同步进行学习计组，今天是第一章——引言。

一、计算机系统介绍

(1) 软硬件概念

计算机系统是由 软件 和 硬件 两大部分组成的。

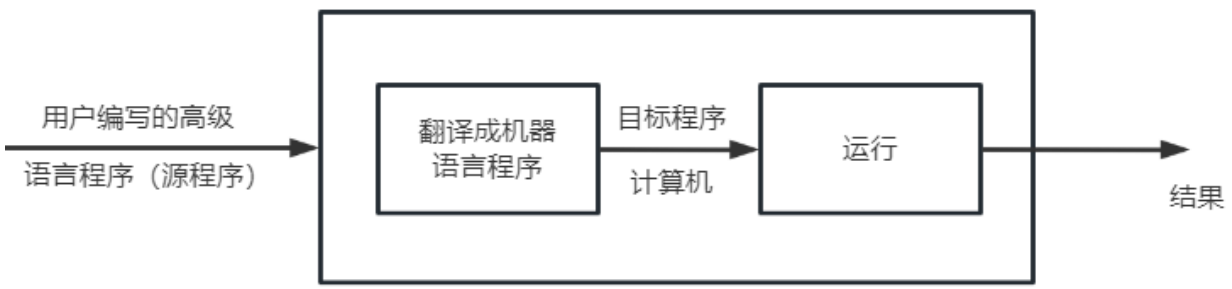
- 硬件：指计算机的实体部分，它由看得见摸得着的各种电子元器件、各类光、电、机设备的实物组成，如主机、外设等等。
- 软件：它是看不见摸不着的，由人们事先编制成具有各类特殊功能的信息组成。通常把这些信息，诸如各类程序寄寓于各类媒体中，如 RAM、ROM、磁带、磁盘、光盘等。计算机的软件通常又可分为两大类：
    - **系统软件：** 又称为系统程序，主要用来管理整个计算机系统，监视服务，使系统资源得到合理调度，确保高效运行。它包括：表中程序库、语言处理程序、操作系统、服务性程序、数据库管理系统、网络软件等等。
    - **应用软件：** 又称为应用程序，它是用户根据任务需要所编制的各种程序。如科学计算程序、数据处理程序、过程控制程序、事务管理程序等等。

硬件是计算机的物理基础，它决定了计算机系统的瓶颈在哪，而软件又决定了可以将硬件的性能发挥到什么样的程度，所以，计算机性能的好坏取决于“软”、“硬”件功能的总和。

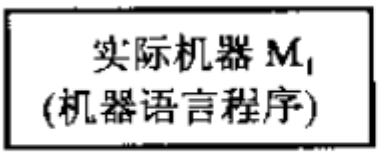
(2) 计算机系统层次结构

现代计算机的解题过程通常是先由用户用高级语言编写程序（称作源程序），然后将它和数据一起送入计算机内，再由计算机将其翻译成机器能识别的机器语言程序（称作目标程序），机器自动运行该机器语言程序，并将计算结果输出。

（OS：这学期正好开了编译原理，正好串起来了）

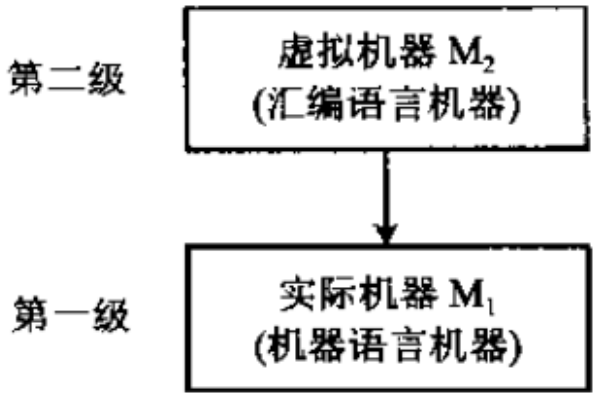


早期只有机器语言，用户编写的机器语言程序，直接在机器上执行，我们把直接执行机器语言的实际机器称 `M1` 。



机器语言程序可
直接在 M_1 上执行

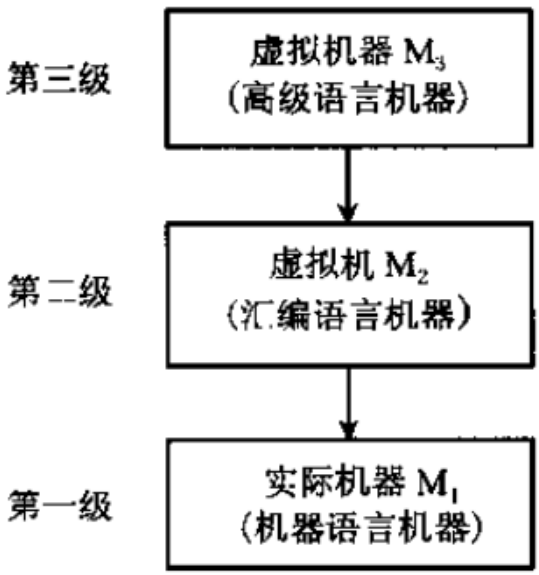
20 世纪 50 年代开始出现了符号式的程序设计语言，即汇编语言，它使程序员摆脱了繁杂而又易错的二进制代码编写程序，但是没有机器能直接识别这种汇编语言程序，必须先将汇编语言程序翻译成机器语言程序后，才能被机器接受并自动运行。我们把具有翻译功能的汇编程序的计算机看作一台`M2`机器。



汇编语言程序先翻译成机器语言程序,再由 M_1 执行

机器语言程序直接在 M_1 上执行

20 世纪 60 年代开始出现了各种面向问题的高级语言，如 FORTRAN、BASIC、Pascal、C 等等。这类高级语言对问题的描述十分接近人们的习惯，给程序员带来了极大的方便，当然，`M1`机器本身是不能识别高级语言的，因此，在进入`M1`机器运行前，必须先将高级语言程序翻译成汇编语言程序，然后再将其翻译成机器语言程序。我们把将高级语言程序直接翻译成汇编或者机器语言程序的机器称为`M3`。

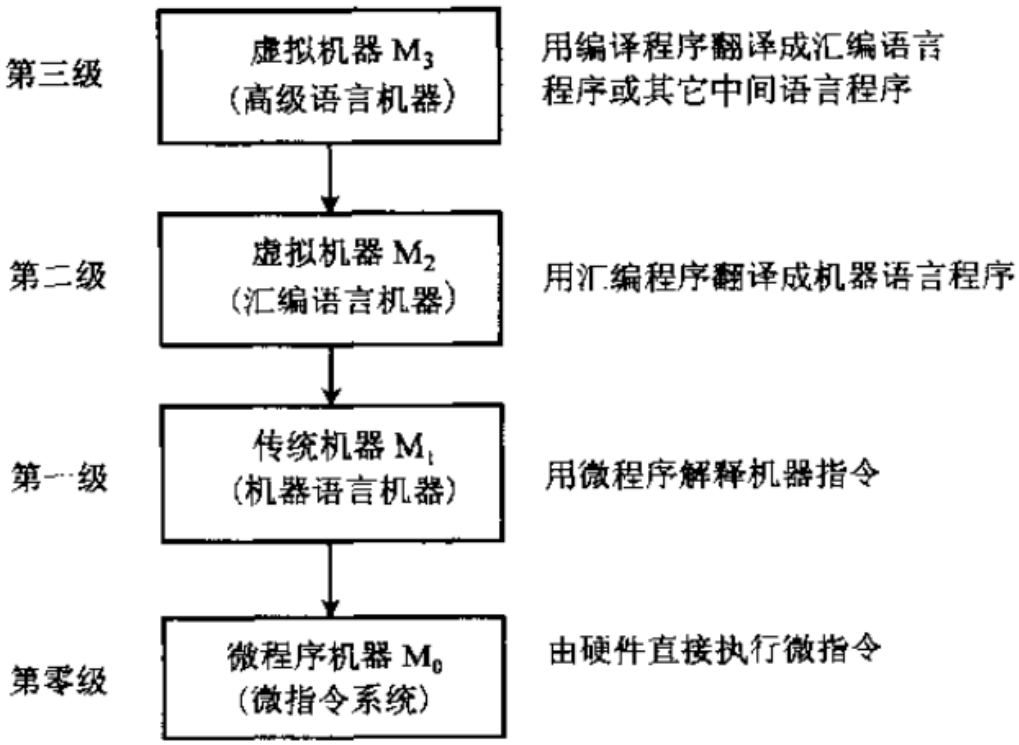


将高级语言程序先翻译成汇编语言程序，
再在 M_2、M_1（或直接到 M_1）上执行

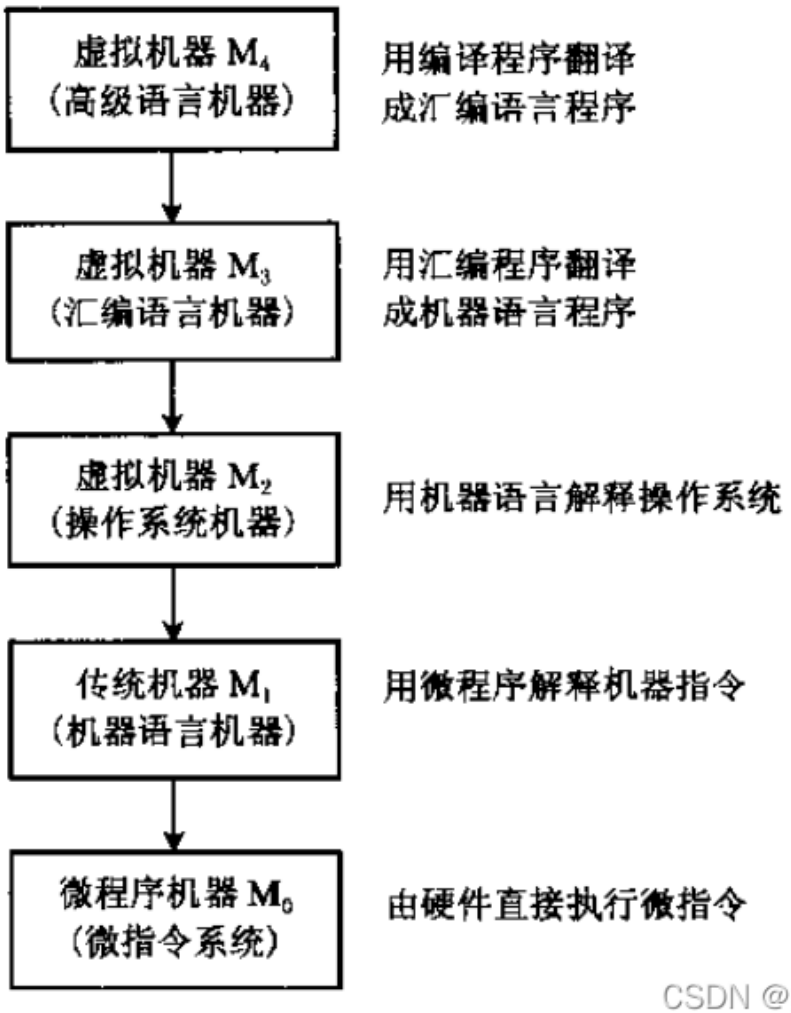
将汇编语言程序先翻译成机器
语言程序，然后在 M_1 上执行

机器语言程序直接在 M_1 上执行

由于软件的发展，使实际机器 M1 向上延伸构成了各级虚拟机器。同理 M1 机器内部也可向下延伸而形成下一级的微程序机器 M0 ， M0 机器是直接将 M1 机器中的每一条机器指令翻译成一组微指令，即构成一个微程序。微程序机器 M0 可看作是对实际机器 M1 的分解，即用 M0 的微程序解释并执行 M1 的每一条机器指令。



实际上再实际机器 M1 与虚拟机器 M2 之间，还有一级虚拟机器，它是由 操作系统 软件构成的。操作系统提供了在汇编语言和高级语言的使用和实现过程中所需的某些基本操作，还起到控制并管理系统硬件和软件全部资源的作用。

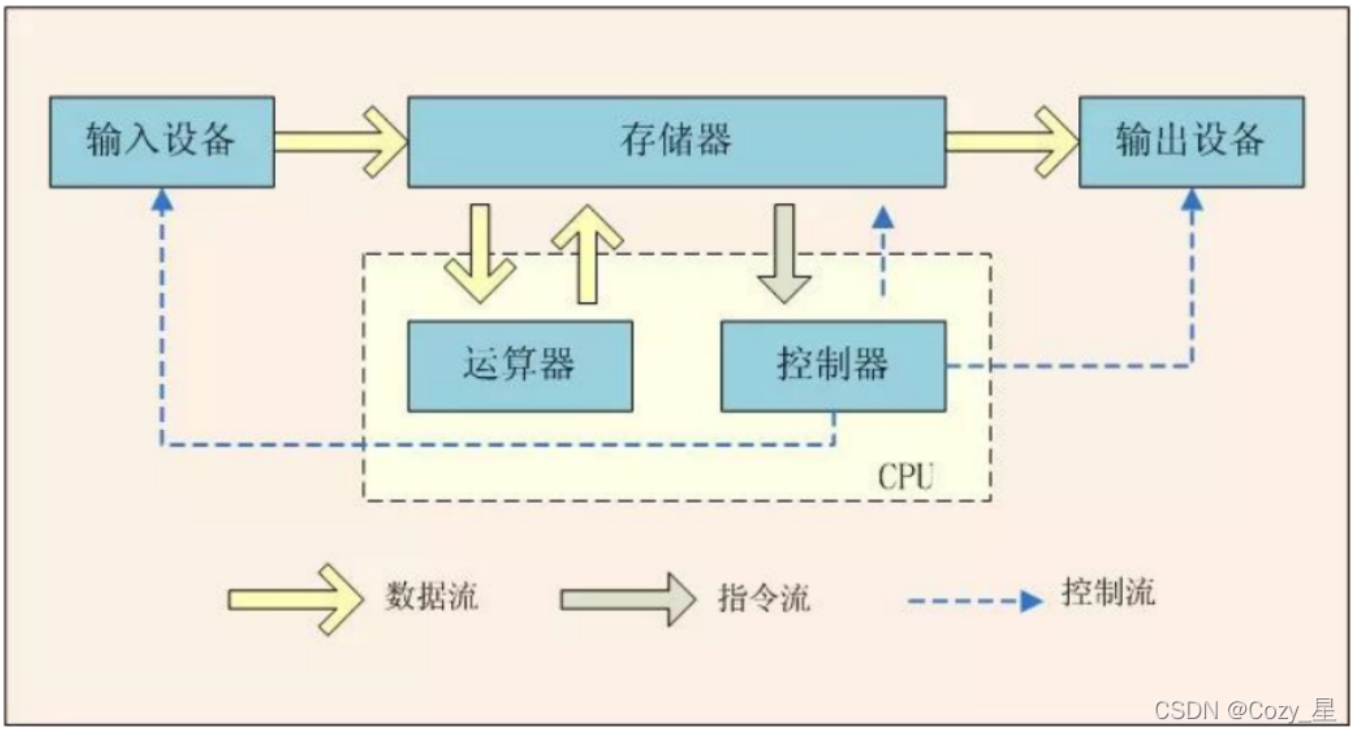


二、计算机的基本组成

(1) 冯 · 诺伊曼计算机的特点

1945 年数学家 冯·诺依曼 等人，在研究 EDVAC 机时，提出了 “存储程序” 的概念。以此概念为基础的各类计算机，统称为`冯 · 诺依曼机`。它的特点可归结为：

- 计算机由运算器、存储器、控制器和输入设备、输出设备**五大部件组成**
- **指令和数据以同等地位存放于存储器**内，并可按**地址寻访**
- 指令和数据均以二进制码表示
- **指令由操作码和地址码组成**，操作码用来表示操作的性质，地址码用来表示操作数所在存储器中的位置
- 存储程序，**指令在存储器内按顺序存放**
- 机器以**运算器**为中心，输入输出设备与存储器的数据传送通过运算器

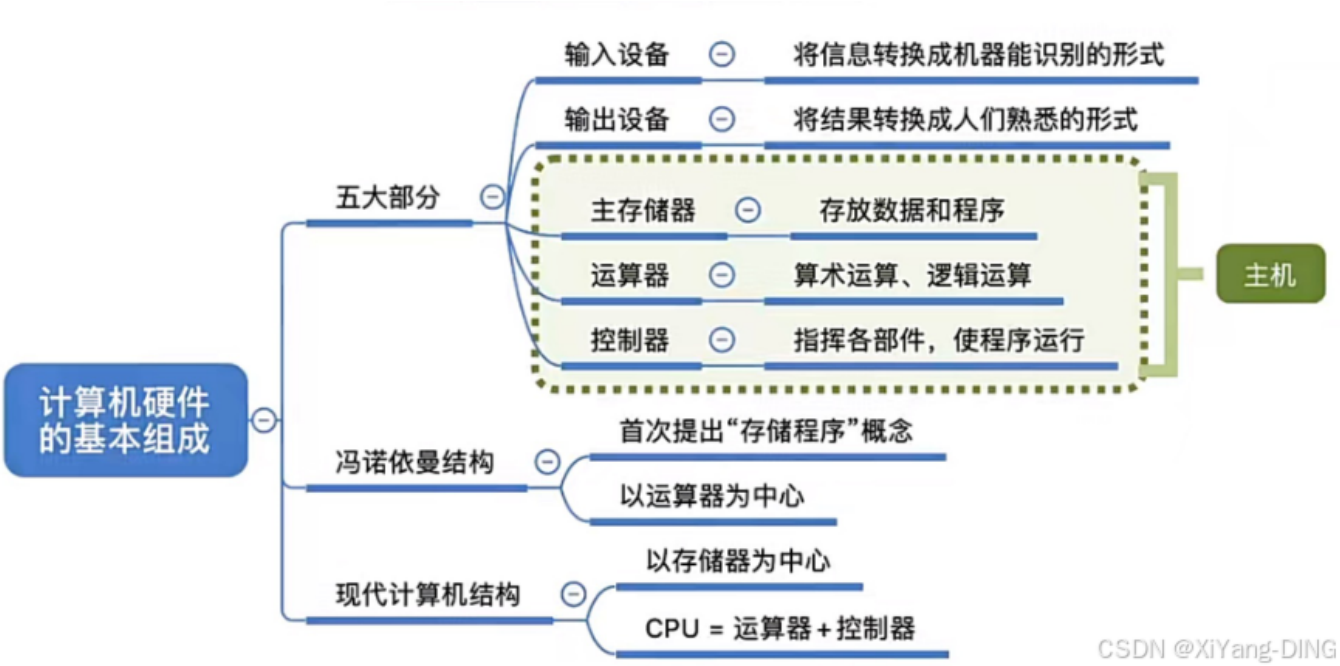


各组件的功能：

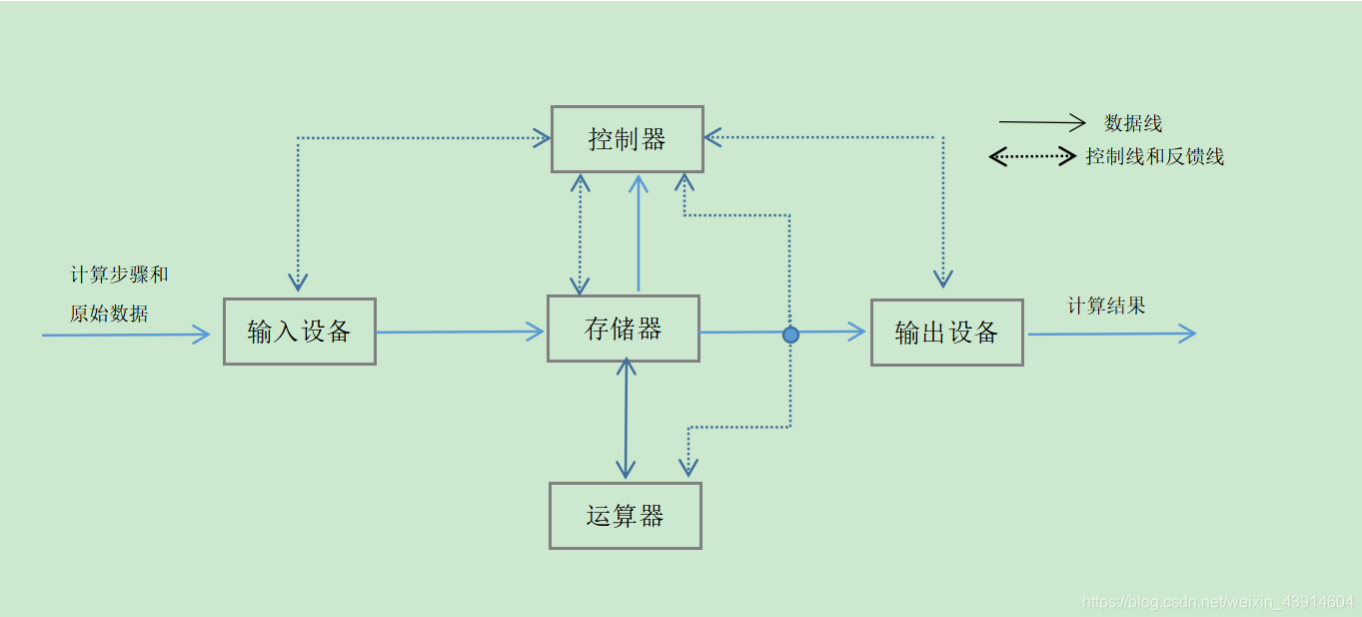
- 运算器：完成算术运算和逻辑运算，并将运算的中间结果暂存在运算器内
- 存储器：用来存放数据和程序
- 控制器：用来控制、指挥程序和数据的输入、运行以及处理运算结果
- 输入设备：将信息转换成机器能识别的形式，比如键盘、鼠标等
- 输出设备：将结果转换成人民熟悉的形式，比如打印机输出，显示器输出等

但也存在两个问题：

（1）以运算器为中心，导致运算器成为系统的瓶颈 （2）不具有层次化的特征

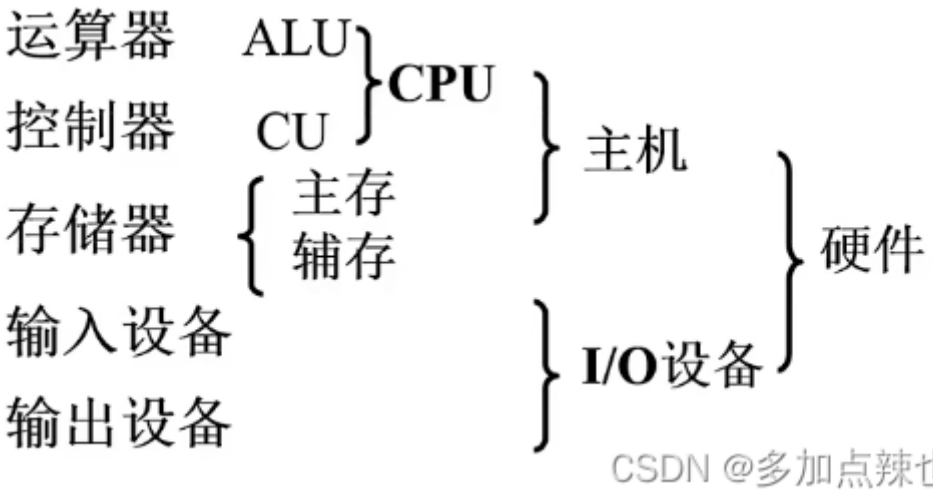


现代计算机硬件框图（以存储器为中心）：



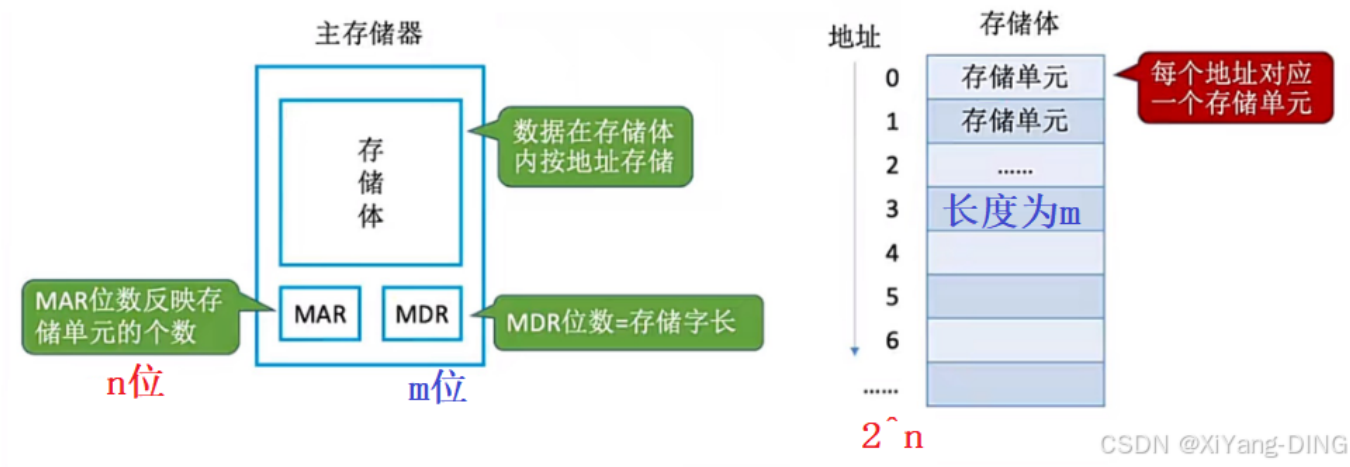
- 由于运算器和控制器在逻辑关系和电路结构的联系十分紧密，尤其在大规模集成电路制作工艺出现后，这两大部件往往制作在同一芯片上，因此，通常将它们合起来统称为中央处理器（Central Processing Unit），简称 CPU。
- 把输入设备与输出设备简称为 IO设备（Input/Output equipment）。I/O设备也受 CU 控制，用来完成相应的输入、输出操作。
- 主存储器M.M（Main Memory），简称主存或内存，用来存放程序和数据，它可以直接与 CPU 交换信息。

这样，现代计算机可认为由三大部分组成：CPU、IO设备及主存储器。CPU 与 M.M 合起来又可称为主机，IO设备 可叫作外部设备。除M.M的另一类叫辅助存储器，简称辅存，又叫外存。



（2）主存储器（M.M）

M.M包括存储体、各种逻辑部件及控制电路等。存储体由许多存储单元组成，每个存储单元又包含若干个存储元件（或称存储基元、存储元），每个存储元件能寄存一位二进制代码 “0” 或者 “1”。一个存储单元可存储一串二进制代码，称这串二进制代码为一个存储字，这串二进制代码的个数叫做存储字长。每个存储单元会被赋予一个编号，叫做存储单元的地址号。主存的工作方式就是按存储单元的地址号来实现对存储字各位的存、取。这种存取方式叫做按地址存取，也即按地址访问存储器（简称访存）。为了能实现按地址访问的方式，主存中还必须配置两个寄存器 `MAR` 和 `MDR` 。



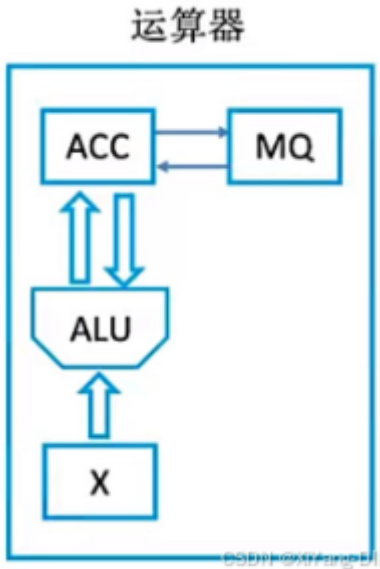
为了能实现按地址访问的方式，主存中还必须配置两个寄存器 `MAR` 和 `MDR` ：

- MAR（Memory Address Register）：存储器地址寄存器，**用于存放要访问的内存地址。** 其位数对应存储单元的个数（如 MAR 为 10 位，则有 210=1024 个存储单元，记为 1K）。

- MDR（Memory Data Register）：存储器数据寄存器，**用于存放从内存中读取或要写入内存的数据。** 其位数与存储字长相等。

（3）运算器

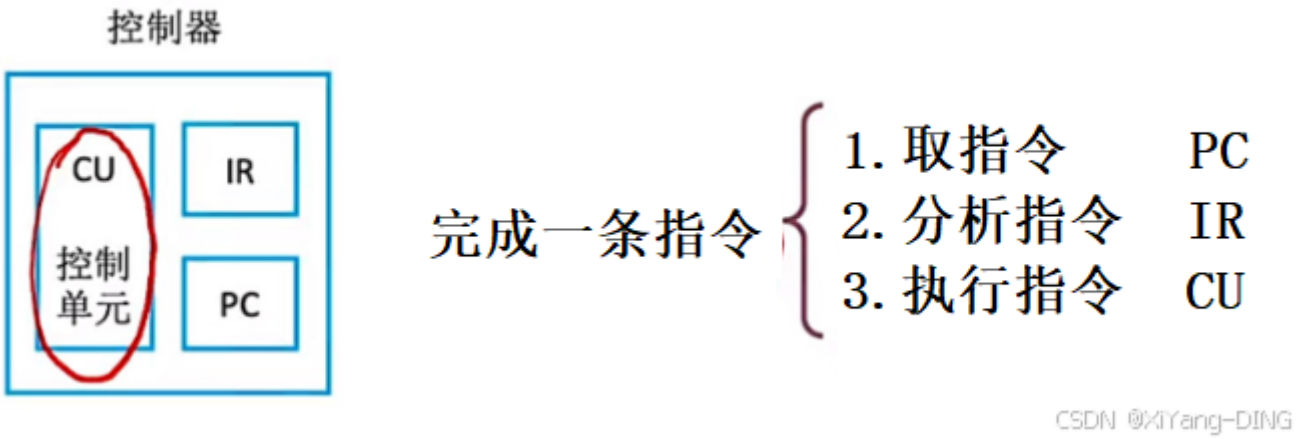
用于实现算术运算（如：加减乘除）、逻辑运算（如：与或非）。运算器包括三个寄存器和一个算逻电路 `ALU`。其中 `ACC (Accumulator)` 为 `累加器`， `MQ (Multiplier-QuotientRegister)` 为 `乘商寄存器`， `X` 为 `操作数寄存器`。这三个寄存器在完成不同运算时，所存放的操作数类别也各不相同。



- ACC:累加器，用于存放操作数，或运算结果。
- MQ:乘商寄存器，在乘、除运算时，用于存放操作数或运算结果。
- X:通用的操作数寄存器，用于存放操作数
- ALU:**算术逻辑单元，通过内部复杂的电路实现算数运算、逻辑运算。**

运算 / 操作数 / 寄存器	加法	减法	乘法	除法
ACC	被加数及和	被减数及差	乘积高位	被除数及余数
MQ			乘数及乘积低位	商
X	加数	减数	被乘数	除数

（4）控制器



- **CU：控制单元，分析指令，给出控制信号**
- IR：指令寄存器，存放当前执行的指令
- PC：程序计数器，有存放下一条指令地址，自动加1功能

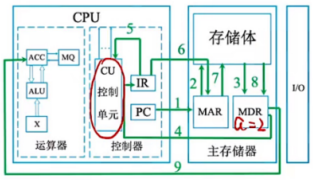
（4）计算机的工作过程

高级语言

```
int a=2,b=3,c=1,y=0;
void main(){
  y=a*b+c;
}
```

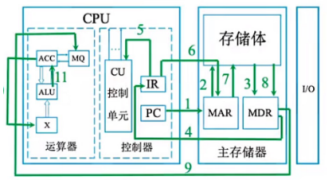
编译
装入主存

主存地址	指令 操作码	指令 地址码	注释
0	000001	0000000101	取数*a*至ACC
1	000100	0000000110	乘*b*得*ab*,存于ACC中
2	000011	0000000111	加*c*得*ab+c*,存于ACC中
3	000010	0000001000	将*ab+c*,存于主存单元
4	000110	0000000000	停机
5	0000000000000010		原始数据*a=2*
6	0000000000000011		原始数据*b=3*
7	0000000000000001		原始数据*c=1*
8	0000000000000000		原始数据*y=0*



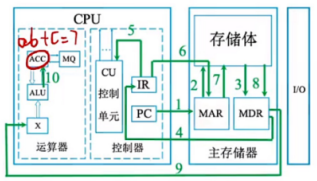
主存地址	指令 操作码	指令 地址码	注释
0	000001	0000000101	取数*a*至ACC
1	000100	0000000110	乘*b*得*ab*,存于ACC中
2	000011	0000000111	加*c*得*ab+c*,存于ACC中
3	000010	0000001000	将*ab+c*,存于主存单元
4	000110	0000000000	停机
5	0000000000000010		原始数据*a=2*
6	0000000000000011		原始数据*b=3*
7	0000000000000001		原始数据*c=1*
8	0000000000000000		原始数据*y=0*

(PC)=0
#1:(PC)→MAR,(MAR)=0
#3:M(MAR)→MDR,(MDR)=000001 0000000101
#4:(MDR)→IR,(IR)=000001 0000000101
#5:OP(IR)→CU,CU分析操作妈000001为取数指令
#6:AD(IR)→MAR,(MAR)=0000000101=5
#8:M(MAR)→MDR,(MDR)=2
#9:(MAR)→ACC,(ACC)=2



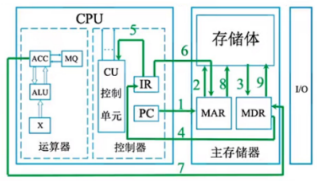
主存地址	指令 操作码	指令 地址码	注释
0	000001	0000000101	取数*a*至ACC
1	000100	0000000110	乘*b*得*ab*,存于ACC中
2	000011	0000000111	加*c*得*ab+c*,存于ACC中
3	000010	0000001000	将*ab+c*,存于主存单元
4	000110	0000000000	停机
5	0000000000000010		原始数据*a=2*
6	0000000000000011		原始数据*b=3*
7	0000000000000001		原始数据*c=1*
8	0000000000000000		原始数据*y=0*

(PC)=1
#1:(PC)→MAR,(MAR)=1
#3:M(MAR)→MDR,(MDR)=000100 0000000110
#4:(MDR)→IR,(IR)=000100 0000000110
#5:OP(IR)→CU,CU分析是乘法
#6:AD(IR)→MAR,(MAR)=0000000110=6
#8:M(MAR)→MDR,(MDR)=3
#9:(MDR)→MQ,(MQ)=3
#10:(ACC)→X,(X)=2
#11:(MQ)*(X)→ACC,(ACC)=6



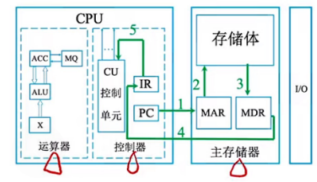
主存地址	指令 操作码	指令 地址码	注释
0	000001	0000000101	取数*a*至ACC
1	000100	0000000110	乘*b*得*ab*,存于ACC中
2	000011	0000000111	加*c*得*ab+c*,存于ACC中
3	000010	0000001000	将*ab+c*,存于主存单元
4	000110	0000000000	停机
5	0000000000000010		原始数据*a=2*
6	0000000000000011		原始数据*b=3*
7	0000000000000001		原始数据*c=1*
8	0000000000000000		原始数据*y=0*

(PC)=2
#1:(PC)→MAR,(MAR)=2
#3:M(MAR)→MDR,(MDR)=000011 0000000111
#4:(MDR)→IR,(IR)=000011 0000000111
#5:OP(IR)→CU,CU分析000011是加法
#6:AD(IR)→MAR,(MAR)=0000000111=7
#8:M(MAR)→MDR,(MDR)=1
#9:(MDR)→X,(X)=1
#10:(ACC)+(X)→ACC,(ACC)=7



主存地址	指令 操作码	指令 地址码	注释
0	000001	0000000101	取数*a*至ACC
1	000100	0000000110	乘*b*得*ab*,存于ACC中
2	000011	0000000111	加*c*得*ab+c*,存于ACC中
3	000010	0000001000	将*ab+c*,存于主存单元
4	000110	0000000000	停机
5	0000000000000010		原始数据*a=2*
6	0000000000000011		原始数据*b=3*
7	0000000000000001		原始数据*c=1*
8	0000000000000111		最终结果*y=7*

上一条指令取指后(PC)=3，执行后，(ACC)=7
#1：(PC)→MAR，导致(MAR)=3
#3：M(MAR)→MDR，导致(MDR)=000010 0000001000
#4：(MDR)→IR，导致(IR)= 000010 0000001000
#5：OP(IR)→CU，指令的操作码送到CU，CU分析后得知，这是“存数”指令
#6：Ad(IR)→MAR，指令的地址码送到MAR，导致(MAR)=8
#7：(ACC)→MDR，导致(MDR)=7
#9：(MDR)→地址为8的存储单元，导致y=7



主存地址	指令 操作码	指令 地址码	注释
0	000001	0000000101	取数*a*至ACC
1	000100	0000000110	乘*b*得*ab*,存于ACC中
2	000011	0000000111	加*c*得*ab+c*,存于ACC中
3	000010	0000001000	将*ab+c*,存于主存单元
4	000110	0000000000	停机
5	0000000000000010		原始数据*a=2*
6	0000000000000011		原始数据*b=3*
7	0000000000000001		原始数据*c=1*
8	0000000000000111		最终结果*y=7*

上一条指令取指后(PC)=4
#1：(PC)→MAR，导致(MAR)=3
#3：M(MAR)→MDR，导致(MDR)=000110 0000000000
#4：(MDR)→IR，导致(IR)= 000110 0000000000
#5：OP(IR)→CU，指令的操作码送到CU，CU分析后得知，这是“停机”指令
（利用中断机制通知操作系统终止该进程）

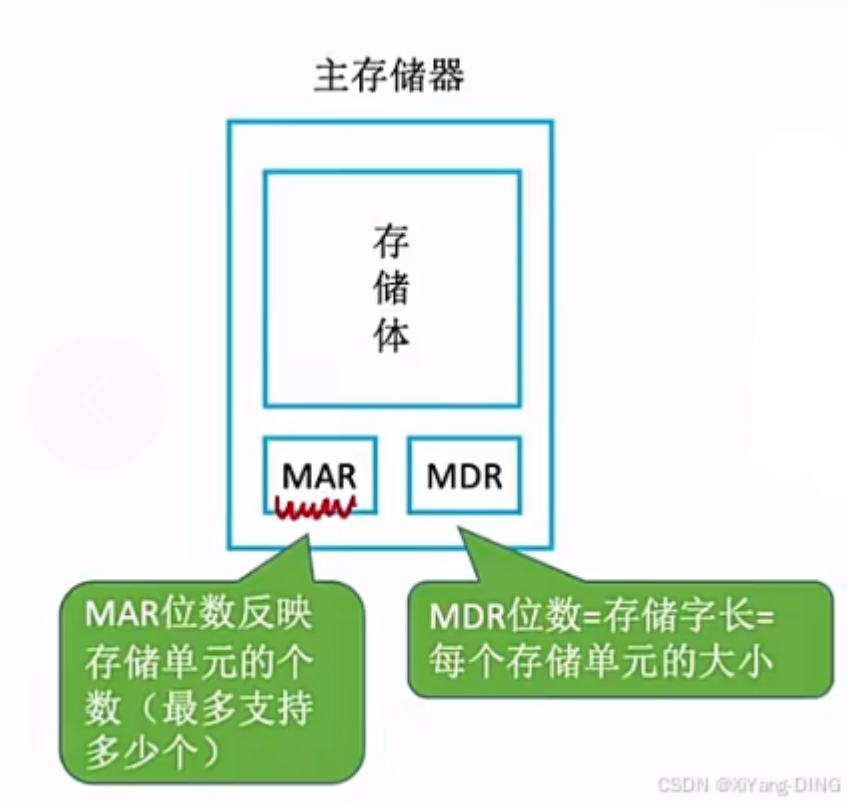
- M：表示**主存储器（Main Memory）**，也就是计算机的内存。
- M(MAR)：表示主存储器中地址为MAR的内容。

- M(MAR) → MDR：表示将主存储器中地址为MAR的数据读取到MDR中。
- OP(IR)：表示取操作码
- AD(IR)：表示去操作数

三、计算机性能

(1) 总容量

$$总容量=存储单元个数*存储字长bit=存储单元个数*存储字长/8Byte$$



e.g.：MAR为32位，MDR为8位 => 总容量= 2^(32)*8 bit = 4GB

(2) CPU性能

a. CPU主频：CPU内数字脉冲信号振荡的频率： $CPU主频（时钟频率）=1/CPU时钟周期$

b. CPI（Clock cycle Per Instruction)：执行一条指令所需的时钟周期数 c. 执行一条指令的耗时=CPI*CPU时钟周期 d. IPs（Instructions Per Second）：每秒执行多少条指令

（3）系统整体性能指标

- 数据通路带宽：数据总线一次所能并行传送信息的位数（各硬件部件通过数据总线传输数据）
- 吞吐量：指系统在单位时间内处理请求的数量
- 响应时间：指从用户向计算机发送一个请求，到系统对该请求做出响应并获得它所需要的结果的等待时间。
- 基准程序是用来测量计算机处理速度的一种实用程序，以便于被测量的计算机性能可以与运行相
- 同权子的其它计算机性能进行比较。（跑分软件）