

RISC-V简介

大家会发现，在实验中会反复提到RISC-V汇编。本节将带大家大致认识一下RISC-V指令系统，方便后续实验的进行。

关于RISC-V的更多介绍，可以参考[RISC-V官网](#)和[RISC-V Book](#)，The RISC-V Instruction Set Manual ([Volume I](#), [Volume II](#))也能帮你详细了解 RISC-V ISA 的各类细节。

RISC-V，读作读作“risk-five”，是一种开放式指令集架构 (ISA)，它是由加州大学伯克利分校开发的。RISC-V 架构的设计目标是提供一种简单、可扩展、开放的 ISA，适用于各种计算机系统和应用场景。以下是 RISC-V 架构的主要构成和指令等信息。

RISC-V 架构构成

寄存器文件

RISC-V 架构包含 32 个通用寄存器，每个寄存器的位数为 32 位或 64 位。同时，它还包含了一些特殊的寄存器，如程序计数器 (PC) 和堆栈指针 (SP) 等。

例如：每个RISC-V CPU都有一组控制寄存器，内核通过向这些寄存器写入内容来告诉CPU如何处理陷阱，内核可以读取这些寄存器来明确已经发生的陷阱。RISC-V文档包含了完整的内容。

以下是最重要的一些寄存器概述：

- `stvec`：内核在这里写入其陷阱处理程序的地址；RISC-V跳转到这里处理陷阱。
- `sepc`：当发生陷阱时，RISC-V会在这里保存程序计数器 `pc`（因为 `pc` 会被 `stvec` 覆盖）。
- `sret`（从陷阱返回）指令会将 `sepc` 复制到 `pc`。内核可以写入 `sepc` 来控制 `sret` 的去向。
- `scause`：RISC-V在这里放置一个描述陷阱原因的数字。
- `sscratch`：内核在这里放置了一个值，这个值在陷阱处理程序一开始就会派上用场。
- `sstatus`：其中的**SIE**位控制设备中断是否启用。如果内核清空**SIE**，RISC-V将推迟设备中断，直到内核重新设置**SIE**。**SPP**位指示陷阱是来自用户模式还是管理模式，并控制 `sret` 返回的模式。

指令集

RISC-V 架构定义了一套完整的指令集，包括基本指令集和标准扩展指令集。基本指令集包含了最基本的计算、存储和分支等指令，而标准扩展指令集则提供了额外的指令，如乘法、除法、浮点运算等。这种模式使得RISC-V更容易支持向后兼容。每一个RISC-V处理器可以声明支持了哪些扩展指令集，然后编译器可以根据支持的指令集来编译代码。

基本指令集

基本指令集包括了最基本的计算、存储和分支等指令，如下所示：

- **add**：加法指令，将两个寄存器的值相加，存储到第三个寄存器中。
- **sub**：减法指令，将两个寄存器的值相减，存储到第三个寄存器中。
- **lw**：加载字（4字节）指令，从内存中读取一个字，并存储到寄存器中。
- **sw**：存储字（4字节）指令，将寄存器中的值存储到内存中。
- **beq**：分支相等指令，如果两个寄存器的值相等，跳转到指定的位置。
- **bne**：分支不相等指令，如果两个寄存器的值不相等，跳转到指定的位置。

标准扩展指令集

标准扩展指令集提供了额外的指令，如乘法、除法、浮点运算等。以下是一些常见的标准扩展指令集：

- **RV32M**：乘法扩展指令集，提供了乘法和除法指令。
- **RV64M**：64位乘法扩展指令集。
- **RV32F**：浮点扩展指令集，提供了单精度浮点数的计算指令。
- **RV64F**：64位浮点扩展指令集。
- **RV32A**：原子扩展指令集，提供了原子操作的指令，用于多线程应用场景。
- **RV64A**：64位原子扩展指令集。

内存模型

RISC-V 架构定义了一种灵活的内存模型，可以支持各种不同的内存访问模式，包括原子操作、虚拟内存和多线程等。

物理地址空间

RISC-V 架构支持 32 位和 64 位的物理地址空间，可以支持各种不同的内存大小和寻址模式。

RISC-V的特点

现在大多数电脑都运行在x86和x86-64处理器上。x86拥有一套不同的指令集，看起来与RISC-V非常相似。但事实上它们并不是如此，RISC-V中的RISC是精简指令集（Reduced Instruction Set Computer）的意思，而x86通常被称为CISC，复杂指令集（Complex Instruction Set Computer）。这两者之间有一些关键的区别：

- 简单易懂：在x86-64中，很多指令都做了不止一件事情。这些指令中的每一条都执行了一系列复杂的操作并返回结果。但RISC-V的指令集非常简单，具有固定大小的指令，使其易于解码和实现，相应的也消耗更少的CPU执行时间。所以相比之下，x86和MIPS等传统指令集架构具有更复杂的指令，这增加了处理器的复杂性和成本。
- 模块化设计：RISC-V的设计是高度模块化的，这意味着它可以根据特定的需求进行定制和优化。相比之下，x86和MIPS等传统指令集架构的设计是固定的，难以进行修改和优化。
- 开放源代码：RISC-V的指令集是市场上唯一的一款开源指令集，这意味着任何人都可以自由地使用、修改和分发它。这使得RISC-V成为一种可扩展的指令集架构，可以在各种不同的硬件上运行。
- 可扩展性：RISC-V的指令集包括基本指令集（RV32I和RV64I）、常见扩展指令集（例如乘法扩展指令集RV32M和RV64M）以及可选扩展指令集（例如向量扩展指令集RV32V和RV64V）。这使得RISC-V可以根据不同的应用需求进行定制和优化。

总结

RISC-V 架构是一种开放式指令集架构，其设计目标是提供一种简单、可扩展、开放的 ISA，适用于各种计算机系统和应用场景。它包含 32 个通用寄存器、一套完整的指令集、一种灵活的内存模型和支持 32 位和 64 位的物理地址空间等。同时，它还提供了多个标准扩展指令集，如乘法扩展、浮点扩展和虚拟内存扩展等，以满足不同应用场景的需求。