© 2014 IJIRT | Volume 1 Issue 7 | ISSN: 2349-6002

A SYSTEMATIC REVIEW REGARDING THE APPLICATIONS OF RISC & CISC

Akash Bhatia, Anirudh, Puneeta Ajmera
Department Of Electronics And Communication Engineering
Dronacharya College Of Engineering Amity University Gurgaon, India

***Abstract-* RISC vs. CISC wars raged in the 1980s when chip area and processor design complexity were the primary constraints and desktops and servers exclusively dominated the computing land-scape. Today, energy and power are the primary design constraints and the computing landscape is significantly different growth in tablets and smartphones running ARM (a RISC ISA) is surpassing that of desktops and laptops running x86 (a CISC ISA). Further, the traditionally low-power ARM ISA is entering the high-performance server market, while the traditionally high-performance x86 ISA is entering the mobile low-power de-vice market. Thus, the question of whether ISA plays an intrinsic role in performance or energy efficiency is becoming important, and we seek to answer this question through a detailed measurement based study on real hardware running real applications. We analyze measurements on the ARM Cortex-A8 and Cortex-A9 and Intel Atom and Sandy bridge i7 microprocessors over workloads spanning mobile, desktop, and server computing. Our methodical investigation demonstrates the role of ISA in modern microprocessors' performance and energy efficiency. We find that ARM and x86 processors are simply engineering design points optimized for different levels of performance.**

***Index Terms-* RISC, CISC, ISA, Cortex-A8, Cortex-A9, Microprocessors.**

I. INTRODUCTION

The question of ISA design and specifically RISC vs. CISC ISA was an important concern in the 1980s and 1990s when chip area and processor design complexity were the primary constraints. It is questionable if the debate was settled in terms of technical issues. Regardless, both flourished commercially through the 1980s and 1990s. In the past decade, the ARM ISA (a RISC ISA) has dominated mobile and low-power embedded computing domains and the x86 ISA (a CISC ISA) has dominated desktops and servers.

Recent trends raise the question of the role of the ISA and make a case for revisiting the RISC vs. CISC question. First, the computing landscape has quite radically changed from when the previous studies were done. Rather than being exclusively desk-tops and servers, today's computing landscape is significantly shaped by smartphones and tablets. . Second, while area and chip design complexity were previously the primary constraints, energy and power constraints now dominate. Third, from a commercial standpoint, both ISAs are appearing in new markets:ARM-based servers for energy efficiency and x86-based mobile and low power devices for higher performance. Thus, the question of whether ISA plays a role in performance, power, or energy efficiency is once again important

II. RELATED WORK

Early ISA studies are instructive, but miss key changes in today's microprocessors and design constraints that have shifted the ISA's effect. We review previous comparisons in chronological order, and observe that all prior comprehensive ISA studies considering commercially implemented processors focused exclusively on performance. Bhandarkar and Clark compared the MIPS and VAX ISA by comparing the M/2000 to the Digital VAX 8700 implementation and concluded: "RISC as exemplified by MIPS provides a significant processor performance advantage." In another study in 1995, Bhandarkar compared the Pentium-Pro to the Alpha 21164 , again focused exclusively on performance and concluded:"...the Pentium Pro processor achieves 80% to 90% of the performance of the Alpha 21164... It uses an aggressive out-of-order design to overcome the instruction set level limitations of a CISC architecture. On floating-point intensive benchmarks, the Alpha 21164 does achieve over twice the performance of the Pentium Pro processor." Consensus had grown that RISC and CISC ISAs had fundamental differences that led to performance gaps that required aggressive microarchitecture optimization for CISC which only partially bridged the gap. Isen et al. compared the performance of Power5+ to Intel Woodcrest considering SPEC benchmarks and concluded x86 matches the POWER ISA. The consensus was that “with aggressive micro architectural techniques for ILP, CISC and RISC ISAs can be implemented to yield very similar performance.” Many informal studies in recent years claim the x86’s “crufty” CISC ISA incurs many power overheads and attribute the ARM processor’s power efficiency to the ISA [1, 2]. These studies suggest that the microarchitecture optimizations from the past decades have led to RISC and CISC cores with similar performance, but the power overheads of CISC are intractable.

In light of the prior ISA studies from decades past, the significantly modified computing landscape, and the seemingly vastly different power consumption of ARM implementations (1-2 W) to x86 implementations (5 – 36) considering the dominance of ARM and x86 and the multi-pronged importance of the metrics of power, energy, and performance, we need to compare ARM to x86 on those three metrics.

Macro-op cracking and decades of research in high-performance micro architecture techniques and compiler optimizations seemingly help overcome x86’s performance and code-effectiveness Bottle necks, but these approaches are not free. The crux of our analysis is the following:

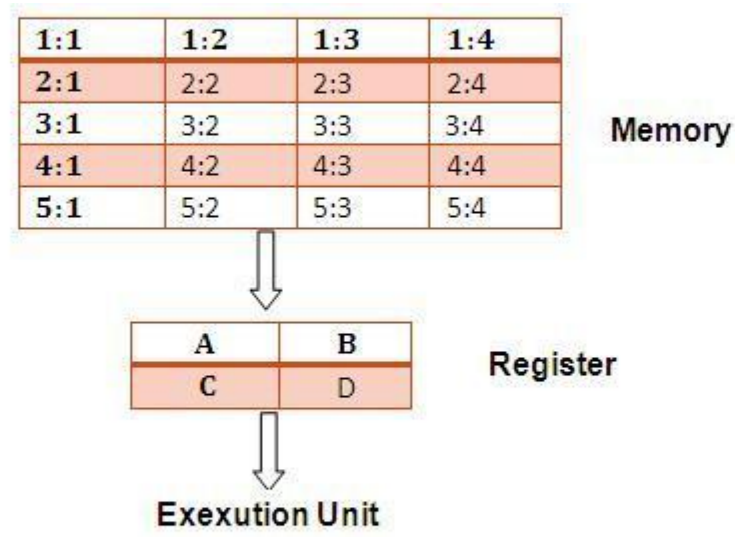
After decades of research to mitigate

CISC performance overheads, do the new approaches introduce

fundamental energy inefficiencies?

III. AN EXAMPLE OF MULTIPLICATION OF TWO NUMBERS

Suppose that the main memory is divided into locations numbered from (row) 1: (column) 1 to (row) 5: (column) 4. The execution unit is responsible for carrying out all computations. However, the execution unit can only operate on data that has been loaded into one of the four registers (A, B, C, or D). Let's say we want to find the product of two numbers - one stored in location 1:3 and another stored in location 4:2 and store back the result to 1:3



IV. CISC APPROACH

CISC design would try to finish the task in the minimum possible instructions by implementing hardware which could understand and execute series of operations. Thus the processor would come with a specific instruction ‘MUL’ in its instruction set. ‘MUL’ will loads the two values from the memory into separate registers, multiplies the operands in the execution unit, and then stores the product in the appropriate location. So, the entire task of multiplying two numbers can be completed with one instruction:

MUL 1:3, 4:2

MUL is referred to as a "complex instruction" as it operates directly on the computer's memory banks and does not require the programmer to explicitly call any loading or storing functions.

V. RISC APPROACH

RISC processors use simple instructions that can be executed within a clock cycle. Thus, ‘MUL’ instruction will be divided into three instructions.

i) "LOAD," which moves data from the memory bank to a register,

ii) "PROD," which finds the product of two operands located within the registers, and

iii) "STORE," which moves data from a register to the memory banks.

In order to perform the task, a programmer would need to code four lines of assembly:

LOAD A, 1:3

LOAD B, 4:2
PROD A, B
STORE 1:3, A

VI. ANALYSIS

1. RISC design uses more lines of code and hence, more RAM is needed to store the assembly level instructions. Also, the compiler must also perform more work to convert a high-level language statement into code of this form.
2. Since each instruction requires only one clock cycle to execute, the entire program will execute in approximately the same amount of time as the multi-cycle "MUL" command.
3. RISC "reduced instructions" require less transistors of hardware space than the complex instructions, leaving more room for general purpose registers.
4. As all of the instructions execute in a uniform amount of time (i.e. one clock), pipelining is possible.
5. Separating the "LOAD" and "STORE" instructions actually reduces the amount of work that the computer must perform. After a CISC-style "MUL" command is executed, the processor automatically erases the registers. If one of the operands needs to be used for another computation, the processor must re-load the data from the memory bank into a register. In RISC, the operand will remain in the register until another value is loaded.

VII. .RISC AND CISC CONVERGENCE

In the late 70s when computer revolution was gaining momentum, the hardware prices were quite expensive. RAM that had a capacity of few megabytes was worth thousands. CISC which is hardware emphasizing was the sole architecture and it made computing expensive and their repair even more. This was the main reason that IBM researched to develop RISC. The hardware prices have dramatically fallen since then and semiconductor processor technology has changed significantly since introduction of RISC chips in the early 80s. Because a number of advancements are used by both RISC *and* CISC processors, the demarcation between the two architectures is getting blurred. In fact, the two architectures almost seem to have adopted the strategies of the other. Because processor speeds have gone high, CISC chips are now able to execute more than one instruction within a single clock. This also allows CISC chips to make use of pipelining. With other technological improvements, it is now possible to fit many more transistors on a single chip. Similarly, RISC processors can make use of more complicated hardware incorporate more complicated, CISC-like commands.

Thus, we are on the verge of "post-RISC/CISC" era wherein two design approaches are converging. A new architecture named EPIC (Explicitly Parallel Instruction Computing) was launched at the beginning of the new millennium. It carried the pros of RISC as well as CISC. EPIC based processor "Itanium" is commercially widely used by giants such as HP-Compaq and Unisys.
Present circumstances and heavy support from Intel have made CISC share the larger part of the smart computing market. However, RISC, due to its power efficient methods has made rapid progress in handheld and portable devices. Nintendo DS and Apple iPod are the most prominent examples for that. Thus both are strongly ahead to a long future unless a better design architecture gets evolved.

VIII. RISC VS CISC COMPARISON

CISC	RISC
Emphasis on hardware	Emphasis on software
Includes multi-clock complex instructions	Single-clock, reduced instruction only
Memory-to-memory: "LOAD" and "STORE" incorporated in Instructions	Register to register: "LOAD" and "STORE" are independent instructions
Small code sizes, high cycles per second	Low cycles per second, large code Sizes
Transistors used for storing complex Instructions	Spends more transistors on memory registers

IX. CONCLUSION

In this work, we revisit the RISC vs. CISC debate considering contemporary ARM and x86 processors running modern workloads to understand the role of ISA on performance, power, and energy. During this study, we encountered infrastructure and system challenges, missteps, and software/hardware bugs. Table 9 outlines these issues as a potentially useful guide for similar studies. Our study suggests that whether the ISA is RISC or CISC is irrelevant, as summarized in Table 10, which includes a key representative quantitative measure for each analysis step. We reflect on whether there are certain metrics for which RISC or CISC matters, and place our findings in the context of past ISA evolution and future ISA and microarchitecture evolution. Appears in the 19th IEEE International Symposium on High Performance Computer Architecture (HPCA 2013) 12 cat by 25%. We feel much of this is explained by simpler core design (in-order vs OOO), and smaller caches, predictors, and TLBs. We also observe that the A9's area is in-between Bobcat

and Atom and is close to Atom's. Further detailed analysis is required to determine how much the ISA and the microarchitecture structures for performance contribute to these differences. A related issue is the performance level for which our results hold. Considering very low performance processors, like the RISC ATmega324PA microcontroller with operating frequencies from 1 to 20 MHz and power consumption between 2 and 50mW [3], the overheads of a CISC ISA (specifically the complete x86 ISA) are clearly untenable. In similar domains, even ARM's full ISA is too rich; the Cortex-M0, meant for low power embedded markets, includes only a 56 instruction subset of Thumb-2. Our study suggests that at performance levels in the range of A8 and higher, RISC/CISC is irrelevant for performance, power, and energy. Determining the lowest performance level at which the RISC/CISC ISA effects are irrelevant for all
metrics is interesting future work.

While our study shows that RISC and CISC ISA traits are irrelevant to power and performance characteristics of modern cores, ISAs continue to evolve to better support exposing workload-specific semantic information to the execution substrate. On x86, such changes include the transition to Intel64(larger word sizes, optimized calling conventions and shared code support), wider vector extensions like AVX, integer crypto
and security extensions (NX), hardware virtualization extensions and, more recently, architectural support for transactions in the form of HLE. Similarly, the ARM ISA has introduced
shorter fixed length instructions for low power targets (Thumb), vector extensions (NEON), DSP and bytecode execution extensions (Jazelle DBX), Trustzone security, and hardware virtual-
ization support. Thus, while ISA evolution has been continuous, it has focused on enabling specialization and has been largely agnostic of RISC or CISC. Other examples from recent research include extensions to allow the hardware to balance accuracy and reliability with energy efficiency and extensions to use specialized hardware for energy efficiency.

REFERENCES

[1] ARM on Ubuntu 12.04 LTS battling Intel x86?
http://www.phoronix.com/scan.php?page=article&item=ubuntu_1204_armfeb&num=1

[2] The ARM vs x86 wars have begun: In-depth power analysis of Atom, Krait & Cortex A15
http://www.anandtech.com/
show/6536/arm-vs-x86-the-real-showdown/

[3] Atmel Datasheet,
http://www.atmel.com/Images/
doc2503.pdf

[4] Chip-architect,
http://www.chip-architect.com/news/
AMD_Ontario_Bobcat_vs_Intel_Pineview_Atom.jpg

[5] M. Baron. The single-chip cloud computer. Microprocessor Report, April 2010.

[6] D. Bhandarkar. RISC versus CISC: a tale of two chips. SIGARCH Comp. Arch. News, 25(1):1–12, Mar. 1997.

[7] D. Bhandarkar and D. W. Clark. Performance from architecture: comparing a RISC and a CISC with similar hardware organization. In ASPLOS '91

[8] N. Binkert, B. Beckmann, G. Black, S. Reinhardt, A. Saidi, A. Basu, J. Hestness, D. Hower, T. Krishna, S. Sardashti, R. Sen, K. Sewell, M. Shoaib, N. Vaish, M. Hill, and D. Wood. The gem5 simulator. SIGARCH Comp. Arch. News, 39(2), Aug. 2011.

[9] W. L. Bircher and L. K. John. Analysis of dynamic power management on multi-core processors. In ICS '08.

[10] E. Blem, J. Menon, and K. Sankaralingam. A detailed analysis of contemporary arm and x86 architectures. Technical report, UW-Madison, 2013.

[11] D. Bruening, T. Garnett, and S. Amarasinghe. An infrastructure for adaptive dynamic optimization. In CGO '03.

[12] R. Colwell, C. Y. Hitchcock, III, E. Jensen, H. Brinkley Sprunt, and C. Kollar. Instruction sets and beyond: Computers, complexity, and controversy.
Computer, 18(9):8–19, Sept. 1985.

[13] M. de Kruijf, S. Nomura, and K. Sankaralingam. Relax: An architectural framework for software recovery of hardware faults. In ISCA ’10.

[14] H. Esmaeilzadeh, T. Cao, Y. Xi, S. Blackburn, and K. McKinley. Looking back on the language and hardware revolutions: measured power, performance, and scaling. In ASPLOS ’11.

[15] H. Esmaeilzadeh, A. Sampson, L. Ceze, and D. Burger. Archi-
tecture support for disciplined approximate programming. In ASPLOS ’12.

[16] M. Ferdman, A. Adileh, O. Kocberber, S. Volos, M. Alisafaee, D. Jevdjic, C. Kaynak, A. D. Popescu, A. Ailamaki, and B. Fal-safi. Clearing the clouds: a study of emerging scale-out work-loads on modern hardware. In ASPLOS ’12
.