计算机漫游

信息：位+上下文

一、一个 hello 程序是如何在计算机中运行的?

1.编译系统:

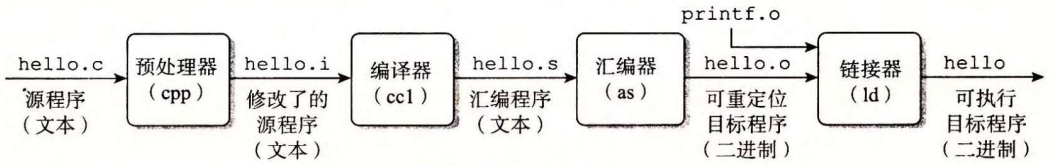

图 1-3　编译系统

2.处理器（CPU）读并解释储存在内存中的指令:

先学习 计算机系统的硬件组成

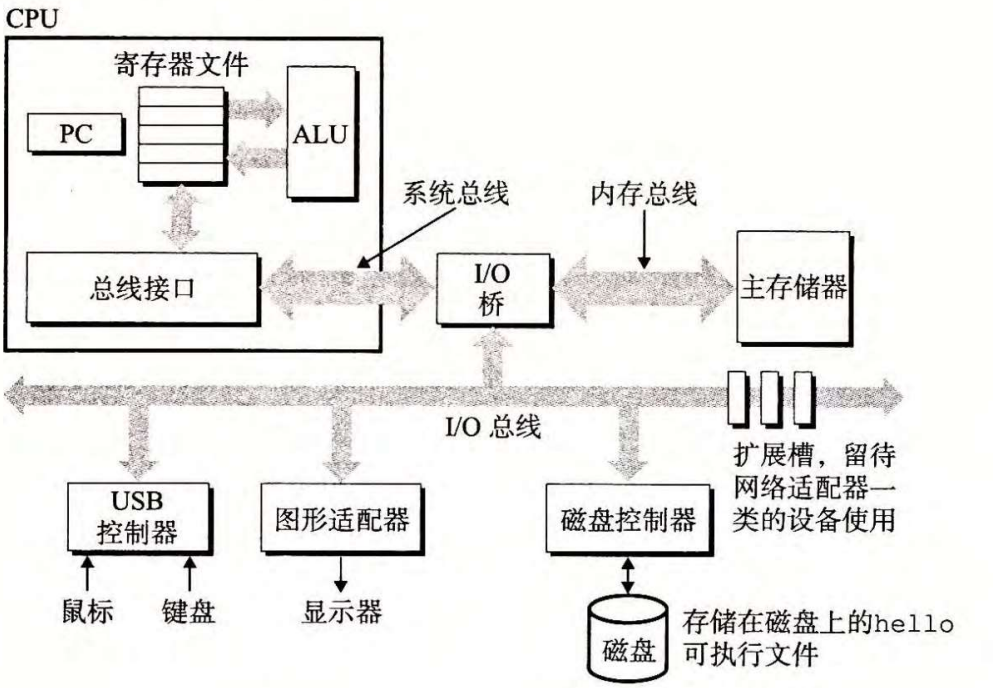

图 1-4　一个典型系统的硬件组成

CPU：中央处理单元；ALU：算术/逻辑单元；PC：程序计数器；USB：通用串行总线

Hello 程序如何工作?

1） shell 程序将命令字符“hello”逐一读入寄存器，再把它存放到内存中

2）然后 shell 执行一系列指令来加载可执行的 hello 文件，将 hello 目标文件中的代码和数据从磁盘复制到主存

3）利用直接存储器存取 CDMA，将在第 章中讨论）技术，数据可以不通过处理器而直接从磁盘到达主存。

4）一旦目标文件 hello 中的代码和数据被加载到主存，处理器就开始执行 hello 程序 main 程序中的机器语 指令。这些指令将 "hello, world n" 字符串中的字节从主存 复制到寄存器文件，再从寄存器文件中复制到显示设备，最终显示在屏幕上。

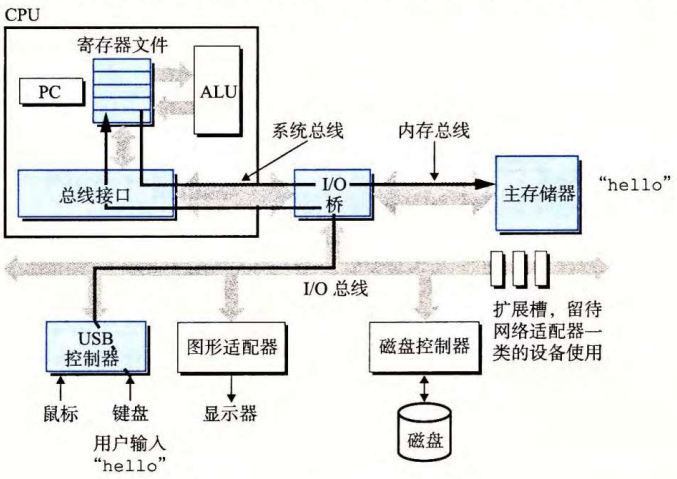

图 1-5　从键盘上读取 hello 命令

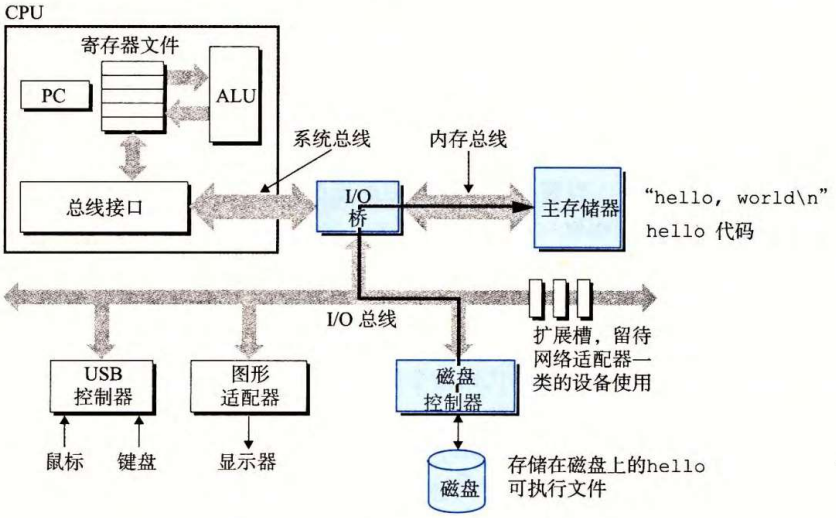

图 1-6　从磁盘加载可执行文件到主存

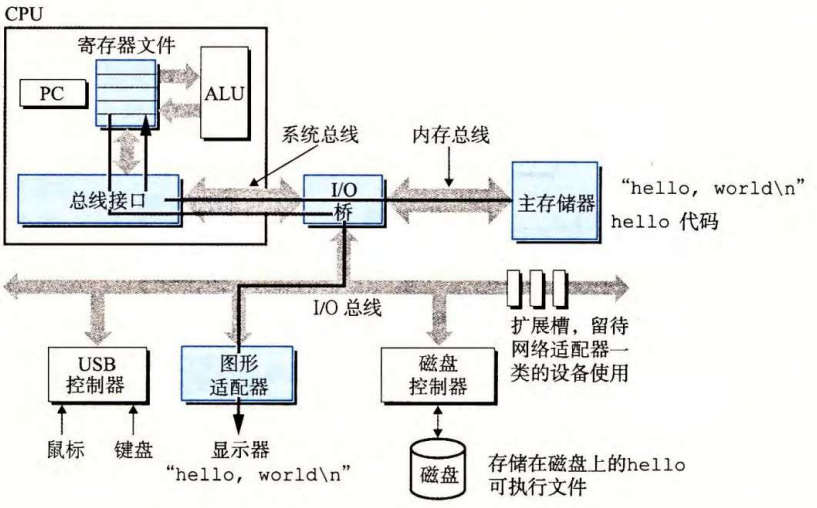

图 1-7　将输出字符串从存储器写到显示器

二、CPU cache 机制以及 cache miss

（一）.为什么引入 cache

引入 Cache 的理论基础是**程序局部性原理（即程序具有访问局部区域里的数据和代码的趋势）**，包括时间局部性和空间局部性。即最近被 CPU 访问的数据，短期内 CPU 还要访问（时间）；被 CPU 访问的数据附近的数据，CPU 短期内还要访问（空间）。因此如果将刚刚访问过的数据缓存在 Cache 中，那下次访问时，可以直接从 Cache 中取，其速度可以得到数量级的提高。

（二）.cache miss

1.定义：CPU 要访问的数据在 Cache 中有缓存，称为“命中” (Hit)，反之则称为“缺失” (Miss)

2.种类：

Cold miss：cache 为空时会发生的 miss

Conflict miss：多个 K+1 级的 blocks 被映射到 K 级中同一个 block

Capacity miss：程序常会分阶段执行（例如循环：内层、外层），每个阶段会 cache blocks 的固定几个块，这几个块所构成的集合称为 working set。 当 working set 超过 cache 大小时所发生的 miss 称为 capacity misses

3.怎么 reduce cache misses?

Change capacity：改变 cache 容量

Change associativity：简化调用关系，减少 cache miss 次数

Change compiler：编译器优化

（三）.cache 一致性

多个处理器对某个内存块同时读写，会引起冲突的问题，这也被称为 Cache 一致性问题。

1. 多个处理器核（core）同时对某个内存块进行读写操作，导致一个内存块同时可能有多个备份，有的已经写回到内存中，有的在不同的处理器核心的一级、二级 Cache 中。由于 Cache 缓存的原因，我们不知道数据写入的时序性，因而也不知道哪个备份是最新的。
2. 假设有两个线程 A 和 B 共享一个变量，当线程 A 处理完一个数据之后，通过这个变量通知线程 B，然后线程 B 对这个数据接着进行处理，如果两个线程运行在不同的处理器核心上，那么运行线程 B 的处理器就会不停地检查这个变量，而这个变量存储在本地的 Cache 中，因此就会发现这个值总也不会发生变化。（story：张三与张三风）

Cache 有两种方案：

两者区别是 Trans(将虚拟地址翻译成物理地址)在 cache 前还是 cache 后。

其中第二种可以规避 cache 一致性问题，但是 trans 花费的时间太长，miss 概率低，不利于性能的提高。