第五章 指令系统

2. 简单回答下列问题。**（参考答案略）**

（1）一条指令中应该明显或隐含地给出哪些信息？

（2）什么是“汇编”过程？什么是“反汇编”过程？这两个操作都需要用到什么信息？

（3）CPU 如何确定指令中各个操作数的类型、长度以及所在地址？

（4）哪些寻址方式下的操作数在寄存器中？哪些寻址方式下的操作数在存储器中？

（5）基址寻址方式和变址寻址方式的作用各是什么？有何相同点和不同点？

（6）为何分支指令的转移目标地址通常用相对寻址方式？

（7）RSIC 处理器的特点有哪些？

（8）CPU 中标志寄存器的功能是什么？有哪几种基本标志？

（9）转移指令和转子（调用）指令的区别是什么？返回指令是否需要有地址码字段？

3. 假定某计算机中有一条转移指令，采用相对寻址方式，共占两个字节，第一字节是操作码，第二字节是相对位移量（用补码表示），CPU 每次从内存只能取一个字节。假设执行到某转移指令时 PC 的内容为 200，执行该转移指令后要求转移到 100 开始的一段程序执行，则该转移指令第二字节的内容应该是多少？

参考答案：

100=200+2+Offset，Offset=100-202=-102=10011010B

（注：没有说定长指令字，所以不一定是每条指令占 2 个字节。）

5. 某计算机字长 16 位，每次存储器访问宽度 16 位，CPU 中有 8 个 16 位通用寄存器。现为该机设计指令系统，要求指令长度为字长的整数倍，至多支持 64 种不同操作，每个操作数都支持 4 种寻址方式：立即（I）、寄存器直接（R）、寄存器间接（S）和变址（X），存储器地址位数和立即数均为 16 位，任何一个通用寄存器都可作变址寄存器，支持以下 7 种二地址指令格式：RR 型、RI 型、RS 型、RX 型、XI 型、SI 型、SS 型。请设计该指令系统的 7 种指令格式，给出每种格式的指令长度、各字段所占位数和含义，并说明每种格式指令需要几次存储器访问？

参考答案：

指令格式可以有很多种，只要满足以下的要求即可。

操作码字段：6 位

寄存器编号：3 位

直接地址和立即数：16 位

变址寄存器编号：3 位

总位数是 8 的倍数

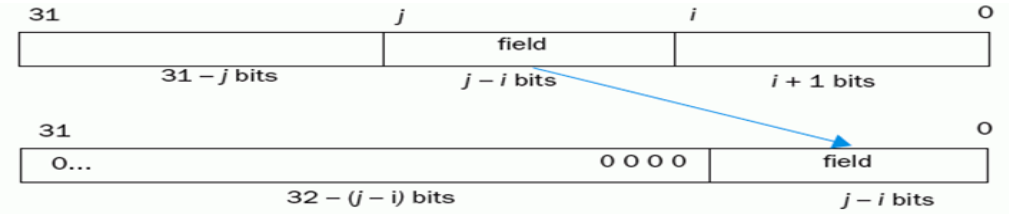
指令格式例 1：（略）

指令格式例 2：（略）

寻址方式字段(2 位)----00：立即；01：寄直；10：寄间；11-变址

6. 有些计算机提供了专门的指令，能从 32 位寄存器中抽取其中任意一个位串置于一个寄存器的低位有效位上，如下图所示。MIPS 指令系统中没有这样的指令，请写出最短的一个 MIPS 指

令序列来实现这个功能，要求 i=5, j=22, 操作前后的寄存器分别为$s0 和$s2。



参考答案：

可以先左移 9 位，然后右移 15 位：

```
sll $s2, $s0, 9
srl $s2, $s2, 15        （sra 算术右移 P.57 表 3.1）
```

若第一条指令中的$s2 改成其他寄存器，则会带来什么问题？

所用寄存器的值被破坏！

7. 以下程序段是某个过程对应的指令序列。入口参数 int a 和 int b 分别置于$a0 和$a1 中，返回参数是该过程的结果，置于$v0 中。要求为以下 MIPS 指令序列加注释，并简单说明该过程的功能。

```
                add   $t0, $zero, $zero
loop:           beq   $a1, $zero, finish
                add   $t0, $t0, $a0
                sub   $a1, $a1, 1
                j     loop
finish:         addi  $t0, $t0, 100
                add   $v0, $t0, $zero
```

参考答案略

9. 用一条 MIPS 指令或最短的指令序列实现以下 C 语言语句：b=25|a。假定编译器将 a 和 b 分别分配到$t0 和$t1 中。

参考答案：ori $t1, $t0, 25

如果把 25 换成 65536，那指令是不是就换成： ori $t1, $t0, 65536 ？

65536（1 0000 0000 0000 0000）不能用 16 位立即数表示，所以不对！

10. 以下程序段是某个过程对应的 MIPS 指令序列，其功能为复制一个存储块数据到另一个存储块中，源数据块和目的数据块的首地址分别存放在$a0 和$a1 中，复制的数据个数存放在$v0 中返回。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。已知程序段中有多个 Bug，请找出它们并修改。

```
      addi $v0, $zero, 0 # Initialize count
loop: lw   $v1, 0($a0)   # Read next word from source
      sw   $v1, 0($a1)   # Write to destination
      addi $a0, $a0, 4   # Advance pointer to next source
      addi $a1, $a1, 4   # Advance pointer to next destination
      beq  $v1, $zero, loop  # Loop if word copied != zero
```

参考答案：

修改后的代码如下：

```
            addi  $v0, $zero, 0
loop:       lw $v1, 0($a0)
            sw $v1, 0($a1)
            beq $v1, $zero, exit
            addi $a0, $a0, 4
            addi $a1, $a1, 4
            addi $v0, $v0, 1
            j   loop
exit:
```

11. 说明 beq 指令的含义，并解释为什么汇编程序在对下列汇编源程序中的 beq 指令进行汇编时会遇到问题，应该如何修改该程序段？

```
here:          beq   $s0, $s2, there
…
there          add   $s0, $s0, $s0
```

参考答案：

beq 是一个 I-Type 指令，可以跳转到当前指令前，也可以跳转到当前指令后。其计算公式为：PC+4+offset（16 位立即数），故 offset 是一个 16 位带符号整数（4 的倍数，用补码表示）。

其正跳范围为：0000 0000 0000 0100（+4）~ 0111 1111 1111 1100（$+2^{15}-4$）

负跳范围为：1000 0000 0000 0000（-2^{15}）~ 1111 1111 1111 1100（- 4）

超过以上范围的跳转就不能用上述指令序列实现。应该改成以下序列：

```
here:     bne $s0, $s2, skip
          j   there
skip:     ……
          ……
there:    add $s0, $s0, $s0
```

12. 以下 C 语言程序段中有两个函数 sum_array 和 compare，假定 sum_array 函数第一个被调用，全局变量 sum 分配在寄存器$s0 中。要求写出每个函数对应的 MIPS 汇编表示，并画出每个函数调用前、后栈中的状态、帧指针和栈指针的位置。

```
1   int sum=0;
2   int sum_array(int num)
3   {
```

```
4           int i，array[10];
5           for (i = 0; i < 10; i ++)
6                 if compare (num, i)    sum+=arrar[i] ;
7           return sum；
8     }
9     int compare (int a, int b)
10    {
11          if ( a >=b)
12                return 1;
13          else
14                return 0;
15    }
```

参考答案：（略）