第四章 指令系统

内容简介

- 指令系统的发展与性能要求
- 指令格式
- 操作数类型
- 指令和数据的寻址方式
- 典型指令

重点内容

- 基本概念

  指令系统、寻址方式、CISC、RISC

- 指令格式和寻址方式辨析

课后作业

P143：3、4、5、6

4.1 指令系统的发展与性能要求

- 指令系统的发展
- 对指令系统性能的要求
- 低级语言与硬件结构的关系

4.1.1 指令系统的发展

指令就是要计算机执行某种操作的命令，本章特指机器指令。一条机器指令可完成一个独立的算术逻辑运算操作。

微指令、机器指令、汇编指令、宏指令、高级语言语句。

若干条机器指令/汇编指令组成的软件指令称为宏指令。

4.1.1 指令系统的发展

一台计算机中所有机器指令的集合，称为这台计算机的指令系统。

指令系统是表征一台计算机性能的重要因素，其格式与功能直接影响机器的硬件结构、系统软件、适用范围等。

4.1.1 指令系统的发展

—— 系列机

系列机指基本指令系统相同、基本体系结构相同的一系列计算机。

兼容问题：同一系列的各机种有共同的基本指令集，新推出的机种指令系统一定包含所有旧机种的全部指令，以确保旧机种上运行的各种软件不加修改便可以在新机种上运行。

4.1.1 指令系统的发展

— CISC

随着超大规模集成电路技术VLSI的飞速发展，计算机的硬件结构越来越复杂、计算机的指令系统越来越复杂，这类计算机称为复杂指令系统计算机（简称CISC），这类指令系统称为复杂指令系统。这是计算机的发展方向之一。

4.1.1 指令系统的发展

— CISC

复杂指令系统计算机CISC的特征：

指令条数越来越多；

指令结构越来越复杂多样；

指令系统功能越来越细化复杂；

指令翻译执行效率越来越低；

研制周期越来越长；

大量使用频率很低的复杂指令浪费了系统硬件资源；

二八规律（最常使用的简单指令占指令总数的20%，但在程序中的使用频率却占80%）。

4.1.1 指令系统的发展

—— RISC

基于复杂指令系统的二八规律，选取使用频率较高的一些简单指令，构成精简指令系统。采用精简指令系统的计算机称为精简指令系统计算机（简称RISC），这是计算机的发展方向之一。

4.1.1 指令系统的发展

—— RISC

精简指令系统计算机RISC的特征：

指令条数较少；

指令类型和结构简单，尽量不采用复杂指令；

指令系统功能简单，复杂功能通过系统级例程实现；

指令翻译执行效率高；

研制周期较短，调试维护简单；

硬件结构相对简单、使用效率高。

4.1.2 对指令系统性能的要求

完备性：指令系统丰富、功能齐全、使用方便；

有效性：指令系统所编写的程序能够高效率运行；

规整性：指令系统的对称性、匀齐性、指令格式和数据格式的一致性，指令翻译效率高

兼容性：向上兼容，低档机上运行的软件可以直接在高档机上运行。

4.1.3 低级语言与硬件结构的关系

低级语言：机器语言、汇编语言，面向具体机器的硬件结构，与具体机器的指令系统密切相关；

高级语言：与计算机的硬件结构和指令系统无关。

4.2 指令格式

- 操作码
- 地址码（操作数）
- 指令字长度
- 指令助记符
- 指令格式举例

4.2 指令格式

指令格式通常由操作码字段和操作数字段（地址码字段）组成。操作码字段表征指令的操作特性与功能，即要求计算机完成什么运算操作。地址码字段 告诉计算机如何取得运算所需的操作数。

ADD AX,BX　　; (AX) + (BX) → (AX)

4.2.1 操作码

1、操作码字段的编码

不同指令的操作码字段用不同编码来表示，如001代表加法ADD、010代表减法SUB、110代表存数STR等。

CPU中控制器（有专门的译码电路）负责解释每个操作码的含义，从而产生相应的控制信号，指挥相关部件完成规定的运算操作。

典型指令	指令助记符	二进制操作码
加　法	ADD	001
减　法	SUB	010
传　送	MOV	011
跳　转	JMP	100
转　子	JSR	101
存　储	STR	110
读　数	LDA	111

4.2.1 操作码

2、操作码字段的位数

操作码字段的位数取决于计算机指令系统所定义的操作类型。

如指令系统有8种操作类型，则操作码字段3位；有32种操作类型，则操作码字段5位。一般地，一个包含n位的操作码最多能表示2^n种操作类型。

4.2.1 操作码

3、操作码字段的结构形式

定长操作码字段：长度固定，译码控制电路简单、执行效率高。

变长操作码字段：长度不固定，一般有多种类型的长度，表示的指令类型多样，但译码控制电路复杂、执行效率低。

4.2.2 地址码

1、地址码的分类

操作码 | A1 | A2 | A3　三操作数（地址码）指令，(A1) OP (A2) → (A3)

操作码 | A1 | A2　双操作数（地址码）指令，(A1) OP (A2) → (A1)

操作码 | A1　单操作数（地址码）指令

OP(A1) → (A1)；(AC) OP (A1) → (AC)

操作码　无操作数（地址码）指令

操作数隐含SCASB、SCMPW、CLD；系统操作指令HLT、WAIT

4.2.2 地址码

2、双操作数（地址码）指令

操作码	目的操作数	源操作数

寄存器-寄存器（RR）型指令。 源操作数、目的操作数、运算结果都在CPU的内部寄存器中，不需要访问内存；

寄存器-存储器（RS）型指令。 一个操作数在寄存器中、另一个操作数在内存单元中，需要访问内存；

存储器-存储器（SS）型指令。 源操作数、目的操作数都在内存单元中，这种指令的长度、执行时间较长，需多次访问内存，一般限制使用。

4.2.3 指令字长度

1、分类

指令字长度是指一条指令中包含的二进制代码的位数，包括操作码和操作数字段。

根据指令字长度与机器字长的关系，可划分为：半字长指令、单字长指令、双字长指令、N字长指令。

4.2.3 指令字长度

2、结构形式

等长指令字结构：各种指令字长度相同，结构形式简单，控制也简单；

变长指令字结构：各种指令字长度不相同的，结构形式灵活，控制复杂；

4.2.4 指令助记符

在汇编语言中，为了便于书写和阅读，指令通常用英文缩写字母来表示，这种英文缩写码称为指令助记符；

指令助记符表示每条指令的操作类型，容易记忆、书写、阅读。

典型指令	指令助记符	二进制操作码
加　法	ADD	001
减　法	SUB	010
传　送	MOV	011
跳　转	JMP	100
转　子	JSR	101
存　储	STR	110
读　数	LDA	111

4.2.5 指令格式举例

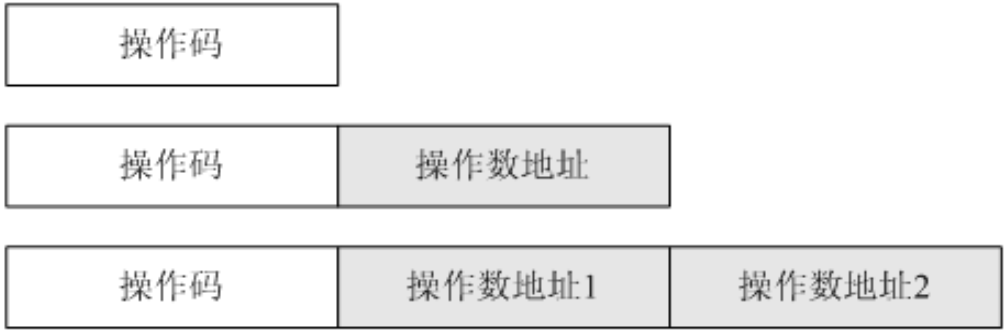
1、八位微型机指令格式

第1机器字长　第2机器字长　第3机器字长

单字长、无操作数指令

双字长、单操作数指令

三字长、双操作数指令



定长操作码字段、变长指令字结构

4.2.5 指令格式举例

2、Intel Pentium指令格式

（1）双操作数指令

操作码：“操作码”

操作数：“Reg或操作码=Reg”

操作数：由（Mod R/M 比例S 变址I 基址B 位移量 立即数）组合

（2）单操作数指令

操作码：“操作码”+“Reg或操作码=操作码”

操作数：由（Mod R/M 比例S 变址I 基址B 位移量 立即数）组合

4.2.5 指令格式举例

2、Intel Pentium指令格式

变长操作码字段

变长指令字结构

N字长指令字（无法确定N）

双操作数指令是RR或RS型指令

4.2.5 指令格式举例

[例] 指令格式如下所示，其中OP为操作码，试分析指令格式的特点。

[解]

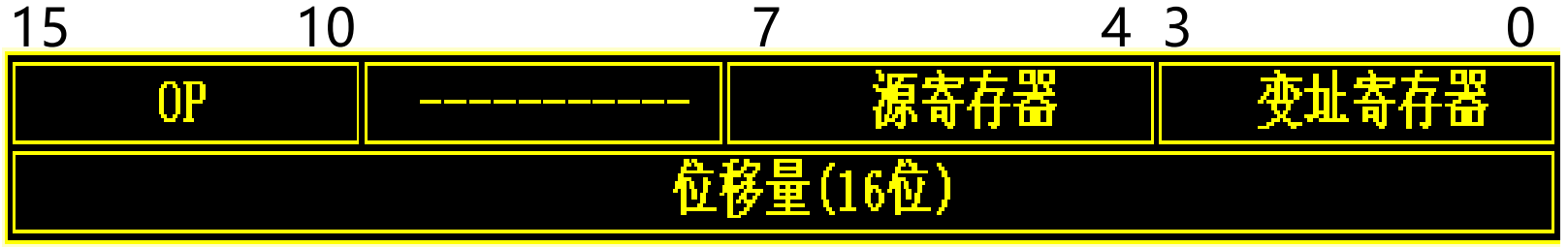
（1）操作码：操作码字段OP=7位，可以指定128类操作；定长操作码字段；

（2）地址码：双操作数指令；源寄存器、目标寄存器可分别指定16个；RR型指令；

（3）指令字长度：单字长指令(假设该机器单字长16bit)，等长指令字结构。

4.2.5 指令格式举例

[例] 指令格式如下所示，其中OP为操作码，试分析指令格式的特点。



[解]

（1）操作码：操作码字段OP=6位，可以指定64类操作；定长操作码字段；

（2）地址码：双操作数指令；源操作数由“源寄存器”指定16个寄存器之一，目的操作数由“变址寄存器（16个之一）+位移量（16位）”指定存储单元；RS型指令；

（3）指令字长度：双字长指令；等长指令字结构。

4.3 操作数类型

- 一般的数据类型
- Intel Pentium数据类型
- 操作数的来源

4.3.1 一般的数据类型

地址数据：指令中的操作数地址（如偏移地址、位移量），往往需要经过一定的变换，形成访问主存储器的物理地址。

数值数据：定点小数/定点整数、浮点数、压缩十进制数BCD码。

字符数据：字符、字符串、文字。广泛使用ASCII码。

逻辑数据：一个单元字中的每一个二进制位代表一种逻辑值（真、假），可按位进行布尔逻辑运算。

4.3.2 Pentium数据类型

数据类型	说明
常规	字节、字（16位）、双字（32位）、四字（64位）
整数	字节、字、双字、四字中的有符号二进制值、补码表示
序数	字节、字、双字、四字中的无符号整数
非压缩BCD码	每个字节表示一个十进制数位，可多字节
压缩BCD码	每个字节表示二个十进制数位，可多字节
近指针	表示段（块）内偏移的32位偏移地址，分段（块）存储器的段内访问
位串	一个连续的二进制位串，每位都是一个独立单位，表示逻辑值
字符串	一个连续的字节、字、双字的序列；ASCII码
浮点数	单精度32位、双精度64位、扩展双精度80位

4.3.3 操作数的来源

寄存器：操作数存放在CPU的某个内部寄存器中，编程时需要给出寄存器的符号名，汇编后转换为寄存器的编码。

存储单元：操作数存放在主存储器的某个单元中，编程时需要给出存储单元的逻辑地址，运行时转换为唯一的物理地址。

立即数：操作数随指令一起保存在程序代码中，即主存储器的某个单元中，通过程序计数器PC形成物理地址。

I/O端口： 操作数在某个I/O接口的I/O端口（接口中的寄存器）中，编程时需要给出I/O端口地址。

4.4 指令和数据的寻址方式

- 指令的寻址方式
- 操作数基本寻址方式
- 寻址方式举例

4.4.1 指令的寻址方式

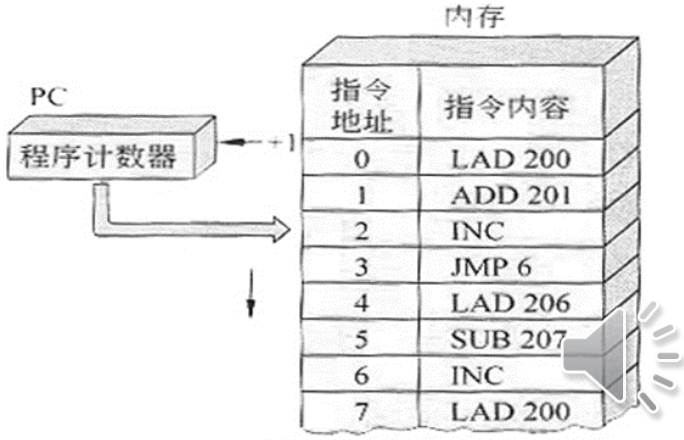
指令的寻址方式：告诉计算机如何获取指令，即如何提供将要取指、执行的指令所在存储单元的物理地址。

4.4.1 指令的寻址方式

1、顺序寻址方式

利用程序计数器PC自动“加1”的功能，控制程序的顺序执行。程序计数器PC实际上就是一个指针，每当一条指令取指阶段结束后，自动增量“加1”，指向下一条待取指令。

第2条取指令结束后（PC）=3



4.4.1 指令的寻址方式

2、跳跃寻址方式

程序跳转执行，如条件转移类指令、循环类指令、子程序调用与返回指令、中断指令。

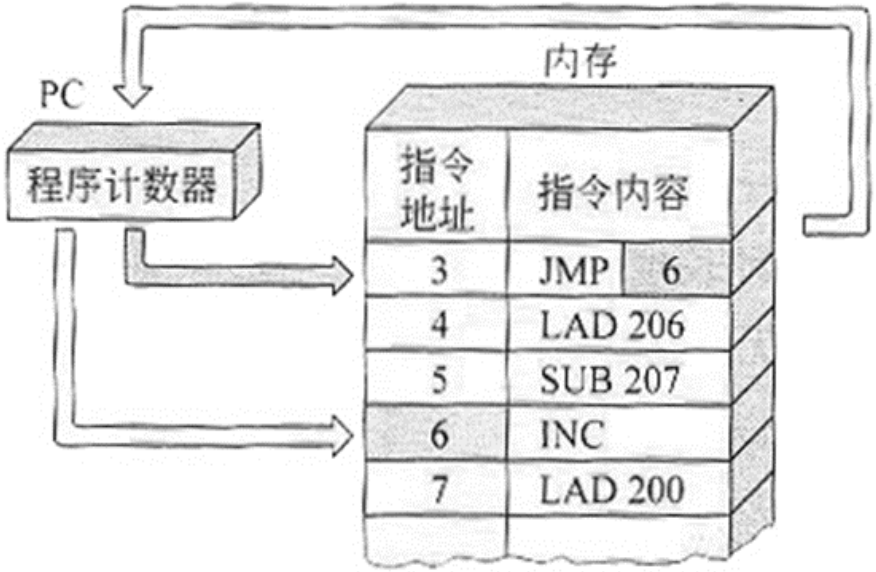
按指定的某种规则（直接寻址、间接寻址、查中断向量表等）生成新的转移地址，并送到程序计数器PC中，从而改变程序的执行顺序。

4.4.1 指令的寻址方式

2、跳跃寻址方式

第3条取指令结束后：（PC）=4

第3条指令执行结束后：（PC）=6



JMP 6：直接寻址方式，在指令中直接给出转移目标位置的地址。

4.4.2 操作数基本寻址方式

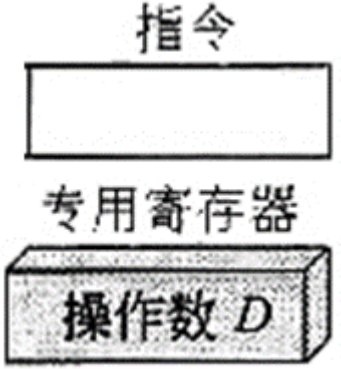
操作数的寻址方式：告诉计算机如何获取运算所需要的操作数，即如何提供运算所需要的操作数所在存储单元的物理地址、或者操作数所在内部寄存器、或者I/O端口的I/O地址。

4.4.2 操作数基本寻址方式

1、隐含寻址

在指令中没有显式的给出，而是指令设计时约定的。

一般是指某些指令中隐含的累加器、循环计数器、变址寄存器等专用寄存器。操作数在专用寄存器中。



LOOP　隐含循环计数器CX

SCAS　隐含串首地址在DI寄存器中、串的长度在CX寄存器中

4.4.2 操作数基本寻址方式

10、段寻址

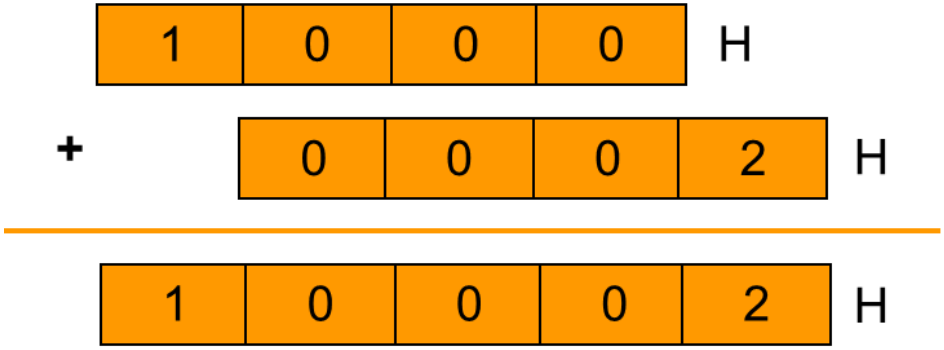
现代计算机的主存空间普遍采用分段技术。

8086CPU，主存地址空间1M，物理地址20位。而寄存器、机器字长、存储单元均典型采用16位，如何解决这个问题呢？采用分段技术，如可划分为16个64K的段，20位物理地址由16位段地址、16位段内地址组成。

地址码字段只给出偏移地址（逻辑地址、有效地址、形式地址、段内地址、位移量），段地址存放到CPU的段寄存器中。

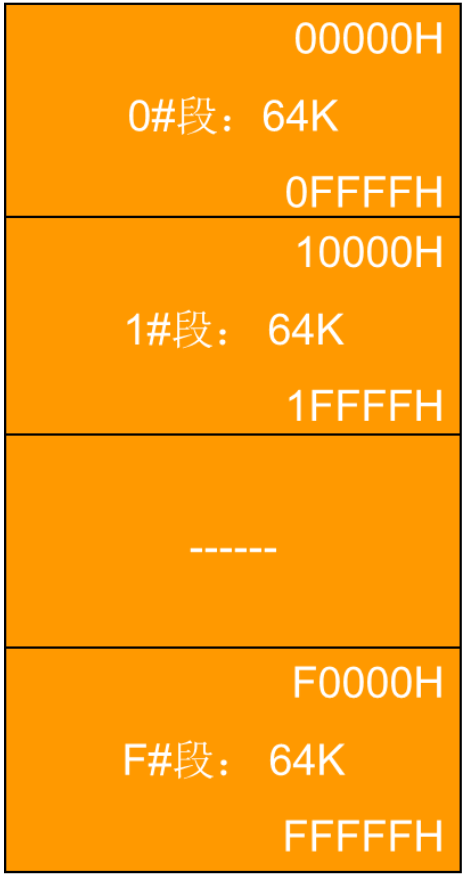
4.4.2 操作数基本寻址方式

10、段寻址



16d×段地址+偏移地址；

段地址、偏移地址均是16位的。



0#段地址：0000H

1#段地址：1000H

F#段地址：F000H

4.4.2 操作数基本寻址方式

10、段寻址

CS：存放当前代码段的段地址， 16d×（CS）+（PC）

DS：存放当前数据段的段地址（随机），16d×（DS）+EA

SS：存放当前堆栈段的短地址（顺序），16d×（SS）+（SP）

ES：存放当前附加段的段地址（随机） 16d×（ES）+EA

---程序和数据分离---

4.4.2 操作数基本寻址方式

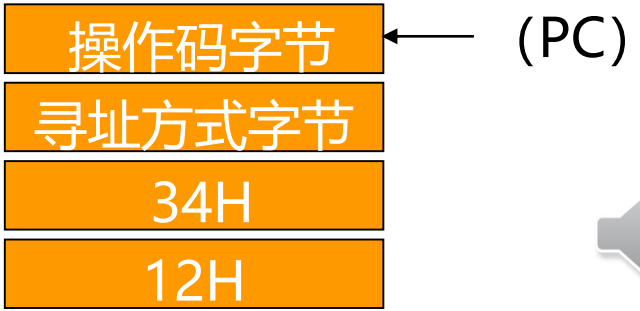
2、立即寻址

指令的地址码字段直接指出操作数本身，类似于高级语言中的常数。操作数在内存单元中。

“操作数D”本身是操作数，随指令操作码一起保存。

ADD AX,1234H

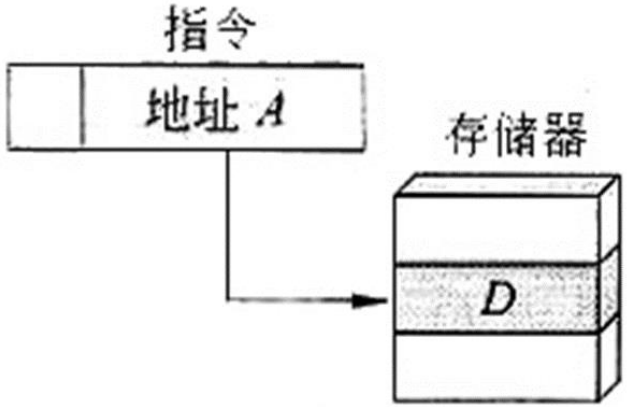
；16d ×（CS）+ （PC）

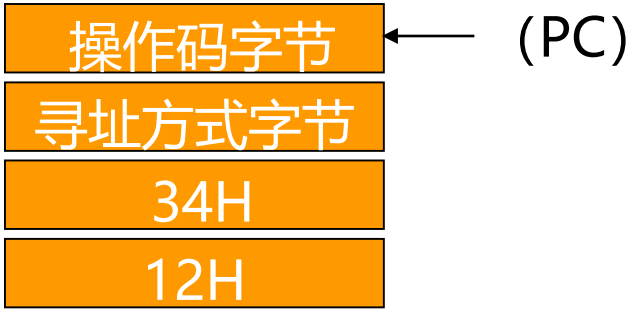


4.4.2 操作数基本寻址方式

3、直接寻址

指令地址码字段中，直接给出操作数所在内存单元的地址A，A往往经过变换形成物理地址。操作数在内存单元中。





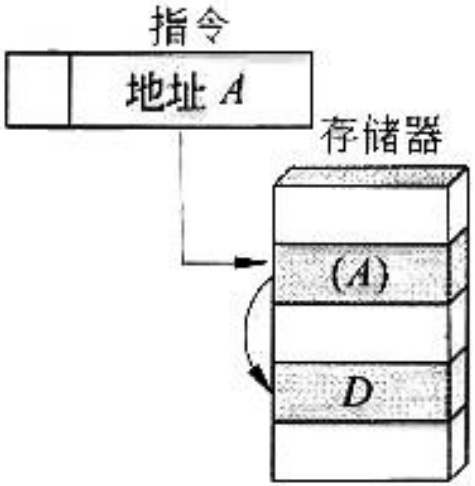
ADD AX，[1234H]

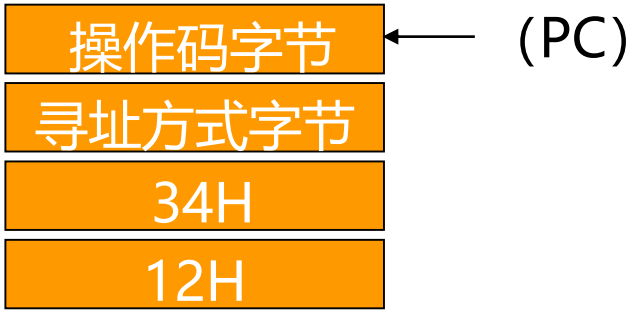
物理地址：16d×(DS)+1234H

4.4.2 操作数基本寻址方式

4、间接寻址

指令地址码字段中的A是操作数地址的地址，理论上可以多次间接寻址。操作数在内存单元中。





ADD AX，[[1234H]]

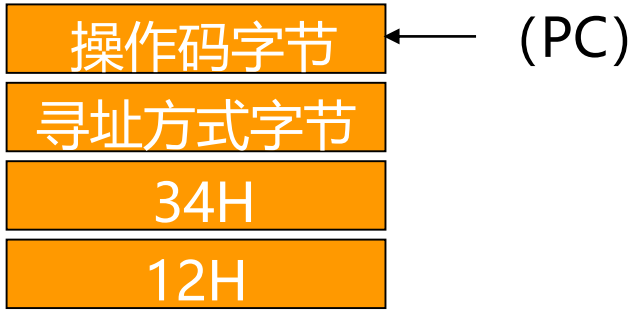
物理地址：16d×(DS) + (16d×(DS)+1234H)

4.4.2 操作数基本寻址方式

ADD AX,1234H

ADD AX,[1234H]

ADD AX，[[1234H]]



执行过程：

(1) 16d ×(CS)+(PC)，取操作码和寻址方式，(PC)+2→(PC)

(2) 16d ×(CS)+(PC)，取“1234H”

(3) 16d ×(DS)+1234H，取出X

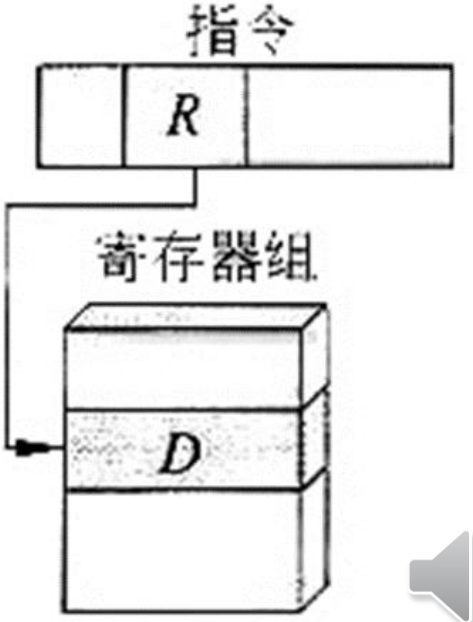
(4) 16d ×(DS)+X，取出操作数D

4.4.2 操作数基本寻址方式

5、寄存器寻址

地址码字段中直接给出寄存器的编号，汇编中采用符号名。

操作数在指定的寄存器中。



ADD AX，BX

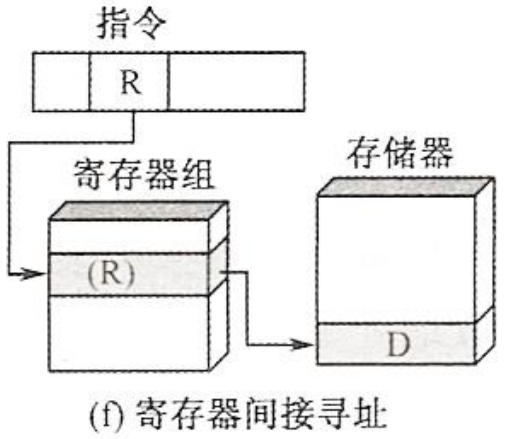
4.4.2 操作数基本寻址方式

6、寄存器间接寻址

在地址码字段中直接给出寄存器编号，该寄存器中的内容是操作数所在内存单元的逻辑地址，经过变换形成物理地址。操作数在内存单元中。

ADD AX，[BX]

物理地址：16d×(DS)+(BX)



(f) 寄存器间接寻址

4.4.2 操作数基本寻址方式

ADD AX,BX

ADD AX，[BX]

ADD AX,[[BX]]

执行过程：

（1）16d ×(CS)+(PC)，取操作码和寻址方式，(PC)+2→(PC)

（2） 根据指定的寄存器编号，取出BX寄存器的内容

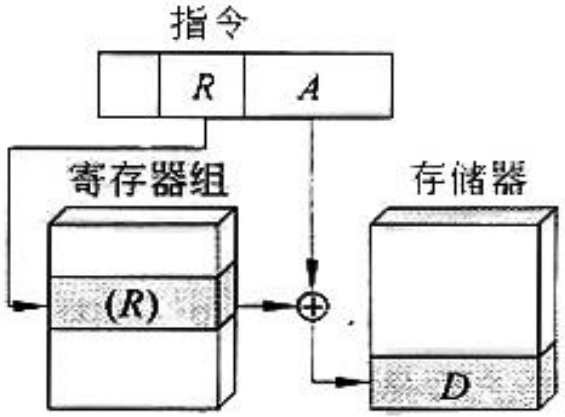
（3） 16d ×(DS)+（BX)，取出X

（4） 16d ×(DS)+X，取出操作数D

4.4.2 操作数基本寻址方式

7、基址寻址

地址码字段中指定基址寄存器R、偏移地址A，（R）+A是操作数所在内存单元的逻辑地址。操作数在内存单元中。



ADD AX,[BX+1234H]

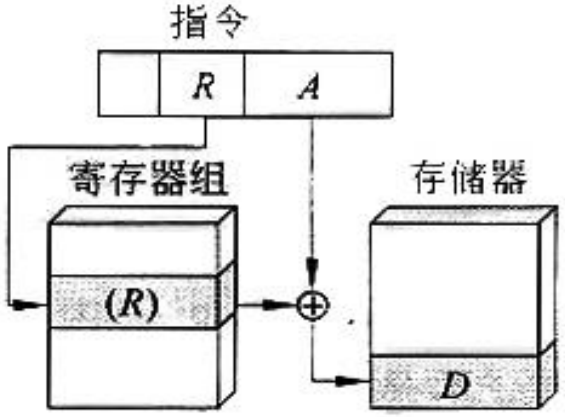
物理地址：16d×(DS)+(BX)+1234H

用于数据块的访问：（BX）是基准点，块首地址，而变化的是位移量。

4.4.2 操作数基本寻址方式

8、变址寻址

地址码字段中指定变址寄存器R、偏移地址A，（R）+A是操作数所在内存单元的逻辑地址。操作数在内存单元中。



ADD AX,[SI+1234H]

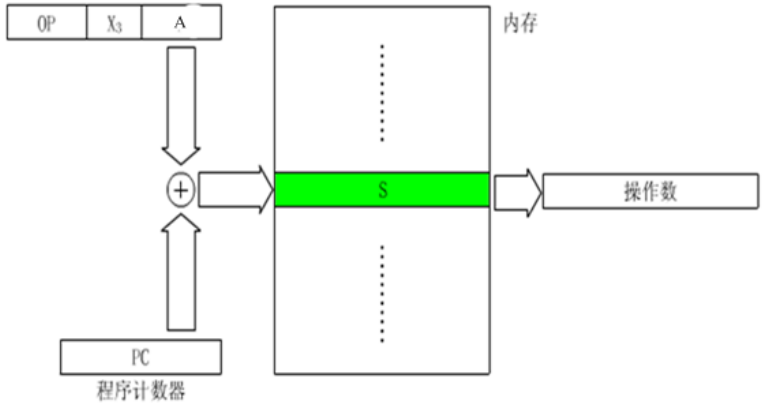
物理地址：16d×(DS)+(SI)+1234H

用于数据块的访问： 偏移地址是基准点，块首地址，而变化的是（SI）。

4.4.2 操作数基本寻址方式

9、相对寻址

程序计数器PC的内容加上地址码字段中的偏移地址A，形成操作数所在内存单元的物理地址。 操作数在内存单元中。



ADD AX,[PC+1234H]

物理地址：16d×(DS)+(PC)+1234H

注意： （PC）=下一条指令的首单元地址

4.4.2 操作数基本寻址方式

ADD AX,[BX+1234H]　　BX基址寄存器，基准点

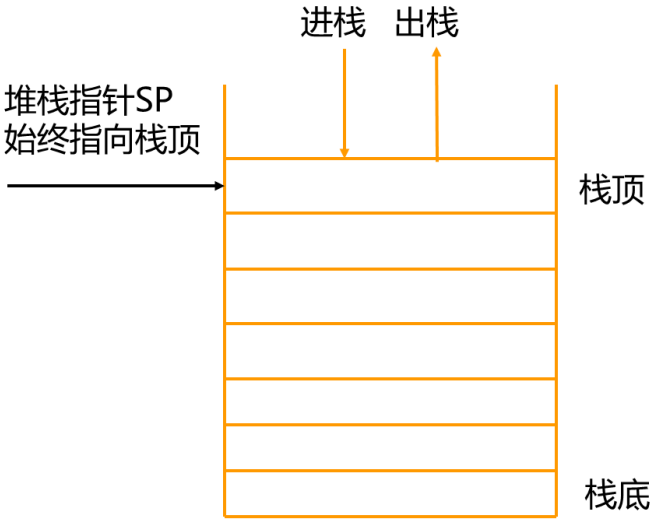
ADD AX,[SI+1234H]　　SI变址寄存器，变化量

ADD AX,[PC+1234H]　　PC程序计数器，相对点

4.4.2 操作数基本寻址方式

11、堆栈寻址

按照“先进后出、后进先出”原则的顺序访问的数据组织结构，通常采用存储器堆栈，因此操作数在存储单元中。



初始化：设置栈底位置，即SP赋初值；

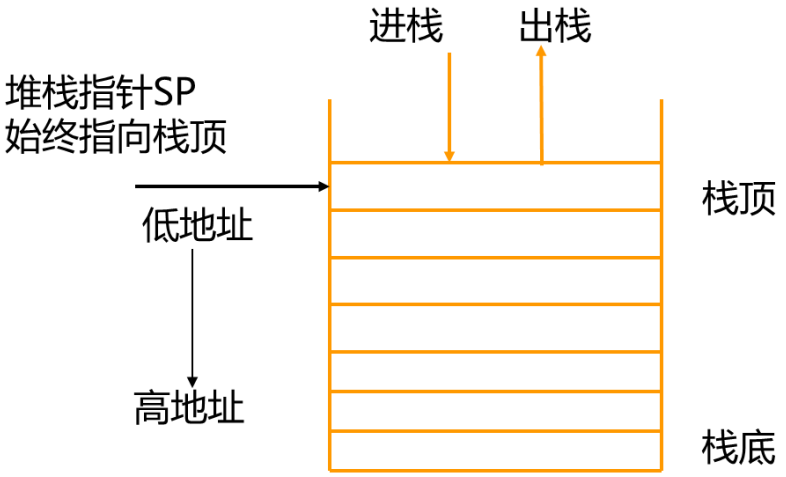
进栈操作：向堆栈PUSH一个数据单元；

出栈操作：从堆栈中POP一个数据单元。

4.4.2 操作数基本寻址方式

11、堆栈寻址

按照“先进后出、后进先出”原则的顺序访问的数据组织结构，通常采用存储器堆栈，因此操作数在存储单元中。



进栈操作：

（SP）-1→（SP），A→（（SP））

出栈操作：

（（SP））→B，（SP）+1→（SP）

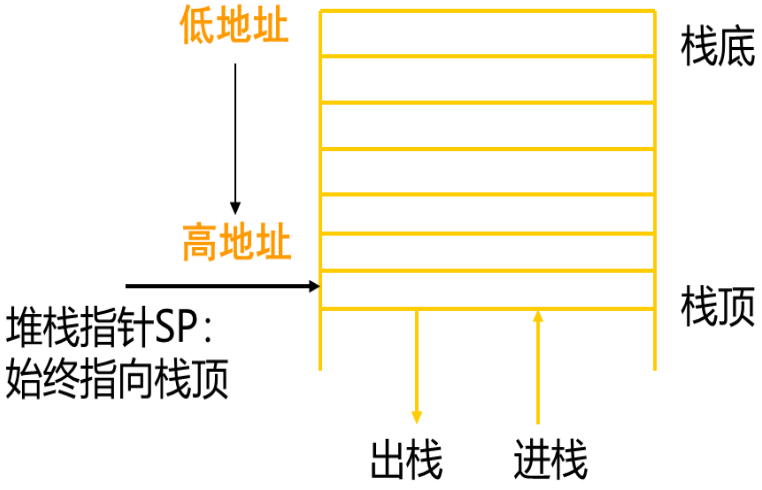
物理地址：

16d×(SS)+(SP)

4.4.2 操作数基本寻址方式

11、堆栈寻址

按照“先进后出、后进先出”原则的顺序访问的数据组织结构，通常采用存储器堆栈，因此操作数在存储单元中。



进栈操作：

（SP）+1→（SP），A→（（SP））

出栈操作：

（（SP））→B，（SP）-1→（SP）

物理地址：

16d×(SS)+(SP)

4.4.3 寻址方式举例

—— Pentium的寻址方式

序号	寻址方式	有效地址算法	说明
1	立即寻址	操作数=A	操作数A在指令中
2	寄存器寻址	EA=R	操作数在寄存器R中
3	偏移量寻址	EA=A	偏移量A在指令中，8位、16位、32位
4	基址寻址	EA=（B）	B是基址寄存器
5	基址+偏移量	EA=（B）+A	
6	比例变址+偏移量	EA=（I）*S+A	I是变址寄存器，比例因子S=1、2、4、8
7	基址+变址+偏移量	EA=（B）+（I）+A	
8	基址+比例变址+偏移量	EA=（B）+（I）*S+A	
9	相对寻址	指令地址=（PC）+A	PC为程序计数器

4.4.3 寻址方式举例

[例]一种二地址RS型指令的结构如下所示：

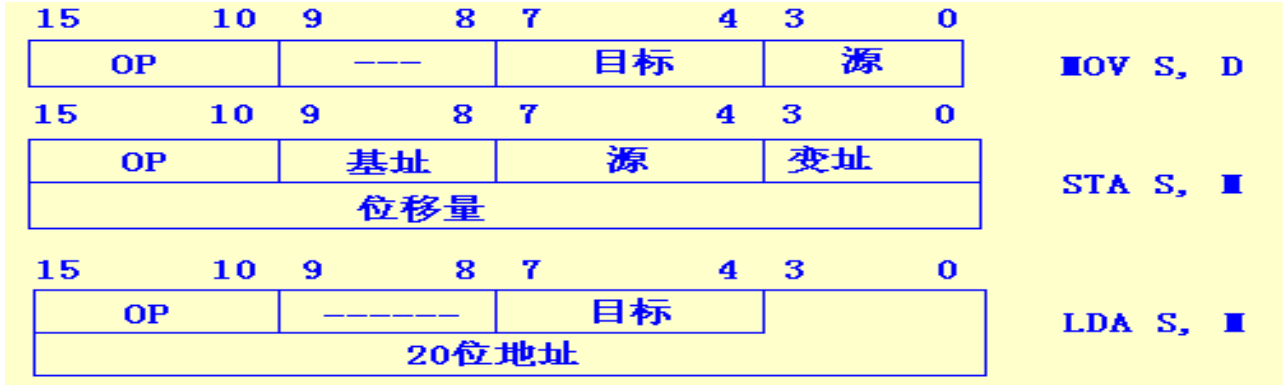
其中I为间接寻址标志位，X为寻址模式字段，D位偏移量字段。通过I，X，D的组合，可构成下表所示的寻址方式。 请写出六种寻址方式的名称。

[解]

寻址方式	I	X	有效地址算法	说明	答案
(1)	0	0	E=D		直接寻址
(2)	0	1	E=PC ± D	PC程序计数器	相对寻址
(3)	0	10	E= (R2) ±D	R2变址寄存器	变址寻址
(4)	1	11	E= (R3)	R3间址寄存器	寄存器间接寻址
(5)	1	0	E= (D)		间接寻址
(6)	0	11	E= (R1) ±D	R1基址寄存器	基址寻址

4.4.3 寻址方式举例

[例] 某16位机器所使用的指令格式和寻址方式如下所示。



要求： (1)分析三种指令的指令格式与寻址方式特点。

(2)CPU完成哪一种操作所花时间最短？哪一种操作所花时间最长？第二种指令的执行时间有时会等于第三种指令的执行时间吗？

4.4.3 寻址方式举例

[解]　格式一

（1）操作码：操作码字段OP=6位，可以指定64类操作；定长操作码字段；

（2）地址码：双操作数指令；源操作数由“源”指定16个寄存器之一，寄存器寻址方式；目的操作数由“目标”指定16个寄存器之一，寄存器寻址方式；RR型指令；

（3）指令字长度：单字长指令；变长指令字结构。

4.4.3 寻址方式举例

[解] 格式二

（1）操作码：操作码字段OP=6位，可以指定64类操作；定长操作码字段；

（2）地址码：双操作数指令；源操作数由“源”指定16个寄存器之一，寄存器寻址方式；目的操作数由“基址寄存器（4个之一）+变址（16个之一）+位移量（16位）”指定存储单元，基地变址寻址方式；RS型指令；

（3）指令字长度：双字长指令；变长指令字结构。

4.4.3 寻址方式举例

[解] 格式三

（1）操作码：操作码字段OP=6位，可以指定64类操作；定长操作码字段；

（2）地址码：双操作数指令；目的操作数由“目标”指定16个寄存器之一，寄存器寻址方式；源操作数由“20位地址”指定存储单元，直接寻址方式；RS型指令；

（3）指令字长度：双字长指令；变长指令字结构。

4.4.3 寻址方式举例

[解] 执行时间比较

第一种指令最短、第二种指令最长、第二种指令的执行时间永远不可能等于第三种指令的执行时间。

第二种指令： （1）需要“基地+变址+位移量”、物理地址的地址变换计算；（2） “基地+变址+位移量”是目的操作数，需保存结果到内存单元；

第三种指令：指令直接给出20位物理地址，无地址变换计算；源操作数，无需保存结果到存储单元。

4.5 典型指令

- 指令的分类
- 基本指令系统的操作

4.5.1 指令的分类

数据传送指令	取数指令、存数指令、传送指令、成组传送指令、字节交换指令、清寄存器指令、堆栈操作指令
算术运算指令	二进制定点加、减、乘、除指令，浮点加、减、乘、除指令，求反、求补、算术移位运算指令、比较运算指令，十进制加、减运算指令、向量运算指令
逻辑运算指令	与、或、非、异或，逻辑移位指令，按位运算指令
程序控制指令	无条件转移指令、条件转移指令、子程序调用与返回指令、中断指令
输入输出指令	输入指令、输出指令
字符串处理指令	字符串传送、比较、查找、替换、转换指令
特权指令	系统资源的分配和管理、改变系统工作方式、管理用户的访问权限、修改虚拟存储器的段表页表、任务的创建和切换
其他指令	状态寄存器复位、置位指令，测试指令、暂停指令、空操作指令、停机指令

4.5.2 基本指令系统的操作

指令类型	指令助记符	指令操作	指令类型	指令助记符	指令操作
数据传送	MOV	传送	算术运算	INC	加1
	STO	存数		DEC	减1
	LAD	取数	逻辑运算	AND	与
	EXC	交换		OR	或
	CLA	清零		NOT	反
	SET	置1		EOR	异或
	PUS	进栈		TES	测试
	POP	出栈		COM	比较
算术运算	ADD	加法		SHI	移位
	SUB	减法		ROT	循环移位
	MUL	乘法	控制传递	JMP	无条件转移
	DIV	除法		JMPX	条件转移
	ABS	绝对值		JMPC	转子程序
	NEG	求负		RET	返回