# 寻址方式

寻址方式分为指令寻址和数据寻址。

**目录**



---

## 一、指令寻址

指令寻址分为顺序寻址和跳跃寻址。

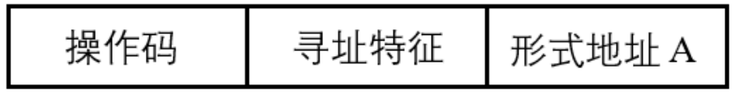
顺序寻址是通过程序计数器PC加1自动形成下一条指令的地址。

跳跃寻址是通过转移类指令实现。

### 二、数据寻址

首先需要知道的是，数据寻址的方式比较多，在指令字中必须设置一个字段来表明是哪种寻址方式。并且指令的地址字段通常都不表示操作数的有效地址，我们把它称为形式地址，记作 A ，有效地址记作 EA，由寻址方式和形式地址共同确定。

指令的格式通常为：




形式地址　　指令字中的地址

有效地址　　操作数的真实地址

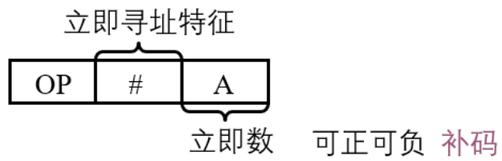
为了方便研究，假设机器字长、存储字长、指令字长都相同。

**1.立即寻址**

立即寻址的特点是操作数就在指令内，因此，也叫做立即数，采用补码的形式存放，在执行阶段不访问内存。

但是指令的位数是有限的，这也就限制了立即数的范围。

形式地址 A 就是操作数




可正可负 补码

- 指令执行阶段不访存
- A 的位数限制了立即数的范围

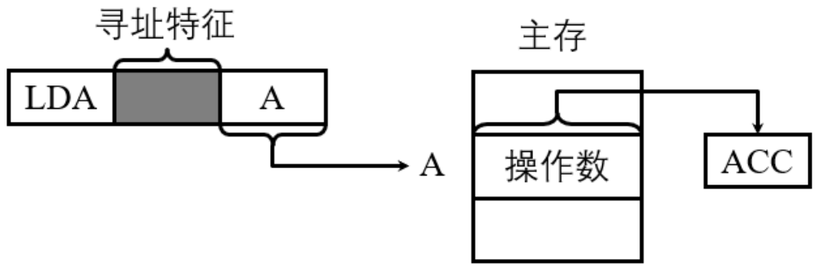


**2.直接寻址**

直接寻址的指令字中的形式地址A就是操作数的有效地址EA，即 EA=A；

他的优点就是，从指令中可以直接的得到操作数的有效地址，且只访问内存一次。缺点呢就是A的位数限制了操作数的寻址范围，而且，必须修改A的值才能修改A的地址。

EA = A　　有效地址由形式地址直接给出




- 执行阶段访问一次存储器
- A 的位数决定了该指令操作数的寻址范围
- 操作数的地址不易修改（必须修改A）

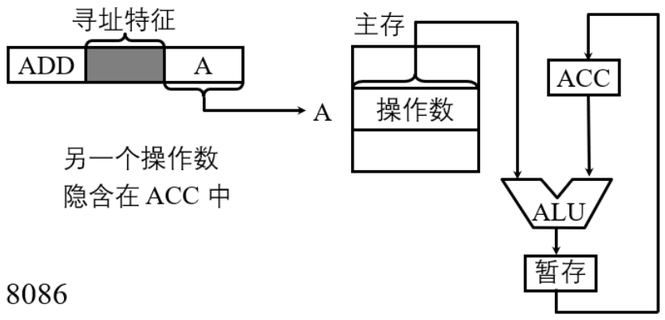


**3.隐含寻址**

隐含，就是不明显。隐含寻址就是不明显的给出操作数的地址，操作数的地址隐含于操作数或某个寄存器内，例如，一地址格式的加法指令只给出一个操作数的地址，另外一个操作数的就隐含在ACC累加器中，这样，累加器的地址就是操作数的地址。

这种隐含寻址的方式在指令字中就少了一个地址，有利于缩短指令字长。

操作数地址隐含在操作码中




如 8086

MUL 指令　被乘数隐含在 AX（16位）或 AL（8位）中

MOVS 指令　源操作数的地址隐含在 SI 中

目的操作数的地址隐含在 DI 中

- 指令字中少了一个地址字段，可缩短指令字长

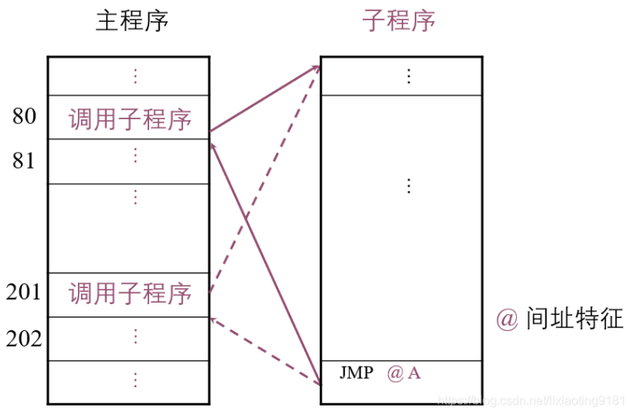


**4.间接寻址**

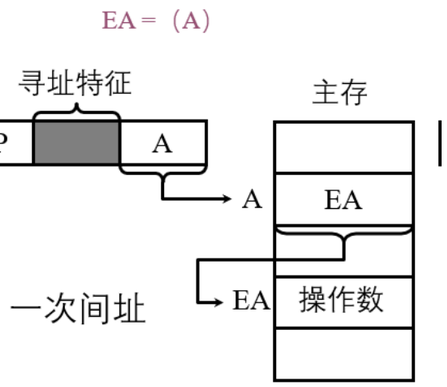
间接寻址的操作数的有效地址是由指令字中的形式地址给出的，也就是指令字中给出的是真实地址的地址，即EA=（A）；

间接寻址的优点有，扩大了操作数的寻址范围，便于编制程序。

例如：




图中表示在调用子程序前先将返回地址保存在子程序最末端指令形式地址A内便可以准确的返回断点处。




- 执行指令阶段 2 次访存
- 可扩大寻址范围
- 便于编制程序

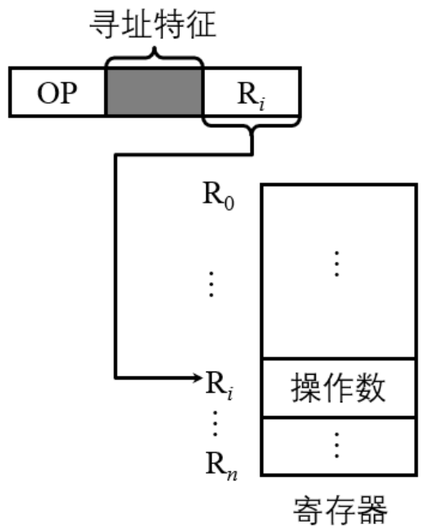
缺点就是，在执行阶段要访问内存两次或多次执行时间过长。

**5.寄存器（直接）寻址**

寄存器直接寻址指令字中的内容是寄存器的编号，操作数在指定编号的寄存器里。

由于寄存器直接寻址不访问内存，执行的速度加快，而且，只需指定寄存器的编号就可以了（计算机内寄存器数量有限），所需指令字较短，节省了储存空间。因此寄存器寻址在计算机中得到广泛应用。

$EA = R_i$　　有效地址即为寄存器编号




- 执行阶段不访存，只访问寄存器，执行速度快
- 寄存器个数有限，可缩短指令字长

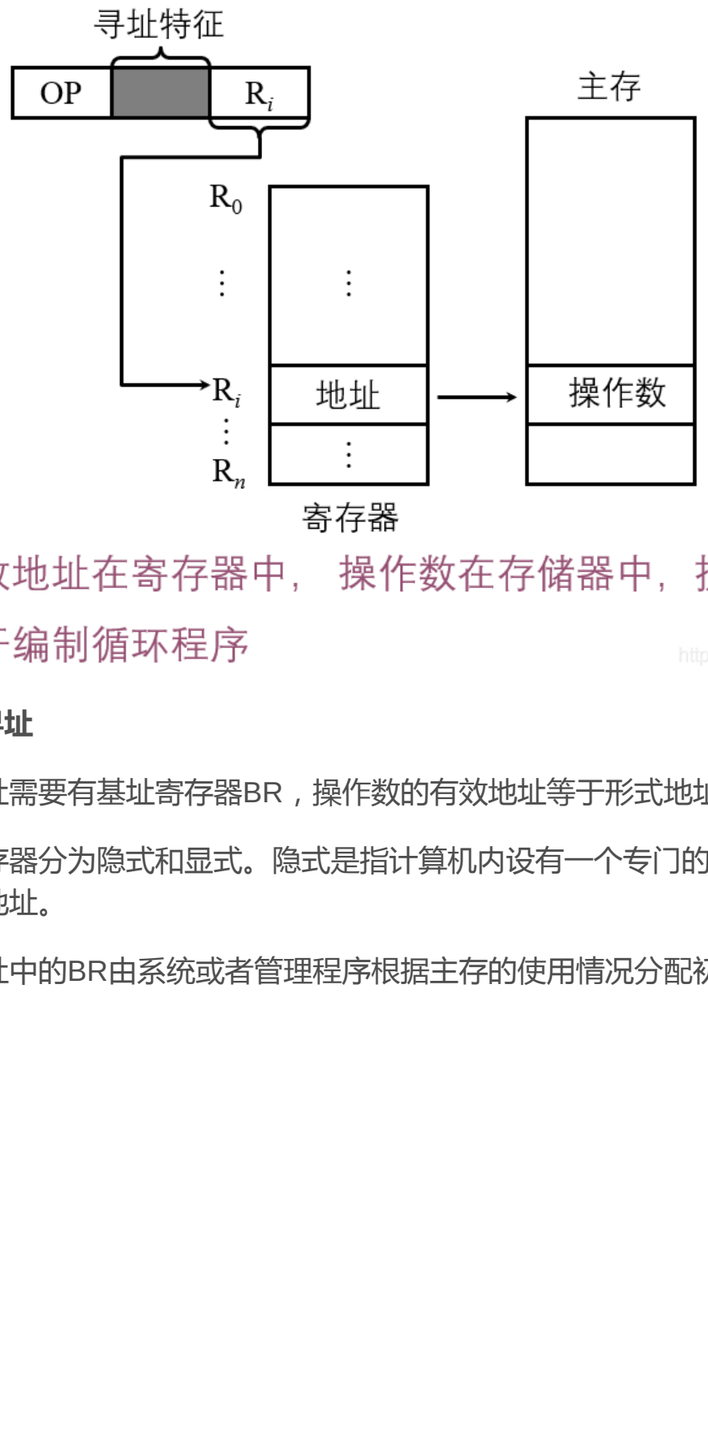


**6.寄存器间接寻址**

寄存器间接寻址也是指定寄存器编号，但是寄存器内存放的并不是操作数，而是操作数的有效地址。

有效地址在寄存器中，操作数在存储器中，执行阶段访存，但访存比间接寻址少一次，便于编制循环程序

EA = ( $R_i$ )　　有效地址在寄存器中




- 有效地址在寄存器中， 操作数在存储器中，执行阶段访存
- 便于编制循环程序



**7.基址寻址**

基址寻址需要有基址寄存器BR，操作数的有效地址等于形式地址加上基址寄存器中的内容（基地址）。

基址寄存器分为隐式和显式。隐式是指计算机内设有一个专门的，用户不必明显指出该基址寄存器，只需由寻址特征位反应基址寻址就可以。而显式是在一组通用寄存器内由用户指定一个基址寄存器存放基地址。

基址寻址中的BR由系统或者管理程序根据主存的使用情况分配初始值，便可将用户程序的逻辑地址转化为主存的物理地址（实际地址）。

**8.变址寻址**

变址寻址与基址寻址极为相似，有效地址等于指令字中的形式地址与变址寄存器IX的内容相加 。

基址寻址主要用于为程序或数据分配内存空间，故基址寄存器的内容通常由操作系统或管理程序确定，在程序的执行过程中其值是不可变的，而指令字中的形式地址A是可变的，在变址寻址中，变址寄存器的内容是用户设定的，在程序执行过程中其值可变，而指令字中的A是不可变的。

变址寻址主要用于处理数组问题，在数组处理过程中，可设定A为数组的首地址，不断改变变址寄存器的IX内的内容，便可以得到数组中任一数据中的地址，特别适合编制循环程序。

### 9、相对寻址

相对寻址的有效地址是将程序计数器PC内容与指令字中形式地址A相加而成。

如图可见，操作数的位置与当前指令的位置有一段距离A。

相对寻址最大的特点就是转移地址不固定，他随PC值的变化而变化，因此，无论主程序在哪段区域，都可正常运行，对编写浮动程序特别有利。

**10、堆栈寻址**

堆栈要求计算机中设有堆栈，可以用寄存器组，也可以主存的一部分空间来做堆栈。

栈只能从栈顶进行操作，堆栈也只能从一个口进行读写，操作数只能从栈顶地址指示的主存单元进行存或取。

SP始终指示着栈顶，所以不论是出栈还是进栈SP都需要发生变化。若栈底元素大于栈顶元素，则每次进栈SP=（SP）-△；每次出栈 SP=（SP）+△；

△的值与主存编址方式有关。若以字编址，△区1；以字节编址，则根据存储字长是几个字节才能确定△。