# 计算机组成原理期末复习笔记

## 写在前面

本篇笔记覆盖了计算机组成原理这门课的主干内容，较为细致地给出一些重点称谓的定义，并在编写中结合自己的理解写了一些理解知识点的心得。但由于笔者考完算法后开始着手制作，在时间上确实比较紧张，因此在某些章节没有事无巨细地交代每一个定义的介绍，大部分只是列出了主干知识点提要（第六章涉及算术运算模式化较强，故这里没有作系统整理），这也决定了笔记不可能代替例题起到帮助理解非常显著的作用。但无论如何，对于系统地温习知识点并查缺补漏，他能够在基础层面给到一定的帮助，为了在考试中取得好成绩，建议大家同时结合 ppt 以及教材习题加以理解。

## 一、 系统总线

### 1.1 总线概述

#### 1.1.1 总线的基本概念

1. 总线定义：总线是连接各个部件的信息传输线，是各个部件共享的传输介质。
同一时刻只能有**一个部件**向总线发送信息；**多个部件**可以同时从总线接受相同信息。
2. 总线上信息的传送可分为两种方式：串行与并行
3. 总线特性：1）机械特性：尺寸、形状、管脚数及排列顺序

2）电气特性：传输方向和有效的电平范围

3）功能特性：每根传输线的功能

4）时间特性：信号的时序关系

#### 1.1.2 总线的分类

1. 片内总线：芯片内部的总线
2. 系统总线：计算机系统内各功能部件之间相互连接的总线，可分为：
   1） 数据总线：用来传递各功能部件间的数据信息（**双向传输总线**，位数与机器字长、存储字长相关）
   2） 地址总线：用来传递地址信息（**单向传输总线**，位数与主存地址空间大小有关）
   3） 控制总线：传输的是控制信息（有出有入，请求和命令下达）
   （**数据通路**表数据流经的路径，**数据总线**是承载数据信息的媒介）
3. 通信总线：也叫外部总线，用于计算机系统之间或计算机系统与其他系统（如控制仪表、移动通信等）之间的通信

#### 1.1.3 总线的结构

1. 单总线结构：
   优点：结构简单成本低，易于接入新设备
   缺点：带宽低、负载重，多个部件争用唯一的总线，且不支持并发传送操作
2. 双总线结构：
   优点：将低速的 I/O 设备从单总线上分离出来，实现了存储器总线和 I/O 总线分离
   缺点：需要增加通道等硬件设备
3. 三总线结构：（新增 DMA 总线用于在内存和高速外设间直接传送数据）
   优点：提高了 I/O 设备的性能，使其快速响应设备，提高系统吞吐量
   缺点：系统工作效率低
   （**任意时刻只能使用一种总线**）

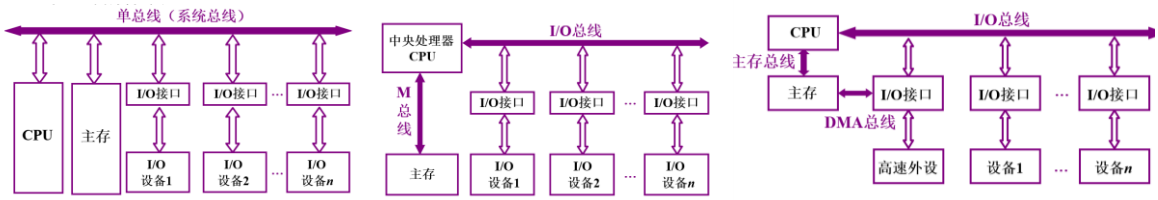


**图 1 单总线结构 双总线结构 三总线结构**

### 1.1.4 常见的总线标准

典型的总线标准有：ISA、EISA、VESA、PCI、AGP、PCI-E、USB 等，他们的主要区别是总线宽度、带宽、时钟频率、寻址能力以及是否支持突发传送等。

ISA（Industry Standard Architecture）总线 支持 24 位地址线 支持 8 位（PC）/16 位（PC/AT）数据线 总线时钟频率 8MHz 用于 80286 计算机

EISA（Extended ISA）总线 支持 32 位地址线和数据线 总线时钟频率 8MHz DMA 方式下可达 33MB/s 传输速率 用于 80386/80486 计算机、

VL-BUS（VESA Local Bus）总线 与 EISA 兼容、外加主存总线以实现高速传输 总线时钟频率与 CPU 外频同步（25-40MHz） 主存总线驱动能力有限，最多接 3 个扩展卡 用于 80486 计算机

PCI（Peripheral Component Interconnect）总线 总线时钟频率 33/66MHz，并与 CPU 独立 总线宽度 32 位/64 位 数据传输率 132MB/s 起，所有设备共享 即插即用（自动分配地址空间、中断号等）

PCI-X（PCI eXtended）总线 与 PCI 总线物理兼容 总线时钟频率支持 66/100/133MHz 支持 DDR 和 QDR 技术，最高传输率 533MB/s 多用于服务器和高端 PC 用户

PCI-E（PCI Express）总线 高速串行总线 可包含多个数据通道 x1/x2/x4/x8/x16 数据传输率 250MB/s 起 软件与传统 PCI 兼容

USB 通用串行总线，是一种连接外部设备的 I/O 总线，属于设备总线，是一种即插即拔、热插拔的特点，有很强的连接能力

（这里建议重点理解一下更新迭代趋势，着重记特点即可）

### 1.1.5 总线的性能指标

总线传输周期：指一次总线操作所需的时间，由若干总线时钟周期构成
总线时钟周期：机器的时钟周期，总线受机器统一时钟控制
总线工作频率：总线上各种操作的频率，为总线周期的倒数
总线时钟频率：机器时钟频率，即总线时钟周期倒数
**总线宽度**：总线位宽，即总线上能同时传输的数据位数
**总线带宽**：即总线的最大数据传输率，单位时间内总线最多可传输的数据位数
总线复用：一种信号线在不同时间传输不同信息

其中，总线最主要性能指标为主线工作频率、主线宽度与主线带宽，其关系为：

$$\text{主线带宽} = \text{总线宽度} \times \text{总线频率}$$

## 1.2 总线事务与总线控制

### 1.2.1 总线通信控制

申请分配阶段：主模块申请，总线仲裁决定
寻址阶段：主模块向从模块给出地址和命令
传数阶段：主模块和从模块交换数据
结束阶段：主模块撤消有关信息

总线通信控制的方式：

1. **同步通信**
   双方由同一时标控制数据传送，优点是规定明确、统一，模块间配合简单一致。缺点是主从设备强制性同步，严重影响总线工作效率。

同步通信适用于总线长度较短、各部件存取时间比较一致的场合

2. **异步通信**
   异步通信克服了同步通信的缺点，允许各模块速度的不一致性。
   其没有公共时钟标准，不要求所有部件严格地统一时间，而是采取应答的方式（握手方式）。
   即当主模块发出请求信号后，一直等待模块反馈回来响应信号后才开始通信。
   应答方式可分为不互锁、半互锁以及全互锁三种
3. **半同步通信**
   保留了同步通信的基本特点，同时又像异步通信那样允许不同速度的模块和谐地工作，为此增设了一条等待响应信号线，采用插入时钟周期的措施协调双方的配合问题。
4. **分离式通信**
   充分挖掘系统总线每个瞬间的潜力
   一个总线传输周期被分为两部分
   子周期 1：主模块申请占用总线，使用完后即放弃总线的使用权
   子周期 2：从模块申请占用总线，将各种信息送至总线上

   分离式通信特点：
   1. 充分提高了总线的有效占用
   2. 采用同步方式通信，不等对方回答
   3. 各模块准备数据时，不占用总线
   4. 总线被占用时，无空闲

### 1.2.2 总线判优控制

可分为集中式和分布式两种，前者将控制逻辑集中在一处，后者将控制逻辑分散在与总线连接的各个部件和设备上

其中集中控制优先权仲裁方式有以下三种：
链式查询、计数器定时查询以及独立请求方式

# 二、 存储系统

## 2.1 概述

### 2.1.1 存储器分类

按存储器介质份分类：
1. 半导体存储器（TTL、MOS）；易失
2. 磁表面存储器（磁头、载磁体）；非易失
3. 磁芯存储器（磁芯是硬磁材料做成的环状元件）；非易失
4. 光盘存储器（磁光材料）；非易失

按存取方式分类：
1. 随机存储器（RAM） 在程序的执行过程中可读可写，其特点是存储器内任何一个存储单元的内容都可以随机存取，且存取时间与存储单元的物理位置无关。
2. 只读存储器（ROM） 在程序的执行过程中只读。早期 ROM 称为掩模型 ROM，存于芯片的信息一旦制成无法更改，但随后制出 PROM/EEPROM/Flash 可进行数据擦除
3. 串行访问存储器
   顺序存取存储器 磁带存储器
   直接存取存储器 磁盘存储器

在计算机中的作用分类
1. 主存储器

RAM（可读可写）；静态 RAM、动态 RAM
ROM（只读）；MROM、PROM、EPROM、EEPROM
2. 辅助存储器 磁盘、磁带、光盘
3. 缓存（Cache）

### 2.1.2 存储器层次结构

存储器有 3 个主要性能指标：速度、容量和每位价格

结合存储器山，缓存-主存层次主要解决 CPU 和主存速度不匹配的问题，主存-辅存层次主要解决存储系统的容量问题。整体分析，辅存速度接近于主存，而容量接近于辅存，平均价位接近于更加低廉的辅存价位，缓存-主存同理，解决了三者矛盾。

## 2.2 主存储器

### 2.2.1 基本知识概述

1. 主存同 CPU 关系见下图：

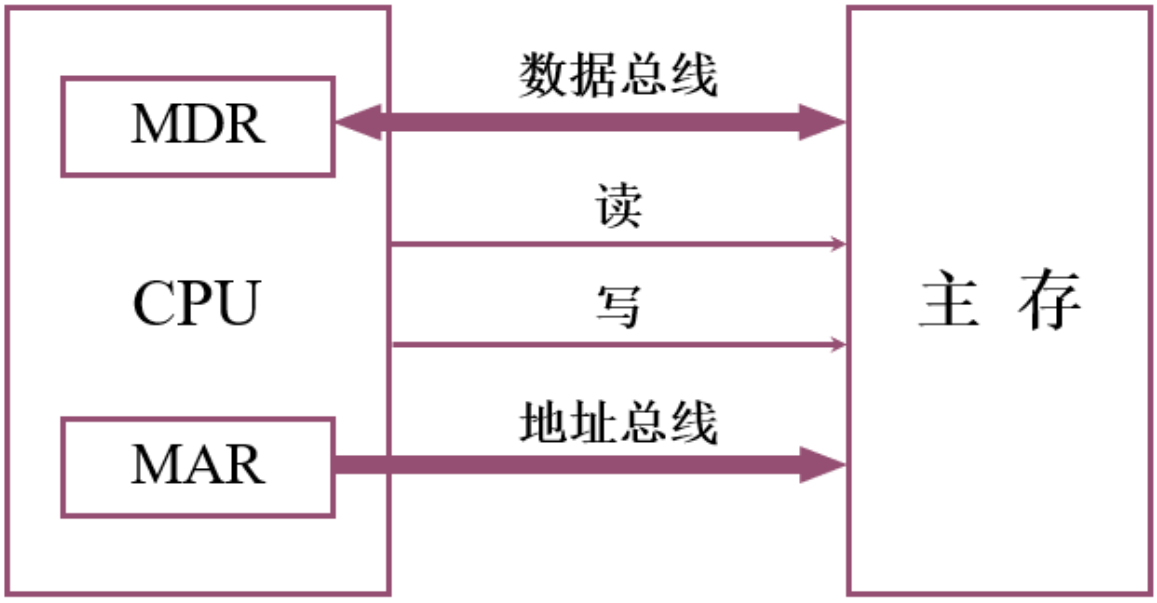


**图 2 主存与 CPU 关系**

2. 主存中存储单元地址的分配：
   这里注意计算机系统**既可按字寻址，也可按字节寻址**，不同要求下，其**寻址范围**也会随之变化，之后关于地址格式设计时会影响 b 的大小
3. 主存的技术指标：存储容量和存储速度
   存储容量：主存存放二进制代码的总位数
   存储速度：
   1、**存取时间**：存储器的访问时间，可分为读出时间和写入时间
   2、**存取周期**：连续两次独立的存储器操作（如连续两次读操作），所需的最小间隔时间
   3、存储器的带宽：单位时间内存储器存取的信息量

### 2.2.2 存储芯片与 RAM

1. 存储芯片结构

存储芯片通过地址总线、数据总线和控制总线与外部连接。
其中地址线是单向输入的，其位数与芯片容量有关。
数据线是双向的，其位数与芯片可读出或写入的数据位数有关。

**地址线与数据线的位数**共同反应存储芯片的容量：
如地址线有 14 根，数据线有 1 根，那么存储器容量为 16K×1 位

2. 译码驱动方式
驱动方式有两种：线选法和重合法
线选法:地址译码信号只选中同一个字的所有位,结构简单,费器材;
重合法:地址分行、列两部分译码,行、列译码线的交叉点即为所选单元。这种方法通过行、列译码信号的重合来选址,也称矩阵译码。可大大节省器材用量,是最常用的译码驱动方式。
3. RAM
按存储信息原理的不同可分为两大类：SRAM 与 DRAM

静态 RAM 是靠双稳态触发器来记忆信息的；
动态 RAM 是靠 MOS 电路中的栅极电容来记忆信息的。
由于电容上的电荷会泄漏，需要定时给与补充，所以动态 RAM 需要设置刷新电路。
但动态 RAM 比静态 RAM 集成度高、功耗低，从而成本也低，适于作大容量存储器。所以主内存通常采用动态 RAM，而高速缓冲存储器（Cache）则使用静态 RAM。

静态 RAM 的特点：是在**不断电**的条件下，其中的信息保持不变，因而不必定期刷新，其中的信息可读可写，但**断电后信息就会丢失**。
1、静态 RAM 用触发器作为存储单元存放 1 和 0，存取速度快，只要不掉电即可持续保持内容不变。
2、，静态 RAM 的集成度较低，并且静态 RAM 无须考虑保持数据而设置的刷新电路，故扩展电路较简单。
动态 RAM：是绝大多数现代台式计算机的标准计算机内存，它是一种易失性存储器，需要用电压定期刷新，否则，它会丢失存储在上面的信息。

**表 1 SRAM 与 DRAM 比较**

| 特性 | SRAM | DRAM |
|---|---|---|
| 存储信息 | 触发器 | 电容 |
| 破坏性读出 | 非 | 是 |
| 需要刷新 | 不要 | 需要 |
| 送行列地址 | 同时送 | 分两次送 |
| 运行速度 | 快 | 慢 |
| 集成度 | 低 | 高 |
| 存储成本 | 高 | 低 |
| 功耗 | 高 | 低 |
| 适用场合 | 高速小容量存储器 | 大容量主存 |

4. DRAM 的**刷新**（这里具体看一下教材 86 页，有可能涉及计算）
刷新:对 DRAM 定期进行的全部重写过程;
刷新原因:因电容泄漏而引起的 DRAM 所存信息的衰减需要及时补充,因此安排了定期刷新操作;
常用的刷新方法有三种:集中式、分散式、异步式：
集中式:在最大刷新间隔时间内,集中安排一段时间进行刷新,存在 CPU 访存死时间。
分散式:在每个读 /写周期之后插入一个刷新周期,无 CPU 访存死时间，但存取周期变长了，整个系统的速度降低了
异步式:是集中式和分散式的结合，既可缩短死时间，又充分利用最大刷新间隔位 2ms 的特点。

## 2.2.3 存储器与 CPU 的连接

这里涉及位扩展、字扩展以及字位同时扩展的知识点
位扩展指增加存储字长，可视为将多块芯片并为一组，每片芯片输出传递到不同的数据线上；
字扩展则是增加存储器字的数量，由片选信号指出工作芯片的序号

关于存储器与 CPU 的连接，建议细读教材 93-94 页，其详细讲述了从地址、数据线的选择、RW、CS 信号的传递连接以及存储芯片的选择方式，结合字位扩展的原理，我们设法涉及芯片组合满足地址要求，并通过设计译码器保证片选信号的正确发出，最后连接 MREQ、G1 信号保证使能端正常工作，从而完成存储器与 CPU 的连接，CPU 通过发出控制信号正确读出有效地址。下面再列出几种设计原则：

1. ROM 用于存放系统程序，RAM 为用户编程而设计
2. 通常将 CPU 地址线的低位与存储芯片的地址线相连，高位在扩充存储芯片时使用，如用来片选
3. CPU 的数据线数若同存储芯片数据线数相同可直接相连，不同则必须进行**位扩展**
4. 最后连线搭接读写命令线与片选线

## 2.2.4 单体多字系统与多体并行系统

为了提高存储器访存速度，除了寻找高速原件和采用层次结构外，调整主存结构也可提高访存速度。

**单体多字系统**允许机器在一个存取周期内，从同一地址读出多条指令，再逐条送至 CPU 执行，这样增大了存储器的带宽，提高了运行速度，但其前提是指令和数据在主存内要是连续存放的，遇转移指令或操作数不连续存放，方法效果不明显。

多体并行系统的存储器采用**低位交叉编址**后，可以在不改变每个模块存取周期的前提下，提高存储器的带宽，实现流水线并行存取。

## 2.2.5 存储器的校验

这里看一下汉明码组成，模式性较强，做几道例题即可

## 2.3 辅助存储器

目前，广泛用于计算机系统的辅助存储器有硬磁盘、软磁盘、磁带、光盘等。
前三种均属于磁表面存储器。

辅存部分概念较多，以重点的磁盘为例：
磁表面存储技术指标：
**记录密度**（道密度、位密度）
**存储容量**（计算方法）
**平均寻址时间**：（平均）寻道时间+等待时间+数据传输时间（有时忽略不计）
**数码传输率**
误码率：出错信息位数与读出信息的总位数的比值

该部分题型主要涉及平均寻址时间的计算、磁盘总存储容量计算、磁盘地址格式设计，相对来说考察变化不大，完成例题即可有较好的理解。

## 2.4 高速缓冲存储器

## 2.4.1 Cache 映射方式

该部分内容是本章的重点内容，Cache 的出现使 CPU 可以不直接访问主存，而与高速 Cache 交换信息。设置 Cache 的作用是解决 CPU 和主存速度不匹配问题。Cache 的基本结构主要由 Cache 存储体、地址映射交换机构、Cache 替换机构以及 Cache 读写操作几大模块组成。

其中，本章考察最为重点的是三种地址映射方式：

1. **直接映射**：主存中的每一块只能被放置到 Cache 中唯一的一个位置。 (循环分配)

空间利用率最低，冲突概率最高，实现最简单。

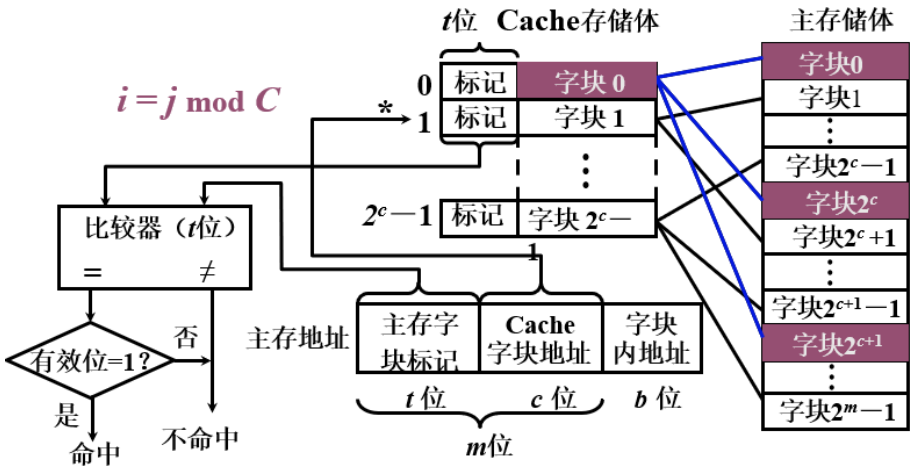


**图 3 直接映射**

2. **全相联**：主存中的任一块可以被放置到 Cache 中的任意一个位置。
特点： 空间利用率最高，冲突概率最低，实现最复杂。

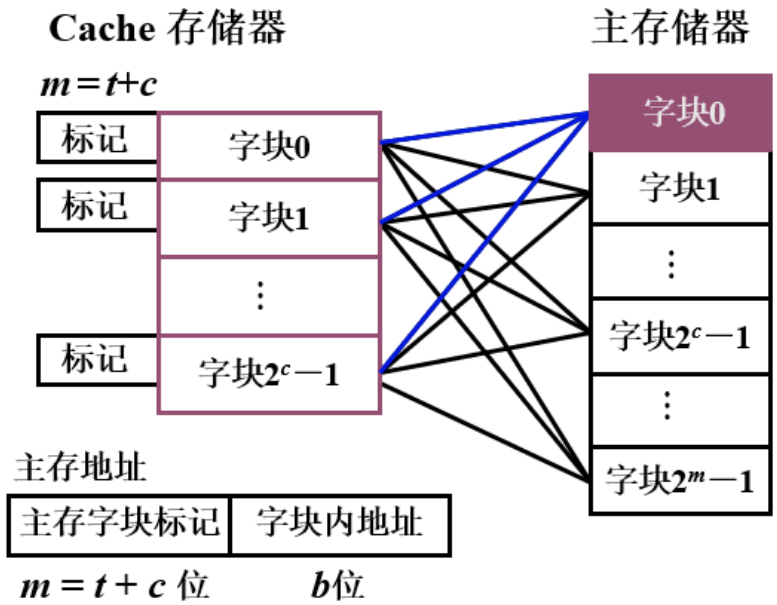


**图 4 全相联映射**

3. **组相联**：主存中的每一块可以被放置到 Cache 中唯一的一个组中的任何一个位置。
特点：值得一提的是，组相联可以视为组间采用直接映射的方法、而组内采用全相联映射的方法，选定合适的组数，可以使组相联映射的成本接近于直接映射，但性能上接近于全相联映射。

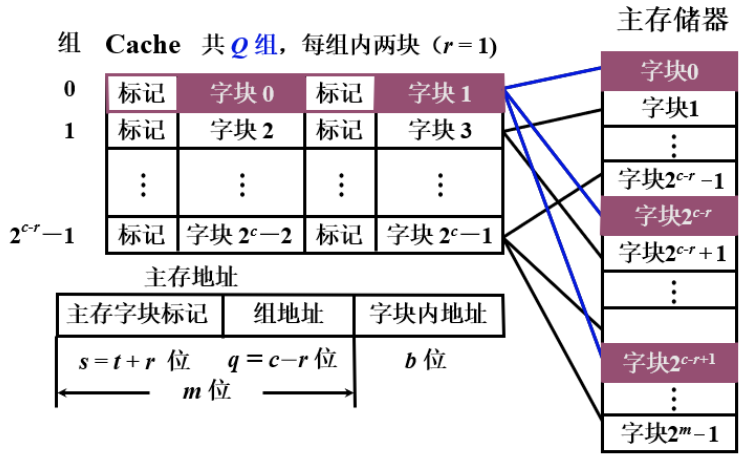


**图 5 组相联映射**

在求解问题的过程中，一般我们习惯上认为每个 Cache 行分为标记项和存储的数据，其中标记项位数由有效位、脏位（视题目要求）以及标记位组成。

具体解题方法视题目要求，一般涉及到组相联映射方法求解较多，如果给定主存地址或主存单元号（可转为主存地址），我们便可通过取模操作确定出组号，进而确定出其他位完成 cache 地址的转换。

### 2.4.2 Cache 的替换算法以及写策略

替换算法常见有三种：随机算法、FIFO 算法以及 LRU 算法
写策略：
如果写命中：
写直达法：写操作时数据既写入主存又写入 cache
写回法：设计脏位，若脏位置 1，则在替换该块时将之写回主存
如果写不命中：
写分配法：加载主存的块到 Cache 中，然后更新这个 Cache 块
非写分配法：只写入主存，不进行调块

**非写分配法常与全写法合用**（多级 Cache 间）
**写分配法常与回写法合用**（下级 Cache 与主存）

## 三、 I/O 设备

本章节重点围绕三种 I/O 方式进行考察，各种方式在代价、性能、解决问题的着重点等方面各不相同，因此应着重复习 I/O 方式，特别程序中断方式（实际上在第九章还做了延申），该部分概念较多，且实现上同 CPU 指令周期有交集，易共同考察。

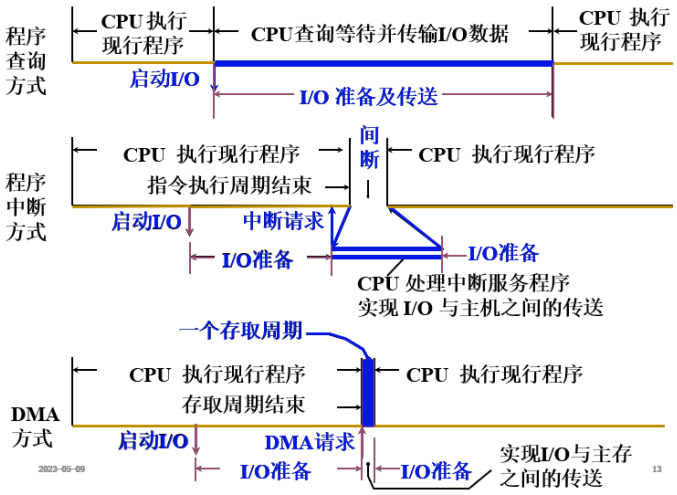


图 6 三种 I/O 方式

## 3.1 I/O 接口

1. I/O 设备编址方式：

1）**独立编址(专用的 I/O 端口编址)**----存储器和 I/O 端口在两个独立的地址空间中

优点：I/O 端口的地址码较短，译码电路简单，存储器同 I/O 端口的操作指令不同，程序比较清晰；存储器和 I/O 端口的控制结构相互独立，可以分别设计

缺点：需要有专用的 I/O 指令，程序设计的灵活性较差

2）**统一编址(存储器映像编址)**----存储器和 I/O 端口共用统一的地址空间，当一个地址空间分配给 I/O 端口以后，存储器就不能再占有这一部分的地址空间

优点：不需要专用的 I/O 指令，任何对存储器数据进行操作的指令都可用于 I/O 端口的数据操作，程序设计比较灵活；由于 I/O 端口的地址空间是内存空间的一部分，这样，I/O 端口的地址空间可大可小，从而使外设的数量几乎不受限制

缺点：I/O 端口占用了内存空间的一部分，影响了系统的内存容量；访问 I/O 端口也要同访问内存一样，由于内存地址较长，执行速度较慢

2. I/O 接口

I/O 接口一般指 CPU 和 I/O 设备间的连接部件；

端口是给信息通讯所划分的通道口是相对于软件来说的，而接口是硬件连接的接口。

外围设备要通过接口与 CPU 相连的原因主要包括：

（1）一台机器通常配有多台外设，它们各自有其设备号（地址），通过接口可实现对设备的选择；

（2）I/O 设备种类繁多，速度不一，与 CPU 速度相差可能很大，通过接口可实现数据缓冲，达到速度匹配；

（3）I/O 设备可能串行传送数据，而 CPU 一般并行传送，通过接口可实现数据串并格式转换；

（4）I/O 设备的入/出电平可能与 CPU 的入/出电平不同，通过接口可实现电平转换；

（5）CPU 启动 I/O 设备工作，要向外设发出各种控制信号，通过接口可传送控制命令；

（6）I/O 需将其工作状态及时报告 CPU，通过接口可监视设备的工作状态，并保存状态信息，供 CPU 查询。

I/O 接口分类方法主要有：

（1）按数据传送方式分有并行接口和串行接口两种；

（2）按数据传送的控制方式分有程序控制接口、程序终端接口和 DMA 接口三种。

## 3.2 I/O 方式

### 3.2.1 程序查询方式

程序查询方式是用户在程序中安排一段输入输出程序，它由 I/O 指令、测试指令和转移指令等组成。CPU 一旦启动 I/O 后，就进入这段程序，时刻查询 I/O 准备的情况，若未准备就绪就踏步等待；

若准备就绪就实现传送。在输入输出的全部过程中，CPU 停止自身操作，效率很低。

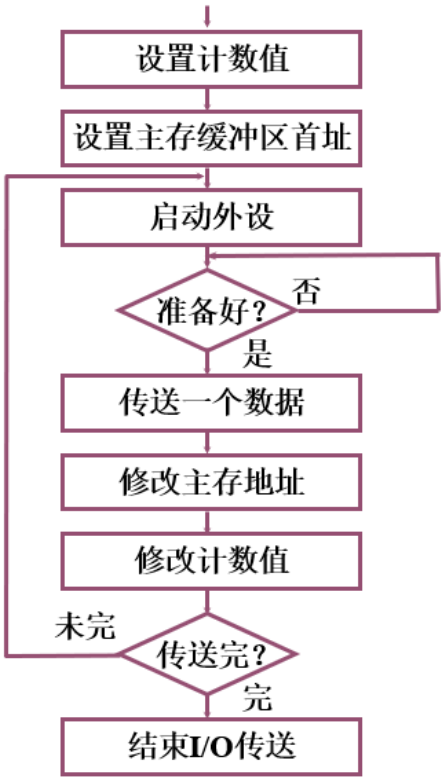


图 7 工作流程

### 3. 2. 2 程序中断方式

程序中断方式虽也要用程序实现外部设备的输入输出，但它只是以中断服务程序的形式插入到用户现行程序中。即 CPU 启动 I/O 后，继续自身的工作，不必查询 I/O 的状态。而 I/O 被启动后，便进入自身的准备阶段，当其准备就绪时，向 CPU 提出中断请求，此时若满足条件，CPU 暂停现行程序，转入该设备的中断服务程序，在服务程序中实现数据的传送。

中断服务程序工作流程：

1. 保护现场（通过中断隐指令完成程序断点的保护）
2. 中断服务
3. 恢复现场（将保存的数据恢复到原寄存器中）
4. 中断返回

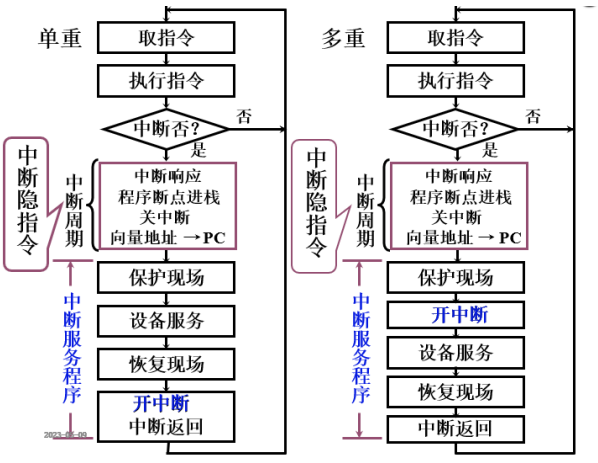


图 8 单重中断和多重中断的服务流程

### 3.2.3 DMA 方式

1. DMA 工作流程

DMA 的数据传送过程分为预处理、数据传送和后处理 3 个阶段：
在预处理阶段，CPU 给 DMA 预置如下信息:
给 DMA 控制路基指明数据传送方向是输入(写主存)还是输出(读主存);
向 DMA 设备地址寄存器送入设备号并启动设备;
向 DMA 主存地址寄存器送入交换数据的主存起始地址；
对字计数器赋予要交换的数据个数；

在数据传送阶段，DMA 方式以数据块为单位传送，通常 DMA 与主存交换数据时采用如下三种方式;停止 CPU 访存、周期挪用以及 DMA 和 CPU 交替访问；

后处理阶段是 DMA 中断请求得到响应后，CPU 停止原程序的执行，转去执行中断服务程序，DMA 做的结束工作包括校验送入主存的数据是否正确，确定是否继续使用 DMA 传送其他数据块。

2. DMA 小结

从数据传送看,程序中断方式靠程序传送,DMA 方式靠硬件传送；
从 CPU 响应时间看,程序中断方式在一条指令执行结束时响应,而 DMA 方式在指令周期内任意存取周期结束时响应,CPU 即将总线控制权让给 DMA 传送；
程序中断方式有处理异常事件的能力,DMA 方式没有这种能力；
程序中断方式需要中断现行程序,故需保护现场,DMA 方式不必中断现行程序,无需保护现场；
DMA 的优先级比程序中断高。

## 四、 指令系统

### 4.1 指令基本格式

指令（机器指令）是指示计算机执行某种操作的命令。一台计算机所有指令的集合构成该机器的指令系统。

一条指令是机器语言的一个语句，通常包括操作码字段与地址码字段两部分，操作码指令中应该执行的操作以及具有何种功能；地址码给出被操作信息的地址。

指令长度指一条指令中包含二进制代码的位数。

指令长度与机器字长没有固定的关系，通常，我们把长度等于机器字长的指令称为单字长指令，指令长度等位两个机器字长的指令称为双字长指令。

根据指令中地址码数目变化可分为如下五种地址格式：

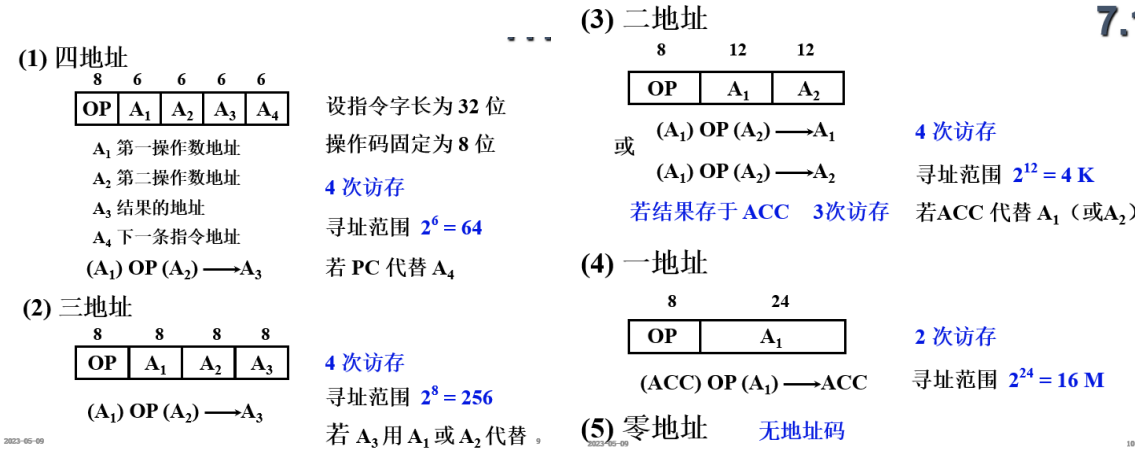


图 9 X 地址指令

这里要注意的是零地址运算类指令以及一地址运算类指令都可以取出隐含地址的操作数进行运算。

除此之位，也需要注意地址内容的存储格式，解题时首先明确地址是以低字节还是高字节作为起始地址进行存放的：


**图 10 地址存放方式**

由于操作码可以是固定的，也可以是可变长的，故我们可以选择扩展操作码的指令格式，从而实现在指令字长有限的前提下仍保持较为丰富的指令种类。

在设计扩展操作码格式时要注意：

1. 不允许短码是长码的前缀（与哈夫曼编码要求同理）
2. 各指令操作码不能重复

这部分主要明确如何设计充分利用操作码位数完成题目的指令设计要求，在这里提一点数目计算方法，如给一个具体的例子，指令规定地址位数为 6 位，那么每减少设计一条二地址指令，可多设计 $2^6$ 条一地址指令或 $2^{12}$ 条零地址指令，操作码的位数随地址数的减少而增加。

## 4. 2 指令的寻址方式

指令系统中采用不同的寻址方式，目的在于为了缩短指令字长，扩大寻址空间，提高编程的灵活性。在这部分主要明确各种寻址方式的特点，如比较寻址速度，基址寻址与变址寻址差别等等。

下面首先给出各种寻址方式的整理：

- **立即寻址**，A 不是操作数地址，而是操作数本身(立即数)，其访存次数=0。
- **隐含寻址**，操作数地址隐含在操作码中，例如 ADD 隐含操作数在 ACC 中，优点是有利于缩短指令长度，其访存次数=0。
- **直接寻址**，EA=A，优点是简单，但 A 字段位数限制寻址范围，只能访问固定地址，不灵活，其访存次数=1。
- **间接寻址**，EA=(A)，实际地址在存储器中，分为 1 次和多次(需要根据存储器最高位来判断是否为最终地址)，可以将操作数寻址范围扩大到存储字长，但需要至少 2 次访存。
- **寄存器寻址**，EA=$R_i$，操作数在指定编号的寄存器中，由于 CPU 中寄存器数量不会太多，故采用这种方式可缩短指令中某个地址段的位数，其访存次数=0。
- **寄存器间接寻址**，EA=(Ri)，比一般间接寻址相比速度更快，常用于循环结构，其访存次数=1。
- **基址寻址**，EA=A+(BR)，BR 是专用的基址寄存器(用户一般不可修改)，或者可显式指定通用寄存器，但其内容仍由操作系统决定，优点为可以扩大寻址范围，有利于设计多道程序，可用于编制浮动程序，其访存次数=1。
- **变址寻址**，EA=A+(IX)，IX 是专用的变址寄存器(用户可修改)，或者可显式指定通用寄存器，地址 A 应该保持不变（作为基地址），可以实现数组访问(A 为数组起始地址，IX 为变化的下标)，其访存次数=1。
- **相对寻址**，EA=(PC)+A，相对当前程序地址进行寻址，A 作为可正可负的位移量使用补码表示，A 的位数决定操作数寻址范围，可以实现位置无关的浮动程序代码，相对寻址广泛应用于转移指令，其访存次数=1。
  注意：这里的 PC 是取完指令进行自增后的 PC！要考虑当前指令长度！
  PC 自增量为整条指令长度，如指令长度如为双字，那么执行该条指令后 PC 值+4
- **堆栈寻址**，堆栈指针寄存器 SP 指向栈顶元素，入栈和出栈会给 SP 增量，增减大小与主存编址方式有关。

本章大题基本也围绕指令格式设计进行出题，其中可能同时考察寻址方式操作码设计。

## 4.3 CISC 与 RISC

关于程序的机器级表示，我们已经在计统这门课中学习了 AT&T 的指令格式，Intel 指令在一些写法和意思理解中与之存在差异，但两种汇编指令的相互转换并不复杂，故在此不再赘述。

考察的主要内容仍围绕于汇编语言的理解，特别在理解寻址命令上，以及要明确各种寄存器的区别和相同点。

本章的最后是关于两种不同风格的指令集：

**RISC（精简指令系统计算机）：**

1. 选用使用频度较高的一些简单指令，复杂指令的功能由简单指令来组合
2. 指令长度固定、指令格式种类少、寻址方式少
3. 只有 LOAD / STORE 指令访存
4. 采用流水技术一个时钟周期内完成一条指令
5. 采用组合逻辑实现控制器
6. CPU 中有多个通用寄存器
7. 采用优化的编译程序，特别重视编译优化工作

**CISC（复杂指令系统计算机）：**

1. 系统指令复杂庞大，各种指令使用频度相差大
2. 指令长度不固定、指令格式种类多、寻址方式多
3. 访存指令不受限制
4. 大多数指令需要多个时钟周期执行完毕
5. 采用微程序控制器
6. CPU 中设有专用寄存器
7. 难以用优化编译生成高效的目的代码

CISC 与 RISC 的比较：

1. RISC 更能充分利用 VLSI 芯片的面积
2. RISC 更能提高计算机运算速度，指令数、指令格式、寻址方式少，通用寄存器多，采用组合逻辑，便于实现指令流水
3. RISC 便于设计，可降低成本，提高可靠性
4. RISC 有利于编译程序代码优化
5. RISC 不易实现指令系统兼容

# 五、 中央处理器

中央处理器是计算机的中心，在本章中，在理解 CPU 的基本构造以及指令周期后，重点在于能够分析典型的指令周期数据流，理解控制器的功能和工作原理。

## 5.1 CPU 的功能与基本结构

CPU 功能主要有：

指令控制（程序顺序控制）、操作控制、时间控制（clock）、数据加工（ALU）、中断处理（对运行中出现的异常情况及特殊请求进行处理）

CPU 基本结构：

中央处理器主要由运算器和控制器两大部分组成

运算器是计算机对数据进行加工处理的中心，主要由算数逻辑单元、暂存寄存器、累加寄存器（ACC）、通用寄存器组（用于存放操作数、目的操作数等，如 AX、BX…）、程序状态字寄存器（PSW）、移位器、计数器等

控制器是整个系统的指挥中枢，其由程序计数器（PC）、指令寄存器（IR）、指令译码器、存储器地址寄存器（MAR）、存储器数据寄存器（MDR）、时序系统和微操作信号发生器等组成

对于各种寄存器，要对其是否对用户透明（用户不可见）加以辨别：
一类为用户可见寄存器，如通用寄存器组、PSW、PC
一类为对用户透明的寄存器，如 MAR、MDR、IR

注意在上述提到的各种存储器中：
PC 位数取决于存储器的容量，即主存容量
IR 位数取决于指令字长
通用寄存器的位数取决于机器字长（为便于操作控制）

## 5. 2 指令周期

**指令周期**定义为取出并执行一条指令所需的全部时间。
一个完整的指令周期应分为四个阶段，并由四个标志触发器加以区别：
**取指周期**为了取出指令，1->FE
**间址周期**为了取出有效地址，1->IND
**执行周期**为了取操作数并执行有关操作，1->EX
**中断周期**为了保存程序断点，1->INT

指令周期的数据流作为贯穿这一章的核心内容，需要对其固定内容加以理解记忆，数据通路基本结构在次基础上细化了每条指令执行时控制信号的流通变化，但内容考察较固定且易于理解，下面给出几种经典的微操作序列：

1. 各指令取指周期基本一样，可直接记忆

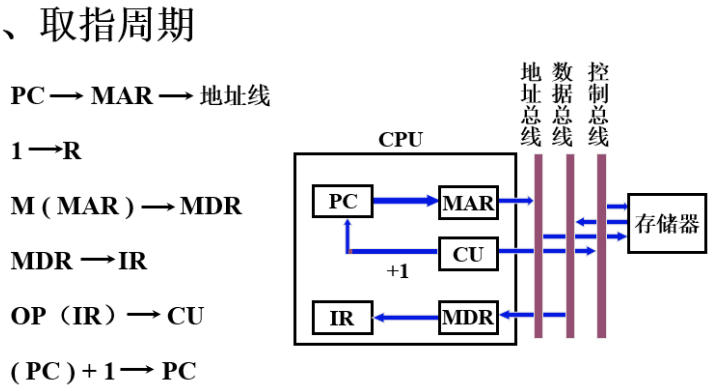


**图 11 取值周期指令**

2. 间址周期：一般涉及要从主存中取地址或操作数的指令会存在间址周期，间址周期中 IND 置 1，间址周期结束后 IND 置 0，EX 置 1 进入执行周期

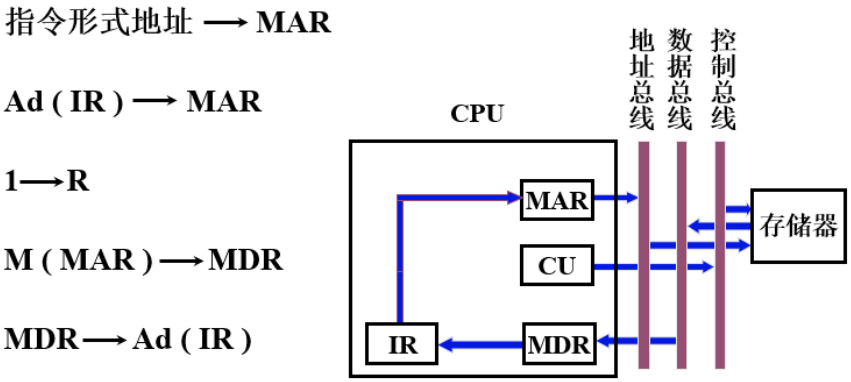


**图 12 间址周期指令**

3. 执行周期：执行周期根据调用指令不同存在一些差别，同时也是指令特点的体现：


**图 13 执行周期指令**

4. 中断周期：中断周期实际实现逻辑为：保存断点，将断点写入主存或压入栈中，再寻找中断服务程序的入口地址，最后将 EINT 置 0（使能端置 0）

| | |
|---|---|
| 程序断点存入 “ 0 ” 地址 | 程序断点 进栈 |
| 0 ⟶ MAR | ( SP ) – 1 ⟶ MAR |
| 1 ⟶ W | 1 ⟶ W |
| PC ⟶ MDR | PC ⟶ MDR |
| MDR ⟶ M ( MAR ) | MDR ⟶ M ( MAR ) |
| 中断识别程序入口地址 M ⟶ PC | |
| 0 ⟶ EINT（置“0”） | 0 ⟶ EINT（置“0”） |

图 14 中断周期指令

## 5.3 组合逻辑设计与微程序设计

多级时序系统中几个概念区别

1） **机器周期**：所有指令执行过程中的一个基准时间
不同指令操作不同，指令周期也不同。
访问一次存储器的时间是固定的，所以常以存取周期为机器周期，即内存中读取一个指令字的最短时间作机器周期。存储字长等于指令字长的前提下，机器周期等于取指周期。

2） **时钟周期**：时钟信号控制节拍发生器，可以产生节拍，每个节拍宽度正好对应一个时钟周期。在每个节拍内机器可完成一个或几个需同时执行的操作。

3） **存取周期**：存储器进行一次读或写操作所需要的时间称为存储器的访问时间，而连续启动两次独立的读或写操作所需的最短时间称为存取周期

### 5.3.1 微程序设计

微程序设计只是对指令周期中的每个机器周期中的操作进行逻辑上的划分，同时出现了一些微操作，但整体上在理解指令运作流程的基础上对个别增加的微操作进行记忆即可基本完成本部分内容。

结合下图可更好帮助理解微程序指令：

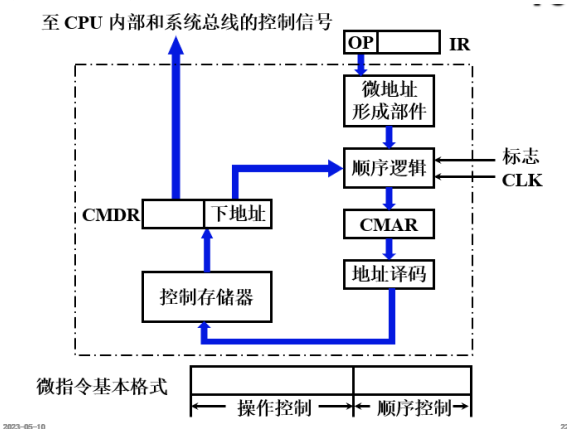


图 15 微程序控制单元的基本框图

微程序设计举例（建议过一遍 415-416 页内容，此处给出取值阶段微程序参考）：

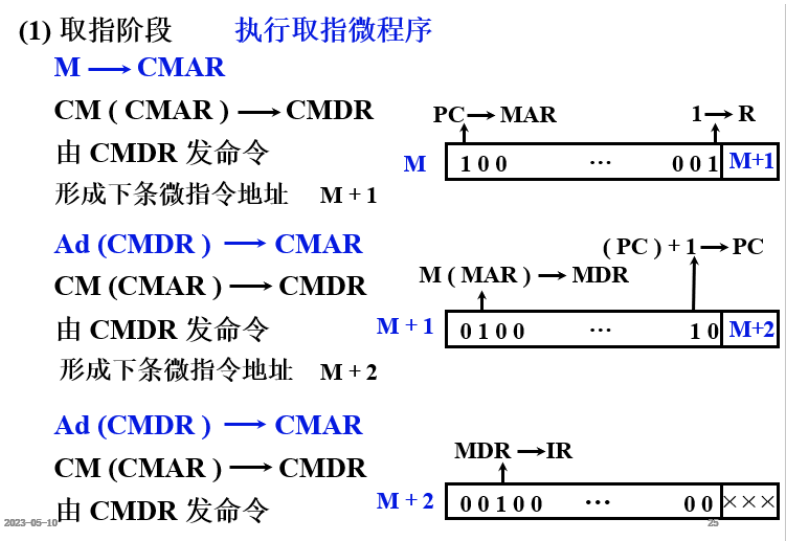


**图 16 取指阶段微程序**

比较：

组合逻辑控制器的设计思想是采用硬连线逻辑。首先根据指令系统，写出对应所有机器指令的全部微操作及其节拍安排，然后列出操作时间表，再写出每一种微操作的逻辑表达式，化简后画出相应的逻辑图，即完成了设计。这种逻辑电路主要由门电路构成的复杂树形网络，一旦构成后，除非在物理上进行重新连线，否则要增加新的控制功能是不可能的。

微程序控制器的设计思想是采用存储逻辑。首先根据指令系统，写出对应所有机器指令的全部微操作及其节拍安排，然后列出操作时间表，再根据微操作的数目，经压缩确定微指令的控制方式、下地址形成方式、微指令格式及微指令字长，编写出全部微指令的代码，即完成了设计。最后将微指令的代码注入到 ROM 中，即可作为微操作的命令信号。

## 写在最后

计算机组成原理涵盖了计算机硬件系统的基本原理和设计方法，为深入了解计算机的工作原理和技术发展奠定了基础。计算机组成原理的内容真的很多，很多细节难以在短时间内全部展现在笔记上，但也希望这篇计算机组成原理的笔记能够对大家的学习有所帮助，祝大家都考试顺利，取得自己满意的成绩。

哈工大计算学部金牌讲师团 沈小荃
2023 年 5 月