CISC vs RISC Architecture

Theory, Characteristics, Instructions, and Comparison

Dr. Debapriya Roy

Agenda

What is RISC?

Reduced Instruction Set Computer (RISC)

- Uses a small set of simple instructions.
- Most instructions execute in a single clock cycle.
- Fixed-length instructions (usually 32 bits).
- Load–store architecture (only LOAD/STORE access memory).
- More general-purpose registers to reduce memory traffic.
- Designed for fast pipelining due to uniform instructions.

What is CISC?

Complex Instruction Set Computer (CISC)

- Large instruction set, many complex operations in a single instruction.
- Variable-length instructions (1 to 15 bytes).
- Memory-to-memory operations allowed.
- Fewer general-purpose registers.
- Many addressing modes.
- Multi-cycle execution of instructions.

RISC: Theoretical Foundations

- Emphasis on reducing **cycles per instruction (CPI)**.
- Instructions are simple $\rightarrow$ faster decoding and execution.
- More registers $\rightarrow$ fewer memory accesses.
- Optimized for compilers and pipelining.
- Program length increases because more instructions are needed.

CISC: Theoretical Foundations

- Emphasis on reducing **instructions per program**.
- Complex instructions perform multiple tasks internally.
- Microcoded control makes instructions powerful but slower.
- Variable-length instructions $\rightarrow$ harder to pipeline.
- Smaller program size due to compact encoding.

CISC ADD Instruction Variants

1. **Register → Register**

```
ADD AX, BX
```

2. **Memory → Register**

```
ADD AX, [BX]
```

3. **Immediate**

```
ADD AX, 5
```

4. **Memory → Memory (classical CISC concept)**

```
ADD [SI], [DI]
```

How Memory Operand Fetch Works in CISC

Example:

```
ADD AX, [SI]
```

Steps:

1. SI holds a memory address.
2. CPU copies SI $\rightarrow$ MAR (Memory Address Register).
3. RAM at that address returns data $\rightarrow$ MDR (Memory Data Register).
4. ALU computes: AX + MDR.
5. Result is stored back in AX.

Key Idea: SI contains a *pointer*, but the actual data lives in RAM.

Why RISC Rejects Memory-to-Memory ADD

- RISC uses strict Load–Store architecture.
- Arithmetic instructions operate only on registers.
- Memory access allowed only with LOAD or STORE.
- Therefore, CISC instruction:

```
ADD [SI], [DI]
```

cannot exist in RISC.

RISC Equivalent: Load–Add–Store

Equivalent sequence in RISC:

```
LOAD R1, 0(SI)
LOAD R2, 0(DI)
ADD R3, R1, R2
STORE R3, 0(SI)
```

Explanation:

- First load both memory operands into registers.
- Perform addition using only registers.
- Store result back into memory.

Instruction Format Differences

CISC:

- Variable-length instructions.
- Many addressing modes.
- One instruction may perform multiple steps.

RISC:

- Fixed-length 32-bit instructions.
- Few addressing modes.
- Each instruction performs a very small task.

Example Comparison: ADD

CISC (x86):

```
ADD EAX, [EBX+8]
```

RISC (MIPS):

```
LW R1, 8(RB)
ADD R2, R2, R1
```

Reason: CISC performs address calculation + fetch + add in one instruction. RISC separates all of them.

CISC vs RISC: Comparison Table

Feature	**RISC**	**CISC**
Instruction Set	Small, simple	Large, complex
Instruction Length	Fixed (32-bit)	Variable
Execution Time	Single-cycle	Multi-cycle
Registers	Many GPRs	Few GPRs
Memory Access	Only Load/Store	Any instruction
Addressing Modes	Few	Many
Code Size	Larger	Smaller
Pipelineability	Easy	Hard
Examples	ARM, RISC-V, MIPS	x86, Intel CPUs

Summary

- RISC focuses on simple, fast, uniform instructions.
- CISC focuses on minimizing instruction count using complex instructions.
- CISC ADD can access memory directly; RISC requires load–add–store.
- RISC is easier to pipeline; CISC is more powerful per instruction.
- Modern CPUs often internally translate CISC instructions into RISC-like micro-ops.

Thank You!