

# Lecture 15: 知识 - CISC 和 RISC 的基本概念

## 基本概念

概念: CISC

- Complex Instruction Set Computer
- 设计思路: 一条指令完成一个复杂的基本功能;
- 代表: x86 架构, 主要用于笔记本、台式机等
- 分析:
  - 一条指令可以由一个专门的电路完成;
  - 采用存储程序的设计思想, 由一个比较通用的电路配合存储部件完成一条指令;

概念: RISC

- Reduced instruction Set Computer
- 设计思路: 一条指令完成一个基本“动作”, 多条指令组合完成一个复杂的基本功能;
- 代表: ARM 架构, 主要用于手机、平板等
- 分析:
  - 一条指令一个电路电路设计相对简单, 功耗更低;
  - 更利于进行指令的并行化以及流水化;
- 图示:

| 对比项目 \ 类别 | CISC               | RISC                |
|-----------|--------------------|---------------------|
| 指令系统      | 复杂, 庞大             | 简单, 精简              |
| 指令数目      | 一般大于200条           | 一般小于100条            |
| 指令字长      | 不固定                | 定长                  |
| 可访存指令     | 不加限制               | 只有Load/Store指令      |
| 各种指令执行时间  | 相差较大               | 绝大多数在一个周期内完成        |
| 各种指令使用频度  | 相差很大               | 都比较常用               |
| 通用寄存器数量   | 较少                 | 多                   |
| ➡ 目标代码    | 难以用优化编译生成高效的目标代码程序 | 采用优化的编译程序, 生成代码较为高效 |
| 控制方式      | 绝大多数为微程序控制         | 绝大多数为组合逻辑控制         |
| 指令流水线     | 可以通过一定方式实现         | 必须实现                |

面试题: 简述精简指令集和复杂指令集的区别

- 指令集架构 (ISA) 是计算机硬件和软件之间的接口, 定义了处理器可以执行的指令类型和操作系统方式;
- 精简指令集 (RISC) 和复杂指令集 (CISC) 是两种主要的指令集架构设计哲学, 它们在设计理念、指令复杂度、执行效率等方面存在显著差异;
- 1. 设计理念
  - CISC 的目标是通过提供丰富的指令集和复杂的指令功能, 使硬件能够直接支持高级语言的复杂操作, 从而减少软件开发的复杂性。CISC 指令集通常包含许多复杂指令, 这些指

令可以在单条指令中完成多个操作；

- **RISC** 的目标是通过简化指令集，减少指令的数量和复杂度，从而提高指令的执行效率和处理器的性能。**RISC** 指令集通常只包含简单且执行时间短的指令，复杂操作通过组合多条简单指令来实现；

## 2. 指令复杂度

### • CISC:

- **指令数量**: 指令集庞大，包含数百条甚至上千条指令。
- **指令功能**: 指令功能复杂，单条指令可以完成多个操作，例如加载、计算和存储操作可以在一条指令中完成。
- **指令长度**: 指令长度不固定，通常包含多个操作码和操作数。

### • RISC:

- **指令数量**: 指令集简洁，通常只有几十条到一百条左右的指令。
- **指令功能**: 指令功能简单，每条指令只完成一个操作，例如加载、计算和存储操作分别由不同的指令完成。
- **指令长度**: 指令长度固定，通常为32位或64位，便于硬件解码和执行。

## 3. 硬件实现

### • CISC:

- **硬件复杂度**: 由于指令复杂，硬件实现复杂，需要更多的微代码支持。
- **执行效率**: 单条指令的执行时间可能较长，因为复杂指令需要多个微操作来完成。
- **适用场景**: 适合对软件开发效率要求较高的场景，例如早期的个人计算机和嵌入式系统。

### • RISC:

- **硬件复杂度**: 硬件实现相对简单，因为指令简单且固定长度，便于流水线化和并行处理。
- **执行效率**: 单条指令的执行时间短，处理器可以通过流水线技术实现高吞吐量。
- **适用场景**: 适合对性能要求较高的场景，例如服务器、高性能计算和现代嵌入式系统。

## 4. 应用

- **CISC**: 虽然传统 **CISC** 架构（如 **x86**）仍然广泛应用于个人计算机和服务端，但现代 **CISC** 处理器（如 **Intel** 和 **AMD** 的 **CPU**）也引入了许多 **RISC** 技术，例如指令流水线化和微操作解码；
- **RISC**: **RISC** 架构在高性能计算和移动设备中占据主导地位。例如，**ARM** 架构（**RISC**）广泛应用于智能手机、平板电脑和服务端，而 **RISC-V** 作为一种开源的 **RISC** 架构，正在迅速发展并得到广泛应用；