# Computer architecture comparison

I will be comparing **ARMv2** and **MIPS R2000** architectures.

## Table of contents

- History
- Physicalities
- Architecture type
- N-address
- Registers
- Flags
- Data width
- Memory layout
- Virtual memory
- ISA
- Addressing modes
- IO
- Interrupts
- Data types
- Speed
- Cache
- Application
- Software
- Emulators
- Sources

## History

### ARMv2

ARM originally stood for Acorn RISC Machine, but later was renamed to Advanced RISC Machines. Arm Holdings develops instruction set architectures (ISAs) and licenses them to others. The first version was tested on 26 April 1985. The latest version that is used in products today is ARMv9.2-A.

### MIPS R2000

This architecture was developed by MIPS Computer Systems in 1986. It was the first commercial implementation of the RISC architecture. R2000 users included Ardent Computer, DEC, Silicon Graphics, Northern Telecom and MIPS's own Unix workstations.

## Physicalities

### ARMv2

It's elementary base was integrated circuits with low scale integration. It had 30,000 transistors and it's power consumption was just a few watts.

### MIPS R2000

It used integrated circuits as it's elementary base and they had large scale integration. It had 110,000 transistors and the size of the main die was 80mm2. I couldn't find information about it's power consumption, but a similar chip from MIPS used 10W.

## Architecture type

### ARMv2

It is a load/store architecture, where data-processing operations operate on registers, not memory.

### MIPS R2000

It was a load/store architecture, which means all instructions operated on the registers, except the load/store incstructions used to access memory.

## N-address

### ARMv2

It had two, three and four address instruction formats.

### MIPS R2000

It used one, two and three address instruction formats.

## Registers

### ARMv2

It had thirty-one general-purpose 32-bit registers. At any time, 16 registers are visible, others are used to speed up exception processing. Two of visible registers have special functions: link register, holds the address of the next instruction; program counter, it is used as a pointer to the instruction which is two instructions after the instruction being executed.

### MIPS R2000

It had thirty-two 32-bit general-purpose registers and three (3) 32-bit special registers: *pc* (program counter) that always contains the address of the next instruction to be executed; *hi* and *lo* hold the result of multiplication or division.

## Flags

### ARMv2

It used condition code flags, which were: negative (N), zero (Z), carry (C) and overflow (V).

### MIPS R2000

As far as I found out, it didn't have any flags. The later released MIPS IV ISA had flags.

## Data width

### ARMv2

It's machine word width was 32 bits. The newest version of the ARM architecture has a machine word width of 64 bits.

### MIPS R2000

It had a machine word width of 32 bits.

## Memory layout

### ARMv2

It had 26-bit by address space. The maximum amount of memory was 4GB.

### MIPS R2000

It had continuous byte-addressable memory, that was using a 32-bit address. This means it could have had a maximum of 4GB of memory. The layout consisted of reserved data, program text, static data, the heap and the stack.

## Virtual memory

### ARMv2

Virtual memory was supported.

### MIPS R2000

Virtual memory was supported.

## ISA

### ARMv2

It used the ARM instructon set architecture. It had 3 branch, 16 data-processing, 6 multiplication, 1 miscellaneous arithmetic, 2 status register access, 17 load/store, 2 semaphore, 2 exception-generating, 5 coprocessor instructions. A total of 54 instructions.

### MIPS R2000

It used the MIPS I instruction set architecture. It had 18 arithmetic, 6 shift, 8 muliplication/division, 12 load/store, 12 jump and branch and 16 other instructions, a total of 72 isntructions. The ISA had three types: R - register, I - immediate, and J - jump.

## Addressing modes

### ARMv2

It had five addressing modes for data-processing and for loading and storing different types of data.

### MIPS R2000

It had one addressing mode: base + offset. The effective address is the offset + content of the specified base register.

## IO

### ARMv2

To perform IO functions, you had to use memory-mapped IO. This uses special memory addresses which supply IO functions when loaded from or stored to.

### MIPS R2000

It had SPIM Input, which allowed reading from the keyboard and SPIM Output, which allowed printing to the console.

## Interrupts

### ARMv2

Software can create a software interrupt exception to occur. These are normally used to make calls to the OS, to request a service.

### MIPS R2000

When an event happened, the address of the instruction that was about to be executed was saved into a special register *EPC*. The interrupt event was handled and the saved instruction was loaded back and continued execution.

## Data types

### ARMv2

It has three data types: a byte, halfword (2 bytes) and word (4 bytes).

### MIPS R2000

It had three data types: a byte, halfword (2 bytes) and word (4 bytes).

## Speed

### ARMv2

The ARM2, which used the ARMv2 architecture, was designed to run at 8 MHz. A year later the speed was increased to 10-12 MHz. It was able to compute one instruction per cycle.

### MIPS R2000

It was available in 8.3, 12.5, and 15 MHz grades. Later an improved version was released, the R2000A. It had two grades of 12.5 and 16.67 MHz. It could process one instruction per cycle.

## Cache

### ARMv2

It didn't use any cache.

### MIPS R2000

It used fast standard SRAM chip cache with one-cycle read latency.

## Application

### ARMv2

The ARM2 processor was used in home and school computers.

### MIPS R2000

The MIPS R2000 microprocessor was used by Ardent Computer, Silicon Graphics, Northern Telecom and MIPS (company) own workstations.

## Software

### ARMv2

It likely had simple software for use at home and school: a text editor, a calculator, software for printing documents.

### MIPS R2000

I found little information on the software that was written for this architecture.

## Emulators

### ARMv2

armulator A python ARM emulator.

### MIPS R2000

SPIM. Versions before 7.0 implemented MIPS I ISA, which was used by MIPS R2000.

## Afterword

Searching information for the MIPS R2000 wasn't as easy as for the ARMv2 and some data I couldn't even find. It might be because it is not as popular as ARM, which results in less available public information. ARMv2 in contrast, has a **LOT** of information, which is good, but has it's own challenge. Different sites sometimes have conflicting information, and because there is a lot of information out there, it is more difficult to find what data is correct.

## Sources

- R2000 microprocessor
- MIPS architecture
- MIPS ISA
- Addressing mode
- MIPS IO and interrupts
- MIPS Data types
- ARM architecture family
- ARM processors
- ARM architecture reference manual