained rapid development by Moore's law, predicted in 2011 [1] that the number of on-chip transistors can reach 14 billion and the chip feature size can reach 50 nm or smaller; On the other hand, with the rapid development of applications such as networks and multimedia, these applications have an urgent need for microprocessors with the abilities of real-time and streaming data processing, higher storage and I/O bandwidth, low power and low design complexity, and etc. In this case, in order to effectively use the massive on-chip transistor resources and to improve the microprocessor performance and reduce power consumption, architects are seeking new microprocessor architectures.

The fundamental parameter for evaluating the performance of microprocessors is the running time of the programs, which can be calculated according to the following equation [2]:

$$\text{CPU-time} = \text{IC} \times \text{CPI} \times \text{CCT}$$

Where, IC is executing instruction counts of programs, CPI is average cycles per instruction and CCT is clock cycle time of microprocessors. In the microprocessor design, architects must try their best to reduce the value of IC, CPI and CCT, in order to shorten the running time of programs and to improve the microprocessor performance.

II. CHALLENGES OF MICROPROCESSORS

An important traditional method to enhance processor performance is to reduce its CCT. For a long time, reducing the feature size of chips is an effective way to increase the frequency of the microprocessor [3]. However, with the chip fabrication technology coming into the nanometer era, gate delays are simultaneously reducing while wire delays are not when fabrication technology becomes smaller. Wire delays, especially global wires, become the main delays of a chip [4]. When the fabrication technology further becomes smaller, the situation will worsen. Another way to increase the frequency of processors is to increase the pipeline stages of microprocessors, dividing the stage with long time into several stages with short time. When the number of gate levels of each stage reaches 8~16, close to the lower threshold, the cost of the super-pipeline techniques is becoming larger and the expenses of wrong branches are increasing. Therefore, increasing stages of pipeline is about to terminate. Research in [5] shows that the performance do not increase instead of decrease after the depth of pipeline of microprocessors has reached 22.

Increasing the microprocessor frequency not only brings performance improvements, however, it also brings several negative effects such as increasing power, widening processor-memory speed gap and etc [6]. As the microprocessor power consumption is proportional to its frequency, the increasing frequency also increases power consumption in proportion at the same time, which leads to chip overheating, signal noise increasing, the stability of the device decreasing, the chip working abnormally even burning. Especially for battery-powered embedded applications, low-power has become the primary consideration for microprocessor design except for performance requirements. Over the past 30 years, processor speed has 60% annual growth, but memory access delays improve only 7% per year. The speed gap between the processor and memory has become one of the obstacles for improving system performance. As a result, the method by increasing the frequency of microprocessors to improve performance of has come to end.

III. ADVANCED MICROARCHITECTURES OF PROCESSORS

A. Instruction Level Parallelism (ILP)

A.1 Superscalar processors

By using larger instruction issue window, the hardware of superscalar processors can automatically find the available parallel instructions in instruction issue window, then issue them into function units to execute. Comparison with the

978-1-4244-2251-7/08/$25.00 ©2008 IEEE

traditional single-issue RISC processor, superscalar processing features are as follows [7, 8]:

- At each clock cycle, several instructions can be issued which are dynamically decided by hardware. The minimum of issued instructions is 0 and the maximum of issued instructions is the issued width of processors.
- The programming model of superscalar processor is still the serial programming model, so it must ensure the serial completion of programs. The instruction execution of superscalar processor is divided into three phases: instruction issuing, instruction executing and instruction completing.
- Employing many function units. The number of function units is at least the width of instruction issue of processors. In addition, due to out-of-order execution of instructions, the internal circuits of superscalar processors for data conflict detection, data bypass and instruction issue are more complex than those of single-issue processors.
- The most important point is that superscalar technique belongs to microarchitecture improvement and does not belong to instruction set architecture improvement.

The leading research work related with superscalar processors include: the advanced superscalar processor [9] proposed by Patt YN, the superspeculative processor proposed by Lipasti MH [10], the trace processor proposed by Smith JE [11], and etc. The common ideas of these studies are to use a broader superscalar, more function units, multi-level cache, more radical forecast for data, instruction and control, to obtain as much as possible ILP. The issues of these methods are: on the one hand, the internal ILP of a single application is limited, and the potential for improved performance will soon reach the limit; on the other hand, the complexity of chips exponentially increases, which will make the cost of chip design, verification and test unacceptable.

A.2 VLIW

VLIW processors find available parallel execution instructions by their compilers and these instructions are packaged as a very long instruction word, and then the hardware issues instructions in the same package into function units to execute at the same time. VLIW processors are mainly used in media applications, for example, the TriMedia [12] processor family of Philips. The main characteristics of VLIW processors are as follows:

- VLIW techniques belong to instruction set architecture techniques. The length of the instructions of VLIW is general from 128bit to 1024bit and each instruction includes a number of parallel operations.
- Long instructions are statically scheduled by the compiler and the number of issued instructions at each cycle is also decided at the compile-time and a central control unit is responsible for in-order issue of long instructions.
- Because the number of short instructions in each long instruction is fixed, when the compiler can not find enough short instructions for parallel execution, the compiler inserts NOP instruction into long instruction which makes their code length increasing.

VLIW processors essentially rely on compilers extracting the greatest possible explicit parallelism from the application and then package available parallel instructions into VLIW for simplifying the complexity of control logics. The main problem of this method is: the complexity of software increasing; some related events at compile time can not be identified; the code compatibility issues. The representative of VLIW is EPIC-based Merced processor jointly developed by HP and Intel [13].

B. Thread level parallelism (TLP)

For the next generation high-performance processor, parallelism should not be limited to the fine-grained ILP of single program. In fact, there exist many forms of coarse-grained thread-level parallelism in a lot of workloads. At present, the use of TLP to improve the performance of the processor can be divided into three categories: multithreaded processor (MTP), chip multiprocessor (CMP) and simultaneous multithreading processor (SMT).

B.1 MTP

Multithreaded processor is designed to reduce the effect on performance of long delays such as cache miss. Usually MTP maintains the independent PC (Program Counter) and registers for each thread at the same time. MTP has a trigger mechanism for thread switch and the cost of thread switch is as possible as small, for example, 0-cycle switch loss. MTP can be divided into fine-grained and coarse-grained multithreaded processor. Fine-grained MTPs carry on thread switch for every clock cycle; coarse-grained MTPs carry on thread switch at meeting long delays in running, otherwise, they execute the instructions from the same thread. Coarse-grained MTPs exist two types of thread-switching mechanism, static and dynamic thread switch. Static thread switch is determined by the compiler and the need for a thread switch is obvious at instruction fetch stage which means very low switch losses. Dynamic thread switch is automatically detected pipeline events by hardware during instruction execution. After meeting switch events, the hardware automatically complete thread switch. Because instructions can be fetched from a number of separate threads, MTP can take advantage of TLP to improve its performance. The goal of MTPs is to speed up processing speed of multi-threaded workload. MTPs are also divided into single-issue or multi-issue processor shown in Figure 1 and 2. The typical processors of MTPs are the fine-grained MTA of Tera [14], loading data based thread switch of Rhamma in Karlsruhe [15], and etc.

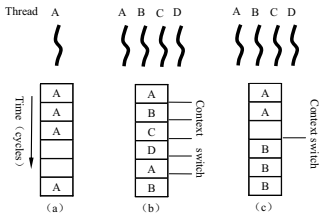

Figure 1. Instruction issue comparison of single-issue processors: (a) Single-issue processor, (b) fine-grained MTP, (c) coarse-grained MTP

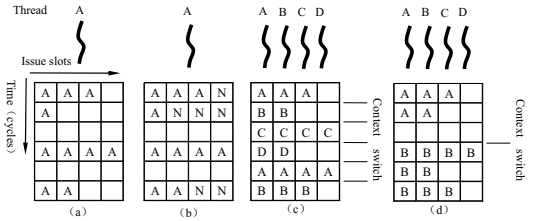

Figure 2. Instruction issue comparison of multi-issue processors: (a) Superscalar, (b) VLIW, (c) fine-grained MTP, (d) coarse-grained MTP

B.2 CMP

With the number of transistors integrated on a single chip increasing, integrating several processors into a chip becomes possible. Based on its memory organization methods, CMP can be divided into three categories: SMP (Symmetric Multi-Processor), DSMP (Distributed Shared Memory Multi-processor) and MPSNMP (Message Passing Shared-nothing Multiprocessor). In the structure of the SMP, all processors share a global memory; every word in the memory has the same address in all processors. In the structure of DSMP, each processor has its own local memory, and can visit local memory of the other processor by connect unit. In general, the speed of visiting its local memory is much higher than that of visiting the other processor's memory. In the structure of MPSNMP, each processor can only access its own local memory, and communication between the processor is completed by the message transmission mechanism. The main features of the CMP are as follows [16]:

- CMP is a microarchitecture technique and benefits from the coarse-grained parallelism of applications. When its applications have several programs, different programs can be as different threads and can be allocated to different processors to run at the same time.
- When a single application is running on CMP, it must be able to extract enough parallel threads, then these parallel threads are allocated to different processor to run. When the application can not be extracted in parallel threads, the hardware resources will be wasted.
- The single processor of CMP architecture is generally relatively simple, for example, the simple single-issue processor or the simple dual-issue processor.
- The difficulty in the hardware design of CMP architecture does not lie in a single processor design and verification, but lies in flexible communications between the processor cores, data transmission and etc; the difficult in the software design is the scheduling and control of task.

B.3 SMT

The concept of SMT is the first proposed by Tullsen [17] in 1995, and the basic idea is: issuing instructions from different threads into function units at each clock cycle and enhancing the utilization of function units. SMT combines features of the superscalar and multithreading processors and can reduce the waste of horizontal and vertical issue:

- SMTs allow executing several instructions from different threads at each clock cycle. Therefore, in a clock cycle, SMT is able to use the TLP and ILP of programs to eliminate the waste of horizontal issue, and increases the real width of issue instruction and enhances the utilization of function units.
- In theory, SMT allow issuing any instruction combination of any active thread. When one issue operation from one thread is stalled because of long delays or resource conflicts, the other threads can use all available issue slots. Therefore, vertical waste can be eliminated through the use of non-blocking instructions of other threads.

Figure 3 is the comparison of instruction issue in SMT and CMP which shows that the issue efficiency of SMT is significantly higher than the efficiency of CMP. The efficiency of SMT is at the cost of the complex microarchitecture design, especially its instruction fetching unit [18]. From a purely microarchitecture point, SMT processor has a very good flexibility. However, when semiconductor technology entering the 0.18-um, the wire delays is greater than the gate delays. A new trend in processor design is employing a number of smaller, more localized basic units to implement complex processors. Compared with SMT, due to the structure of the CMP has been divided into a number of cores and each core is relatively simple, CMP is easy to optimize the design and is more promising.

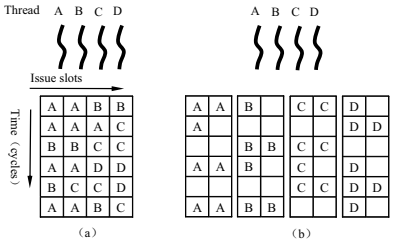

Figure 3. The comparison of instruction issue in SMT and CMP: (a) SMT, (b) CMP

C. Processor in memory (PIM)

Using ILP, TLP techniques can greatly increase instruction execution parallelism, however, the supply of instructions and data play an important role for the performance of these techniques. Traditional processor-centric design has led to the complex solution to the issue of memory access long delays (such as the complex cache mechanism). Many studies have already committed to reduce or mitigate memory access long delays, such as lookup-free cache, data and instructions pre-fetching by hardware and software, out-of-order instruction fetching, speculative execution, multi-threading execution, and so on. Even so, the speed gap between processor and memory is still increasing, which makes memory access speed become a major bottleneck for improving processor performance in the future. Based on current trends in technology development, in the near future when the processor issues hundreds or even thousands of instructions, it fetches only one data or instruction into on-chip memory. According to these observations as well as the ability of semiconductor technology in the future, PIM technique proposes integrating processor and memory into the same chip. The main features of PIM are as follows:

- On-chip memory support high-bandwidth, low latency instructions and data access.

- In most cases, the entire application can be put in on-chip memory during program running. Therefore, it can reduce performance loss due to access to external memory.
- After memory is integrated into chip, the pins which are originally for memory bus can be saved for increasing the I/O bandwidth of processors.
- Due to the reduction of off-chip memory access, the power consumption of chips will reduce significantly.

The most representative of PIM research is IRAM [19], proposed by Patterson at the Berkeley University. Their basic design ideas are: facing the vector processing to satisfy the needs of a variety of multimedia applications; integrating DRAM and calculation logic on a single chip to satisfy the requirements of low power, small size, light weight.

D. Processors with reconfigurable computing

At present, chips for calculation include two categories: dedicated ASIC and microprocessors. Microprocessor-based architecture can be used in almost all kinds of applications through various software algorithms, which resulted in a wide range of flexibility but also resulted in low performance and high power consumption for many applications. Dedicated ASIC is designed for a specific algorithm and its internal structure can perfectly match algorithm, so the calculation is fast and efficient. However, ASIC circuits can not be amended after they are manufactured. If the new algorithm appears, ASIC must be re-designed and re-manufactured. In recent years, the rapid development of reconfigurable devices greatly makes up deficiencies of the dedicated ASIC. Through large-scale FPGA, CPLD can get closer to the computing power of dedicated ASIC, but also can re-configure new functions after the application environment changes. Therefore, they can possess both merits of microprocessor and ASIC and become a new trend in processor design.

Reconfigurable microprocessors are usually organized as the microprocessor with reconfigurable logic. The combination of reconfigurable logic and microprocessor has four ways according to the close degree of integration, which are: reconfigurable logic as a function unit; reconfigurable logic as the co-processor; reconfigurable logic as the external subsidiary processing unit; reconfigurable logic independent of the processor, through the I/O interface to complete collaborative computing. The representative of reconfigureable systems is Garp proposed at Berkeley University [20].

IV. CONCLUSIONS

Due to consideration of wire delays and the chip power consumption, the traditional method by increasing the frequency of the processor to improve performance has come to end. The new methods mainly use the ILP, DLP and TLP of applications; solve processor-memory speed gap; employ reconfigurable computing; and so on.

The goal of microarchitecture design of current processors has changed from the high frequency clock to the high throughput. The design emphasis has become from pursuit of peak performance of single application to consideration of the performance, power consumption, adaptability, and many other factors and architects pay more attention to the chip resource utilization and the ratio between performance and energy.

The new processor architectures have higher demands for compiler techniques, for example, the CMP and multithreaded processors require the use of multi-threaded programming model; reconfigurable computing require hardware and software hybrid programming model; and etc. The emergence of new programming model increases dependence on the compiler and their performance improvement is directly related to the performance of compiler.

REFERENCES

[1] Doug Burger and James R. Goodman. “Billion-Transistors Architectures: There and Back Again”. IEEE Computer, Vol. 37, No. 22, pp. 22-28, 2004.

[2] J. L. Hennessy and D. A. Patterson, *Computer Architecture: A Quantitative Approach*, 3rd edition, Morgan Kaufmann Publishers, Inc, 2002.

[3] J. M. Rabaey, A. Chandrakasan, and B. Nikolic, *Digital Integrated Circuits: A design Perspective*, 2nd edition, Prentice Hall, 2003.

[4] D.Sylevester and K. Keutzer. “A Global wiring Paradigm for Deep Submicron Design”. IEEE Trans on CAD/ICAS, Vol. 19, No. 2, pp. 242-252, 2000.

[5] Michael Flynn and Patrick Hong. “Microprocessor Design Issues: Thoughts on the Road Ahead”. IEEE Micro, Vol. 25, No. 3, pp. 16-31, 2005.

[6] Silc J, Ungerer T, and Robic B. “A Survey of New Directions in microprocessors”, Microprocessors and Microsystems, Vol. 24, pp. 175-190, 2000.

[7] John Paul Shen and Mikko H.Lipasti, *Modern Processor Design: Fundamentals of Superscalar Processors*, McGraw-Hill, 2003.

[8] Silc J, Robic B, and Ungerer T, *Processor Architecture: From Dataflow to Superscalar and Beyond*, Springer-Verlag, March 1999.

[9] Patt Y.N, Patal S.J, and Evers M, "One Billion Transistors, One Uniprocessor, One Chip", *Computer*, Vol. 30, No. 9, pp. 51-57, 1997.

[10] Lipasti M.H and Shen J.P, "Superspeculative Microarchitecture for Beyond AD 2000", *Computer*, Vol. 30, No. 9, pp. 59-66, 1997.

[11] Smith J.E and Vajapeyam S, "Trace Processors: Moving to Fourth Generation Micro- architevtures", *Computer*, Vol. 30, No. 9, pp. 68-74, 1997.

[12] TriMedia processor, http://www.trimedia.philips.com/.

[13] Schlansker M.S and Rau B.R, "EPIC: Explicitly Parallel Instruction Computing", *Computer*, Vol. 33, No. 2, pp. 37-45, 2000.

[14] Bianchini, R.P.; Kim, H.S. “The Tera project: a hybrid queueing ATM switch architecture for LAN”, IEEE Journal on Selected Areas in Communications, Vol. 13, No. 4, pp. 673 – 685, 1995.

[15] Grunewald, W.; Ungerer, T. “A multithreaded processor designed for distributed shared memory systems”. Proceedings Advances in Parallel and Distributed Computing, pp. 206 – 213, 1997.

[16] Flachs B, Asano S, Dhong S.H, and et al. “The microarchitecture of the synergistic processor for a cell processor”, IEEE Journal of Solid-State Circuits, Vol. 41, No. 1, pp. 63-70, 2006.

[17] Tullsen, D.M.; Eggers, S.J.; Levy, H.M. “Simultaneous multithreading: Maximizing on-chip parallelism”, Proceedings 22nd Annual International Symposium on Computer Architecture, pp. 392 – 403, 1995.

[18] Yingmin Li, Skadron K, Brooks D, and Zhigang Hu. “Performance, energy, and thermal considerations for SMT and CMP architectures”. The 11th International Symposium on High Performance Computer Architecture, pp. 71-82, 2005.

[19] Patterson D, Anderson T, Cardwell N, and et al. “A case for intelligent RAM”, IEEE Micro, Vol. 17, No. 2, pp. 34-44, 1997.

[20] Callahan, T.J.; Hauser, J.R.; Wawrzynek, J. “The Garp architecture and C compiler”, Computer, Vol. 33, No. 4, pp. 62 – 69, 2000.