Does the world need another ISA?

Ross Alexander

September 9, 2020

Contents

1 The IBM S/360 and ISAs

In the early 1960s IBM decided to create a new general purpose computer. Rather than a single system a family with half a dozen different models would be produced with a 50 times performance difference between the lowest and the top models. The difference between this an earlier systems was the same application binary would run on any of the models (assuming enough memory). The theory was customers would buy a low end model to begin with and then upgrade as their requirements grew without needing to recode or recompile any of their applications.

By defining the system in terms of an Instruction Set Architecture (ISA) rather than a particular implementation allowed the a wide variety of hardware solutions to be chosen for the various models.

This model of having a single ISA and multiple implementations was copied by many other vendors, allowing rapid improvement in performance and cost allowed by Moore's Law without the need for software to be constantly updated. The downside of this is poor design decisions made early on in an ISA are very difficult to rectify or remove.

2 RISC vs CISC

Early ISAs such as the S/370 (the follow on from the S/360), Digital's VAX or the Intel IA32 ISA are described as Complex Instruction Set Computers. In particular the often have complex addressing models which can be mixed and matched for many arithmetic operations. This worked well when most programs were hand written in assembly but with high level languages such as C replacing assembly as the language of choice many of these complex instructions were never used.

In the mid 1970s within IBM a project was undertaken to produce a much simpler ISA which would be a good target for a C compiler, ultimately leading to the IBM 801 processor. This was dubbed a Reduced Instruction Set Computer (RISC), and would

go on to influence design ideas in UC Berkeley and Stanford, leading to the SPARC and MIPS RISC designs in the 1980s. These ideas were also taken up by the designers of the BBC Micro in their Acorn RISC Machine (ARM).

3 ISA timeline

Historical timelines of various RISC ISAs.

- IBM 801 (1980) -> RS/6000 (1990) -> POWER10 (2020)
- MIPS (MIPS1 R2000 1985) -> MIPS32/64 Relase 6 (2017) [Stanford]
- SPARC V7 (1987) -> SPARC V9 1993) -> SPARC M8 (2017)
- Berkeley RISC1 (1981) -> RISC V (2010) -> 32/64 Frozen (2019)
- ARM v2 (1983) -> AArch64 (ARMv8-A 2011) -> ARMv8.6-A (2019)

4 Scaling Digital Logic

CPU	Count
Intel 4004	2500
Intel 80386	275,000
Freedom U500	250,000,000
IBM z14	6,100,000,000
Apple A14X	15,000,000,000
IBM Power 10	18,000,000,000
AMD Eypc Rome	39,540,000,000
NVIDIA GA100	54,000,000,000
Cerebras WSE	1,200,000,000,000

4.1 Systems on a Chip (SoC)

The original IBM PC consisted of a considerable number of separate Integrated Circuits (ICs aka chips). A rough list of the most important chips includes

- 8086 CPU
- RAM Chips
- ROM
- Memory controller
- Programmable Interrupt Controller
- DMA Controller
- Programmable Interval Timer
- Keyboard Controller
- Floppy Controller
- Video Controller

For many functions all these functions (and others) are integrated onto a single silicon die. These chips then just have pin outs to peripherals, such as HDMI, USB, SD, Ethernet, Wifi etc. SoCs are often packaged with the peripheral heads as Single Board Computers (SBCs) such as the Raspberry Pi or Arduino. Most mobile phones are SBCs based on a SoC.

The least powerful SoCs are often described as mircocontrollers. These tend to be extremely limited in their compuation and memory capabilities. Instead they often focus on simple I/O such as Analog to Digital Converters (ADCs) and General Purpose IO (GPIO) pins for driving power relays, LED displays etc.

5 History of RISC-V

The RISC-V instruction set was developed at UC Berkeley in 2010-2011 and the first manual was released in May 2011. The instruction set origins go back to DLX, an academic ISA design used for teaching, itself an iteration on the original MIPS design from 1985. In 2015 the ISA was official made open source and moved to the RISC-V Foundation. In 2019 the RISCV32I and RISCV64I ISAs were frozen so that no more changes could be made. To allow for expansion the ISA consists of either the base

32-bit or 64-bit Integer ISA with extensions as required. Some of these extensions have been standardized, others are in the process of standardization while others are purely implementation specific.

6 Structure of the ISA

The base unprivileged ISA has four Integer versions, based on the width of the integer registry set. These are RV32I, RV64I and RV128I. There is cut down version of RV32I called RV32E, designed for embedded and very low power devices.

6.1 Base Integer Instructions (RV32I and RV64I)

These include integer arithmetic (but not multiple/divide), condition checking, branching and load/store operations. The M extension contains integer multiple/divide, which was broken out as not required for very simple embedded devices, such as micro controllers.

6.2 Floating Point Instructions

All modern CPUs support the IEEE 754 floating point standard. This standard covers single precision (32 bits) and double precison (64 bits) formats. There are also extended precision (80 and 128 bit) formats but they are not common.

The RISC-V F[loat] and D[ouble] extensions over single and double precisions floating point operations. Adherence to the standard is critical so that numerical computation on gives identical results on all qualified CPUs. Early Pentium chips infamously had a buggy FPU [Floating Point Unit].

6.3 Atomic Instructions

With the rise of multi-core (and multi-thread) CPUs has come multi-threaded user space applications. There are cases where applications need to synchronize between threads and this requires support from CPU to ensure memory consistancy between CPU hardware threads (which can be over multiple sockets). Atomic Instructions are critical to SMP systems and their use and implementation can dramatically effect system performance.

7 Privilege Modes

To run a full OS like Windows or Unix a CPU must support a minimum of two privilege modes, User and Supervisor. At any one time a CPU thread is running in a particular mode. Should an operation occur that violates the rights of the mode (such as executing a privileged instruction or accessing memory not in the current page table) then an exception will occur and be trapped at a high privilege level. These are defined in the U,S and H extensions. Because only the OS needs to execute Privileged Instructions they can be much more implementation specific. There is a standard Prvileged ISA, which is supported in-tree by Linux and FreeRTOS, but it not yet frozen.

8 OS and tooling support

RISC-V is a supported architecture within the Linux kernel. This required the GCC toolchain (GCC/binutils/glibc) to support the ISA. With the rise of LLVM as an alterative to GCC adding RISC-V support to LLVM was also important as many languages use it as a code generation backend. Both Go and Rust have limited RISC-V support.

9 Implementations and Industrial Support