An Overview of RISC vs. CISC

Alan D. George
Department of Electrical Engineering
FAMU/FSU College of Engineering
Tallahassee, FL 32304

ABSTRACT

One of the most significant recent developments in computer processor technology is the RISC (Reduced Instruction Set Computer) microprocessor. Under certain circumstances, RISC devices offer significant advantages over their conventional CISC (Complex Instruction Set Computer) counterparts. This paper presents a brief comparison of the principal features of both RISC and CISC processors.

Introduction

As hardware technology has changed, so too have the architectural strategies applied to processor design. For many years, the accepted strategy had been to design processors with minimal registers, elaborate addressing modes, and extremely large and complex instruction sets often supported by increasingly enhanced microcode. These early techniques have continually evolved, but much of the original design goals have remained the same even today. Modern CISC processors such as the 68040 and 80486 were designed to operate with hundreds of different variable-length instructions, and they represent the flagships of the two most widely used general purpose microprocessor families in existence.

As microprocessors have evolved, their instruction sets have tended to increase in capability and functionality. For example, the Intel 8080, which was one of the most dominant of the 8-bit microprocessors, has no multiply instruction, limited addressing modes, and a single accumulator register [6]. Today, microprocessors such as the 32-bit chip used in the DEC MicroVAX series of computers have many accumulator-capable registers, as well as a highly robust set of addressing modes and instructions, even including an instruction for polynomial evaluation [4].

However, as semiconductor memory, registers, and optimizing compiler technology has improved, an alternative approach has arisen. Researchers have found that, for certain applications, a processor designed using hard-coded logic (instead of microcode) based on a small and simple instruction set can achieve superior performance. Optimizing compilers are written to take full advantage of these compact and extremely fast instructions, resulting in better optimization and improved execution time. This philosophy, known as RISC, represents a clear divergence from typical CISC processor architectural strategies. Unlike CISC machines, much of the emphasis in RISC devices has been on simplicity and efficiency.

RISC Features and Potential Advantages

RISC machines are characterized by many distinct features [5]. Some of these include:

- large register files
- emphasis on register-oriented operations
- instructions that primarily execute in a single cycle
- simple LOAD/STORE instructions for memory access
- limited addressing modes
- fixed-length instructions that do not span word boundaries
- hard-coded logic (as opposed to microcode-driven)
- pipelined instruction cycle (typically uniform-delay pipelines)

While some processors that are referred to as RISC machines conform to all of these characteristics, others in fact adhere to merely most them. These features in turn can lead to many potential enhancements. Some of these include:

- fast instruction execution (simple compact instructions; surveys show most often used)

- simple control unit (less instructions and addressing modes to be handled)
- fast decode (limited instructions and addressing modes; fixed-size instructions)
- highly efficient pipelined parallel execution (fixed-length and simple instructions)
- faster processor design, development, and test (simpler design)
- improved optimizing compiler support (simple machine language generally preferred)
- reduced pipeline branching penalties (due to delayed branch technique used in many RISCs)
- improved subroutine parameter passing speed (register windows)

CISC Features and Potential Advantages

Most processors currently in existence, as well as most in the last few decades, may be classified as CISC machines. While details vary, some typical characteristics of these devices are:

- limited number of registers
- richer instruction sets
- emphasis on memory-oriented operations
- instructions that primarily execute in a number of clock cycles (e.g. 8086: from 2 to 200+) [3]
- individual instructions that often require dozens of RISC instructions to represent (e.g. floating point operations)
- memory access capability with a large majority of instructions
- extensive and highly sophisticated addressing modes
- variable-length instructions that span word boundaries
- microcode-driven logic, hard-coded logic, or a combination of both
- pipelined instruction cycles on many recent machines

These typical CISC characteristics often lead to a number of advantages over many RISC machines. Some of these include:

- fast context switching (smaller process environment to handle)
- powerful assembly language programming facility
- reduced requirements on compiler design (machine language forms a layer of abstraction)
- flexibility of processor operation via microcode modifications (writable control store or ROM change)
- powerful and fast floating-point operations (highly sophisticated instructions)
- reduced memory requirements (programs require less memory)
- improved cache performance (due to smaller program size)
- reduced bus traffic (highly sophisticated instructions require less memory access to do the same job)
- ability to achieve some of RISC advantages via use of a subset of the instruction set
- compatibility with widely-used CISC predecessors (e.g. 8086, 68000, VAX, 370)

Conclusions

A comparison of RISC and CISC approaches leads to an interesting dilemma. Clearly, high performance systems of tomorrow will rely on highly parallel architecture, hardware, and software [1,2]. With this trend in mind, should the RISC or CISC paradigm be the emphasis? Some of the issues that will help determine the answer are:

- supercomputer designs have shown that RISC techniques are effective (e.g. Cray machines)
- the RISC design process is typically completed in much less time than CISC designs
- RISC machines provide improved scaling ability in VLSI implementations and new logic families
- upward code compatibility still strongest with CISC (smaller number of RISC machines in use to date)
- optimizing compiler technology continues to improve (which generally implies the RISC model)
- frequency and necessity of assembly and machine language programming continues to decrease
- future developments in parallelizing compilers will play a key role, and tend to imply RISC

These issues tend to indicate that while CISC will dominate in the short-term because of compatibility reasons, RISC processors will play the key role in the highly parallel computing environments to come. However, it should be noted that RISC is more of an evolutionary change due to technology development than it is a revolutionary change. It merely represents the next step in the constantly changing body of knowledge used for processor design.

REFERENCES

[1] Deitel, H.M., *Operating Systems*, 2nd Edition, Addison-Wesley, Reading, MA, 1990.

[2] Hwang, K. and F.A. Briggs, *Computer Architecture and Parallel Processing*, McGraw-Hill, New York, NY, 1984.

[3] *iAPX 86/88, 186/188 User's Manual: Programmer's Reference*, Intel Corporation, Santa Clara, CA, 1985.

[4] Levy, H.M. and R.H. Eckhouse, Jr., *Computer Programming and Architecture: The VAX-11*, Digital Press, Bedford, MA, 1980.

[5] Patterson, D., "Reduced Instruction Set Computers," *Communications of the ACM*, January 1985, pp. 8-21.

[6] Uffenbeck, J., *Microcomputers and Microprocessors*, Prentice-Hall, Englewood Cliffs, NJ, 1985.