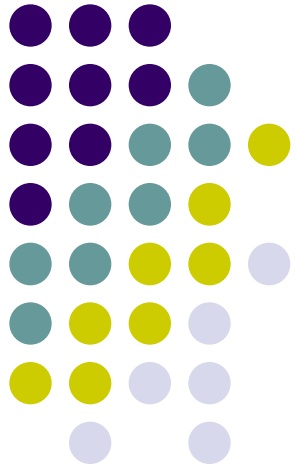
第四章 指令系统

4.1 指令系统的发展与性能要求
4.2 指令格式
4.3 操作数类型
4.4 指令和数据的寻址方式
4.5 典型指令
4.6 ARM汇编语言



返回

4.1 指令系统的发展与性能要求

4.1.1 指令系统的发展

4.1.2 对指令系统性能的要求

4.1.3 低级语言与硬件结构的关系

4.1.1 指令系统的发展

指令系统基本概念

- 指令：就是要计算机执行某种操作的命令。从计算机组成的层次结构来说，计算机的指令有微指令、机器指令和宏指令之分。微指令是微程序级的命令，它属于硬件；
- 宏指令：由若干条机器指令组成的软件指令，它属于软件；
- 机器指令：介于微指令与宏指令之间，通常简称为指令，每一条指令可完成一个独立的算术运算或逻辑运算操作。
- 本章所讨论的指令，是机器指令。
- 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，它的格式与功能不仅直接影响到机器的硬件结构，而且也直接影响到系统软件，影响到机器的适用范围

4.1.1 指令系统的发展

发展情况

- 复杂指令系统计算机，简称CISC。但是如此庞大的指令系统不但使计算机的研制周期变长，难以保证正确性，不易调试维护，而且由于采用了大量使用频率很低的复杂指令而造成硬件资源浪费。
- 精简指令系统计算机：简称RISC，人们又提出了便于VLSI技术实现的精简指令系统计算机。

4.1.2 对指令系统性能的要求

- 完备性：完备性是指用汇编语言编写各种程序时，指令系统直接提供的指令足够使用，而不必用软件来实现。完备性要求指令系统丰富、功能齐全、使用方便。一台计算机中最基本、必不可少的指令是不多的。许多指令可用最基本的指令编程来实现。例如，乘除运算指令、浮点运算指令可直接用硬件来实现，也可用基本指令编写的程序来实现。采用硬件指令的目的是提高程序执行速度，便于用户编写程序。
- 有效性：有效性是指利用该指令系统所编写的程序能够高效率地运行。高效率主要表现在程序占据存储空间小、执行速度快。一般来说，一个功能更强、更完善的指令系统，必定有更好的有效性。
- 规整性：规整性包括指令系统的对称性、匀齐性、指令格式和数据格式的一致性。对称性是指：在指令系统中所有的寄存器和存储器单元都可同等对待，所有的指令都可使用各种寻址方式；匀齐性是指：一种操作性质的指令可以支持各种数据类型，如算术运算指令可支持字节、字、双字整数的运算，十进制数运算和单、双精度浮点数运算等；指令格式和数据格式的一致性是指：指令长度和数据长度有一定的关系，以方便处理和存取。例如指令长度和数据长度通常是字节长度的整数倍。
- 兼容性：系列机各机种之间具有相同的基本结构和共同的基本指令集，因而指令系统是兼容的，即各机种上基本软件可以通用。但由于不同机种推出的时间不同，在结构和性能上有差异，做到所有软件都完全兼容是不可能的，只能做到“向上兼容”，即低档机上运行的软件可以在高档机上运行。

4.1.3 低级语言与硬件结构的关系

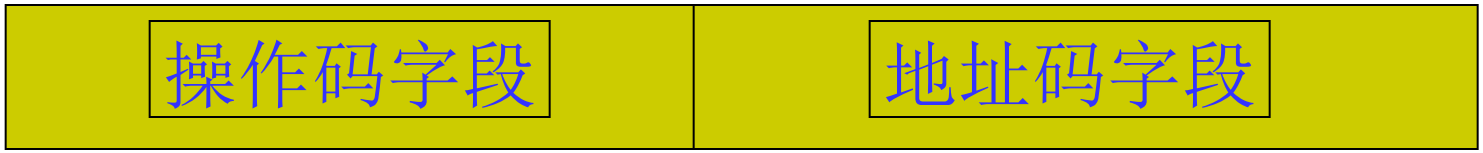
低级语言与高级语言关系

	比较内容	高级语言	低级语言
1	对程序员的训练要求 (1)通用算法 (2)语言规则 (3)硬件知识	有 较少 不要	有 较多 要
2	对机器独立的程度	独立	不独立
3	编制程序的难易程度	易	难
4	编制程序所需时间	短	较长
5	程序执行时间	较长	短
6	编译过程中对计算机资源的要求	多	少

4.2 指令格式

4.2.1 操作码
4.2.2 地址码
4.2.3 指令字长度
4.2.4 指令助记符
4.2.5 指令格式举例

- 指令格式包括两个方面：



4.2.1 操作码

- 设计计算机时，对指令系统的每一条指令都要规定一个操作码。指令的操作码OP表示该指令应进行什么性质的操作，如进行加法、减法、乘法、除法、取数、存数等等。不同的指令用操作码字段的不同编码来表示，每一种编码代表一种指令。
- 组成操作码字段的位数一般取决于计算机指令系统的规模。较大的指令系统就需要更多的位数来表示每条特定的指令。
  - 等长（指令规整，译码简单）
    - 例如IBM 370机，该机字长32位，16个通用寄存器R0～R15，共有183条指令；指令的长度可以分为16位、32位和48位等几种，所有指令的操作码都是8位固定长度。
    - 固定长度编码的主要缺点是：信息的冗余极大，使程序的总长度增加。

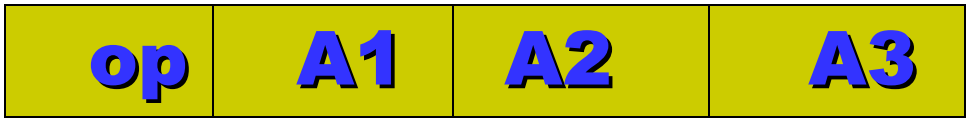
4.2.2 地址码

- 根据一条指令中有几个操作数地址，可将该指令称为几操作数指令或几地址指令。
  - 三地址指令
  - 二地址指令
  - 单地址指令
  - 零地址指令

操作码（4位）	A1（6位）	A2（6位）	A3（6位）
操作码（4位）	A1（6位）	A2（6位）	
操作码（4位）	A1（6位）		
操作码			

4.2.2 地址码

- 三地址指令
  - 指令格式如下：



  - 操作码op　第一操作数A1　第二操作数A2　　结果A3
  - 功能描述：
  - (A1) op(A2)→A3
  - (PC) +1→PC
  - 这种格式虽然省去了一个地址，但指令长度仍比较长，所以只在字长较长的大、中型机中使用，而小型、微型机中很少使用。

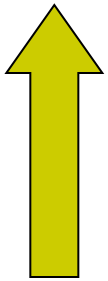
4.2.2 地址码

- 二地址指令
  - 其格式如下：

  - 操作码op第一操作数A1第二操作数A2
  - 功能描述：
  - (A1) op(A2)→A1
  - (PC)+1→PC
  - 二地址指令在计算机中得到了广泛的应用，但是在使用时有一点必须注意：指令执行之后，A1中原存的内容已经被新的运算结果替换了。

4.2.2 地址码

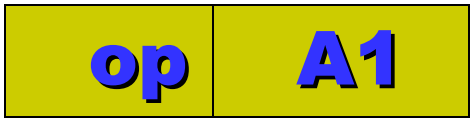
- 二地址地址根据操作数的物理位置分为：
  - SS　存储器-存储器类型
  - RS　寄存器-存储器类型
  - RR　寄存器-寄存器类型



慢

4.2.2 地址码

- 一地址指令
  - 指令格式为：



  - 操作码op　　第一操作数A1
  - 功能描述：
  - (AC) op(A1) →A1
  - (PC)+1→PC
  - 单操作数运算指令，如“+1”、“-1”、“求反”
  - 指令中给出一个源操作数的地址

4.2.2 地址码

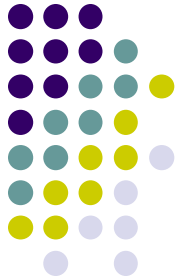
- 零地址指令
  - 其格式为：

  - 操作码op
  - “停机”、“空操作”、“清除”等控制类指令。

4.2.3 指令字长度

- 概念
  - 指令字长度（一个指令字包含二进制代码的位数）
  - 机器字长：计算机能直接处理的二进制数据的位数。
  - 单字长指令
  - 半字长指令
  - 双字长指令
- *多字长指令*的优缺点
  - 优点提供足够的地址位来解决访问内存任何单元的寻址问题；
  - 缺点必须两次或多次访问内存以取出一整条指令，降低了CPU的运算速度，又占用了更多的存储空间。
- 指令系统中指令采用*等长指令*的优点：各种指令字长度是相等的，指令字结构简单，且指令字长度是不变的；
- 采用*非等长指令*的的优点：各种指令字长度随指令功能而异，结构灵活，能充分利用指令长度，但指令的控制较复杂。

4.2.4 指令助记符

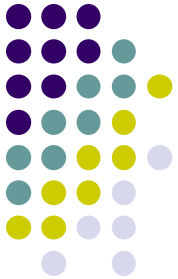


- 由于硬件只能识别1和0，所以采用二进制操作码是必要的，但是我们用二进制来书写程序却非常麻烦。
- 为了便于书写和阅读程序，每条指令通常用3个或4个英文缩写字母来表示。这种缩写码叫做指令助记符
  - 用3～4个英文字母来表示操作码，一般为英文缩写
  - 不同的计算机系统，规定不一样
  - 必须用汇编语言翻译成二进制代码

4.2.5 指令格式举例

- 8位微型计算机的指令格式
  - 如8088，字长8位，指令结构可变
  - 包括单字长指令、双字长指令和三字长指令
  - 操作码长度固定

4.2.5 指令格式举例



- MIPS R4000指令格式
- RISC计算机系统，32位字长，32个通用寄存器。

  R型（寄存器）指令：

  6位 5位 5位 5位 5位 6位

  op rs rt rd shamt funct

  I型（立即数）指令：

  6位 5位 5位 16位

  op rs rt 常数或地址

4.2.5 指令格式举例：ARM指令格式

2. ARM指令的格式

ARM是字长32位的嵌入式处理机，2008年生产了4亿片，它具有世界上最流行的指令集。下面是ARM指令集的指令一种格式：

cond	F	I	opcode	s	R_n	R_d	operand 2
4 位	2 位	1 位	4 位	1 位	4 位	4 位	12 位

各字段的含义如下：

opcode　指明指令的基本操作，称为操作码。

R_d　指明目的寄存器的地址（4位），共16个。

R_n　指明源寄存器地址（4位），共16个。

Operand2　指明第2个源操作数。

I　指明立即数，如果I=0，第2个源操作数在寄存器中；如果i=1，第2个源操作数

是12 位的立即数。

S　设置状态，该字段涉及条件转移指令。

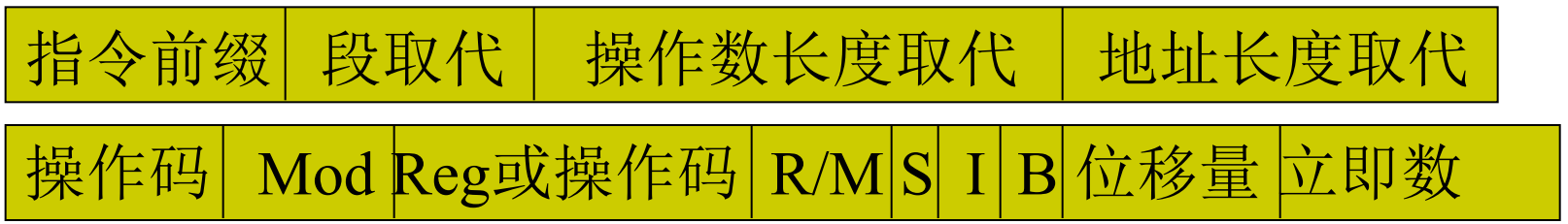
cond　指明条件，该字段涉及条件转移指令。

F　说明指令类型，当需要时该字段设置不同的指令。

4.2.5 指令格式举例

Pentium指令格式

- 指令长度可变，最短1个字节，最长12个字节，典型的CISC指令系统



4.2.5 指令格式举例

[例3] MIPS R4000汇编语言中，寄存器$s0～$s7对应寄存器号为十进制16～23，寄存器$t0～$t7对应的寄存器号为8～15。请将下表4条汇编语言翻译成对应的机器语言十进制数表示。

类别	指令	示例	语义	说明
算术运算	加	add $s1,$s2,$s3	$s1=$s2+$s3	三个操作数：数据都在寄存器中
	减	sub $s1,$s2,$s3	$s1=$s2-$s3	三个操作数：数据都在寄存器中
数据传送	取字	lw $s1,100($s2)	$s1=Memory[$s2+100]	数据从存储器到寄存器
	存字	sw $s1,100($s2)	Memory[$s2+100]=$s1	数据从寄存器到存储器

4.3 操作数类型

4.3.1 一般的数据类型

4.3.2 Pentium数据类型

4.3.3 Power PC数据类型

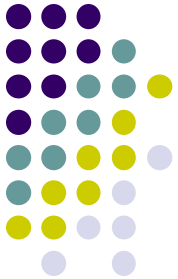
4.3.1 一般的数据类型

- 地址数据:地址实际上也是一种形式的数据。
- 数值数据:计算机中普遍使用的三种类型的数值数据。
- 字符数据:文本数据或字符串，目前广泛使用ASCII码。
- 逻辑数据:一个单元中有几位二进制bit项组成，每个bit的值可以是1或0。当数据以这种方式看待时，称为逻辑性数据。

4.3.2 Pentium数据类型

常规
整数
序数
未压缩的BCD
压缩的BCD
近指针
位串
字符串
浮点数

4.3.3 Power PC数据类型



无符号字节
无符号半字
有符号半字
无符号字
有符号字
无符号双字
字节串
浮点数

4.4 指令和数据的寻址方式

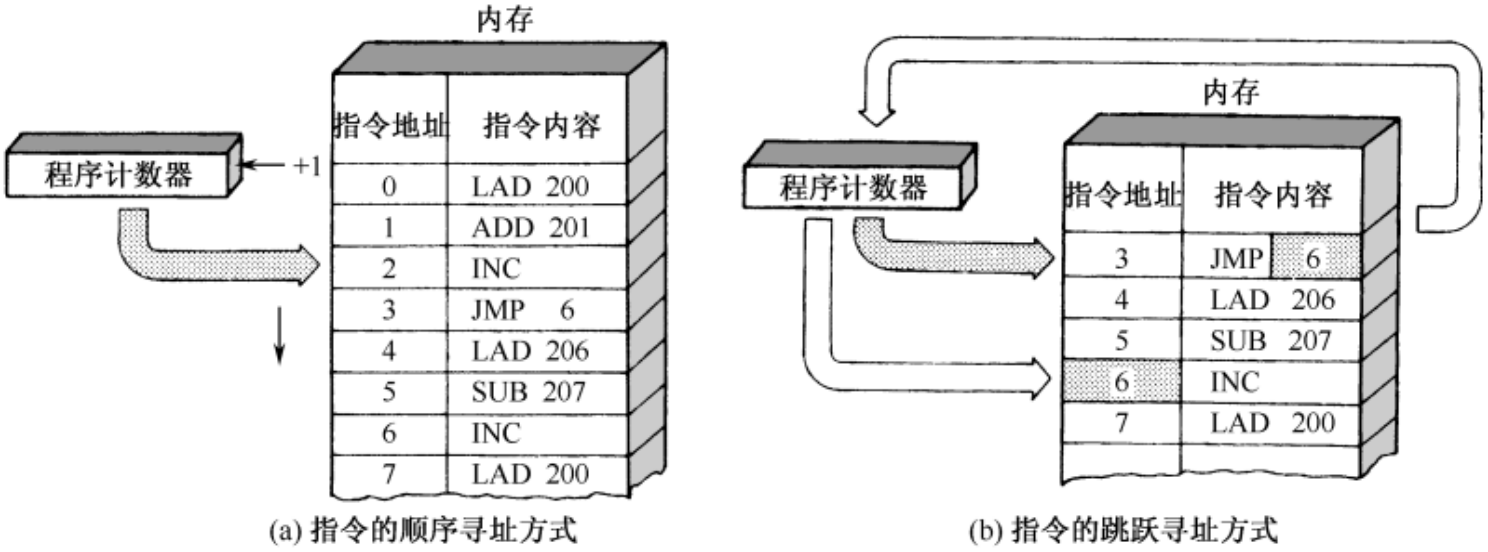
4.4.1 指令的寻址方式

4.4.2 操作数基本寻址方式

4.4.3 寻址方式举例

4.4.1 指令的寻址方式

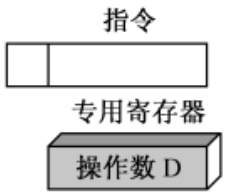
- 顺序方式
  - PC
- 跳跃方式



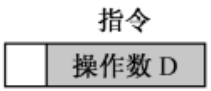
4.4.2 操作数基本寻址方式

- 形成操作数有效地址的方法，称为寻址方式。
- 例如，一种单地址指令的结构如下：
  操作码OP、变址X、间址I、形式地址A

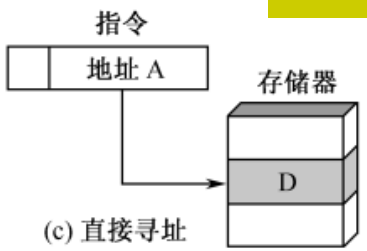
4.4.2 操作数基本寻址方式



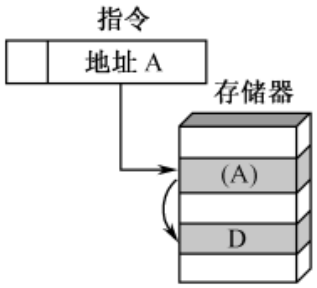
(a) 隐含寻址



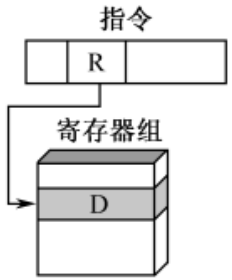
(b) 立即寻址



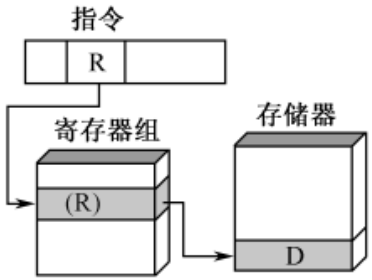
(c) 直接寻址



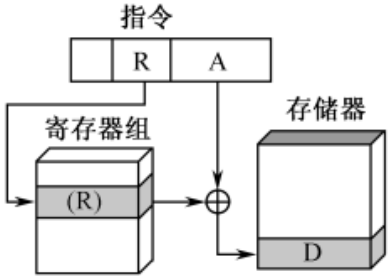
(d) 间接寻址



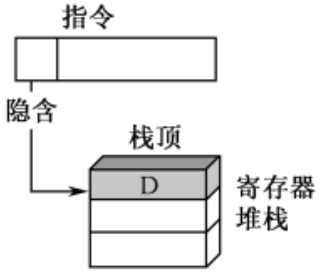
(e) 寄存器寻址



(f) 寄存器间接寻址

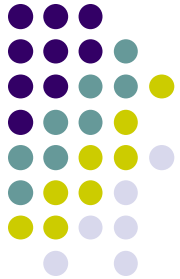


(g) 偏移寻址



(h) 堆栈寻址

4.4.2 操作数基本寻址方式



1、隐含寻址

- 指令中隐含着操作数的地址
- 如某些运算，隐含了累加器AC作为源和目的寄存器
  - 如8086汇编中的STC指令，设置标志寄存器的C为1

4.4.2 操作数基本寻址方式

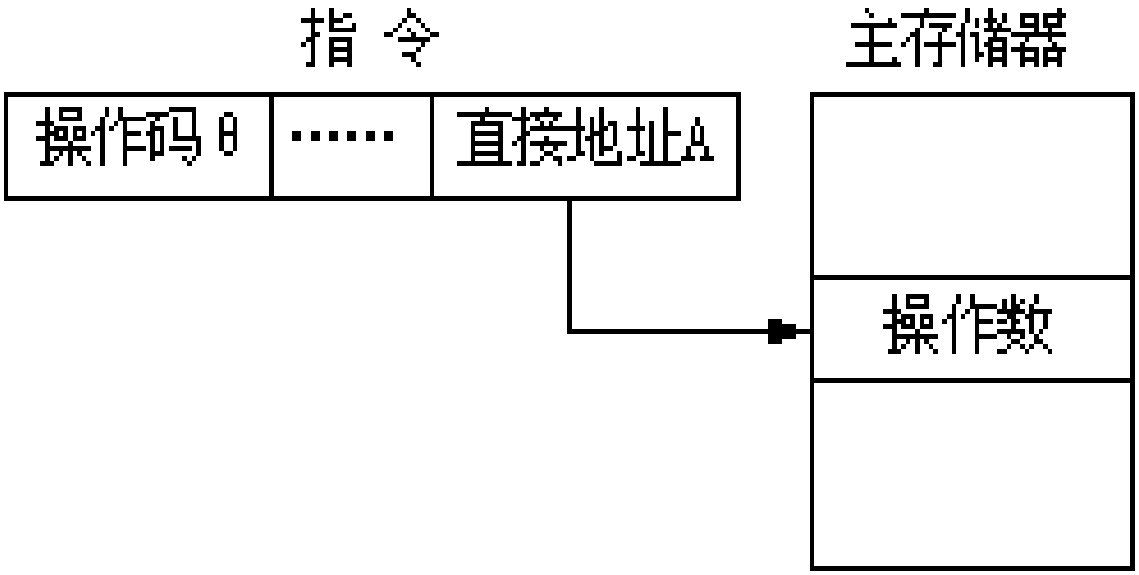
2、立即寻址

- 立即寻址是一种特殊的寻址方式，指令中在操作码字段后面的部分不是通常意义上的操作数地址，而是操作数本身，也就是说数据就包含在指令中，只要取出指令，就取出了可以立即使用的操作数，因此，这样的操作数被称为立即数。
- 指令格式：操作码OP 操作数A

4.4.2 操作数基本寻址方式

3、直接寻址

- 指令中地址码字段给出的地址A就是操作数的有效地址EA(Effective Address)，即EA＝A。

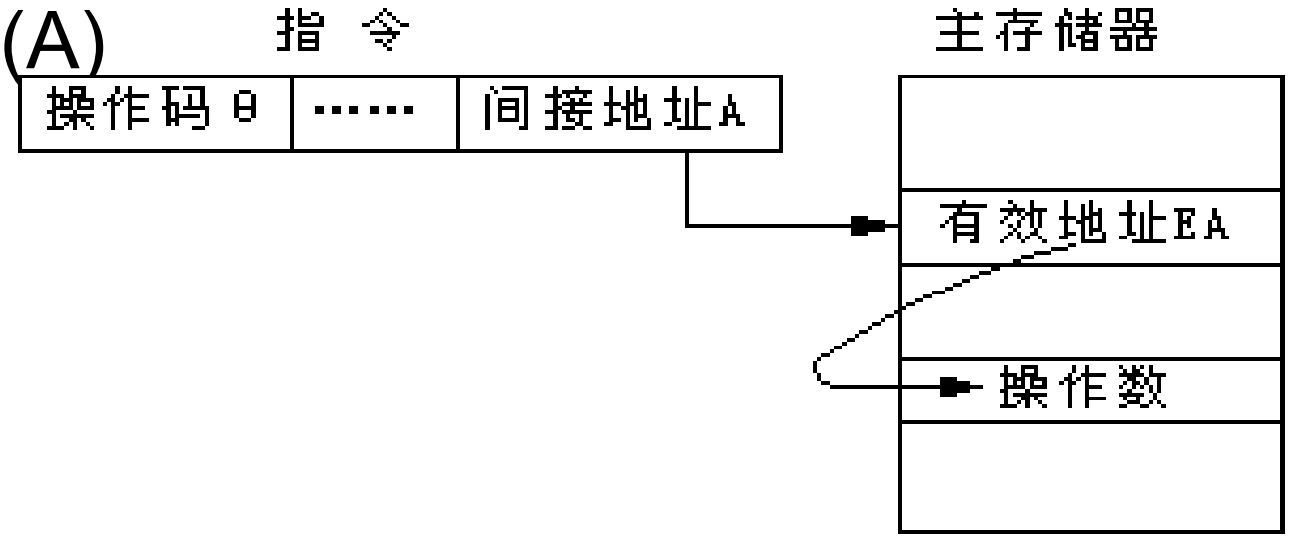


4.4.2 操作数基本寻址方式

4、间接寻址

- 间接寻址意味着指令的地址码部分给出的地址A不是操作数的地址，而是存放操作数地址的主存单元的地址，简称操作数地址的地址。操作数的有效地址的计算公式为：

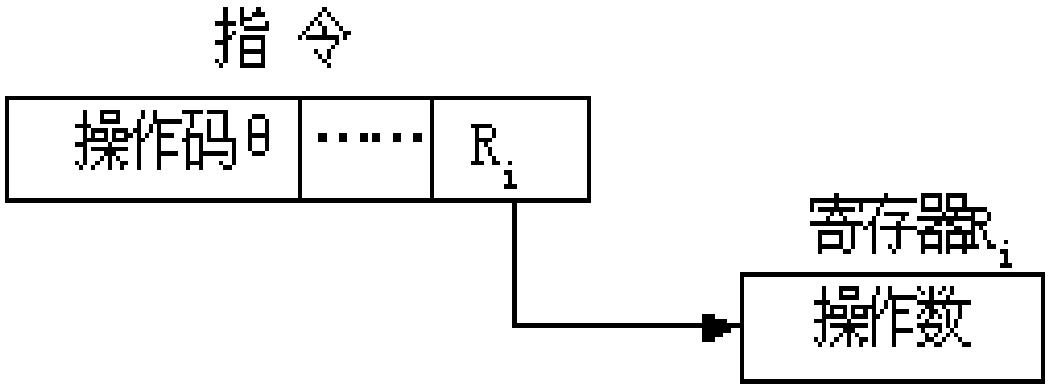
EA＝(A)



4.4.2 操作数基本寻址方式

5、寄存器寻址

- 在指令的地址码部分给出CPU内某一通用寄存器的编号，指令的操作数存放在相应的寄存器中，即EA=Ri



优点：

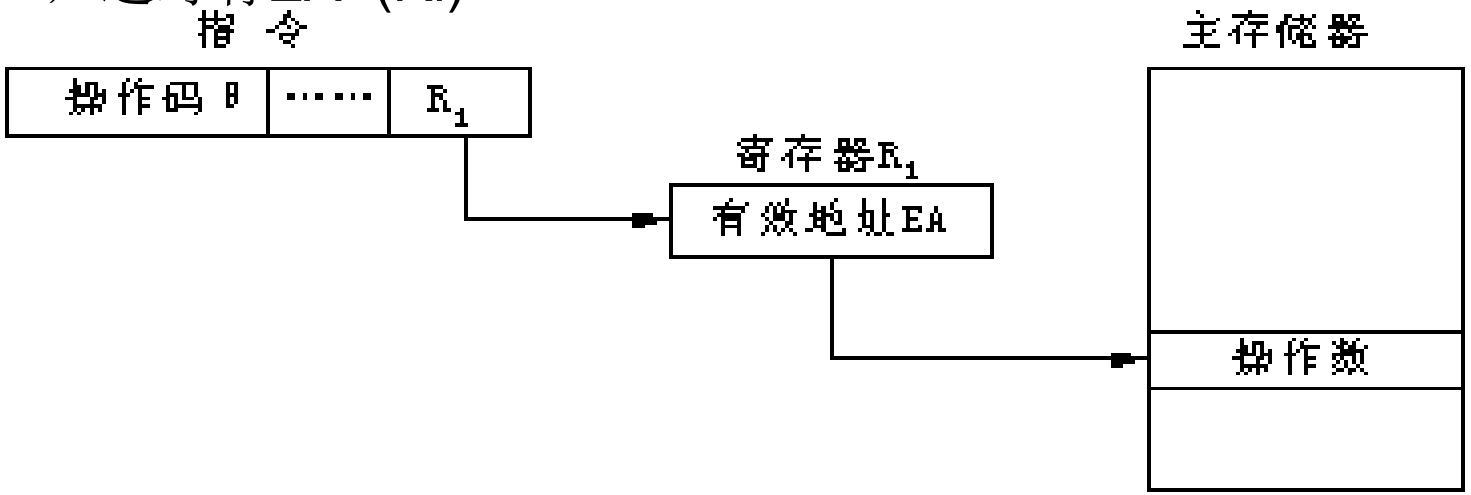
(1)由于寄存器在CPU的内部，指令在执行时从寄存器中取操作数比访问主存要快得多；

(2)由于寄存器的数量较少，因此寄存器编号所占位数也较少，从而可以有效减少指令的地址码字段的长度。

4.4.2 操作数基本寻址方式

6、寄存器间接寻址

- 为了克服间接寻址中多次访存的缺点，可采用寄存器间接寻址，即将操作数放在主存储器中，而操作数的地址放在某一通用寄存器中，然后在指令的地址码部分给出该通用寄存器的编号，这时有EA=(Ri)

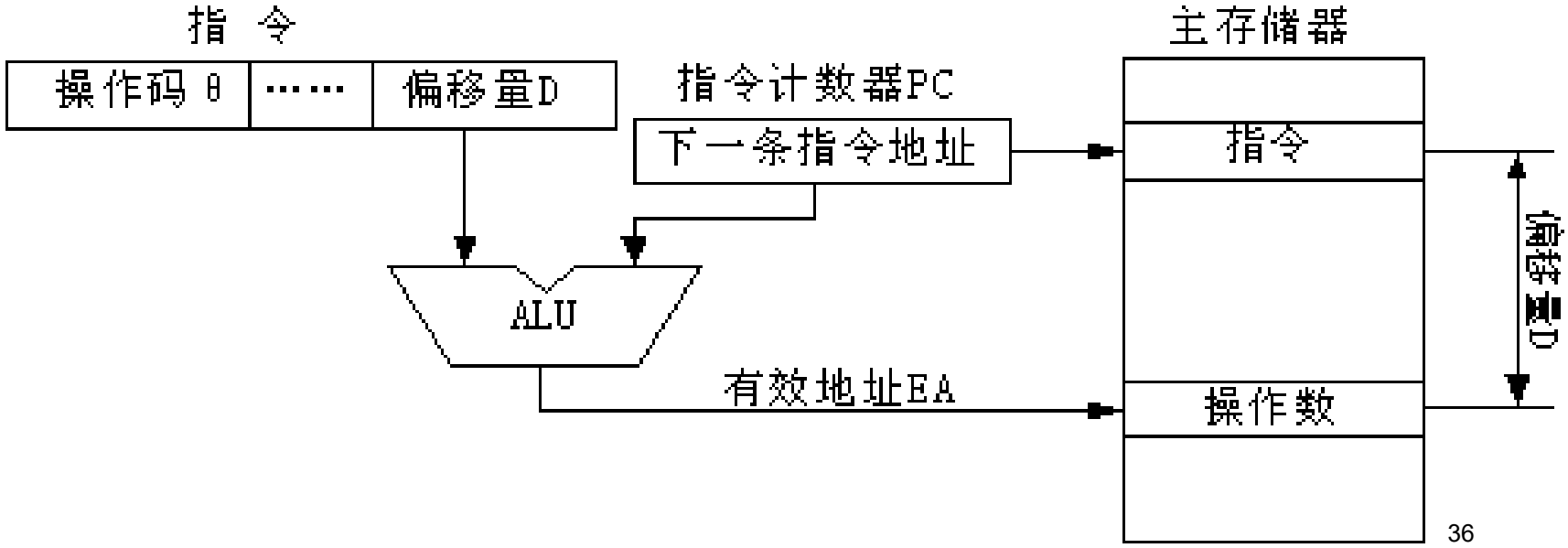


- 这种寻址方式的指令较短，并且在取指后只需一次访存便可得到操作数，因此指令执行速度较前述的间接寻址方式要快，也是目前在计算机中使用较为广泛的一种寻址方式。

4.4.2 操作数基本寻址方式

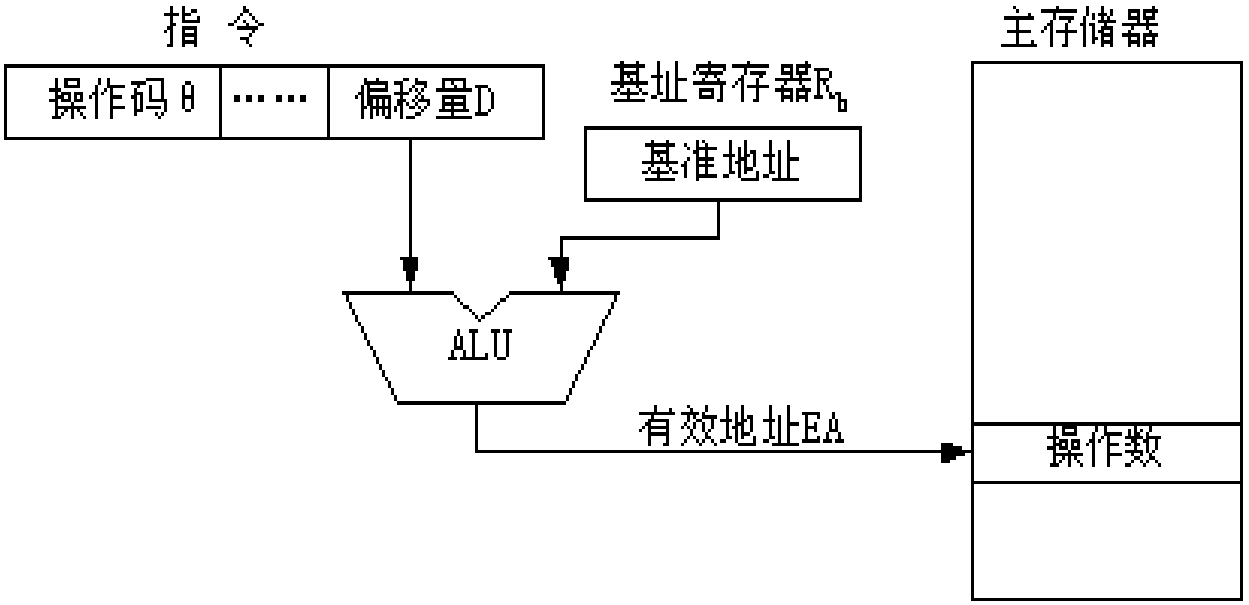
7、偏移寻址

- 相对寻址：由程序计数器**PC**提供基准地址，而指令的地址码部分给出相对的位移量**D**，两者相加后作为操作数的有效地址，即：**EA＝(PC)＋D**。



4.4.2 操作数基本寻址方式

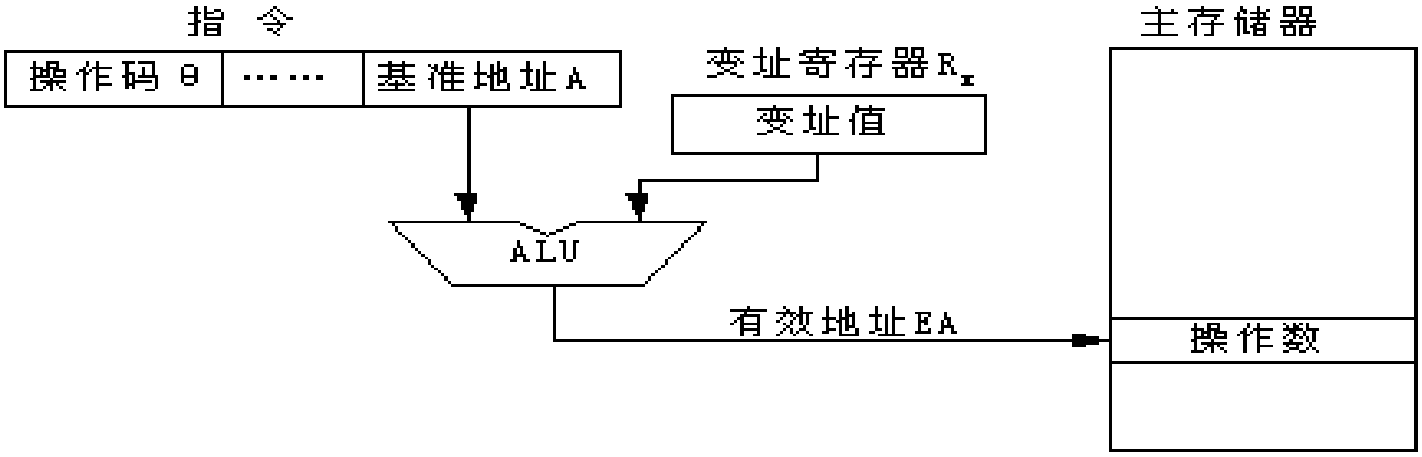
- **7**、偏移寻址：基址寻址



- 基址寄存器的位数可以设置得很长，从而可以在较大的存储空间中寻址。

4.4.2 操作数基本寻址方式

- 7、偏移寻址：变址寻址

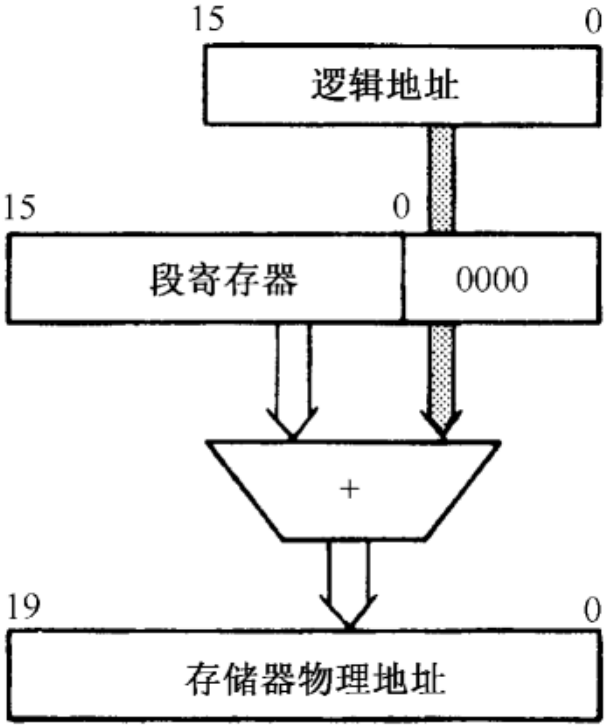


- 变址寻址就是将指令的地址码部分给出的基准地址A与CPU内某特定的变址寄存器Rx中的内容相加，以形成操作数的有效地址。

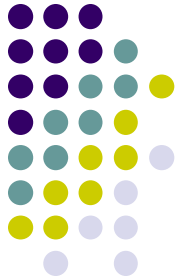
4.4.2 操作数基本寻址方式

8、段寻址方式

存储空间划分为多段



4.4.2 操作数基本寻址方式



- 分寄存器堆栈、存储器堆栈
以先进后出原理存储数据

4.4.2 操作数基本寻址方式

例**4.4**：将**ARM**汇编语言翻译成机器语言。已知**5**条**ARM**指令格式译码如下表所示：

指令名称	cond	F	I	opcode	S	R_n	R_d	operand2
ADD（加）	14	0	0	4	0	reg	reg	reg
SUB（减）	14	0	0	2	0	reg	reg	reg
ADD（立即数加）	14	0	1	4	0	reg	reg	constand（12 位）
LDR（取字）	14	1	–	24	–	reg	reg	address（12 位）
STR（存字）	14	1	–	25	–	reg	reg	address（12 位）

设**r3**寄存器中保存数组**A**的基值，**h**放在寄存器**r2**中。**C**语言程序语句 **A[30]=h+A[30]** 可编译成如下**3**条汇编语句指令：

```
LDR r5 , [r3, #120]        ;寄存器r5中获得A[30]
ADD r5 , r2 , r5 ,         ;寄存器r5中获得h+A[30]
STR r5 , [r3, #120]        ;将h+A[30]存入到A[30]
```

请问这**3**条汇编语言指令的机器语言是什么？

4.4.2 操作数基本寻址方式

解：首先利用十进制数来表示机器语言指令，然后转换成二进制机器指令。从表**4.3**中我们可以确定**3**条机器语言指令：

LDR指令在第**3**字段（**opcond**）用操作码**24**确定。基值寄存器**3**指定在第**4**字段（R_n），目的寄存器**5**指定在第**6**字段（R_d），选择**A[30]**（**120**＝**30×4**）的**offset**字段放在最后一个字段（**offset12**）。

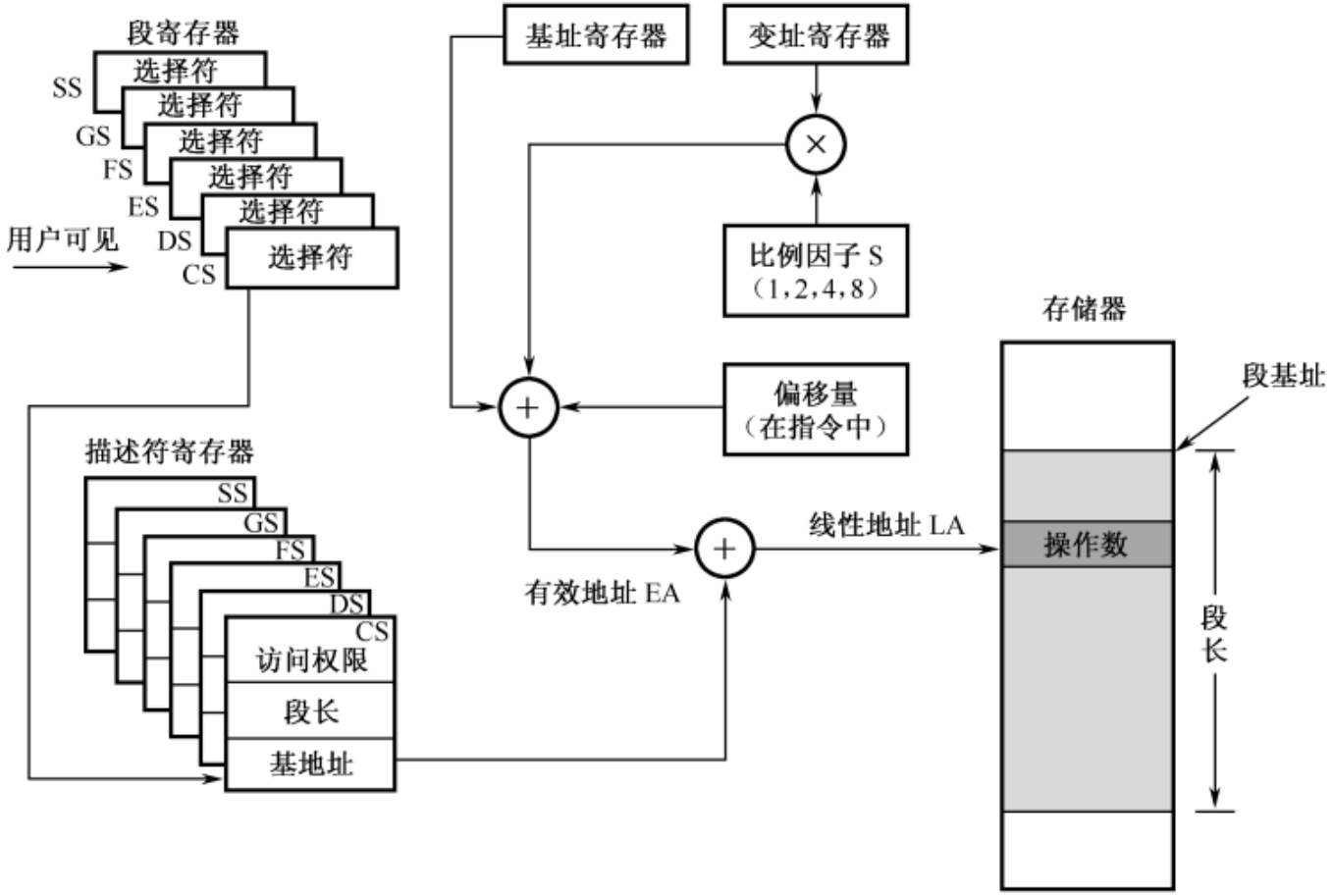
ADD指令在第**4**字段（**opcode**）用操作码**4**确定。**3**个寄存器操作（**2**、**5**和**5**）分别被指定在第**6**、**7**、**8**字段。

STR指令在第**3**字段用操作码**25**确定，其余部分与**LDR**指令相同。

	cond	F	opcode			Rn	Rd	offset
			I	opcode	S			operand
十	14	1	24			3	5	120
进	14	0	0	4	0	2	5	5
制	14	1	25			3	5	120
二	1110	1	11000			0011	0101	0000 1111 0000
进	1110	0	0	100	0	0010	0101	0000 0000 0101
制	1110	1	11001			0011	0101	0000 1111 0000

4.4.3 寻址方式举例

Pentium的寻址方式



4.4.3 寻址方式举例

Pentium的寻址方式

方式	算法
立即	作数=A
寄存器	LA=R
偏移量	LA=(SR)+A
基址	LA=(SR)+(B)
基址带 偏移量	LA=(SR)+(B)+A
比例变 址带 偏移量	LA=(SR)+(I)×S+A
基址带变 址和偏移量	LA=(SR)+(B)+(I)+A
基址带 比例变 址和偏移量	LA=(SR)+(B)+(I)×S+A
相对	LA=(PC)+A

4.4.3 寻址方式举例

[例4] 一种二地址RS型指令的结构如下：

6位 4位 1位 2位 16位

OP 通用寄存器 I X 偏移量D

其中I为间接寻址标志位，X为寻址模式字段，D为偏移量字段。通过I，X，D的组合，可构成如下寻址方式：

寻址方式	I	X	有效地址E算法	说明
(1)	0	00	E=D	
(2)	0	01	E=(PC)±D	PC为程序计数器
(3)	0	10	E=(R_2)±D	R_2为变址寄存器
(4)	1	11	E=(R_3)	
(5)	1	00	E=(D)	
(6)	0	11	E=(R_1)±D	R_1为基址寄存器

请写出6种寻址方式的名称。

4.4.3 寻址方式举例

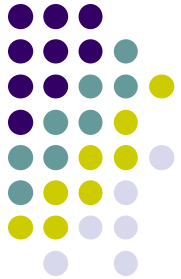
[例5] 将ARM汇编语言翻译成机器语言。已知5条ARM指令格式译码如下表所示：

指令名称	cond	F	I	opcode	S	Rn	Rd	operand 2
ADD(加)	14	0	0	4	0	reg	reg	reg
SUB(减)	14	0	0	2	0	reg	reg	reg
ADD(立即数加)	14	0	1	4	0	reg	reg	constant(12位)
LOR(取字)	14	1	–	24	–	reg	reg	address(12位)
STR(存字)	14	1	–	25	–	reg	reg	address(12位)

设r3寄存器中保存数组A的基址，h放在寄存器r2中。C语言程序语句A[30]=h+A[30] 可编译成如下3条汇编语言指令：

```
LDR r5,[r3,#120]      ;寄存器r5中获得A[30]
ADD r5,r2,r5          ;寄存器r5中获得h+A[30]
STR r5,[r3,#120]      ;将h+A[30]存入到A[30]
```

请问这3条汇编语言指令的机器语言是什么？



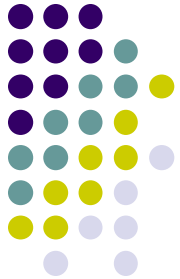
4.5 典型指令

4.5.1 指令的分类
4.5.2 基本指令系统的操作
4.5.3 精简指令系统

4.5.1 指令的分类

- 数据传送类指令
  - 一般传送指令： MOV AX，BX
  - 数据交换指令： XCHG
  - 堆栈操作指令： PUSH，POP
- 运算类指令
  - 算术运算指令： 加、减、乘、除以及加1、减1、比较
  - 逻辑运算指令：
  - 移位指令
- 程序控制类指令
  - 程序控制类指令用于控制程序的执行方向，并使程序具有测试、分析与判断的能力。
- 输入和输出指令、字符串处理指令、特权指令、其他指令

4.5.2 基本指令系统的操作



- 20%和80%规律：CISC中大约有20%的指令使用频率高，占据了80%的处理机时间，而有80%的不常用指令只占用处理机的20%时间。
- VLSI技术发展引起的问题
  - VLSI工艺要求规整性，而大量复杂指令控制逻辑极其不规整，给VLSI工艺造成了很大的困难。
  - 现在用微程序实现复杂指令与用简单指令组成的子程序相比，没有多大的区别。因为现在控制存储器和主存的速度差缩小。
- CISC中，通过增强指令系统的功能，简化了软件，增加了硬件的复杂程度。然而指令复杂了，指令的执行时间必然加长，从而使整个系统的执行时间反而增加，因而在计算机体系结构设计中，软硬件的功能分配必须恰当

4.5.3 精简指令系统

- 特点（采用流水线技术）
  - 简单而统一格式的指令译码；
  - 大部分指令可以单周期执行
  - 只有LOAD/STORE可以访问存储器
  - 简单的寻址方式
  - 采用延迟转移技术
  - 采用LOAD延迟技术
  - 三地址指令格式
  - 较多的寄存器
  - 对称的指令格式

4.6 ARM汇编语言

汇编语言是计算机机器语言（二进制指令代码）进行符号化的一种表示方法，每一个基本汇编语句对应一条机器指令。

表4.11列出了嵌入式处理机ARM的汇编语言。其中操作数使用16个寄存器（r0 , r1～r12 , sp , lr , pc ）,2^{30}个存储字（字节编址，连续的字的地址间相差4）。

表4.11(上)

表4.11(下)

返回

4.6 ARM汇编语言

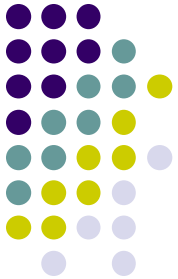
在进行汇编语言程序设计时，可直接使用英文单词或其缩写表示指令，使用标识表示数据或地址，从而有效地避免了记忆二进制的指令代码。

不用由程序设计人员对指令和数据分配内存地址，直接调用操作系统的某些程序段完成输入输出。

用编辑程序建立好的汇编语言源程序，需要经过系统软件中的“汇编器”翻译为机器语言程序之后，才能交付给计算机硬件系统去执行。

本章小结

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，它的格式与功能不仅直接影响到机器的硬件结构，而且也影响到系统软件。指令格式是指令字用二进制代码表示的结构形式，通常由操作码字段和地址码字段组成。



返回

本 章 小 结

操作码字段表征指令的操作特性与功能，而地址码字段指示操作数的地址。目前多采用二地址、单地址、零地址混合方式的指令格式。指令字长度分为：单字长、半字长、双字长三种形式。高档微机采用32位长度的单字长形式。

本 章 小 结

形成指令地址的方式，称为指令寻址方式。有顺序寻址和跳跃寻址两种，由指令计数器来跟踪。形成操作数地址的方式，称为数据寻址方式。操作数可放在专用寄存器、通用寄存器、内存和指令中。数据寻址方式有隐含寻址、立即寻址、直接寻址、间接寻址、寄存器寻址、寄存器间接寻址、相对寻址、基值寻址、变址寻址、块寻址、段寻址等多种。按操作数的物理位置不同，有RR型和RS型。前者比后者执行的速度快。

本 章 小 结

按结构不同，分为寄存器堆栈和存储器堆栈。不同机器有不同的指令系统。一个较完善的指令系统应当包含数据传送类指令、算术运算类指令、逻辑运算类指令、程序控制类指令、I/O类指令、字符串类指令、系统控制类指令。RISC指令系统是目前计算机发展的主流，也是CISC指令系统的改进，它的最大特点是：①指令条数少；②指令长度固定，指令格式和寻址方式种类少；③只有取数/存数指令访问存储器，其余指令的操作均在寄存器之间进行。

本 章 小 结

汇编语言与具体机器的依赖性很强。为了解该语言的特点，列出了目前流行的嵌入式处理机ARM的汇编语言。