

# 计算机组成

## 简单计算机模型

### CPU 基本组成

1. 基本知识：负责提取程序指令，并对指令进行译码，然后按程序规定的顺序对正确的数据执行各种操作
2. 基本组成：
  1. **数据通道**：它由存储单元 (寄存器) 和算数逻辑单元 (对数据执行各种操作) 所组成的网络。这些组件通过总线 (传递数据的电子线路) 连接起来，并利用时钟来控制事件。
  2. **控制单元**：该模块负责对各种操作进行排序并保证各种正确的数据适时地出现在所需的地方。

### Register - 寄存器

- **寄存器** 是一种存储二进制数据的硬件设备，位于处理器的内部，因此处理器可以快速访问寄存器中存储的各种信息。
- 计算机通常处理的是一些存储在寄存器中，具有固定大小的二进制字的数据。
- 可以向寄存器写入或读取信息，信息也可以在不同的寄存器中传递。寄存器的编址方式与存储器不同，寄存器是由 CPU 内部的控制单元进行编址和处理。
- 现代计算机系统中由各种不同类型的专用寄存器，或者只能存储数据，只能存放地址或各种控制信息。还有一些通用寄存器，可以在不同时刻存储不同信息。

### ALU - 算术逻辑单元

- **算术逻辑单元** 在程序执行过程中用于进行逻辑运算。ALU 的各种操作存储会影响状态寄存器的某些数据位的数值。通过控制单元发出的信号，控制 ALU 执行各种运算。

### Control Unit - 控制单元

- **控制单元** 负责监视所有指令的执行和各种信息的传送过程。
- 控制单元负责从内存提取指令，对这些指令进行译码，确保数据适时出现在正确的位置。
- 控制单元还负责通知 ALU 应该使用哪个寄存器，执行哪些中断服务程序，以及对所需执行的各种操作接通 ALU 中的正确电路。

### Bus - 总线

- **总线** 是一组导电路路的组合，它作为一个共享和公用的数据通道将系统内各个子系统连接在一起。
- 在任意时刻，只能有一个设备使用总线。这种共享总线的方式可能会引起通信上的瓶颈。更通常的情况下，各种设备分为 **主设备** 和 **从设备**。主设备是最初启动的设备，从设备是响应主设备请求的设备。
- 总线可以实现 **点对点 (point-to-point)** 的方式连接两个特定设备，或者将总线用作一条 **公用通道 (common pathway)** 来连接多个设备，作为 **多点总线**。
- 典型总线：
  1. **数据总线 (data bus)**：用于数据传递的总线，数据总线传递的是必须在计算机的不同位置之间移动的实际信息。
  2. **控制总线 (control line)**：计算机通过控制总线指示哪个设备允许使用总线，以及使用总线的目的。

控制总线也传递有关总线请求、终端和时钟同步信号的相应信号。

### 3. 地址总线 (address line): 指出数据读写的位置

- 由于总线传递的信息类型不同, 使用设备不同, 因此总线还能细分称不同种类。
  1. **处理器 - 内存总线** 是长度较短的高速总线, 这种总线连接处理器和与机器匹配的内存系统, 并最大限度地扩充数据传递的带宽。
  2. **I/O 总线** 通常比处理器 - 内存总线长, 可以连接带宽不同的各种设备, 兼容多种不同的体系结构。
  3. **主板总线** 可将主板上的所有部件连接在一起, 让所有设备共享一条总线。
- 各种信息的传递都发生在一个 **总线周期 (bus cycle)** 中, 总线周期是完成总线信息传送所需的时钟脉冲间的时间间隔。
  - **同步总线 (synchronous)**: 由时钟控制, 各种事件只有在时钟脉冲到来时才会发生, 也就是事件发生的顺序由时钟脉冲来控制。
  - **异步总线 (asynchronous)**: 各种控制线都是使用异步总线来负责协调计算机的各种操作, 采用较为复杂的 **握手协议 (handshaking protocol)** 来强制实现与计算机其他操作的同步。

## Clock - 时钟

- 每台计算机中都有一个内部时钟, 用来控制指令的执行速度; 时钟也用来对系统中的各个部件的操作进行协调和同步。
- CPU 的每条指令执行都是使用固定的时钟脉冲数目。因此计算机采用 **时钟周期 (clock cycle)** 数目来量度系统指令的性能。

## I/O 输入/输出子系统

- 输入 / 输出是指各种外围设备和主存储器 (内存) 之间的数据交换。通过输入设备可以将数据输入计算机系统; 而输出设备则从计算机中获取信息。
- 输入输出设备通常不直接与 CPU 相连, 而是采用某种 **接口 (interface)** 来处理数据交换。接口负责系统总线和各外围设备之间的信号转换, 将信号变成总线和外设都可以接受的形式。
- CPU 通过输入输出寄存器与外设进行交流, 这种数据的交换通常有两种工作方式。
  1. 在 **内存交换的输入输出 (memory-mapped I/O)** 方式中, 接口中的寄存器地址就在内存地址的分配表中。在这种方式下, CPU 对 I/O 设备的访问和 CPU 对内存的访问是完全相同的。这种转换方式的速度很快, 却需要占用系统的内存空间。
  2. 在 **指令实现的输入输出 (instruction based I/O)** 方式中, CPU 需要有专用的指令来实现输入和输出操作。尽管不占用内存空间, 但需要一些特殊的 I/O 指令, 也就是只有那些可以执行特殊指令的 CPU 才可以使用这种输入输出方式。

## 存储器组成和寻址方式

- 存储器的地址几乎都是采用无符号整数来表示。正常情况下, 存储器采用的是 **按字节编址 (byte-addressable)** 的方式, 也就是说每个字节都有一个唯一的地址。计算机的存储器也可以采用 **按字编址 (word-addressable)** 的方式, 即每个机器字都具有一个自己的唯一地址。
- 如果计算机系统采用按字节编址的体系结构, 并且指令系统的结构字大于一个字节, 就会出现 **字节对齐 (alignment)** 的问题。在这种情况下, 访问所查询的地址时无需从某个自然对齐的边界线开始。
- 存储器由随机访问存储器 (**RAM**) 芯片构成。存储器通常使用符号 **L × W (长度 × 宽度)** 来表示。长度表示字的个数, 宽度表示字的大小。
- 主存储器通常使用的 **RAM** 芯片数目大于 1, 通常利用多块芯片来构成一个满足一定大小要求的单一存储器模块。
  - 单一共享存储器模块可能会引起存储器访问上的顺序问题。 **存储器交叉存储技术 (memory interleaving)**, 即从多个存储器模块中分离存储器单元的技术。

## Interrupt - 中断

- **中断**：反应负责处理计算机中各个部件与处理器之间相互作用的概念。中断就是改变系统正常执行流程的各种事件。多种原因可以触发中断：
  1. I/O 请求
  2. 出现算术错误 (除以 0)
  3. 算数下溢或上溢
  4. 硬件故障
  5. 用户定义的中断点 (程序调试)
  6. 页面错误
  7. 非法指令 (通常由于指针问题引起)
  8. ...
- 执行各种中断的操作称为中断处理，不同中断类型的中断处理方法各有不同。但是这些操作都是由中断来操作的，因为这些过程都需要改变程序执行的正常流程。
- 由用户或系统发出的中断请求可以是 **屏蔽中断 (maskable)** (可以被禁止或忽略) 或 **非屏蔽中断 (nonmaskable)** (高优先级别的中断，不能被禁止，必须响应)。
- 中断可以出现在指令中或指令之间，可以是同步中断，即与程序的执行同步出现；也可以是异步中断，即随意产生中断。中断处理可能会导致程序的终止执行，或者当中断处理完毕后程序会继续执行。

---

## MARIE

### 寄存器和总线

- CPU 中的 ALU 部件负责执行所有进程 (算数运算、逻辑判断等)。执行程序时，寄存器保存各种暂存的数值。在多种情况下，计算机对寄存器的引用都隐含在指令中。
- 在 MARIE 中，有 7 种寄存器；
  1. **AC**：累加器 (*accumulator*)，用来保存数据值。它是一个 **通用寄存器 (general purpose register)**，作用是保存 CPU 需要处理的数据。
  2. **MAR**：存储器地址寄存器 (*memory address register*)，用来保存被引用数据的存储器地址。
  3. **MBR**：存储器缓冲寄存器 (*memory buffer register*)，用来保存刚从寄存器中读取或者将要写入存储器的数据。
  4. **PC**：程序计数器 (*program counter*)，用来保存程序将要执行的下一条指令的地址。
  5. **IR**：指令寄存器 (*instruction register*)，用来保存将要执行的下一条指令。
  6. **InREG**：输入寄存器 (*input register*)，用来保存来自输入设备的数据。
  7. **OutREG**：输出寄存器 (*output register*)，用来保存要输出到输出设备的数据。
    - **MAR、MBR、PC 和 IR** 寄存器用来保存一些专用信息，并且不能用作上述规定之外的其他目的。另外，还有一个 **标志寄存器 (flag register)**，用来保存指示各种状态或条件的信息。
- **MARIE** 需要利用总线来将各种数据或指令传入和移出寄存器。在 **MARIE** 中，使用一条公共总线，每个设备都被连接到这条公用总线。
- 在 **MAR** 和存储器之间设计一条通信路线，**MAR** 通过这条电路为存储器的地址线提供输入，指示 CPU 要读写的存储器单元的位置。
- 从 **MBR** 到 **ALU** 还有一条特殊的通道，允许 **MBR** 中的数据也可以应用在各种算数运算中。同时，各种信息也可以从 **AC** 流经 **ALU** 再流回 **AC**，而不需要经过公用总线。
  - 采取这三条额外通道的好处是，在同一个时钟周期内，既可以将信息放置在公用总线，又可以同时将数据放置到这些额外的数据通道上。

## 指令系统体系结构

- 计算机的 **指令系统体系结构** (*instruction set architecture ISA*) 详细规定了计算机可以执行的每条指令及其格式。从本质上来说 **ISA** 是计算机软件和硬件之间的接口。
- 指令由 **操作码** 和 **地址** 构成。大多数 **ISA** 由处理数据、移动数据和控制程序的执行序列的指令组成。