

P300 指令格式

指令是由操作码和地址码两部分组成的

- 操作码；用来指明该指令所要完成的操作
- 地址码；地址码用来指出该指令的源操作数的地址、结果的地址

操作码

操作码的长度可以是固定的，也可以是变化的

操作码长度不固定会增加指令译码和分析的难度，使控制器的设计更加复杂

通常采用扩展操作码技术，使操作码的长度随地址数的减少而增加，不同地址数的指令可以具有不同长度的操作码，从而在满足需要的前提下，有效地缩短指令字长

P302 例7.1

假设指令字长为16位，操作数的地址为6位，指令有零地址、一地址、二地址三种格式

- 设操作码固定，若零地址指令有P种、一地址指令有Q种，则二地址指令最多有几种？
- 采用扩展操作码技术，若二地址指令有X种，零地址指令有Y种，则一地址指令最多有几种？

P306 操作类型

数据传送

数据传送包括

- 寄存器与寄存器
- 寄存器与存储单元
- 存储单元与存储单元

算术逻辑操作

- 算术运算；（加、减、乘、除）
- 逻辑运算；（与、或、非）

移位

移位可分为

- 算术移位；对有符号数乘以 2^n （左移）或整除（ 2^n ）的运算
- 逻辑移位；对无符号数乘以 2^n （左移）或整除（ 2^n ）的运算
- 循环移位；将移出的低位放到该数的高位（循环右移）或把移出的高位放到该数的低位（循环左移）

转移

- 无条件转移; JMP X
- 条件转移; JZ (为0转移) ;JO (溢出转移) ;JC (进位转移) 等等
- 调用与返回; CALL (调用 子程序) 与RETURN (返回)
- 陷阱; Trap

输入输出

P310 寻址方式

指令寻址

顺序寻址

程序计数器加一, 自动形成下一条指令的地址

跳跃寻址

通过转移类指令实现

数据寻址

立即寻址

形式地址A不是操作数的地址, 而是操作数本身, 又称之为立即数;

假设指令字长=机器字长=存储字长, 操作数为3



立即寻址: 形式地址A就是操作数本身, 又称为立即数, 一般采用补码形式。
#表示立即寻址特征。

一条指令的执行:

取指令 访存1次

执行指令 访存0次

暂不考虑存结果

共访存1次

优点: 指令执行阶段不访问主存, 指令执行时间最短

缺点:

A的位数限制了立即数的范围。

如A的位数为n, 且立即数采用补码时, 可表示的数据范围为 $-2^{n-1} \sim 2^{n-1} - 1$

https://blog.csdn.net/qq_41587740

- 数据采用补码形式存放;
- 用#表示立即寻址特征
- 优点在于取出指令便可获取操作数, 不必访存;
- 缺点是显然形式地址A的位数限制了这类指令所能表述的立即数的范围

直接寻址

指令字种的形式地址A就是操作数的真实地址EA

EA = A

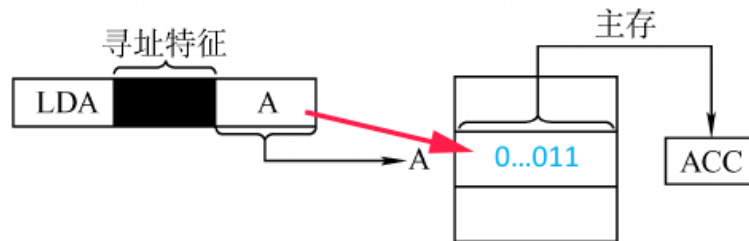
假设指令字长=机器字长=存储字长，操作数为3

一地址指令

操作码 (OP)

1001...0111

直接寻址：指令字中的形式地址A就是操作数的真实地址EA，即EA=A。



一条指令的执行：

取指令 访存1次

执行指令 访存1次

暂不考虑存结果

共访存2次

优点：简单，指令执行阶段仅访问一次主存，不需专门计算操作数的地址。

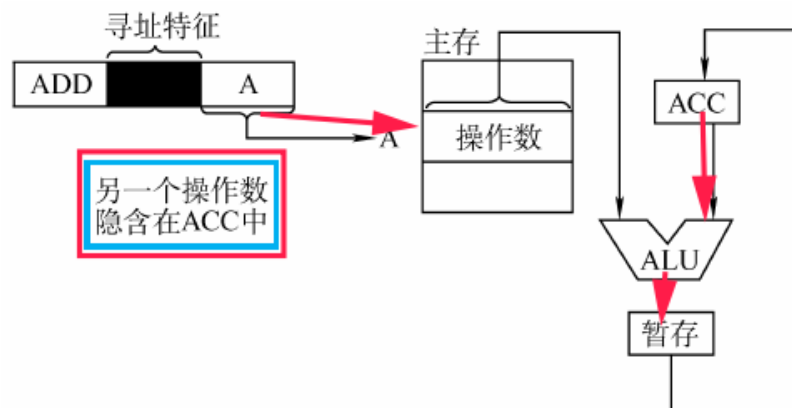
缺点：
A的位数决定了该指令操作数的寻址范围。
操作数的地址不易修改。

https://blog.csdn.net/qq_41587740

隐含寻址

不是明显地给出操作数的地址，而是在指令中隐含着操作数地址

隐含寻址：不是明显地给出操作数的地址，而是在指令中隐含着操作数的地址。



优点：有利于缩短指令字长。

缺点：需增加存储操作数或隐含地址的硬件。

https://blog.csdn.net/qq_41587740

间接寻址

指令中的地址字段给出的形式地址不是操作数的真正地址，而是操作数有效地址所再的存储单元的地址

EA = (A)

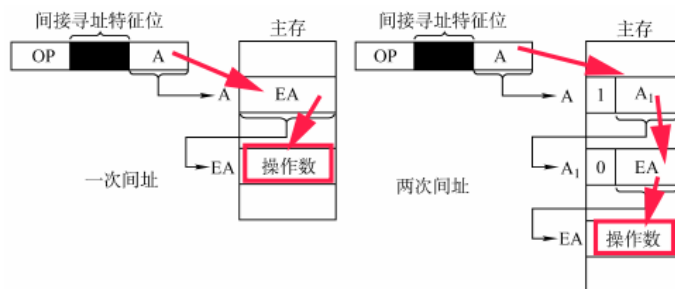
假设指令字长=机器字长=存储字长，操作数为3

一地址指令

操作码 (OP)

1001...0111

间接寻址：指令的地址字段给出的形式地址不是操作数的真正地址，而是操作数有效地址所在的存储单元的地址，也就是操作数地址的地址，即 $EA=(A)$ 。



优点：

可扩大寻址范围(有效地址EA的位数大于形式地址A的位数)。
便于编制程序(用间接寻址可以方便地完成子程序返回)。

缺点：

指令在执行阶段要多次访存(一次间址需两次访存，多次寻址需根据存储字的最高位确定几次访存)。

https://blog.csdn.net/qq_41587740

寄存器寻址

在指令字中直接给出操作数所在的寄存器编号，操作数在由 R_i 所指的寄存器内

$EA = R_i$

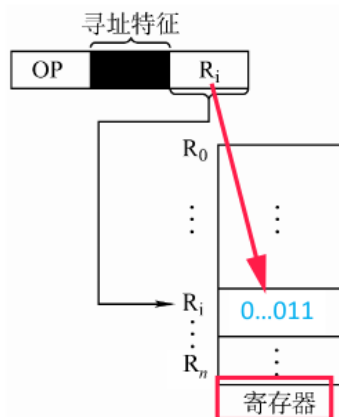
假设指令字长=机器字长=存储字长，操作数为3

一地址指令

操作码 (OP)

1001

寄存器寻址：在指令字中直接给出操作数所在的寄存器编号，即 $EA = R_i$ ，其操作数在由 R_i 所指的寄存器内。



一条指令的执行：

取指令 访存1次
执行指令 访存0次
暂不考虑存结果
共访存1次

优点：

指令在执行阶段不访问主存，只访问寄存器，指令字短且执行速度快，支持向量/矩阵运算。

缺点：

寄存器价格昂贵，计算机中寄存器个数有限。

https://blog.csdn.net/qq_41587740

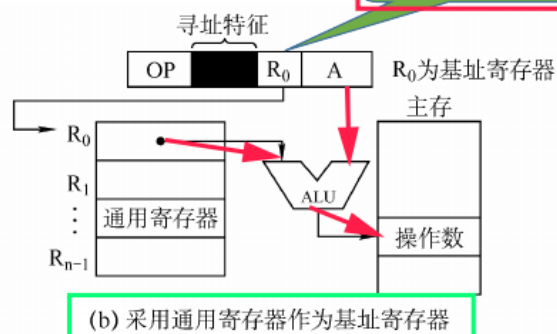
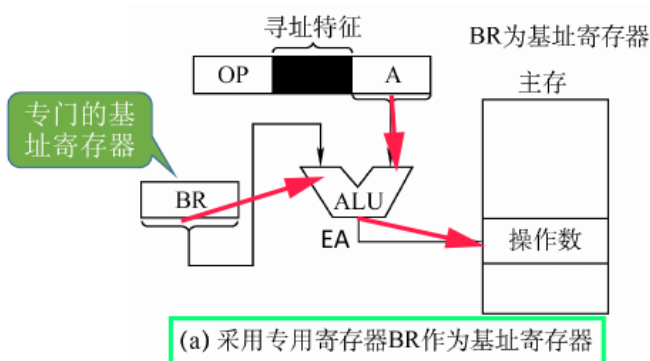
基址寻址

将CPU中基址寄存器 (BR) 的内容加上指令格式中的形式地址A，而形成操作数的有效地址

$EA = (BR) + A$

基址寻址：将CPU中**基址寄存器（BR）**的内容加上指令格式中的形式地址A，而形成操作数的有效地址，即 **$EA = (BR) + A$** 。

在指令中指明，
要将哪个通用
寄存器作为基
址寄存器使用



Tips: 可对比操作系统第三章第一节学习，OS课中的“重定位寄存器”就是“基址寄存器”

要用几个bit
指明寄存器？

根据通用寄存
器总数判断



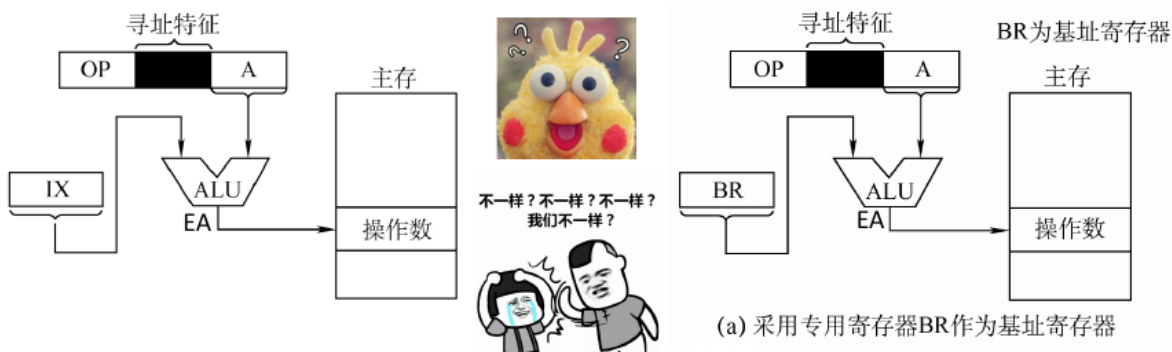
https://baidu.com/q_41587740

变址寻址

有效地址EA等于指令字中的形式地址与变址寄存器IX的内容相加之和

$$EA = (IX) + A$$

变址寻址：有效地址EA等于指令字中的形式地址A与**变址寄存器IX**的内容相加之和，即 **$EA = (IX) + A$** ，其中IX可为**变址寄存器（专用）**，也可用**通用寄存器作为变址寄存器**。



注：变址寄存器是**面向用户**的，在程序执行过程中，**变址寄存器的内容可由用户改变**（IX作为偏移量），形式地址A不变（作为基地址）。

基址寻址中，BR保持
不变作为基地址，
A作为偏移量

https://baidu.com/q_41587740

P318 例7.2

P319 例7.3

P323 例7.4

P323 例7.5

P323 例7.6

P323 例7.7
