DESIGN PRINCIPLES FOR TODAY'S MICROPROCESSORS

Noiki Oluwamuyiwa A.

Adekunle Y.A

Department of Computer Science, Babcock University, Ilishan-Remo, Ogun State, Nigeria

Abstract

This article presents an overview of issues to address before embarking on the production of any processor. Processors used in systems must provide highly energy-efficient operation, with regards to the importance of battery utilization, without compromising high performance when the user requires it.

Today's microprocessors are the powerful descendants of the von Neumann computer. The so called von Neumann architecture is characterized by a sequential control flow resulting in a sequential instruction stream. A program counter addresses the next instruction if the preceding instruction is not a control instruction such as a jump, branch, subprogram call or return. An instruction is coded in an instruction format of fixed or variable length, where the op-code is followed by one or more operands that can be data, addresses of data, or the address of an instruction in the case of a control instruction. The op-code defines the types of operands. Code and data are stored in a common storage that is linear, addressed in units of memory words (bytes, words, etc.).

Keywords: Microprocessors, Computer components, Computer architecture

Introduction

The sequential operating principle of the von Neumann architecture is still the basis for today's most widely used high-level programming languages, and even more astounding, of the instruction sets of all modern microprocessors. While the characteristics of the von Neumann architecture still determine those of a contemporary microprocessor, its internal structure has considerably changed. The main goal of the von Neumann design - minimal hardware structure - is today far outweighed by the goal of maximum performance. However, the architectural characteristics of the von Neumann design are still valid since the sequential high-level programming languages that are used today follow the von Neumann architectural paradigm.

Current superscalar microprocessors are a long way from the original von Neumann computer. However, despite the inherent use of out-of-order execution within superscalar microprocessors today, the order of the instruction flow as seen from outside by the compiler or assembly language programmer still retains the sequential program order - often coined result serialization - defined by the von Neumann architecture. At the same time today's microprocessors strive to extract as much fine-grained or even coarse-grained parallelism from the sequential program flow as can be achieved by the hardware. Unfortunately, a large portion of the exploited parallelism is speculative parallelism, which in the case of incorrect speculation, leads to an expensive rollback mechanism and to a waste of instruction slots. Therefore, the result serialization of the von Neumann architecture poses a severe bottleneck. At least four classes of future possible developments can be distinguished; all of which continue the ongoing evolution of the von Neumann computer:

1. Micro-architectures that retain the von Neumann architecture principle (the result serialization), although instruction execution is internally performed in a highly parallel fashion. However, only instruction-level parallelism can be exploited by the contemporary microprocessors. Because instruction-level parallelism is limited for sequential threads, the exploited parallelism is enhanced by speculative parallelism. Besides the superscalar principle applied in commodity microprocessors, the super--speculative, multi-scalar, trace, and data-scalar processor principles are all hot research topics. All these approaches belong to the same class of implementation techniques because result serialization must be preserved. A reordering of results is performed in a retirement or commitment phase in order to fulfill this requirement.
2. Processors that modestly deviate from the von Neumann architecture but allows the use of sequential Von Neumann languages. Programs are compiled to the new instruction set principles. Such architectural deviations include very long instruction word (VLIW), SIMD in the case of multimedia instructions, and vector operations.
3. Processors that optimize the throughput of a multiprogramming workload by executing multiple threads of control simultaneously. Each thread of control is a sequential thread executable on a von Neumann computer. The new processor principles are the single-chip multiprocessor and the simultaneous multithreaded processor.

4. Architectures that break totally with the von Neumann principle and that need to use new languages, such as dataflow with dataflow single-assignment languages, or hardware - software co-design with hardware description languages. The processor-in-memory, reconfigurable computing, and the asynchronous processor approaches also point in that direction.

Review of Literature

The microprocessor consists of a number of units/components that work together in making sure all instructions done on the computer are carried out in the right manner and altogether constitute any microprocessor unit. These include: the register, the arithmetic and logic unit (ALU), and logic gates.

The Arithmetic logic unit (ALU) is a digital circuit that performs arithmetic and logical operations (Maini, 2007). It is the heart of the internal architecture of each microprocessor. The internal architecture of ALU is composed of arithmetic units each providing a mathematical function, for example division or subtraction. It has two inputs and one output for the result.

A Register is a small amount of memory integrated into the CPU. However its content is available on the CPU, it can be accessed more quickly than from anywhere else (Wikipedia, 2011a). The length of registers is measured in bits, for example an "8-bit register or a "16-bit register.

A register consists of small memory units called flip-flops. The flip-flop is composed of logic gates to achieve a memory effect (Mano, 1993). Each register is made of number of flip-flops which are logic circuits capable of remembering the previous state. Each flip-flop holds one bit of information. Joining these flip-flops together produces a larger memory called register. (Barboriak, 2004).

The Evolution of Microprocessors

Processors are the brains of computers. Other components allow a computer to store or retrieve data and to input or output data, but the processor perform computations and do something useful with the data. Today microprocessors are everywhere. Supercomputers are designed to perform calculations using hundreds or thousands of microprocessors. Even personal computers that have a single central processor use other processors to control the display, network communication, disk drives, and other functions. In addition, thousands of products we don't think of as computers make use of microprocessors. Cars, stereos, cell phones, microwaves, and washing machines all contain microprocessors.

Some computer chips are designed to perform a single very specific function, but microprocessors are built to run programs. By designing the processor to be able to execute many different instructions in any order, the processor can be programmed to perform whatever functions needed at the moment. The possible uses of the processor are limited only by the imagination of the programmer. This flexibility is one of the keys to the microprocessor’s success. Another is the steady improvement of performance.

Over the last 30 years, as manufacturing technologies have improved, the performance of microprocessors has doubled roughly every 2 years. For most products, built to perform a particular function, this amount of improvement would be unnecessary. Microwave ovens are an improvement on conventional ovens mainly because they cook food more quickly, but what if instead of heating food in a few minutes; they could be improved even more to only take a few seconds? There would probably be a demand for this, but what about further improvements so that it took only tenths of a second, or even just hundredths of a second. At some point, further improvements in performance of a single task become meaningless because the task being performed is fast enough. However, the flexibility of processors allows them to constantly make use of more performance by being programmed to perform new tasks.

All a processor can do is run software, but improved performance makes new software practical. Tasks that would have taken an unreasonable amount of time suddenly become possible. New functionality drives the need for improved performance.

Being designed to run programs allows microprocessors to perform many different functions, and rapid improvements in performance are constantly allowing for new functions to be found. Continuing demand for new applications funds manufacturing improvements, which make possible these performance gains.

Despite all the different functions a microprocessor performs, in the end it is only a collection of transistors and wires. The job of microprocessor design is ultimately deciding how to connect transistors to be able to quickly execute the commands that run programs. As the number of transistors on a processor has grown from thousands to millions that job has become steadily more complicated, but a microprocessor is still just a collection of transistors connected to operate as the brain of a computer. The story of the first microprocessor is therefore also the story of the invention of the transistor and the integrated circuit.

The computers of the 1960s stored their data and instructions in “core” memory. These memories were constructed of grids of wires with metal donuts threaded onto each intersection point. By applying current to one vertical and one horizontal wire a specific donut or “core” could be magnetized in one direction or the other to store a single bit of information. Core memory was reliable but difficult to assemble and operated slowly compared to the transistors performing computations. A memory made out of transistors was possible but would require thousands of transistors to provide enough storage to be useful. Assembling this by hand wasn’t practical, but the transistors and connections needed would be a simple pattern repeated many times, making semiconductor memory a perfect market for the early integrated circuit business.

In 1968, Bob Noyce and Gordon Moore left Fairchild Semiconductor to start their own company focused on building products from integrated circuits. They named their company Intel® (from INTegrated ELectronics). In 1969, Intel began shipping the first commercial integrated circuit using MOSFETs (Metal Oxide Semiconductor Field-Effect Transistor), a 256-bit memory chip called the 1101. The 1101 memory chip did not sell well, but Intel was able to rapidly shrink the size of the new silicon gate MOSFETs and add more transistors to their designs. One year later Intel offered the 1103 with 1024 bits of memory, and this rapidly became a standard component in the computers of the day. Although focused on memory chips, Intel received a contract to design a set of chips for a desktop calculator to be built by the Japanese company Busicom. At that time, calculators were either mechanical or used hard-wired logic circuits to do the required calculations. Ted Hoff was asked to design the chips for the calculator and came to the conclusion that creating a general purpose processing chip that would read instructions from a memory chip could reduce the number of logic chips required. There would be four chips altogether: one chip controlling input and output functions, a memory chip to hold data, another to hold instructions, and a central processing unit that would eventually become the world’s first microprocessor.

The computer processors that powered the mainframe computers of the day were assembled from thousands of discrete transistors and logic chips. This was the first serious proposal to put all the logic of a computer processor onto a single chip. However, Hoff had no experience with MOSFETs and did not know how to make his design a reality. The memory chips Intel was making at the time were logically very simple with the same basic memory cell circuit repeated over and over. Hoff’s design would require much more complicated logic and circuit design than any integrated circuit yet attempted. Months passed and Hoff’s idea could not be implemented yet and Intel struggled to find someone who could implement this idea.

In April 1970, Intel hired Faggin, the inventor of the silicon gate MOSFET. Faggin worked at a fast pace to help validate the design and by February 1971 he had all four chips working. The chips processed data 4 bits at a time and so were named the 4000 series. The fourth chip of the series was the first microprocessor, the Intel 4004.

The 4004 contained 2300 transistors and ran at a clock speed of 740 kHz, executing on average about 60,000 instructions per second. This gave it the same processing power as early computers that had filled entire rooms, but on a chip that was only 24 mm2. It was an incredible engineering achievement, but at the time it was not at all clear that it had a commercial future. The 4004 might match the performance of the fastest computer in the world in the late 1940s, but the mainframe computers of 1971 were hundreds of times faster. By the end of 1971, Intel was marketing the 4004 as a general purpose microprocessor. Busicom ultimately sold about 100,000 of the series 4000 calculators before going out of business in 1974. Intel would go on to become the leading manufacturer in what was for 2003—a $27 billion a year market for microprocessors. The incredible improvements in microprocessor performance and growth of the semiconductor industry since 1971 have been made possible by steady year after year improvements in the manufacturing of transistors.

Moore's law

Since the creation of the first integrated circuit, the primary driving force for the entire semiconductor industry has been process scaling. Process scaling is shrinking the physical size of the transistors and the wires interconnecting them, allowing more devices to be placed on each chip, which allows more complex functions to be implemented. In 1975, Gordon Moore observed that shrinking transistor dimensions were allowing the number of transistors on a die to double roughly every 18 months. This trend has come to be known as **Moore's law**. For microprocessors, the trend has been closer to a doubling every 2 years, but amazingly this exponential increase has continued now for 30 years and seems likely to continue through the foreseeable future. The 4004 used transistors with a feature size of 10 microns (μm). This means that the distance from the source of the transistor to the drain was approximately 10 μm. Ahuman hair is around 100 μm across. In 2003, transistors were being mass produced with a feature size of only 0.13 μm. Smaller transistors not only allow for more logic gates, but also allow the individual logic gates to switch more quickly. This has provided for even greater improvements in performance by allowing faster clock rates. Perhaps even more importantly, shrinking the size of a computer chip reduces its manufacturing cost. The cost is determined by the cost to process a wafer and the smaller the chip, the more that are made from each wafer. The importance of transistor scaling to the semiconductor industry is almost impossible to overstate. Making transistors smaller allows for chips that provide more performance, and therefore sell for more money, to be made at a lower cost. This is the fundamental driving force of the semiconductor industry.

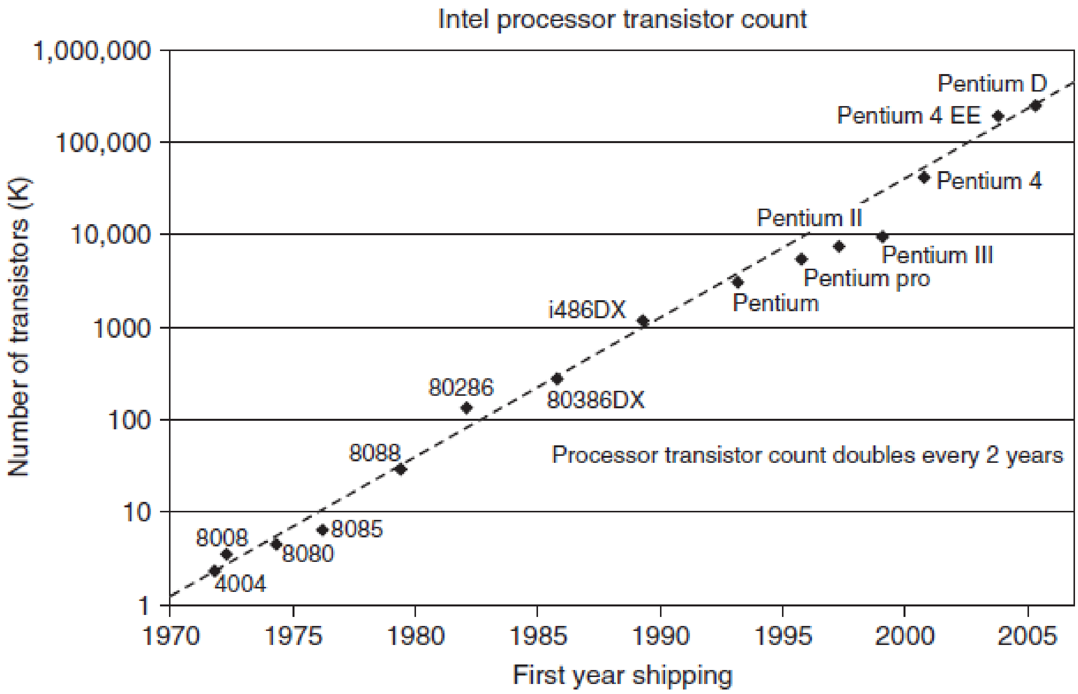

Figure 1: Illustrating Moore's law (source: intel processors)

Microprocessor Design Planning

Transistor scaling and growing transistor budgets have allowed microprocessor performance to increase at a dramatic rate, but they have also increased the effort of microprocessor design. As more functionality is added to the processor, there is more potential for logic errors. As clock rates increase, circuit design requires more detailed simulations. The production of new fabrication generations is inevitably more complex than previous generations. Because of the short lifetime of most microprocessors in the marketplace, all of this must happen under the pressure of an unforgiving schedule. A microprocessor, like any product, must begin with a plan, and the plan must include not only a concept of what the product will be, but also how it will be created. The concept would need to include the type of applications to be run as well as goals for performance, power, and cost. The planning will include estimates of design time, the size of the design team, and the selection of a general design methodology.

Defining the architecture involves choosing what instructions (instruction set) the processor will be able to execute and how these instructions will be encoded. This will determine whether already existing software can be used or whether software will need to be modified or completely rewritten. Because it determines the available software base, the choice of architecture has a huge influence on what applications ultimately run on the processor. Design planning and defining the architecture to be used is the design specification stage of the design process, since completing these steps allows the design implementation to begin. The performance and capabilities of the processor are also in part determined by the instruction set.

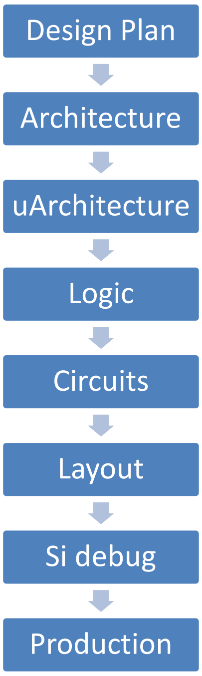

Figure 2 Typical Microprocessor design flow

Processor Roadmaps

The design of any microprocessor has to start with an idea of what type of product will use the processor. In the past, designs for desktop computers went through minor modifications to try and make them suitable for use in other products, but today many processors are never intended for a desktop PC. The major markets for processors are divided into those for computer servers, desktops, mobile products, and embedded applications. Servers and workstations are the most expensive products and therefore can afford to use the most expensive microprocessors. Performance and reliability are the primary drivers with cost being less important. Most server processors come with built-in multiprocessor support to easily allow the construction of computers using more than one processor. To be able to operate on very large data sets, processors designed for this market tend to use very large caches. The caches may include parity bits or Error Correcting Codes (ECC) to improve reliability.

Until recently mobile processors were simply desktop processors repackaged and run at lower frequencies and voltages to reduce power, but the rapid growth of the mobile computer market has led to many designs created specifically for mobile applications. Some of these are designed for “desktop replacement” notebook computers.

Embedded processors are used inside products other than computers. Mobile handheld electronics such as Personal Digital Assistants (PDAs), MP3 players, and cell phones require ultralow power processors, which need no special cooling. The lowest cost embedded processors are used in a huge variety of products form microwaves to washing machines. Many of these products need very little performance and choose a processor based mainly on cost.

Market	**Product**	**Priorities**
Server	High-end server	Performance, reliability and multiprocessing
Desktop	High-end desktop Mainstream Desktop Value desktop	Performance Balanced performance and cost Lowest cost at required performance
Mobile	Mobile desktop replacement Mobile battery optimized	Performance within power limit Power and performance
Embedded	Mobile handheld Consumer electronics and appliances	Ultralow power Lower cost at required performance

Table 1 Microprocessor Markets

In addition to targets for performance, cost, and power, software and hardware support are also critical. Ultimately all a processor can do is run software, so a new design must be able to run an existing software base or plan for the impact of creating new software. The type of software applications being used changes the performance and capabilities needed to be successful in a particular product market. The hardware support is determined by the processor bus standard and chipset support. This will determine the type of memory, graphics cards, and other peripherals that can be used. More than one processor project has failed, not because of poor performance or cost, but because it did not have a chipset that supported the memory type or peripherals in demand for its product type.

Design Types and Design Time

As earlier mentioned, Designs that start from scratch are called lead designs. They offer the most potential for improved performance and added features by allowing the design team to create a new design from the ground up. Of course, they also carry the most risk because of the uncertainty of creating an all-new design. It is extremely difficult to predict how long lead designs will take to complete as well as their performance and die size when completed. Because of these risks, lead designs are relatively rare.

Most processor designs are compactions or variations. Compactions take a completed design and move it to a new manufacturing process while making few or no changes in the logic. The new process allows an old design to be manufactured at less cost and may enable higher frequencies or lower power. Variations add some significant logical features to a design but do not change the manufacturing process. Added features might be more cache, new instructions, or performance enhancements.

Proliferations change the manufacturing process and make significant logical changes. The simplest way of creating a new processor product is to repackage an existing design. A new package can reduce costs for the value market or enable a processor to be used in mobile applications where it couldn't physically fit before. In these cases, the only design work is revalidating the design in its new package and platform.

Design Type	**Typical design time**	**Reuse**
Lead	4 years	Little to no reuse
Proliferation	3 years	Significant logic changes and new manufacturing process
Compaction	2 years	Little or no logic changes, but new manufacturing process
Variation	2 years	Some logic changes on same manufacturing process
Repackage	6 months	Identical die in different package

Table 2 Processor design types and timing

The size of the design team needed will be determined both by the type of design and the designer productivity with team sizes anywhere from less than 50 to more than 1000.

The larger the design team, the more additional personnel will be needed to manage and organize the team, growing the team size even more. For design teams of hundreds of people, the human issues of clear communication, responsibility, and organization become just as important as any of the technical issues of design. The headcount of a processor project typically grows steadily until tape-out when the layout is first sent to be fabricated. The needed headcount drops rapidly after this, but silicon debug and beginning of production may still require large numbers of designers working on refinements for as much as a year after the initial design is completed. One of the most important challenges facing future processor designs is how to enhance productivity to prevent ever-larger design teams even as transistors budgets continue to grow.

The design team and manpower required for lead designs are so high that they are relatively rare. As a result, the vast majorities of processor designs are derived from earlier designs, and a great deal can be learned about a design by looking at its family tree. Because different processor designs are often sold under a common marketing name, tracing the evolution of designs requires deciphering the design project names. For design projects that last years, it is necessary to have a name long before the environment into which the processor will eventually be sold is known for certain. Therefore, the project name is chosen long before the product name and usually chosen with the simple goal of avoiding trademark infringement.

Positions	**Responsibilities**
Computer Architect	Define instruction set and micro-architecture
Logic designer	Convert micro-architecture into RTL
Circuit designer	Convert RTL in transistor level implementation
Mask designer	Convert circuit design into layout
Validation engineer	Verify logical correctness of design at all steps
Design automation engineer	Create and/or support design CAD tools

Table 3 Processor design team jobs

Methodology

Microprocessor Production

Pipelining

Pipelining provides higher performance by allowing execution of different instructions to overlap. The earliest processors did not have sufficient transistors to support pipelining. They processed instructions serially one at a time exactly as the architecture defined.

A simple processor might break down each instruction into four steps (a cycle): fetch, decode, execute, and write. All modern processors use clock signals to synchronize their operation both internally and when interacting with external components.

A pipelined processor improves performance by noting that separate parts of the processor are used to carry out each instruction step. With some added control logic, it is possible to begin the next instruction as soon as the last instruction has completed the first step.

FPGA

Field-Programmable Gate Arrays (FPGA) are programmable logic elements. FPGAs can be designed using a hardware description language (HDL) such as Verilog or VHDL, and that design can be mapped to a hardware design by the HDL synthesizer. FPGAs are quick to design, and because they are reprogrammable, troubleshooting is quick and easy.

Photolithography

The current state-of-the-art process for manufacturing processors and small ICs in general is to use photolithography. Photolithography is a complicated multi-step process.

Wafers

A wafer is a large circular disk, typically made of doped silicon. Each wafer can hold multiple chips arranged like tiles. The number of chips per wafer is known as the yield.

Basic Photolithography

In photolithography, there are typically two important chemicals: an acid and a resist. A photo negative of the design is exposed to light, and the pattern is projected onto the wafer. Resist is applied to the wafer, and it sticks to the portions of the wafer that are exposed to light. Once the resist is applied to the wafer, it is dipped in the acid. The acid eats away a layer of everything that is not covered in resist.

After that, layers of polysilicon, silicon oxide, and metal are added, coating the entire wafer. After each layer of desired material is added, resist and acid are used to "pattern" the layer, keeping the desired regions and removing the undesired regions of that layer.

After all the layers specified by the design have been applied, the wafer is "diced" into individual rectangular "die". Then each die packaged.

Power Dissipation

In addition to power and performance, another useful metric for examining processors is in terms of the amount of power used. Power is a valuable commodity, especially in mobile or embedded environments. Processors that utilize less power are more highly prized in these areas then processors with more capability and better performance.

Heat

In microprocessors, power is mostly dissipated as heat energy. This conversion to heat energy is a function of the size of the wires and transistors, and the operating frequency of the processor. As transistors get smaller, the depletion region gets smaller and current leaks through the transistor even when it is off. This leakage produces additional heat, and wastes additional power.

Heat can also cause materials to expand, which can alter the electrical characteristics of the tiny transistors and wires. Many small microcontrollers do not need to worry about heat because they generate so little, but larger general purpose processors typically need to be accompanied by heat sinks and fans to help cool the processor. If a processor is running too hot, typically it can be slowed down to a lower clock rate to help prevent heat buildup.

Conclusion

Every processor begins as an idea. Design planning is the first step in processor design and it can be the most important. Design planning must consider the entire design flow from start to finish and answer several important questions.

1. What type of product will use the processor?

2. What is the targeted performance and power of the design?
3. What will be the performance and power of competing processors?
4. What previous design (if any) will be used as a starting point and how much will be reused?
5. How long will the design take and how many designers are needed?
6. What will the final processor cost be?

Errors or poor trade-offs in any of the later design steps can prevent a processor from meeting its planned goals, but just as deadly to a project is perfectly executing a poor plan or failing to plan at all. Although in general these steps do flow from one to the next, there are also activities going on in parallel and setbacks that force earlier design steps to be redone. Even planning itself will require some work from all the later design steps to estimate what performance, power, and die area are possible. No single design step is performed entirely in isolation. The real challenge of design is to understand enough of the steps before and after your own specialty to make the right choices for the whole design flow.

References:

Adtek Photomask. “Photomask Manufacturing: Concepts and Methodologies.” 2002.

Advanced RISC Machines, Ltd., *ARM710 Data Sheet,* Technical Document, Dec. 1994.

Amdahl, Gene. “Validity of the single-processor approach to achieving large scale computing capabilities.” *AFIPS Conference Proceedings,* Atlantic City, NJ: 1967.

Baker, Jacob. *CMOS Circuit Design, Layout, and Simulation*. 2d ed., New York: Wiley-IEEE Press, 2004.

Baker, Jacob. *CMOS: Circuit Design, Layout, and Simulation*. 2d ed., New York: Wiley Interscience, 2005.

Bakoglu, H. B. *Circuits, Interconnections, and Packaging for VLSI*. Reading MA: Addison Wesley, 1990.

C.E. Kozyrakis, D.A. Patterson, New direction for computer architecture research, (1998).

Campbell, Stephen. *The Science and Engineering of Microelectronics Fabrication.* 2d ed., New York: Oxford University Press, 2000.

Davis, E. et al. “Solid Logic Technology: Versatile, High Performance Microelectronics.” *IBM Journal of Research and Development*, vol. 8, April 1964.

F. Gabbay, A. Mendelson, The effect of instruction fetch bandwidth on value prediction, in: Proceedings of the ISCA 25, Barcelona, Spain, 1998.

Grant McFarland. “Microprocessor Design: A practical guide from design planning to manufacturing.

Hannemann, Robert et al. *Semiconductor Packaging: AMultidisciplinary Approach*. New York: Wiley-Interscience, 1994.

S. Petters. *Worst Case Execution Time Estimation for Advanced Processor Architectures*. PhD thesis, Technische Universit¨at M¨unchen, Aug. 2002.

S. Sze, *Physics of Semiconductor Devices*, New York: Wiley, 1981.

S. Vajapeyam, T. Mitra, Improving superscalar instruction dispatch and issue by exploiting dynamic code sequences, in: Proceedings of the ISCA 24, Denver, CO, 1997.

S.-M. Kang, Y. Leblebici: *CMOS Digital Integrated Circuits: Analysis and Design*. McGraw-Hill, 1996.

S.W. Moore: *Multithreaded Processor Design*. Kluwer Academic Publishers, 1996.

Semiconductor International Capacity Statistics (SICAS). http://www.sicas.info.

Singer, Pete. “The Interconnect Challenge: Filling Small, High Aspect Ratio Contact Holes.” *Semiconductor International*, 1994.