1. 计算机系统简介

1.1. 计算机系统的组成

```
  组成计算机的实际装置
            ↕          相辅相成
计算机系统 = 计算机硬件 <-----------> 计算机软件
                                        ↕
                                  程序(指令的集合)+数据
```

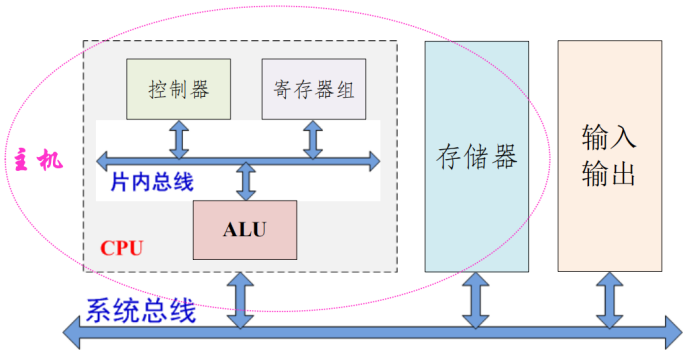
1.2. 从软件到硬件

```
            编译器            汇编器(优化)      链接器+动/静态库                    加载器
            ↕                 ↕                 ↕                  ↕
高级语言(.cpp)----->汇编语言(.s)----->机器语言(.o)---->链后机器语言(.exe/a.out)---->存储器
```

1.3. 存储程序计算机

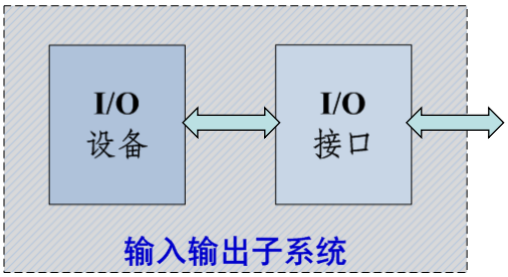
```
1.特点：指令和数据以同等地位存放于存储器内，并可按地址访问，用二进制表示
2.组成：运算器，控制器，存储器，I/O
3.指令：由操作码(例如10101001代表ADD)和地址码(寄存器码)组成
```

2. 计算机硬件的组成：总线结构

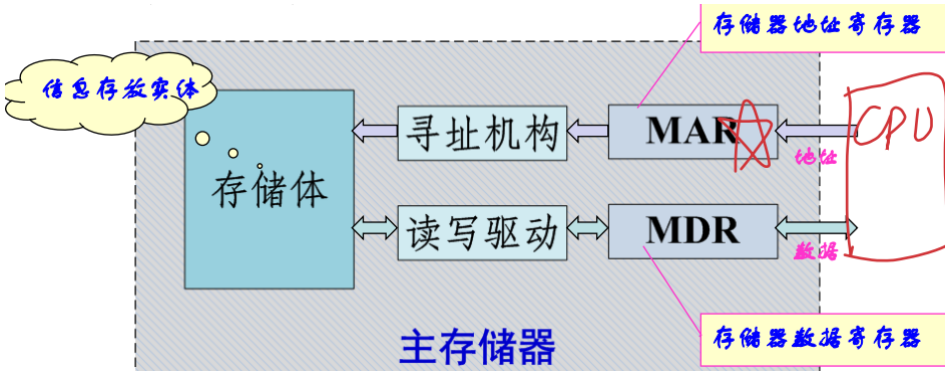


2.1. IO系统：IO设备+接口

完成人机交互的功能，当然也有完成机机交互的设备比如网卡



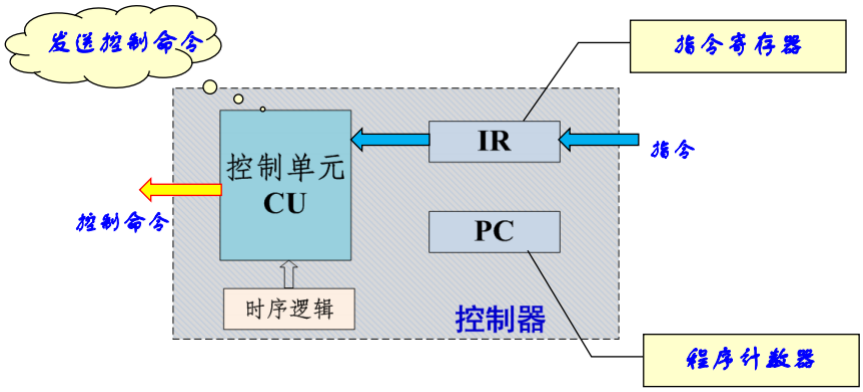
2.2. 主存储器：存放正在运行的程序和数据



1. MAR是存储器的地址寄存器，用来接收CPU传输的地址信号
2. MDR是存储器的数据寄存器，用来给CPU传输从存储体读出的数据
3. 存储体=一系列存储单元，存储单元=8个存储元(1位)

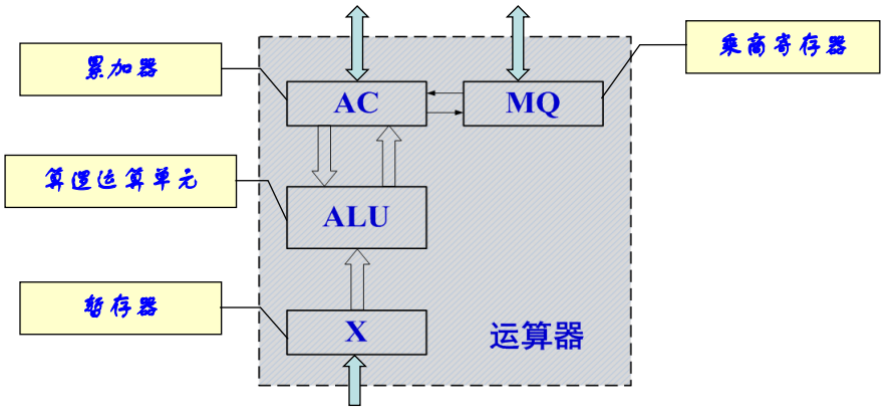
2.3. CPU中的两个模块

2.3.1. 控制器：指挥中心



1. PC：相当于8086中的IP指针，永远指向下一条要执行的指令
2. IR：存储当前被执行的指令
3. CU：解码指令，发出控制信号

2.3.2. 运算器：基于ALU的数据处理核心



1. AC：存储算术或逻辑操作的结果，直接参与加减计算
2. MQ：乘除操作时，一个操作数会存储在MQ中(另一个在AC中)
3. ALU：负责加减乘除算数与逻辑运算

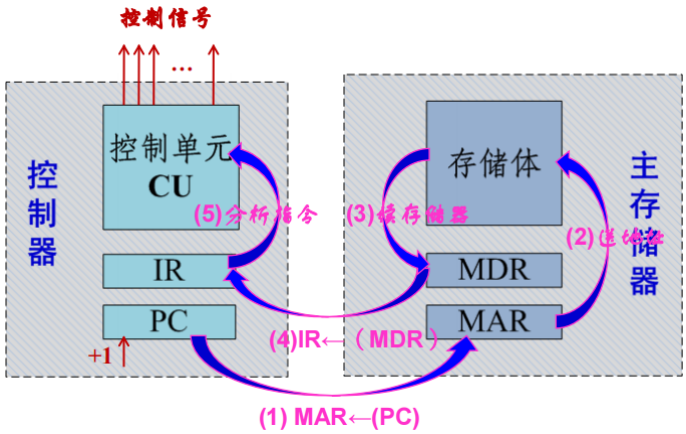
4 暂存器：高速缓存

2.4. 基于计算机硬件的指令执行过程

2.4.1. 执行指令的预备

程序已经为二进制(.exe/a.out)，可执行程序已经被加载，PC已经指向起始位置

2.4.2. 取指令&分析指令



1 PC所指向的指令的地址送MAR寄存器

2 MAR在存储体中找到指令的地址

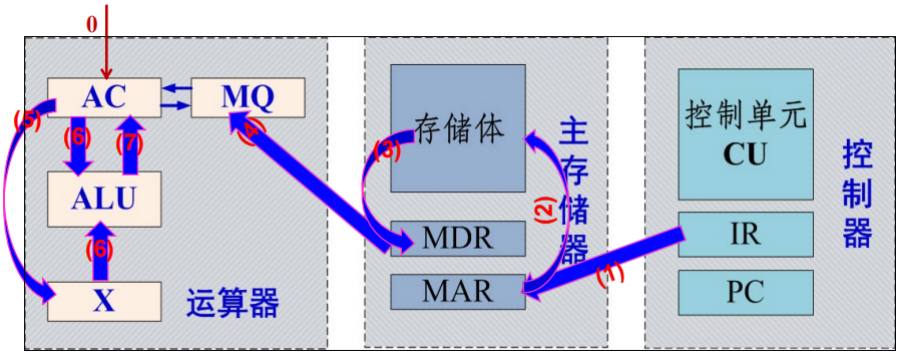
3 MDR读取该地址处对应的指令/数据

4 MDR把指令送IR，当前就开始执行该指令了

5 IR将指令给CU去分析然后生成控制信号，然后PC+1

2.4.3. 执行指令：以imul eax,dword ptr [m (00404010)]为例

```
1 imul eax,dword ptr [m (00404010)]
2 ; imul,带符号整数乘法,乘两个整数,将结果存储在指定的目标操作数中
3 ; dword ptr表示这是一个32位数据引用
4 ; [m(00404010)]表示一个内存地址，m为变量名，具体的地址是00404010
5 ; 执行操作：取eax的值和内存地址00404010中的32位值相乘，结果存放回eax中
```



假设前一条指令已经将EAX的值送AC寄存器(MDR可送AC也可送MQ)

1 将地址[00404010]从IR中取出来送给MAR

2 MAR把地址给存储体，存储体把m取出给MDR

3 把m送MQ，AC中EAX的值送X腾地方

4️⃣ 令AC=00000000，使得AC&MQ一起构成一个64位数

5️⃣ EAX和m在ALU完成运算，结果送AC，然后再送寄存器

3.计算机系统的性能(功能+质量)

3.1. 字长

机器字长：**CPU**同时处理的数据位数(=数据通路宽度/寄存器位数/ALU位数)

存储字长：一个存储单元可存放的二进制代码位数(按字/字节编址字长为一字/字节)

指令字长：一条指令所具有的二进制代码位数(= 字节的整数倍)

3.2. 计算速度

3.2.1. 吉普森法

$T_M = \sum f_i t_i$，程序运行时间=$\sum$第i种指令占全部指令百分比x第i种指令执行时间

3.2.2. 常用单位

MIPS：每秒执行百万条指令数

CPI：执行一条指令所需的时钟周期数

FLOPS：每秒浮点运算次数