# Understanding the Impact of ISAs on Performance and Power of Modern Embedded Systems


## ABSTRACT

RISCV is an emerging Instruction Set Architecture (ISA) and has become an important option for both academia and industry when considering new microprocessor designs. Features like modularity, extensibility, simplicity, and being open and free to use, make RISCV an attractive option for next generation of processors especially in embedded systems domain where new, customized, low-power, and efficient cores are needed. In this paper we present a comparative study on impacts of three well-known Instruction Set Architectures (ISAs) (MIPS, ARM, and RISCV) on Performance, Power, and Area (PPA) of state-of-the-art embedded processors through a systematic measurement campaign using several different toolchains and frameworks and several standard benchmark suites. We particularly study the impact of these ISAs on important metrics such as static and dynamic Instruction Count (*icount*), Cycle Count, Microarchitectural Statistics (e.g. MPKI, Branch Prediction Accuracy, etc.), Dynamic Power, and Core's Area and report our key findings on impacts of using different ISAs on each of these metrics. We find that some of these metrics are ISA-dependent and others are dependent on other factors such as compiler, runtime libraries, and specific microarchitectural features. Our main conclusion is that while comparing to MIPS and ARM, RISCV has some shortcomings and design/toolchain issues that should be addressed and fixed, due to its intrinsic features such as modularity it provides a great opportunity for designing customized PPA-efficient cores.


## 1. INTRODUCTION

Instruction Set Architectures (ISAs) has a key role in designing cores for different domains, where x86 ISA has become dominant in desktop and server domains, and ARM has become the dominant ISA in mobile, tablet, and embedded system domain. The question of impact of ISA design on different Performance, Power, Area (PPA) metrics has traditionally been an important concern for designers and semiconductor industry especially in the 1980s and 1990s when chip area and processor design complexity were the primary constraints [24, 12, 17, 7]. In the past decade, we radical changes in computing landscape and rise of mobiles and tables and increasing popularity of ARM ISA this question again becomes an important issue.

Today, with proliferation of embedded and cyber-physical systems (e.g. IoTs) and increasing popularity of domain-specific languages and emerging applications like machine-learning and more importantly, introduction of a new, open-source, modular ISA (RISCV), this question once again becomes an interesting topic for research.

To answer this question, in this paper we present a comparative study on impacts of using three different ISAs (MIPS, ARM, and RISCV) on important metrics such as static and dynamic instruction count (icount), total execution time (cycle count), dynamic power, and area. We show which of these metrics are ISA-dependent and what are the other important factors on PPA. Using these experiments we pinpoint the shortcomings, issues, and advantages of using RISCV ISA over ARM and MIPS ISAs.

## 2. METHODOLOGY

## 3. BACKGROUND

RISC-V is an emerging open-source software and hardware ecosystem that has gained in popularity in both industry and academia [2, 11]. At the heart of the ecosystem, the RISC-V ISA is designed to be open, simple, extensible, and free to use. The RISC-V software tool chain includes open-source compilers (e.g., GNU/GCC and LLVM), a full Linux port, a GNU/GDB debugger, verification tools, and simulators. On the hardware side, several RISC-V prototypes (e.g., Celerity [4]) have been published. The rapid growth of the RISC-V ecosystem enables computer architects to quickly leverage RISC-V in their research.

## 4. RELATED WORK

Early ISA studies are instructive but miss key changes in todayâĂŹs microprocessors and design constraints that have shifted the ISAâĂŹs 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 theMIPS and VAX ISA by comparing theM/2000 to the Digital VAX 8700 implementations [Bhandarkar and Clark

1991] and concluded: âĂIJRISC as exemplified by MIPS provides a significant processor performance advantage.âĂİ In another study in 1995, Bhandarkar compared the Pentium-Pro to the Alpha 21164 [Bhandarkar 1997], again focused exclusively on performance and concluded: âĂIJthe 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.âĂİ Consensus had grown that RISC and CISC ISAs had fundamental differences that led to performance gaps that required aggressive microarchitecture optimization for CISC that only partially bridged the gap. Isen et al. [2009] compared the performance of Power5+ to Intel Woodcrest considering SPEC benchmarks and concluded that x86 matches the POWER ISA. The consensus was that âĂIJwith aggressive microarchitectural techniques for ILP, CISC and RISC ISAs can be implemented to yield very similar performance.âĂİ Many informal studies in recent years claim the x86âĂŹs âĂIJcruftyâĂİ CISC ISA incursmany power overheads and attribute the ARM processorâĂŹs power efficiency to the ISA.1 These studies suggest that the microarchitecture optimizations from the past decades have led to RISC and CISC cores with similar performance but that the power overheads of CISC are intractable. In light of the ISA studies from decades past, the significantly modified computing landscape, and the seemingly vastly different power consumption of RISC implementations (ARM: 1âĂŞ2W, MIPS: 1âĂŞ4W) to CISC implementations (x86: 5âĂŞ36W), we feel there is need to revisit this debate with a rigorous methodology. Specifically, considering the multipronged importance of the metrics of power, energy, and performance, we need to compare RISC to CISC on those three metrics. Macro-op cracking and decades of research in high-performance microarchitecture techniques and compiler optimizations seemingly help overcome x86âĂŹs performance and code-effectiveness bottlenecks, 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?

## 5. REFERENCES