第6章 指令系统 作业参考题解

6.1 指令包括哪几部分？各表示什么含意？

参考答案：

- 指令包括操作码和地址码两部分。
- 操作码：用来指明该指令所要完成的操作，即定义指令的功能。
- 地址码：用来寻找执行指令所需要的操作数，即操作数的地址信息。

6.2 在一地址指令、二地址指令中，如何指定二个操作数地址？如何存放操作结果？

参考答案：

- 一地址指令：由指令中的地址码提供源操作数，另一操作数隐含指定（一般指累加器 ACC）；操作结果也存放在隐含规定的寄存器（一般指累加器 ACC）中，即目的操作数隐含指定，源操作数由指令中的地址码指定。
- 二地址指令：由指令的两个地址码分别指定两个操作数；操作结果也存入其中一个地址码指定的操作数（目的操作数）。

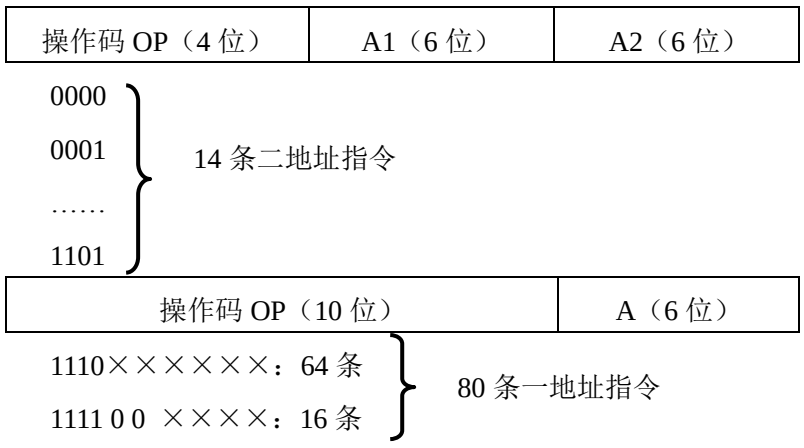
6.3 简述指令操作码的扩展技术的基本方法。

参考答案：

指令系统既有单地址指令，也有双地址指令或无地址指令。操作码扩展技术的基本方法就是将指令中不使用的地址码扩展为操作码，操作码的长度随地址数的减少而增加，即不同地址数的指令可以具有不同长度的操作码，从而有效地缩短指令字长。

6.4 某机器字长 16 位，采用单字长指令，每个地址码 6 位。试采用操作码扩展技术，设计 14 条二地址指令，80 条一地址指令，60 条零地址指令。请给出指令编码示意图。

参考答案：不唯一，其他不冲突和重复的编码方式也是正确的。



操作码 OP（16 位）

1111 111111 000000

1111 111111 000001

……

1111 111111 111011

} 60 条零地址指令

6.5　什么是指令字长？什么是机器字长？它们之间有何关系？

参考答案：

- 指令字长是指一条机器指令的位数，不同机器的指令字长是不相同的；一般它主要取决于操作码的长度、操作数地址的长度和操作数地址的个数。
- 机器字长是指 CPU 一次能处理的数据位数，它决定了寄存器、运算部件、数据总线的位数，机器字长一般是字节长度的整数倍，即 8、16、32 或 64 位。
- 指令字长一般与机器字长有着简单而不固定的对应关系：指令字长可以等于机器字长；指令字长可以小于或等于机器字长，这种指令称为短格式指令；指令字长也可以大于机器字长（譬如双字指令），这类指令称为长格式指令。

6.6　确定寻址方式的目的是什么？

参考答案：

确定寻址方式的目的是：确定本条指令的操作数地址，以及下一条将要执行的指令地址，即为了找到操作数和下条指令。

6.7　请说明间接寻址和直接寻址的不同。

参考答案：

间接寻址和直接寻址方式的操作数均存放在存储器中；但是直接寻址的操作数的 EA 由指令的地址码字段指出，而间接寻址的操作数的 EA 存放在另一个存储单元，该单元的地址由指令的地址码字段指出。即直接寻址方式下，用指令的地址码字段访问一次存储器就得到操作数，而对于间接寻址，用指令的地址码字段访问两次存储器才得到操作数，见图 6.1。

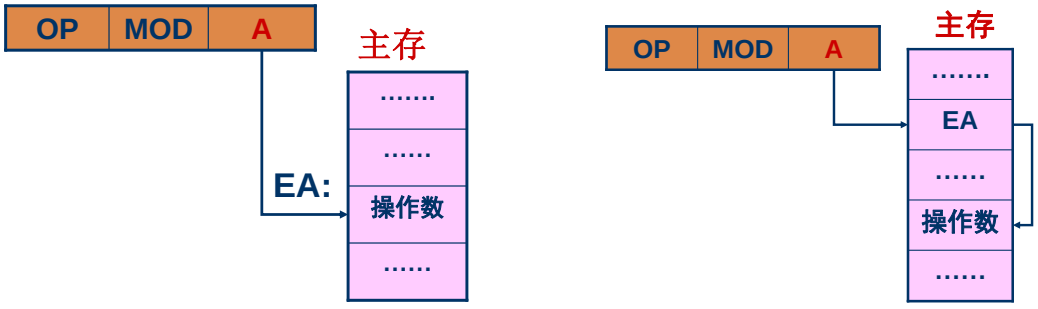


图 6.1　间接寻址和直接寻址

6.8　简述变址寻址和基址寻址的主要区别。

参考答案：

变址寻址和基址寻址的有效地址 EA 均是由一个寄存器（变址寄存器和基址寄存器）加上指令的地址码字段得到的，但是应用的场合不同和使用方法不同。

变址寻址主要用于实现程序块的规律变化。通常指令中的形式地址 A 作为基准地址，而变址寄存器的内容作为修改量。在应用场合上，变址寻址面向用户编程，特别适合于对字符串处理、数组运算等成组数据处理。

基址寻址适合于多用户计算机系统，当操作系统为多道程序分配主存空间，将用户程序装入主存时，给每个用户一个基地址并放入其相应的基址寄存器，在程序执行时，以基址为基准自动进行地址的变换。基址寄存器的内容是基准量，指令中的地址码字段（形式地址）是位移量，因此其位数较短。在应用场合上，基址寻址面向系统，可以用来解决程序在主存中的重定位和扩大寻址空间等问题。

6.9　选择题：

（1） 寄存器间接寻址方式中，操作数在 D 中。

A. 程序计算器　　B. 堆栈　　C. 寄存器　　D. 主存

（2） 堆栈常用于 B、C 。

A. 数据移位　　B. 程序转移　　C. 保护程序现场　　D. 输入、输出

（3） 单地址指令中，为了完成两个数的算术运算，除地址码指明的一个操作数外，另一个数常需采用 C 。

A．堆栈寻址　　B. 立即寻址　　C. 隐含寻址　　D. 间接寻址

（4） 用于对某个寄存器中操作数的寻址方式称为 C 。

A．直接寻址　　B. 间接寻址　　C. 寄存器直接寻址　　D. 寄存器间接寻址

（5） 指令中采用不同的寻址方式，其主要目的是 C 。

A. 可以实现操作码的扩展　　B. 实现存储程序和程序控制

C. 缩短指令长度、扩大寻址空间、提高编程的灵活性。

D. 降低指令译码的难度。

（6） 指令寻址和数据寻址的不同在于 C 。

A. 前者是访问存储器，后者是访问寄存器。

B. 前者是确定程序转移地址，后者取操作数。

C. 前者是确定程序执行顺序，后者是取操作数地址。

D. 前者是短指令，后者是长指令。

（7） 变址寻址方式中，操作数的有效地址为 C 。

A. 程序计数器的内容加上形式地址。

B. 基址寄存器的内容加上形式地址。

C. 变址寄存器的内容加上形式地址。

D. 变址寄存器的内容加上基址寄存器的内容。

（8） CISC 指令系统与 RISC 指令系统相比具有 B 等特点。

A. 前者指令条数少，后者指令条数多。

B. 前者执行速度慢，后者执行速度快。

C. 前者有利于编译生成优化代码，后者不便于编译。

D. 前者指令功能简单，后者指令功能复杂。

6.10 某机 16 位字长指令格式如下：

OP	M	D
5 位	3 位	8 位

其中：D 是形式地址，采用补码表示（包括一位符号位）；

M 是寻址方式 M=0 立即寻址；

M=1 直接寻址（这时 D 为地址，是无符号数）；

M=2 间接寻址；

M=3 变址寻址（变址寄存器 RI，16 位）；

M=4 基址寻址（基址寄存器 Rb，16 位）；

M=5 相对寻址。

（1）该指令格式最多可以定义多少种不同的指令？立即寻址操作数范围是多少？

（2）写出各种寻址方式的有效地址的计算表达式。

（3）各种寻址方式时能访问的最大主存空间范围是多少？

参考答案：

（1）该指令格式最多可以定义 **2^5=32** 种不同的操作，立即寻址操作数范围是 -128~+127 。（8 位补码的表示范围）

（2）寻址方式的有效地址：

- M=0 立即寻址：无有效地址
- M=1 直接寻址（这时 D 为地址，是无符号数）：EA=D
- M=2 间接寻址：EA=（D）
- M=3 变址寻址（变址寄存器 RI，16 位）：EA=（RI）+D
- M=4 基址寻址（基址寄存器 Rb，16 位）：EA=（Rb）+D
- M=5 相对寻址：EA=（PC）+D

（3）访问的最大主存空间：

- M=0 立即寻址：无
- M=1 直接寻址（这时 D 为地址，是无符号数）：EA=D，0~255

- M=2 间接寻址：EA=（D），主存单元是 16 位的，0~2^{16}-1，即 0~65535
- M=3 变址寻址（变址寄存器 RI，16 位）：EA=（RI）+D

  （RI）：0~2^{16}-1，D：0~255，所以：**0~2^{16}+254=65790**
- M=4 基址寻址（基址寄存器 Rb，16 位）：EA=（Rb）+D

  （Rb）：0~2^{16}-1，D：0~255，所以：**0~2^{16}+254=65790**
- M=5 相对寻址：EA=（PC）+D

  （PC）：0~2^{16}-1，D：-128~127，所以：-128~2^{16}+126=65662

  由于地址是无符号数，不可能为负，所以：**0~65662**

6.11 一个较完整的指令系统应该包括哪些类型的指令？

参考答案：

一个较完整的指令系统应该包括数据传送类指令、算术逻辑运算指令、程序控制类指令、移位操作指令、堆栈操作指令、输入输出指令、处理器控制类指令、特权指令等等。

6.12 假设相对寻址的转移指令占两个字节，第一个字节是操作码和寻址方式，第二个字节是相对偏移量，用补码表示。若当前转移指令的第一字节所在地址为 0019H，且 CPU 每取出一个字节指令便会自动执行（PC）+1→PC 操作。请问：若当前指令分别是相对转移指令 JMP 0006H 和 JMP 0025H 时，转移指令第二字节的内容是什么？

参考答案：

EA＝PC＋Disp，因为 PC＝001BH，所以：

- 对于 JMP 0006H：EA＝0006H，Disp＝EA－PC＝0FFEBH，8 位的偏移量：0EBH
- 对于 JMP 0025H：EA＝0025H，Disp＝EA－PC＝000AH，8 位的偏移量：0AH

6.13 某机器内共有 16 个 32 位的通用寄存器，设计一种有 60 种操作，8 种寻址方式的指令系统。假设指令字长等于机器字长（32 位），请回答：

（1）若主存可直接寻址或间接寻址，存储器字长 32 位，采用“寄存器－存储器”型指令，能寻址最大存储空间是多少？画出指令格式并说明各字段的含意。

（2）若采用通用寄存器作基址寄存器，则“寄存器－存储器”型指令的指令格式是怎样？能寻址最大存储空间是多少？

参考答案：

（1）“寄存器－存储器”型指令格式：

操作码	寻址方式码	寄存器号	地址码
（6 位）	（3 位）	（4 位）	（19 位）

若采用直接寻址，则能寻址最大存储空间是 0～2^{19}－1。

若采用间接寻址，则能寻址最大存储空间是 0～2^{32}－1。

（2）若基址寄存器号在指令中指出，则指令格式为：

操作码	寻址方式码	寄存器号	基址寄存器号	地址 ADDR
（6 位）	（3 位）	（4 位）	（4 位）	（15 位）

EA＝R_B＋ADDR，R_B 的取值 0～2^{32}－1，ADDR 有 15 位，取值 0～2^{15}－1

则 EA 能寻址最大存储空间是 0～2^{32}＋2^{15}－2。

◆ 若基址寄存器号由操作码或寻址方式码隐含指出，则指令格式为：

操作码	寻址方式码	寄存器号	地址 ADDR
（6 位）	（3 位）	（4 位）	（19 位）

EA＝R_B＋ADDR，R_B 的取值 0～2^{32}－1，ADDR 有 19 位，取值 0～2^{19}－1

则 EA 能寻址最大存储空间是 0～2^{32}＋2^{19}－2。

6.14 什么叫堆栈？它的操作特点是什么？堆栈主要用在哪里？

参考答案：

堆栈：是由若干个连续主存单元组成的先进后出（First in Last out，即 FILO）存储区，第一个放入堆栈的数据存放在栈底，最近放入的数据存放在栈顶。栈底是固定不变的，而栈顶是随着数据的入栈和出栈在时刻变化。栈顶由堆栈指针（Stack Pointer，,即 SP）指向。

堆栈的操作特点是：堆栈只有两种操作——压入 PUSH 和弹出 POP，从堆栈栈顶读出一个数据叫弹出，把操作结果写入堆栈栈顶单元叫压入；这两种操作都只能通过 SP 堆栈指针访问实现。

堆栈主要用来暂存中断和子程序调用时的现场数据及返回地址。

6.15 简述 RISC 的主要优缺点。

参考答案：

1. 指令系统中的指令简单、使用频率较高。
2. 指令长度固定，指令格式种类少，寻址方式种类少。
3. 只有取数和存数指令访问存储器，其余指令的操作都在寄存器之间进行。
4. 采用流水线技术，使得一条指令的平均指令执行时间小于一个机器周期。
5. CPU 中设置大量通用寄存器，以减少访存次数。
6. 以硬布线控制逻辑为主，不用或少用微码控制。
7. 采用优化的编译程序，能有效地支持高级语言程序。

6.16 设某机寄存器字长 16 位，用 16 进制表示，已知：变址寄存器内容为 0004H，PC 的内容为 0003H，内存中部分单元内容如下：

地址：内容	地址：内容
0002H：000AH	0007H：000AH

0003H：0002H　　0008H：0002H

0004H：0007H　　0009H：0003H

0005H：0004H　　000AH：0009H

0006H：0005H　　000BH：0008H

指令为双字长指令，格式如下：

操作码，寻址方式码，寄存器号(16 位)
直接地址/间接地址/立即数/相对位移量/形式地址(16 位)

若**当前指令**分别为下列寻址方式时，试求出操作数填入下表。

参考答案：

寻址方式	操作数
直接	**000AH**
间接	**0009H**
立即	**0007H**
变址	**0008H**

6.17 试从指令格式、寻址方式和每条指令的周期数（CPI）等方面比较 RSIC 和 CSIC 的差异。

参考答案：

RISC 的指令格式种类少、指令字长固定，寻址方式简单且少，CPI 一般=1；

CISC 的指令格式种类多、指令字长可变，寻址方式丰富，CPI 一般≥1；

6.18 乘法运算可以用加法指令和移位指令来实现，在不考虑溢出的情况下，若要将$S0 的内容与 6 相乘，乘积放入$S1 中，请写出一段指令条数最少，且不包括乘法指令的 MIPS 代码。

参考答案：

$6x = 4x + 2x$,2x 和 4x 通过移位得到，因此 MIPS 代码可以写成：

```
sll $s1,$s0,2        #将 $s0 左移 2 位，送入$s1
sll $s0,$s0,1        #将 $s0 左移 1 位，送入$s0
addu  $s1,$s1,$s0    #将 $s0 和$s1 相加（不考虑溢出），结果送入$s1
```

6.19 以下程序段是某个过程对应的 MIPS 指令序列，其功能是复制一个存储块数据到另一个存储块中，存储块中每个数据的类型为 float，源数据块和目的数据块的首地址分别存放到$a0 和$a1 中，复制的数据个数存放在$v0 中，作为返回参数返回给调用过程。在复制过程中遇到 0 则停止，最后一个 0 也需要复制，但不被计数。已知该程序段中有多个 bug，请找出它们并改正。

```
        addi $v0, $zero, 0
loop:   lw $v1,0($a0)
        sw $v1,0($a1)
        addi $a0, $a0, 4
        addi $a1, $a1, 4
        beq $v1, $zero, loop
```

参考答案：

修改后的代码：

```
        addi $v0,$zero,0
loop:   lw $v1,0($a0)
        sw $v1,0($a1)
        beq $v1,$zero,exit      #红色是不同之处，即修改 BUG
        addi $a0,$a0,4
        addi $a1,$a1,4
        addi $v0,$v0,1
        j loop
exit:
```

6.20 以下 C 语言程序段：

```
for (i=0;i<=100;i=i+1)
a[i]=b[i]+c;
```

假如 a 和 b 的每个元素都是 int 型变量，首地址分别存放在寄存器$a0 和$a1 中，寄存器$t0 和$s0 分别对应变量 i 和 c。要求写出与之对应的 MIPS 指令代码，并计算这段代码运行过程中所执行的指令条数和数据的访存次数。

参考答案：

每个数组元素占 4 字节，所以循环体内每一步地址增量是 4，代码如下：

```
        add $t0,$zero,$zero     #i=0
loop:   add $t4,$a1,$t0         #$t4=address of b[i]
        lw $t5,0($t4)           #$t5=b[i]
        add $t6,$t5,$s0             #$t6=b[i]+c
        add $t7,$a0,$t0         #$t7=address of a[i]
```

```
sw $t6,0($t7)          #a[i]=b[i]+c
addi $t0,$t0,4         #i=i+4
slti $t8,$t0,401           #if(i<401) then $t8=1 else $t8=0
bne $t8,$zero,loop     #if($t8=1) goto loop
```

6.21 某高级语言源程序中的语句“while(save[i]==k)i+=1;”若对其编译时，编译器将 i 和 k 分别分配在寄存器$s3 和$s5 中，数组 save 的基址存放在$s6 中，则生成的 MIPS 汇编代码段如下：

```
loop:   sll $t1, $s3, 2      #R[$t1]←R[$s3]<<2,即 R[$t1]=i×4
        add $t1, $t1, $s6    #R[$t1]←R[$t1]+R[$s6],即 R[$t1]=Address of save[i]
        lw $t0, 0($t1)       #R[$t0]←M[R[$t1]+0],即 R[$t0]=save[i]
        bne $t0, $s5, exit   #if R[$t0]≠R[$s5] then goto exit
        addi $s3, $s3, 1     #R[$s3]←R[$s3]+1,即 i=i+1
        j   loop                 #goto loop
exit:
```

假设从 loop 处开始的指令序列存放在内存 80000 处，则上述循环对应的 MIPS 机器码如表 6.7 所示。

表 6.7 程序中的机器级代码

	6 位	5 位	5 位	5 位	5 位	6 位
80000	0	0	19	9	2	0
80004	0	9	22	9	0	32
80008	35	9	8	0		
80012	5	8	21	2		
80016	8	19	21	1		
80020	2	20000				
80024	...					

根据以上叙述，请回答下列问题，要求说明理由或给出计算过程。

（1） MIPS 的编址单位是多少？数组 save 每个元素占几个字节？

MIPS 编址单位是字节，因为从表 6.7 看出，指令为 32 位，但是后继指令地址+4，所以以字节为单位编址；

每次循环取数组元素，下标地址都(<<2)即乘以 4，所以 save 数组每个元素占 4 个字节。

（2） 为什么指令“sll $t1, $s3, 2”能实现 4×i 的功能？

Sll 是左移指令，<<2 即乘以 4。

（3） 该循环指令序列中哪些是 R 型指令？哪些是 I 型指令？

从表 6.7 看，第 1,2 条是 R 型指令，第 3,4,5 条是 I 型指令.

（4） $t0 和$s6 的编号各是多少？

从第 3、4 条指令看，$t0 的编号是 8，$s6 编号是 22.

（5） 指令“j loop”的操作码是什么（用二进制表示）？

二进制操作码是 000010B

（6） 标号 exit 的值是多少？如何根据指令计算得到？

标号 exit 的值是 80024，如果根据 bne 指令计算，公式是 PC+4+offset*4=80012+4+2*4=80024，2 是相对位移量。

（7） 标号 loop 的值是多少？如何根据指令计算得到？MIPS 中跳转指令的跳转范围是多少？

由程序可看出，标号 loop 的值是 80000；

如果由跳转指令 J loop 计算，则 80020 的高 4 位（0000B)，与指令低 26 位（20000）拼接成 30 位地址，然后再在低位补 00，相当于乘以 4，即 20000*4=80000；

MIPS 中跳转指令的转移目标地址等于 PC 高四位和 26 位 address 及 2’b00 拼接而成，可见，转移目标地址的高四位保持与 PC 一致，低 28 位的寻址范围是 2^{28}，即 0~256M-1，所以跳转目标地址范围的大小是 256M，即假定跳转指令的高 4 位为 X，则跳转目标地址范围是 X0000000H~XFFFFFFCH。