Asian Journal of Information Technology 6 (11): 1125-1131, 2007
ISSN: 1682-3915

A New Trend for CISC and RISC Architectures

Hasan Krad and Aws Yousif Al-Taie
Department of Computer Science and Engineering, College of Engineering, Qatar University, Qatar

Abstract: The comparative study between CISC (Complex Instruction Set Computer) and RISC (Reduce Instruction Set Computers) has been a well known research area for many years. In this study, we try to address the new trend of these two architectures, which is CRISC (Complex-Reduce Instruction Set Computer). We chose the Intel Core Duo processor, Intel's most recent processor, to be the focus of our study. The Core Duo processor features is highlighted, focusing on pipelining stages, clock speed, number of transistors, Instruction Set Architecture (ISA) and the improvement in cache technology.

Key words: Computer architecture, RISC, CISC, CRISC

INTRODUCTION

From the architecture point of view, the microprocessor chips can be classified into two categories: Complex Instruction Set Computers (CISC) and Reduce Instruction Set Computers (RISC). In either case, the objective is to improve system performance. The debates between these two architectures made this research area very interesting, challenging and some times confusing.

CISC computer architecture is based on a complex instruction set in which instructions are executed by microcode. Microcode allows developers to change hardware designs and still maintain backward compatibility with instructions for earlier computers by changing only the microcode, thus make a complex instruction set possible and flexible. Although CISC designs allow a lot of hardware flexibility, the supporting of microcode slows microprocessor performance because of the number of operations that must be performed to execute each CISC instruction. A CISC instruction set typically includes many instructions with different sizes and execution cycles, which makes CISC instructions harder to pipeline (Yi *et al*., 2000).

Since the 60's CISC microprocessors became prevalent; processors continue to have more and more complicated hardware and more and more complex instruction sets. This trend started with Intel 80486, Pentium MMX to Pentium 3.

RISC chips evolved around the mid-1970 as a reaction at CISC chips. In 70's, John Cocke at IBM's T.J. Watson Research Center provided the fundamental concepts of RISC, the idea came from the IBM 801 minicomputer built in 1971 which is used as a fast controller in a very large telephone switching system. This chip contained many traits a later RISC chip should have: few instructions, fix-sized instructions in a fixed format, execution on a single cycle of a processor and a load/store architecture. These ideas were further refined and articulated by a group at University of California Berkeley led by David Patterson, who coined the term RISC. They realized that RISC promised higher performance, less cost and faster design time (Yi *et al*., 2000). Simple load/store computers, such as MIPS, are commonly referred to as RISC architectures. David A. Patterson was the finder of the term RISC, after that John L. Hennessy invented the MIPS architecture to represent RISC (John and David, 2006).

RISC VERSUS CISC

When designers create a new generation of processors, improving performance is often the key goal. The three main factors, that affect the performance, are (Marc, 1998):

- How fast you can crank up the clock.
- How much work you can do per cycle.
- How many instructions you need to perform a task.

Corresponding Author: Hasan Krad, Department of Computer Science and Engineering, College of Engineering, Qatar University, Qatar

In this study, our comparison based on the above key factors, to highlight the major differences between the two architectures, CISC and RISC.

A CISC processor has most of the following properties (Yi *et al*., 2000):

- Richer instruction set, some simple, some very complex.
- Instructions generally take more than 1 clock to execute.
- Instructions of a variable size.
- Instructions interface with memory in multiple mechanisms with complex addressing modes.
- No pipelining.
- Upward compatibility within a family.
- Microcode control.
- Work well with a simpler compiler.

As time passed, one of the non-RISC architecture with large market was the Intel x86 family. It has some specific characteristics that became associated with CISC (Yi *et al*., 2000):

- Segmented memory model.
- Few registers.
- Crappy floating point performance.

Typically, CISC chips have a large amount of different and complex instructions. It was believed that hardware is always faster than software; therefore, one should make a powerful instruction set, which provides programmers with assembly instructions to do a lot with short programs. Commonly speaking, CISC chips are relatively slow per instruction compared to RISC chips, but use fewer instructions than RISC.

Most actual RISC machines such as the RISC I and RISC II from the University of California at Berkeley and the MIPS from Stanford University have most of the following common properties (Yi *et al*., 2000):

- Simple primitive instructions and addressing modes.
- Instructions execute in one clock cycle.
- Uniformed length instructions and fixed instruction format.
- Instructions interface with memory via fixed mechanisms (load/store).
- Pipelining.
- Instruction set is orthogonal (little overlapping of instruction functionality).
- Hardwired control.
- Complexity pushed to the compiler.

Additional properties of CISC and RISC, regarding cost and performance together (Jim, 2003) are:

- RISC design is approximately twice as cost-effective as CISC.
- RISC architectures are designed for a good cost/performance, whereas CISC architectures are designed for a good performance on slow memories.

Also, more properties added to a new RISC technology, including (Dileep, 1997):

- Superscalar and out-of-order execution.
- Large number of registers.
- Fast floating point performance.
- Larger Cache.
- Higher bandwidth.

The essence of RISC architecture is that it allows the execution of more operations in parallel and at a higher rate than possible with a CISC architecture employing similar implementation complexity. It can not only improve parallelism, using pipelining, but also make superscalar and out-of -order execution possible (Richard and William, 1989). RISC was also called a scalable architecture because it is possible to go from one technology to another with practically the same design (Margarita and Rojas, 1991).

Back in the middle to late 80's, the battle over RISC and CISC is mainly non-Intel versus Intel x86 and RISC seemed to have a clearly upside, until the appearance of i486, Pentium and P2, P3. Now Intel's machines still run the old instruction set, but they adopt some RISC-like characteristics such as one clock execution, clean memory models, deep pipelining, superscalar operations, lots of registers and even out-of-order execution. They run faster and faster with a decent floating point performance. On the other hand, some RISC machines added more instructions to their architectures for new data types. So, it seems the RISC-CISC gap is narrowed down.

So, nowadays, the difference between RISC and CISC is no longer one of instruction sets, but of the whole chip architecture and system. The designations RISC and CISC are no longer meaningful in the original sense. What counts, in a real world, is always how fast a chip can execute the instructions it is given and how well it runs existing software (Yi *et al*., 2000).

RISC's original goals were to limit the number of instructions on the chip so that each could be allocated enough transistors to make it execute in one cycle. Rather than provide a mul instruction, for example, the microprocessor's designers might make sure that add

executes in one clock. Then a compiler could multiply a and b by adding a to itself b times or b to itself a times. A CISC could multiply 5 by 10 as follows (Jeff, 1995):

```
Mov ax,10
Mov bx,5
Mul bx
```

But a RISC chip might do it like this:

```
Mov ax,0
Mov bx,10
Mov cx,5
Begin:
Add ax,bx
Loop Begin ; loop cx times
```

The two architectures, CISC and RISC, can be compared based on instruction set, which is an important feature of computer architecture. The instruction set chosen for a particular processor determines the way that machine language programs are constructed. Another way for comparing these two architectures is by studying the available addressing modes. This point will give us an idea about the memory or register referencing, which will be one of the important factors in performances comparison. Other factors for comparing these two architectures can be the integer and floating point units. Recently one of the most important factors is instruction pipelining. The more pipelining stages that a processor has, the faster the processor will execute the instructions. Several researchers have been working on instruction pipelining, because of the impact of this feature on the overall performance. The cache and the main memory were also primary factors affecting processor performance. Microprocessor industries were always greedy for speed, while memory industry were greedy for capacity, causing a big gab between CPU and memory speeds (Carlos, 2002). The number of transistors in each processor can affect the speed of the processor. If the processor contains more transistors, this means it will have more gates. This makes the design of the processors at the gate level more compact and as a result it will be faster and has a better performance (Table 1 and 2).

Table 1: Some examples of CISC and RISC processors

CISC Processors	RISC Processors
IBM 370/168	MIPS R2000
VAX 11/780	Sun SPARC
Microvax II	Intel i860
Intel 80386	Motorola 8800
Intel 80286	Power PC 601
Sun-3/75	IBM RS/6000
PDP-11	MIPS R4000

Table 2: System comparisons between selected CISC and RISC processors (William, 2006)

	CISC examples		RISC examples	
	IBM 370/168	VAX 11/780	88000	R4000
Year developed	1973	1978	1988	1991
The no. of instruction	208	303	51	94
Instruction size (bytes)	2-6	2-57	4	4
Addressing modes	4	22	3	1
Number of GRP's	16	16	32	32
Cache size (KB)	64	64	16	128

TECHNIQUES USED FOR EVALUATING CISC AND RISC ARCHITECTURES

Combining features of CISC processors and RISC processors: One method, proposed and implemented by Simon Garth, to combine two different processors with two different operating systems into a one PC. There were some hardware and software issues involved in that study. The CISC technology in the PC based on the Intel 80×86 processor, with DOS operating system. While the RISC technology in the PC was based on the i860 processor, with UNIX operating system (Fig. 1) (Simon, 1991).

The researcher concluded that the resulting ability to increase computation power by adding extra hardware when software was added provides an effective means of accommodating new, more demanding, applications within the confines of existing machine (Simon, 1991).

Experimental comparisons between CISC processors and RISC processors: Performance comparison between CISC and RISC processors using integer and floating point benchmarks was a very popular technique for many years. A study done by Gao, Tang and Ding that compares MIPS R2000 instruction set, which represents a pure RISC and 80386 instruction set, which represents a pure CISC. Based on some experiments, the researchers obtained some meaningful statistical results.

The following conclusions have been drawn (Yi *et al*., 2000):

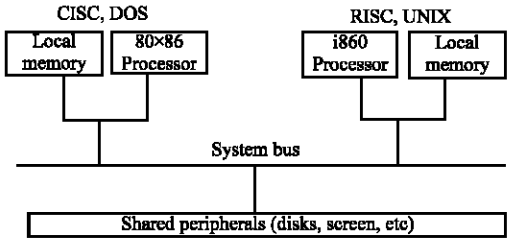

Fig. 1: The system architecture for the combined machine at high level of abstraction

Table 3: Summary of MIPS R2000 and 80386 architectures (Yi *et al*., 2000)

	MIPS R2000	Intel 80386
Date announced	1986	1985
Instruction size (bits)	32	Variable
Address space (size, model)	32 bits, flat	32 bits, segmented with paging support
Data alignment	Aligned	No
Data addressing modes	2	11
Protection	Page	Segmented scheme
Integer registers (number, model, size)	31 GPR*32 bits	8 GPR*32 bits, 6 segment registers*16 bits, 2 other* 16 bits
Separate floating-point registers	16*32 or 16*64 bits	8*80 bits
Floating-point format	IEEE 754 single, double	IEEE 754 single, double, extended

- CISC has richer instruction set.
- CISC has better code density, 80386's average instruction length is less than MIPS R2000's.
- 80386 has richer addressing modes, but in both integer and floating-point case, 3 addressing modes used for about 90%'s addressing, some addressing modes even never used.
- For 80386, the operand types can be memory, while in MIPS R2000, only load/store will access memory, all operands are in registers (Table 3).

Another experimental study has been done by Dileep and Douglas (1991). The study compared an example implementation of the RISC and CISC architectures (a MIPS M2000 and a VAX 8700) on nine of the ten SPEC (Standard Performance Evaluation Corporation) benchmarks. The organizational similarities of these machines provides an opportunity to examine the purely architectural advantages of RISC. The RISC approach promises many advantages over CISC architectures, including superior performance, design simplicity, rapid development time (Dileep and Douglas, 1991). The researcher's founding was that the RISC, MIPS M2000, has significantly higher architecturally determined performance than the CISC, the Digital VAX 8700.

The impact of system software on CISC and RISC processors: Shifting from a CISC design strategy to a RISC design strategy isn't without its problems. Software engineers should be aware of the key issue, which arises when moving code from a CISC processor to a RISC processor.

The performance of a RISC depends greatly on the code quality that it is executing. If the programmer (or compilers) does a poor job of instruction scheduling, the processor can spend quit a bit of time stalling, waiting for the result of one instruction before it can proceed with a subsequent instruction.

Since CISC machine perform complex action with single instructions, where RISC machines may require multiple instructions for the same action, code expansion can be a problem.

In the nineties, people at Sun developed the java language and also the Java Virtual Machine, which contains a complete, stack-based, instruction list. It can be seen as a CISC instruction list. But compared to RISC it is opposite in every detail (Stefan, 2006).
The JVM byte code architecture is:

- Not reduced instruction set.
- Not register based.
- Not load-store (Can combine memory access and execution).

The instructions are:

- Not fixed-length.
- Not one per cycle.
- Not simple (Can do multiple memory accesses and/or multiple execution cycles).

There have been several research papers that covered the impact of system software on the processor and memory performance. John (1989) evaluated several hardware platforms (MIPS M2000, VAX 8800, Sun-3/75 and Sun-4/280) and several operating systems platforms such as (Ultrix, SunOS, RISC/os and Spriite). A set of benchmarks was used in his research. The study highlight issues for both hardware designers and operating systems people to think about (John, 1989).

The improvement of computing performance, without any hardware update, can be achieved by either leaving optimization to the compiler, which todays recent ones are pipelined (splitting a complex operation in to several simple operations), or to the programmer (Carlos, 2002).

COMPLEX-REDUCE INSTRUCTION SET COMPUTERS CRISC

CRISC the future of processor design: Intel's Pentium series chips are sometimes described as CRISC (Complex-Reduce Instruction Set Computers), because they are hybrid between the two architectures. This first started when Intel released its i486 processor and it was widely repeated as having a RISC integer unit (Paul, 2000).

The term, like its antonym RISC, has become less meaningful with the continued evolution of both CISC and RISC designs and implementations. The first pipelined CISC CPUs, such as 486s (Paul, 2000) from Intel, AMD, Cyrix and IBM, certainly supported every instruction that their predecessors did, but achieved high efficiency only on a fairly simple ×86 subset (resembling a non load/store RISC instruction set).

In present study, we focused on the Intel family microprocessors, to illustrate the future trend of CISC and RISC, which is CRISC. An example of CRISC is the the Intel Pentium-Pro, which is an interesting blend of the two architectures. It still executes the CISC instruction set, but the internal implementation is a high performance Post RISC CPU (Mark *et al*., 1996).

Architecture of Intel Core 2 Duo: In this study, we explain briefly the architectural aspects of the Intel's recent processor, the Intel Core 2 Duo processor. The Core 2 brand refers to a range of Intel's mobile, desktop dual- and quad-core 64-bit ×86 CPUs based on the Intel Core microarchitecture. The Core 2 brand for desktop, laptop and workstation PCs was introduced on July 27, 2006 comprising Duo (dual-core), Quad (quad-core) and extreme (dual- or quad-core CPUs with higher speeds and unlocked multiplier) branches (Intel Group, 2007). Unlike the architecture of Pentium 4 or Pentium D branded processors, the Core architecture does not stress extremely high clock speeds, but rather improvements in the processor's usage of both available clock cycles and power (Intel Group, 2007). This translated into more efficient decoding stages, execution units, caches and buses, etc., reducing the Core 2 CPU's power consumption, while enhancing their processing capacity. The design of Intel Core 2 Duo is chosen to maximize performance and minimize power consumption. It emphasizes mainly on cache efficiency and does not stress on the clock frequency for high power efficiency. Although clocking at a slower rate than most of its competitors, shorter stages and wider issuing pipeline compensates the performance with higher Inter-Process Communications (IPC's). In addition, the Core 2 Duo processor has more ALU units (Table 4 and 5) (Tribuvan, 2007).

Intel Core 2 Duo processor uses the following instruction sets: x86, MMX, SSE, SSE2, SSE3, SSSE3, x86-64. The most resent instruction set that the Core 2 Duo processor is using is Supplemental Streaming SIMD Extension 3 (SSSE3), Intel's name for the SSE instruction set's fourth iteration. The previous state of the art was SSE3 and Intel have added an S rather than increment the version number, as they appear to consider it merely a revision of SSE3. SSSE3 contains 16 new discrete instructions over SSE3 (Intel Group, 2007).

The Core 2 Duo processor has 14 pipelining stages. The slightly deeper pipeline enables increased clock speeds and techniques such as memory disambiguation and improved prefetch logic also help offset any advantage an integrated memory controller offers. It includes the shared 4 MB L2 cache and the L1 cache with size 32KB+32 KB. Intel has chosen not to follow suit in their Core 2 processors, preferring to use any additional on-die transistors to increase the Level 2 cache to 4MB, it has 291 million transistors (Intel Group, 2007). The improvements implemented in the Intel Core 2 chips make them unquestionably the most efficient processors, able to decode and process more instructions per clock cycle (Fig. 2).

Table 4: Some information about the Intel Family (Intel Group, 2007)

Microprocessor (Intel family)	No. of transistors	Year of development
4004	2300	1971
8008	3500	1972
8080	6000	1974
8086	29000	1978
8088	29000	1981
286	134000	1982
386	275000	1985
486	1.2 Million	1989
P1	3.1 Million	1993
P2	7.5 Million	1995
P3	9.5 Million	1997
P4	42 Million	2000
Core 2 Duo	291 Million	2006

Table 5: Additional information about the Intel Family (Intel Group, 2007)

Microprocessor (Intel family)	Speed	Year of development
4004	108 KHz	1971
8008	200 KHz	1972
8080	2 MHz	1974
8086	5 MHz	1978
8088	8 MHz	1981
286	12.5 MHz	1982
386	16 MHz	1985
486	33 MHz	1989
PI	66 MHz	1993
PII	233 MHz	1995
PIII	1.2 GHz	1997
P4	1.8 GHz	2000
P4 with HT.	3.06 GHz	2002
Pentium D	2.66×2 GHz	2005
Core 2 Duo	3.0×2 GHz	2006

The five main features of Intel Core 2 Duo contributing towards its high performance are (Tribuvan, 2007):

- Intel's wide dynamic execution.
- Intel's advanced digital media boost.
- Intel's intelligent power capability.
- Intel's advanced smart cache.
- Intel's smart memory access.

We believe that debate about the two architectures CISC and RISC is of no more interest in today's computer technology. Computers today, though they bear the brand of RISC or CISC, are in fact not pure implementations of either one. They are truly hybrid systems. RISC architects have adopted a larger set of

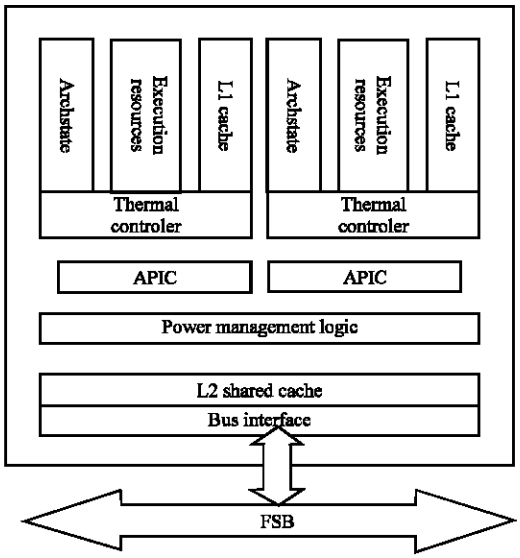

Fig. 2: The block diagram of Intel Core 2 Duo Processor

instructions and CISC architects have realized the benefits of implementing a core set of instructions that can execute in a single CPU cycle. From our observation Intel's Core Duo processor has improved the cache system. It has two high speed caches of size (L1 and L2) in each core (processor, which will store most recently used main memory locations. It typically requires only one to two processor cycles to access data as compared with 10 to 200 cycles for main memory access. We must confess that bigger caches will only improve the hit rate, but decrease the access speed. The Intel's Core Duo processor utilizes 14 stage pipelining, to speedup execution, although the data dependencies will remain one of the problems in processors using pipelining. Another fact in Intel's Core Duo processor is the increasing number of transistors (291 Million), which means closer circuits and as a result a better performance (Table 6).

Most of the researchers, who have done some research in this field and focused on the hardware or software aspects of the two architectures CISC and RISC, have, in fact, chosen two old processors, one to represent a pure CISC and the other one to represent a pure RISC. This type of research does not have much impact on the design of the current chips. Secondly, the two architectures are not pure CISC and RISC processors. The comparison between them will lead us to wrong conclusions, because the recent processors are CRISC, a hybrid processor of CISC and RISC features. These types of processors have been focusing on the increasing number of pipelining stages and the number of instruction sets.

Table 6: Additional information about two different Intel's Core processors (Intel Group, 2007)

	Core 2 Duo (Merom)	Core Duo (Yonah)
Manufacturing process	65 nm	65 nm
Die size	143 mm^2	90 mm^2
Transistors	291 million	151 million
Clock speeds	1.06 GHz - 2.4GHz+	1.20 GHz - 2.33 GHz
FSB frequency	533 MHz-800 MHz	533 MHz -667 MHz
L1 cache size	32 KB+32 KB	32 KB+32 KB
L2 cache size	2 MB-4 MB Shared	2 MB Shared
Pipeline stages	14	12
Decoders	1 complex+3 simple	1 complex+2 simple
Maximum decode rate	4+1	3
Reorder buffer	96	80
SSE units	3	1
Socket interface	Socket-M (PGA/BGA) Socket-P (PGA/BGA)	Socket-M (PGA/BGA)

CONCLUSION

A new trend of CISC and RISC architectures is addressed. Some of previous research was highlighted and a new technology is presented, Intel's Core 2 Duo processor. For the best performance and scalability of the Intel Core 2 Duo processor, the following are important factors: Fast cache-to-cache communication, Large L2 or shared capacity, Fast L2 access delay and Fair resource (cache) sharing.

REFERENCES

Carlos, J.L., 2002. Improve Computing Performance, without any Hardware Update.

Carlos Carvalho, 2002. The Gap between Processor and Memory Speeds.

Dileep Bhandarkar and Douglas W. Clark, 1991. Performance from Architecture Comparing a RISC and a CISC with Similar Hardware Design.

Dileep, B., 1997. RISC versus CISC: A Tale of Two Chips.

Intel Group, 2007. Intel Core 2 Duo Processor Specification. http://www.intel.com/core2duo.

Jeff Prosise, 1995. RISc vs. CISC: The Real Story, PC Magazine.

Jim, T., 2003. Evaluation of Performance Instruction Set Architectures: RISC versus CISC.

John Ousterhout, 1989. Why are not Operating Systems Getting Faster as fast as hardware.

John, L.H. and D.A. Patterson, 2006. Computer Architecture A Quantitative Approach. 3rd Edn.

Marc, T., 1998. Challenges and Trends in Processor Design.

Margarita Esponda and Ra'ul Rojas, 1991. The RISC Concept-A Survey of Implementations.

Mark, B., T. Doom and R. Enbody, 1996. Beyond RISC-The Post RISC Architecture.

Paul DeMone, 2000. RISc vs CISC Stilll Matters.

Richard, S. Piopho and William S. Wu, 1989. A Comparison of RISC Architectures.

Simon, C.J. Garth, 1991. Combining RISC and CISC in PC System.

Stefan Blixt, 2006. Processors Designs for Embedded Systems.

Tribuvan, K.P., 2007. Performance Analysis of Intel Core 2 Duo Processor, Thesis.

William Stallings, 2006. Computer Architecture Designing for Performance. 7th Edn.

Yi Gao, Shilang Tang and Zhangli Ding, 2000. Comparison between CISC and RISC.