D1

slug	note-1
上次编辑	@2026年2月10日 21:38
创建时间	@2026/02/01
标识	csapp-note
状态	完成

摘要

- 信息就是位 + 上下文
- 程序被其他程序翻译成不同形式（编译过程）
- 处理器读并解释存储在内存中的指令
- 存储设备形成层次结构
- 操作系统管理硬件
- 网络在系统中的角色

NOTE

编译过程

hello.c → 预处理器（cpp） → hello.i → 编译器（cc1） → hello.s → 汇编器（as） → hello.o → 链接器（ld） → a.out

💡 分别对应：宏展开、生成汇编、生成机器码、符号解析与重定位。
与 OS 的关系：

1. 可执行文件和进程的关系（程序 + 数据 被 OS 载入为进程）；
2. 链接后的可执行文件里有“代码段、数据段、符号表”等。

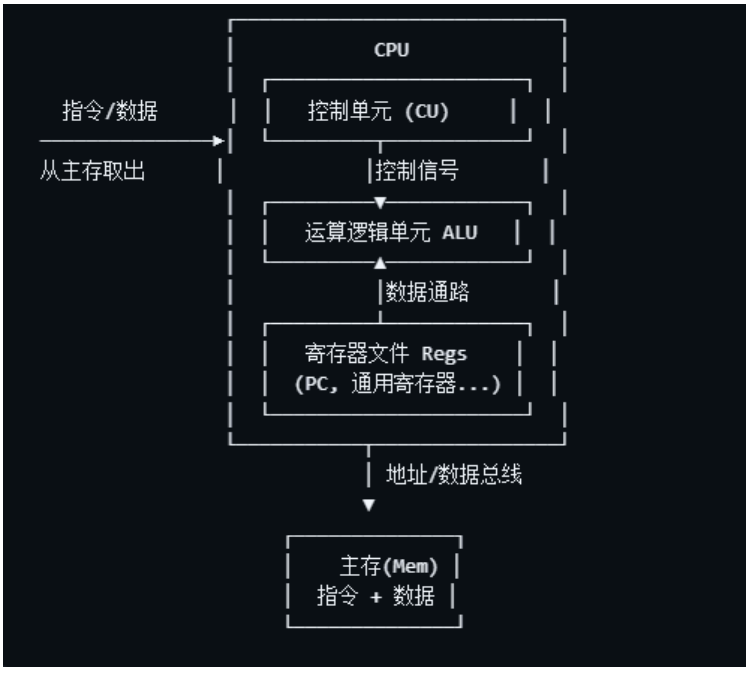
```
gcc -E hello.c -o hello.i          # 预处理
gcc -S hello.i -o hello.s          # 编译成汇编
gcc -c hello.s -o hello.o          # 汇编成目标文件
gcc hello.o -o hello               # 链接成可执行文件
```

处理器执行指令

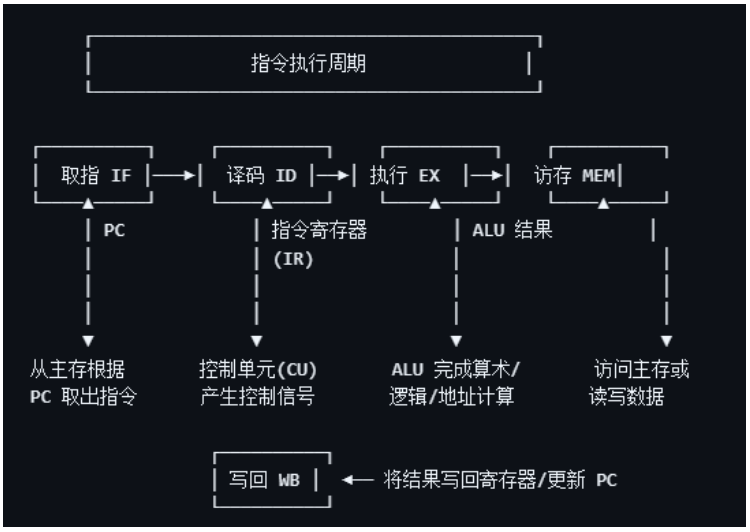
- CPU：寄存器文件 + ALU + 控制单元
- 主存：存放指令和数据
- CPU 周期：取指 – 译码 – 执行 – 访存 – 写回

简图：

1. 整体结构：



2. “取指–译码–执行”周期视角：



- IF（取指）：
  - PC 在寄存器文件里；
  - 通过地址总线去主存取指令；
  - 取回来的指令放到指令寄存器（IR）中。
- ID（译码）：
  - 控制单元（CU）读取 IR，分析操作码和操作数位置；
  - 产生对寄存器、ALU、存储器的控制信号。
- EX（执行）：
  - 寄存器中的操作数送入 ALU；
  - ALU 执行加减、逻辑、移位、地址计算等。
- MEM（访存）：
  - 若指令需要访问内存（load/store），用 EX 阶段算��的地址访问主存。
- WB（写回）：
  - 把 ALU 结果或访存结果写回某个寄存器，或更新 PC（比如跳转）。

存储器层次结构

- 寄存器
- L1 Cache、L2 Cache（有的书拆更细）
- 主存（DRAM）
- 磁盘/SSD
- 可能还有网络存储等

> 💡 从上到下：速度变慢、容量变大、单价变低；

操作系统的角色

1. 三层结构
    - 顶层：应用程序（C 程序、Shell、浏览器等）
    - 中间：操作系统（进程管理、存储管理、文件系统、I/O 管理）
    - 底层：硬件（CPU、内存、磁盘、网络接口）
2. 交互：
    - 应用程序 调用 **系统调用** → OS；
    - OS 控制 CPU 调度、内存分配、I/O 设备等；
    - OS 最终通过访问硬件寄存器、内存映射 I/O、总线等操纵硬件。
3. OS组件对应的关键字：
    - 进程管理：创建/撤销进程、调度、上下文切换；
    - 存储管理：分页、分段、换页算法；
    - 文件系统：目录结构、inode、磁盘调度；
    - I/O 管理：中断、DMA、设备驱动。