1.4 衡量微机系统性能的主要指标有哪些?若某微机系统的运算速度为 100 MIPS，试问其平均指令周期为多长?

解:衡量微机系统性能的主要指标有字长、存储器容量、运算速度、外设扩展能力和软件配足情况。

平均指令周期=1/平均指令执行速度

= 1/(100 MIPS)

= 1/(1 00 x 10^6)，

= 0.01 μs

1.8 试填写出表 1.2 中各数对应的 8 位原码、反码、补码。

解:见表 1.2

十进制数	原码	反码	补码
+0	00000000	00000000	00000000
+15	00001111	00001111	00001111
+62	00111110	00111110	00111110
+127	01111111	01111111	01111111
-0	10000000	11111111	00000000
-10	10001010	11110101	11110110
-18	10010010	11101101	11101110
-100	11100100	10011011	10000001
-127	11111111	10000000	10000001
-128	无	无	10000000

1.9 请将下列各 8 位补码扩展为 16 位补码，并从中总结出补码位扩展的规律。

(1)00101101B (2)01110010B (3) 01010111B

(4) 10110101B (5) 10000100B (6) 11001011B

解:先求出 8 位补码对应的真值，再用 16 位补码表示，结果如表 1.3 所示。正数是用 0 充填扩充的高位。而负数是用 1 充填扩充的高位，由此可以总结得出补码位扩展的规律:用符号位充填扩展的高位。

表 1.3 将 8 位补码扩充为 16 为补码

8 位补码		16 位补码
00101101	45	00000000 00101101
01110000	114	00000000 01110010

8 位补码	真值	16 位补码
01010111	87	00000000 01010111
10110101	-75	11111111 10110101
10000100	-124	11111111 10000100
11001011	-53	11111111 11001011

1.10 填写表 1.4 中各机器码分别为原码、反码、补码、无符号二进制码和压缩 BCD 码时对应的十进制真值。

解:见表 1.4

表 1.4 机器数对应的十进制真值

机器数	对应的十进制真值				
	原码	反码	补码	无符号二进制数	压缩 B 码 CD
10001001	-9	-118	-119	137	89
10000000	-0	-127	-128	128	80
10010110	-22	-105	-106	150	96
01010011	83	83	83	83	53
00000111	7	7	7	7	7
11111111	-127	-0	-1	255	无效

1.12 已知某徽机的浮点数格式为:阶码 6 位.尾数 10 位.其中各含 1 位符号位;阶码为补码定点整数，尾数为原码规格化小数。

(1)该浮点数所能表示的数值范围是多少?

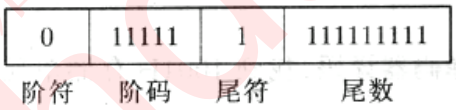
(2) 1100100110111010 代表的十进制数是多少?

(3)$(-86.57)_{10}$的机器数(浮点数)是什么?

解:(1)该浮点数所能表示的最大数是:阶码正最大(补码)，尾数正最大(原码)》，即如下数:

此时，阶码真值=2^5一 1=31，所以，最大真值=十$(1-2^{-9})*2^{31}$，

该浮点数所能表示的最小数为:阶码正最大(补码)。尾数负最大(原码)，即如下数;



此时，阶码真值=2^5一 1=31，所以，最大真值=+$(1-2^{-9})$ ×2^{31}

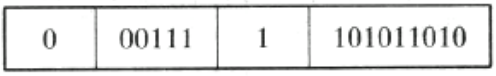
所以，该浮点数所能表示的数值范围为-$(1-2^{-9})$ ×2^{31}~~十$(1-2^{-9})$ ×2^{31}

（2）阶码=-14

十进制数=（0.11011101）$_2$×2^{-14}=（11011101）$_2$=221×2^{-22}=5.269×10^{-5}

（3）（-86.57）$_{10}$ ^{4}=（-1010110.100100011）$_2$≈（-0.101011010）$_2$×2^7

机器数为:



1.18“微型计算机中，程序执行的时间就是程序中各条指令执行时间的总和。”这种说法是否一定对?为什么?试谈谈你的理解。

解:这种说法不一定对。

在采用流水线结构的微型计算机中，由于不同指令的取指、分析和执行三个阶段可并行处理,相当于不同指令的执行时间被流水线所吸收,从而从整体上使程序的执行时间减少了,即程序执行的时间不是程序中各条指令执行时间的总和。

1.20 微处理器中采用流水线技术后，是否意味着每条指令的执行时间明显缩短了?为什么?

解:不能缩短每条指令的执行时间。

采用流水线技术后，并没有加速单条指令的执行，每条指令的操作步骤(取指、分析、执行)一个也不能少，只是多条指令的不同操作步骤可由多个独立的功能部件同时执行，因而从总体看加快了指令流速度，缩短了程序执行时间。

1.21 请自行写出主教材第 1.4.5 节中给出的程序的详细执行步骤。

解:MOV A,SCH 详细执行步骤如下:

(1)将 PC 内容 1000H 送地址寄存器 MAR o

(2) PC 值自动加 1，为取下一个字节机器码作准备。

(3) MAR 中内容经地址译码器译码，找到内存储器 1000H 单元。

(4) CPU 发读命令。

(5)将 1000H 单元内容 BOH 读出，送至数据寄存器 MDR o

(6)由于 BOH 是操作码，故将它从 MDR 中经内部总线送至指令寄存器 IRo

(7)经指令译码器 ID 译码，由操作控制器 OC 发出对应于操作码的控制信号。

;;;以下取操作数

(8)将 PC 内容 1001H 送 MAR o

(9) PC 值自动加 1。

(10) MAR 中内容经地址译码器译码，找到 1001iH 存储单元。

(11) CPU 发读命令。

(12)将 1001H 单元内容 5CH 读至 MDR

;;;以下执行指令

(13)因 5CH 是操作数，将它经内部总线送至操作码规定好的累加器 A。

ADD A，2EH 详细执行步骤如下:

(1) 将 PC 内容 1002 H 送地址寄存器 MAR 。

(2) PC 值自动加 1.为取下一个字节机器码作准备。

(3) MAR 中内容经地址译码器泽码，找到内存储器 1002H 单元;

(4) CPU 发读命令。

(5)将 1002H 单元内容 04 H 读出，送至数据寄存器 MDR o

(5)由于 1002H 是操作码，故将它从 MDR 中经内部总线送至指令寄存器 IRo

(7) 经指令译码器 1D 译码，由操作控制器 OC 发出对应于操作码的控制信号。

(8)将 YC 内容 1003H 送 MAR

(9)PC 值自动加 1

(10) MAR 中内容经地址译码器译码。找到 1003H 存储单元。

(11)CPU 发读命令。

(12)将 1003H 单元内容 2EH 读至 MDR

(13)因 2EH 是操作数，将它经内部总线送至 ALU。

(14)执行:2.EH+5CH 送 A 累加器，修改标志寄存器。

JO 100AH 详细执行步骤如下:

(1)将 PC 内容 1004H 送地址寄存器 MAR o

(2) PC 值自动加 1，为取下一个字节机器码作准备。

(3) MAR 中内容经地址译码器译码。找到内存储器 1004H 单元。

(4) CPU 发读命令。

(5)将 1004H 单元内容 70 H 读出，送至数据寄存器 MDR

(6)由于 70H 是操作码。故将它从 MaR 中经内部总线送至指令寄存器 IR}

(7)经指令译码器 ID 译码，由操作控制器 OC 发出对应于操作码的控制信号。

;“以下取操作数
(8)将 PC 内容 1005H 送 MAR o
(9) PC 值自动加 1。
(10) MAR 中内容经地址译码器译码，找到 1005H 存储单元。
(11)CPU 发读命令。
(12)将 1005H 单元内容 0AH 读至 MDR。

(13)因 0AH 是操作数，将它经内部总线暂存作为低 8 位地址。
(14)将 PC 内容 1006H 送 MAR 。
(15) PC 值自动加 1。
(16) MAR 中内容经地址译码器译码，找到 1006H 存储单元。
(17) CPU 发读命令。
(18)将 1006H 单元内容 1OH 读至 MDR
(19)因 1OH 是操作数，将它与暂存的低 8 位地址形成 16 位地址。
;;;以下执行指令
(20)因加法(SCH + 2EH = 8AH)有溢出，将 100AH 送 PC。
(下一步将跳过下一条指令，执行 100AH 单元指令)
MOV (0200H),A 详细执行步骤如下(此指令因加法溢出不执行):
(1)将 PC 内容 1007H 送地址寄存器 MAR。
(2) PC 值自动加 1，为取下一个字节机器码作准备。
(3) MAR 中内容经地址译码器译码，找到内存储器 1007H 单元。
(4) CPU 发读命令。
(5)将 1007H 单元内容 A2H 读出，送至数据寄存器 MDR 。
(6)由于 A2H 是操作码，故将它从 MDR 中经内部总线送至指令寄存器 IRo
(7)经指令译码器 ID 译码，由操作控制器 OC 发出相应于操作码的控制信号。
;;;以下取操作数
(8)将 PC 内容 1008H 送 MAR 。
(9) PC 值自动加 1
(10) MAR 中内容经地址译码器译码，找到 1008H 存储单元。
(11) CPU 发读命令。
(12)将 1008H 单元内容 00H 读至 MDR 。
(13)因 00H 是操作数，将它经内部总线暂存作为低 8 位地址。
(14)将 PC 内容 1009H 送 MAR 。
(15) PC 值自动加 1
(16) MAR 中内容经地址译码器译码，找到 1009H 存储单元。
(17) CPU 发读命令。
(18)将 1009H 单元内容 02H 读至 MDR
(19)因 02H 是操作数，将它与暂存的低 8 位地址形成 16 位地址 0200H 送 MAR 。
;;;以下执行指令
(20) CPU 发写命令。
(21)将 A 内容送 0200H 单元。
HLT 详细执行步骤如下:
(1)将 PC 内容 100AH 送地址寄存器 MAR.
(2) PC 值自动加 I，为取下一个字节机器码作准备。

（3） MAR 中内容经地址译码器译码，找到内存储器 100AH 单元。
(4) CPU 发读命令。
(5)将 100AH 单元内容 F4 H 读出，送至数据寄存器 MDR。
（6） 由于 F4 是操作码，故将他从 MDR 中经内部总线送至指令寄存器 IR。
(7)经指令译码器译码，由操作控制器 OC 发出对应于操作码的控制信号
(8)停机。

2.9 如果 GDT 寄存器值为 0013000000FFH，装人 LDTR 的选择符为 0040H，试问装人描述符高速缓存的 LDT 描述符的起始地址是多少?

解:GDT 寄存器的高 32 位和低 16 位分别为 GDT 的基址和段限，所以:GDT 的基址=00130000H

LDTR 选择符的高 13 位 D_{15}~D_3=000000001000B 是该 LDT 描述符在 GDT 中的序号，所以:

LDT 描述符的起始地址= GDT 的基址

十 LDT 描述符相对于 GDT 基址的偏移值

=00130000H+8×8=00130040H

2.10 假定 80486 工作在实模式下，(DS)=1000H, (SS)=2000H, (SI)=E107FH, (BX)=0040H, (BP) = 0016H，变量 TABLE 的偏移地址为 0100H。请问下列指令的源操作数字段是什么寻址方式?它的有效地址(EA)和物理地址(PA)分别是多少?

(1)MOV AX，[1234H] (2) MOV AX, TABLE

(3) MOV AX，[BX+100H] (4) MOV AX,TABLE[BPI[SI]

解:(1)直接寻址，EA=1234H , PA =(DS)×16+EA=11234H。

(2)直接寻址，EA= 0100H,PA= (DS)×16+EA=10100H。

(3) 基址寻址，EA=(EBX)+100H =0140H,PA= (DS) × 16+EA=10140H。

(4)带位移的茎址加变址寻址。(EA) = (BP)+[SI]十 TABLE 的偏移地址=0195H

PA=(SS)×16+EA=20195H}

2.11 下列指令的源操作数字段是什么寻址方式?

(1)MOV EAX , EBX (2)MOV EAX，[ECX] [EBX]

(3) MOV EAX，[ESI][EDX * 2] (4) MOV EAx，[ESI*8]

解:(1) 寄存器寻址。

(2) 基址加变址寻址。

(3)基址加比例变址寻址。

(4)比例变址寻址。

2.12 分别指出下列指令中源操作和目的操作数的寻址方式。

式表示出 EA 和 PA。

(1)MOV SI,2100H (2) MOV CX, DISP[BX]

(3) MOV [SI] ,AX (4) ADC AX，[BX][SI]

(5) AND AX，DX (6) MOV AX，[BX+10H]

(7) MOV AX,ES:[BX] (8) MOV Ax, [BX+SI+20H]

(9) MOV [BP].CX (10) PUSH DS

解:(1) 源操作数是立即数寻址;目的操作数是寄存器寻址。

(2) 源操作数是基址寻址，EA=(BX)+DISP，PA=(DS)×16+(BX)+DISP

目的操作数是寄存器寻址。

(3)源操作数是寄存器寻址;

目的操作数是寄存器间接寻址，EA=(SI).PA=(DS) × 16十(SI)。

(4) 操作数是基址加变址寻址，EA= (BX)+(SI).PA= (DS) × 16十(BX)+(SI)

目的操作数是寄存器寻址。

(5)源操作数和目的操作数均为寄存器寻址。

(6) 源操作数是基址寻址，EA=(BX)+10H.PA= (DS) × 16十(BX)+10H

目的操作数是寄存器寻址。

(7) 源操作数是寄存器间接寻.EA= (Bx).PA= (ES) × 16+(BX)

目的操作数是寄存器寻址。

(8)源操作数是带位移的基址加变址寻址:

EA=（BX）+（SI)+20H.PA=（DS）×16+（BX）+（SI）+20H ;

目的操作数是寄存器寻址。

(9)源操作数是寄存器寻址;

目的操作数是寄存器间接寻址，EA=（BP）.PA=（SS）×16十(BP)。

(10)源操作数是寄存器寻址;

目的操作数是堆栈寻址,EA=（SP)-2.PA=（SS）×16十(SP)-2。

2.13 已知80486工作在实地址方式下。其中一些寄存器的内容和一些存储单元的内容如图2.1所示，试指出下列各条指令执行后。AX中的内容。

(1) MOV AX , 2010H

(2) MOV AX, BX

(3) MOV AX , [1200H]

(4) MOV AX , [BX]

(5) MOV AX, 1100H[BX]

(6) MOV AX, [BX][SI]

(7) MOV AX, 1100H[Bx+SI]

(8) LEA AX, [SI]

解:(1) (AX) =2010H;

(2) (AX)=（BX）=0100H;

(3) (AX)= [32100H]=4C2AH:

(4)(AX)= [30100H]=3412H;

(5)(AX)= [31202H]=4C2AH;

(6)（AX）= [30102H]=7856H;

(7) (AX)= [31202H]=65B7H;

(8)（AX）=（SI）=0002H

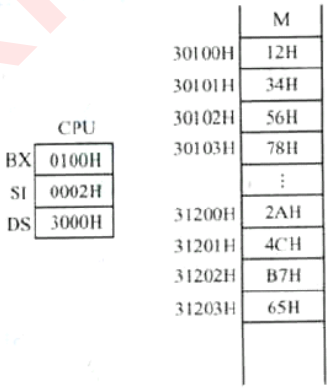


图2.1 习题2.13图

2.15 已知数据如图2.2所示，数据是低位在前，按下列要求编写程序段:

(1)完成NUM1和NUM2的两个字数据相加，和存放在NUM1中。

(2)完成NUM 1单元开始的连续4个字节数据相加，和不超过一字节，放在RES单元。

(3)完成NUM1单元开始的连续8个字节数据相加，和为Ifi位，放在RES和RES十1两个单元中。

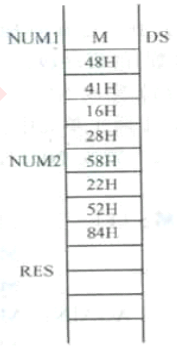


图 2.2　习题 2.15 图

(4)完成 NUMI 和 NUM2 的双倍精度字数据相加。和存放在 NUM2 开始的双字单元中。

```
解：（1）  MOV     AX,WORD PTR NUM2
           ADD      WORD PTR NUMI,AX
     (2)   LEA      SI，NUM1
           MOV      AL，0
           MOV      CX,4
AGAIN ：   ADD      AL，[S1]
           INC      SI
           LOOP     AGAIN
           MOV      RES，AL
     （3）  LEA      SI,NUM1
           MOV      AX，0
           MOV      CX，8
AGA I N：  ADD      AL，[SI]
           ADC      AH，0
           INC       SI
           LOOP     AGAI N
           MOV      WORD PTR RES，AX
           MOV      AX，WORD PTR NUM 1
           ADD      WORD PTR NUM2，AX
           MOV      AX，WORD PTR NUM 1［2］
           ADC      WORD PTR NUM2［2］，AX
```

2.16　已知的 BCD 数如图 2,2 所示，低位在前，按下列要求编写计算 BCD 数据(为组合 BCD 数)的程序段:

(1)完成 NUM I 单元开始的连续 8 个组合型.SCD 数相加，和(超过一字节)放在 RES 和 RES + 1 两个单元中。

(2)完成 NUMI 单元和 NUM2 单元的两个 BCD 数相减，其差存人 RES 单元，差=?，CF=?。

```
解:(1)LEA      SI，   NUM 1
               MOV    WORD PTR RES,0          ;和清 0
               MOV      CX，8                 ;置循环次数
     AGAIN ;   MOV    AL，RES                 ;取和低字节
               ADD     AL，[SI]               ;与当前 BCD 数相加
```

```
                DDA                         ;BCD 调整
                MOV   RES, AL               ;保存和低字节
                JNC   NEXT                  ;无进位不处理和高字节
                MOV   AL, RES + 1           ;有进位，和高字节加 1
                ADD    AL , 1
                DAA
                MOV   RES+ 1 ,  AL          ;保存和高字节
        NEXT:   INC   SI
                LOOP   AGAIN
    （2）       MOV AL, NUM1
                SUB  AL, NUM2
                DAS
                MOV     RES, AL
```

差=(RES)=90H,CF=1

2.17 已知数据如图}.z 所示，低位在前，按下列要求编写程序段:

（1）RIUM1 和 NUM2 两个数据相乘(均为无符号数)，乘积放在 RES 开始的单元。

（2）NUM1 和 NUM2 两个字数据相乘《均为带符号数)，乘积放在 RES 开始的单元。

（3）NUM1 单元的字节数据除以 46（均为无符号数)，商和余数依次放人 RES 开始的两个字节单元。

（4）NUM1 字单元的字数据除以字单元的宇，商和余数依次放人 RES 开始的两个字单元。

解(1)

```
MOV ,    AL,BYTE PTR NUM1
MUL      BYTE,PTR,  NUM2
MOV      WORD PTR RES,  AX
```

（2）

```
MOV      AX,WORD PTR  NUM1
IMUL     WORD PTR, NUM2
MOV:  WORD PTR RES, AX
MOV      WORD PTR RES 十 2, DX
```

（3）

```
MOV      AX,46
DIV      BYTE PTR NUMI
MOV      WORD PTR RES,  AX
MOV      AX, NUM2
```

（4）

```
CWD/MOV DX, 0
IDIV/DIVWORD  PTRNUM1
MOV WORD PTR RES, AX
MOV WORD PTR RES +2 , DX
```

2.18 已知:(SS)=0A8E0H,(SP)=06C0H，(AX)二 8881 H, (CX)=0E245H.试画出下列指令执行到位置 1 和位置 2 时堆栈区和 SF 指针内容的变化示意图。图中应标出存储单元的实际地址 PA。

```
PUSH  AX
PUSH  CX
POPF
```

执行到位置 1 和位置 2 时堆栈区和 SF 指针内容分别如图 2.3(a)和 2.3（b）所示。

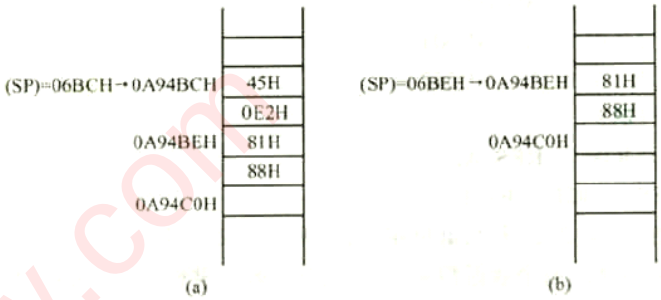


图 2.3　习题 2.18 图

2.19 识别下列指令的正确性，对错误的指令、说明错误的原因。

(1)MOV　DS，100　　(2)MOV［1000H］，23H

(3)MOV　［1 000H］，［2000H］　　(4) MOV　DATA,1133H

(5)MOV　1020H，DX　　(6) MOV　AX，［0100H+BX+BI］

(7) MOV　CS，AX　　(8) PUSH　AL

(9) PUSH　WORD PTR[SI]　　(10) IN　Al,[80H]

(11)OUT　CX ， AL　　(12) IN　AX, 380H

(13) MOV　CL, 3300H　　(14) MOV　Ax, 2100H[BP]

(15) MOV　DS，ES　　(16) MOV　IP, 2000H

(17) PUSH　CS　　(18)POP　CS

(19) LDS　CS, [BX]　　(20) MOV　GAMMA ， CS

(21)XCHG　AH，AL　　(22) XCHG　200，AL

(23) JMP　BX　　(24)SHR　Dx,2

解:(1)错误。立即数不能直接斌给段寄存器。

(2)错误。目的操作数长度不确定。

(3)错误。源、目的操作数不能同为存储器寻址。

(4)依赖于 DATA 的定义。若 DATA 定义为字或双字变童，则正确:若 DATA 定义为字节变量.则源操作数超出目的操作数的数据范围.则错误;若 DATA 定义为常 t，则目的操作数为立即数。则错误。

(5) 错误。立即数不能直接赋给段寄存器。

(6)错误。16 位(带位移)基址加变址寻址方式中的地址寄存器不能同为基址寄存器。

(7)错误。不能用传送指令改变代码段寄存器。

(8)错误。堆栈指令的操作数不能为字节操作数。

(9)正确。

(10) I/O 端 A 只能用立即数或 DX 间接寻址。

(11)错误。CX 不能用于寻址 I/O 端口。

(12)错误。采用立即数寻址 I/O 端口时，端口号不能大于 255。

(13)错误。源操作数超出目的掸作数的数据范围。

(14)正确。

(15)错误。源、目的操作数不能同为段寄存器。

(16)错误。不能用传送指令改变 IP 寄存器。

(17)正确.:

(18)错误。不能用堆找指令改变 CS 寄存器。

(19)错误。CS.（E）IP 寄存器只能通过程序控制类指令改变。

(20)依赖于 GAMMA 的定义.见(4)。

(21)正确。

(22)错误。交换指令的源、目的操作数均不能为立即数。

(23)正确。

(24)若为 80486 指令，则正确:若为 8086/8088 指令，则错误，8086/8088 源操作数用立即数只能为 1。

2.20 设程序在数据段中定义的数组如下:

```
NAMES DB      `GOUP MORNING!
      DW      2050H
      DB      'PRINTER
      DB      48
      DB      'MOUS.EXE
      DW      3080H
```

请指出下列指令是否正确，如正确，累加器 A 中的结果是多少?

```
(1) MOV  ESX , OFFSET NAMES
    MOV  EAX, [EBX+13 }
(2) MOV  EAX, NAMES
(3) MOV  AX, WORD PTR NAMES+S
(4) MOV  BX ,12.
    MOV  SI , 6
    MOV  AX , NAMES[ BX ] [SI ]
(5) MOV EBX,16*2
    MOV  ESI , 4
    MOV  EAX, OFFSET NAMES[ EBX][ ESI]
    INC [AX]
(6) MOV BX, 12
    MOV SI,6
    LEA    DI, NAMES[ BX ] [ SI ]
    MOV    AL, [SI]
```

解:(1)两条指令都是合法指令。

第一条指令执行(EBX)=NAMES 的偏移地址。

第二条 MOV 指令原操作数有效地址 EA=(EBX)十 13= NAMES 的偏移地址+13.

所以.(EAX)= [EBX + 13] = [NAMES + 13)=52542454H

(2)指令不合法。变量 NAMES 的类型为字节，而目的寄存器是 EAX，所以源操作数与目操作数类型不一致。

(3)指令正确。

[NAMES+5]='M' =4DH, (NAMES+6]=.'O' =4FH

所.(AX)=[NAMES+5]=4F4DH

(4)前二条指令正确;第三条指令错误，源操作数与目的操作数类型不一致。

(5)前二条指令正确;第三条指令错误,OFFSET 运算符只能作用于直接的存储器变量,第四条指令错误，在 15 位寻址中，AX 不能作为间接寻址寄存器。

(6)均为合法指令。

前两条指令执行:[BX]=12,(SI)=6.

第三条指令执行:(DI)=(BX)+(SI)+ OFFSET NAMES = OFFSET NAMES + 1 8 .

所以，(AL) =(DI)= [NAMES+ 18] = 'N '= 4EH.

2.22 已知(AL)=0C4H ， DATA 单元的内容为 5AH.写出下列每条指令单独执行后的结果。

(1)AND AL,DATA (2)OR AL,DATA

(3)XQR AL ， DATA (4)NOT DATA

(5)AND AL,0FH (6)OR AL,01H

(7)XOR AL,0FFH (8)TEST AL,80H

解:(1)(AL)=01000000B (2) (AL)=11011110B

(3) (AL)=10011110B (4) (AL)=0A5H

(5) (AL_)=04H (6) (AL) =0C5H

(7) (AL)=3BH (8) (AL)=0C4H

2.23 设(IP)=3D8FH, (CS)=4050H,(SP)=0F17CH，当执行 CALL 2000 :009AH 后，试指出 IP.CS.SP.[SP].[SP+1].[SP+2][SP+3]的内容。

解:CALL 指令的执行过程是:先将返回地址(当前 CS:IP 寄存器的值)压栈,然后将调用程序的人口地址送 CS:IP 寄存器，从而转人子程序执行。压栈时 CS 在先，IP 在后。此题，执行 CALL 2000: 0009AH 指令

后，堆栈变化如图 2.4 所示。相应寄存器和堆栈单元内容为:

(IP)=0009AH;

(CS)=2000H:

(SP)=0F17CH-4=0F178H;

[SP]=8FH;

〔SP+1〕= 3DH;

[SP+2]=50H;

[5P+3]=40H.

2.24 将 DS 寄存器的内容传送到 ES 段寄存器。

(1)用传送指令

解:(1)MOV AX, DS (2) PUSH DS

MOV ES, AX POP ES

2.25 将 BX 和 CX 中的两个 16 位数(其中 AX 是高 16 位)组成 32 位数传送到 EAx 寄存器中。

(1]用堆栈操作指令。(2)用移位指令。

解(1)将两个}6 位数通过堆栈组成一个 32 位数时，要先压高位，后压低位。

```
PUSH    BX                  ;先压高 15 位
PUSH    CX                  ;后压低 15 位
POP     EAX
( 2)    MOV     AX, BX
        SAL     EAX,10H     ;将 BX 内容移至 EAX 的高 15 位
        MOV     AX,CX
```

2.26 执行以下三条指令后 AL 寄存器的值和 CF 标志位的值是多少?

```
MOV  AL, 58H
ADD  AL, 64H
DAA
```

解：此程序完成两 BCD 数的相加：58+64=122。

所以，(AL) =22H，CF=1。

2.27 执行指令 ADD A1, 72H.前，(AL)=8EH,标志寄存器中的状态标志 OF，SF，ZF，AF，PF,

CF 全为 0。指出该指令执行后各状态标志的值

解：8EH+72H=10001110B+01110010B=10000000B

OF	SF	ZF	AF	PF	CF
0	0	1	1	1	1

2.28 已知寄存器 DX:AX 的内容为 32 位带符号数，编写一段程序使 DX:AX 的内容成为原来数据的绝对值。

解：对双精度数求补可用指令 NEG 实现，又可用指令 NOT 实现，假定用 NEG 指令实现

程序段如一下：

```
        TEST    DX，8000H       ；（DX：AX）≥0?
        JZ      NEXT            ；大于等于 0，则不求补
        NEG     DX              ；以下对 DX；AX 内容求补
        NEG     AX
        SBB     DX，0
NEXT：HLT
```

2.29 两个 ASCII 码数据串定义如下：

```
STR1   DB   ‘352678’
STR2   DB   ‘123400’
```

编写一程序段，计算 STR1+ STR2，要求结果仍然是 ASCII 码数据串。

解：假定 ASCII 码数据串是高位在前，低位在后，结果存于 STR3 开始的数据区，程序段如下：

```
        MOV   SI，5                 ；SI 指向 ASCII 码数据串个位
        CLC                         ；清 CF，保证个位相加不带进位
        MOV   CX，6
DONE：  MOV   AL，STR1[SI]
        ADC    A L ，STR2 [ SI ]    ；ASCI I 码数相加
        AAA                         ;非压缩 13CD 码调整
        PUSHF                       ;保存进位。以便向高位进位
        AND AL,01FH                 ;非压缩 BCD 码转换为 ASCII 码
        ADD       AL,30H
        MOV       STR3 [SI],AL      ;保存 ASCI I 码数结果
        DEC       SI                ；S1 指向下一串数据
        POPF                        ;恢复低位来的 CF,以向高位进位
      LOOP DONE
```

2.30 假定程序中数据定义如下:

```
FIRST       DB   30 DUP(?)
SECOND      DB   30 DUP[?)
THIRD       DB   30 DUP(?)
```

编写一个程序把 FIRST 与 SECOND 中的 30 个字节数分别相加，结果存放到 THIRD。

(1)假定数据为无符号数，如果和大于 255 则保存结果为 255。

(2)假定数据为带符号数，如果有溢出(大于+127 或小于-128)则保存结果为 0。

解：（1）使用串操作指令。程序如下:

```
        LEA     SI ，  FIRST
        LEA     BX，SECOND
        LEA     DI,THIRD
```

```
                MOV     CX,30                   ;Cx 作循环计数
                CLD                             ;DF=4，地址递增
        NEXT:   LODSB                           ;从数组 FIRST 取一字节数.指向下
一元素
                ADD     AL，[BX]                ;数组元索相加
                JNC     STORE;小于 255          ;跳过
                MOV     AL,0FFH                 ;溢出，结果为 2S5
        STORE ：STOSB                           ；保存结果至 THIRD，并指向下一
元素
                INC    BX                       ;指向数组 SECOND 的下一元素
                LOOP    NEXT
```

(2)用变址寻址。程序如下:

```
                MOV     SI,0                    ;SI 作数组下标
                MOV     CX,30                   ;CX 作循环计数
        NEXT:   MOV     AL, FIRST[ SI ]         ;从数组 FIRST 取一字节数
                ADD     AL , SECOND[ BX ]       ;与 SECOND 数组元素相加
                JNO    NO__OVER                 ;无溢出，跳过
                MOV     AL,0                    ；溢出，结果为 0
    NO_ OVER:   MOV     THIRD[ SI ] , AL        ;保存结果至 THIRD
                INC     SI                      ;指向下一数组元素
                LOOP    NEXT
```

2.31 编写一个程序计算 $Z=X^2+34\ X\ Y-X/Y$(X,丫为字节变量).

假定 X,Y.Z 为有符号数.Z 为字变量程序段如下:

```
                MOV     AL, X
                MOV     BL,AL                   ;X 保存至 BL.以便后用
                IMUL    BL                      ;X²→AX
                MOV     DX,AX                   ;保存 X²
                IMUL    BL
                MOV     CX, 34
                IMUL    AX, CX
                ADD     DX, AX
                MOVSX   AX, BL
                IDIV    Y
                CBW
                SUB     DX, AX
                MOV     Z, DX
```

2.32 字符串 STR1 保存着 100 个字节的 ASCII 码,试编写一个程序统计该字符串中空格(20H)个数。

解:使用串扫描指令查找关健字时.可用 REPNZ 或 REPNE 重复前级，但要注意执行指令 REPNZ SCASB 的结束条件有两个,一是找到关键字,此时 ZF 标志位为 1,二是重复计数器(CX)值为 D，未找到关键字，此时 ZF 标志位为 0。所以串指令结束时，可判别 ZF 标志来识别是否找到关键字。程序如下:

```
        LEA     DI，STR 1
        MOV     CX, 100
```

```
                MOV     DL, 00H                 ;统计空格计数器初值为。
                CLD
                MOV     AL,2DH                  ;字符
        DONE:    REPNZ  SCASB                   ;扫描字符串，直到找到空格或扫描完
毕
                 JNZ    EXIT                    ;搜索完，结束
                 INC    DL.                     ;找到。计数器加 1
                 JMP    DONE                    ;未搜素完，继续
        EXIT:    HLT
```

2. 33 用循环移位指令，编写完成以下功能的程序段(结果放回原处)。

(1)将无符号数 83D 乘以 2 和除以 2.

(2)将带符号数-47D 乘以 2 和除以 2.

解(1)用带进位的循环移位指令‘无论乘还是除，均要保证进位 CF= 0

```
    MOV       AX, 83
    CLC
  RCL/RCR     AX,1                ;AX 乘 2/除 2→AX
```

(2)用带进位的循环移位指令实现带符号数的倍乘和倍除时，乘法与无符号数一样.要保证进位 CF=0;而对除法则要根据符号位设定。正数 CF=0，负数 CF=1.

```
   MOV   AX,-47
  CLC/STC;乘 CF= 0/除 CF=1
  RCL/RCR    AX ,1;(AX)乘 2/除 2→AX
```

2.34 设(AX)=0C3H，用循环移位指令实现以下功能。

(1)设(CL)=8，移位前后 AX 内容不变·

(2)设(CL]=9。移位前、后 Ax 内容不变。

(3)将 AX 中高 4 位和低 4 位交换位置。

(4)将 AX 中高 4 位放到低 4 位上.高 4 位清 0.

解(1)用一条不带进位的循环移位指令实现:

```
  ROL/ROR        AL,CL 或 ROL/ROR      AH , CL
```

(2)用一条带进位的循环移位指令实现:

```
  RCL/RCR        AL,CL 或 RCL/RCR    AH, CL
```

```
(3) MOV   CL,4 或 MOV   CL, 4
    ROR   AH , CL   或    ROL   AL, CL
    ROR    AX , CL   或   ROL   AX , CL
    ROR    AL. CL     或  ROL   AH, CL
(4) MOV    CL,4
    ROL      AX,CL                .
    ROL       AH,CL
    AND       AX.0FFFH
```

2.35 用乘法指令和用传送、移位、相加指令分别实现 y =10X 的运算，设 X=12345678H ，分别编写这两个程序段。

解:(1)用乘指令实现 10X

```
  MOV       EAX,10
  MOV        EBX ,12345678H
  MUL      EBX
```

(2)用移位和加法指令实现 10X = 8X+2X

```
MOV        EAX, 12345678H
SAL/SHL    EAX，1
MOV        EBX , EAX
SAL/SHL    EAX,2
ADD        EAX , EBX
```

2.36 已知一个关于 0- 9 的数字的 ASCII 码表首地址是当前数据段的 0A80H 现要找出数字 5 的 ASCII 码，试写出用指令 XLAT 进行翻译的指令序列。

解:

```
MOV     BX,0A80H
MOV     AL,5
XLAT
```

2.37 试编写完成(EAX)×9/4 的程序段。

解:可用乘、除法指令或移位和加法指令实现。

(1)用乘、除法指令。注意单操作数乘、除法指令不能用立即数寻址。

```
MOV       EBX , 9
MUL/IMUL  EBX
MOV       EBX,4
DIV/IDIV
```

(2)用移位和加法指令实现。(EAX)×9/4=[(EAX)×8+(EAX)]/4

```
MOV       EBX，EAX
SAL/SHL   EAX,3
ADD       EAX. EBX
SAR/SHR   EAX.2
```

2.38 若(AL)=0FFH.(BL)=03H，指出下列指令执行后标志 OF,SF,ZF,PF.CF 状态。

```
(1)ADD    BL, AL          (2)INC    BL
(3) SUB   BL,AL           (4) NEG    BL
(5)CMP    BL,AL           (6) MUL    BL
(7)AND    BL,AL           (8)IMUL    BL
(9) OR    BL,AL           (10) SHL   BL,1
(11)XOR   BL,BL           (12) SAR   AL,1
(13) SHR  AL，1
```

解:指令执行后的结果和标志如表 2.3 所示。对各指令执行后的条件码设置要注意:

(1)I NC/DEC 指令执行.后不影响 CF 标志。

(2)IMUL/MUL 指令的执行仅影响 CF 和 OF 标志，设置方法如下:对 IMUL 指令，若结果的高一半部分((8 位乘时为 AH.16 位乘时为 DX, 32 位乘时为 EDX)不是结果的低一半部分(AL/AX/EAX)的符号扩展，即高一半部分是有效数字，则 CF 和 OF 为 1;否则.CF 和 OF 为 0。此题是 8 位乘法，所以表示 AH 含有乘积结果的有效数字。而 MUL 指令执行后，若结果的高一半部分不为。，则 CF 和 OF 置 1;否则，CF 和 OF 为 0.

(3)补码减法可用补码加法实现，所以减法滋出的判别也可用补码加法的抽双进位法”来判别，但此时。减法的借位 CF 与加法进位 CF 相反。

(4)对移位指令，当移位位数为 1 时，SAR 指令执行后，OF=0; SHR 指令执行后。OF 被置成与执行前目标操作数的最高有效位 MSB 相同的值;SAL/SHL/ROL/ROR/RCL.指令执行后。若 CF 与目的操作数(结果)的最高位相等则 OF=0 否.则 OF= 1;而对 RCR 指令，OF 的测试是在循环

移位前进行，设里方法同 SAL/SHL/ROL/ROR/RCL

(5) NEG 指令的标志位设置同 SUB 指令。

表 2.3 指令执行后标志状态

指　令	结　果	OF	SF	ZF	PF	CF
(1) ADD BL,AL	(BL) = 0000,0010	0	0	0	0	1
(2) INC BL	(BL) = 0000,0100	0	0	0	0	
(3) SUB BL,AL	(BL) = 0000,0100	0	0	0	0	1
(4) NEG BL	(BL) = 1111,1101	0	1	0	0	1
(5) CMP BL,AL	(BL)=03H (BL) - (AL) = 0000,0100	0	0	0	0	1
(6) MUL BL	(AX) = 2FDH	1	—	—	—	1
(7) AND BL,AL	(BL) = 0000,0011	0	0	0	1	0
(8) IMUL BL	(AX) = FFFDH	0	—	—	—	0
(9) OR BL,AL	(BL) = 1111,1111	0	1	0	1	0
(10) SHL BL,1	(BL) = 0000,0110	0	0	0	1	0
(11) XOR BL,BL	(BL) = 0000,0000	0	0	1	1	0
(12) SAR AL,1	(AL) = 1111,1111	0	1	0	1	1
(13) SHR AL,1	(AL) = 0111,1111	1	0	0	0	1

2.39 试编写程序段，根据 AL 中的内容决定程序走向，若位 0 是 1.其他位为 0.转向 LAB1 ;若位 1 是 1.其他位为 0，转向 LAB2:若位 2 是 1，其他位为 0，转向 LAB3 ;若位 0 至位 2 都是 0，顺序执行。假定所有的转移都是短转移。

```
解:     TEST   AL,07H
        JNZ    NEXT
          …
NEXT:   CMP    AL,01H
        JZ     LAB1
        CMP    AL,02H
        JZ     LAB2
        CMP    AL,04H
        JZ     LAB3
LAB1:    …
LAB2:    …
LA83:    …
```

2.40 有以下的调用嵌套试函出下列各项调用或返回时的堆栈状态示意图·

(1) MAIN 调用 NEAR 的 SUBA 过程(返回的偏移地址为 0500H)

(2) SUBA 调用 NEAR 的 SUBB 过程(返回的偏移地址为 0810H)

(3) SUBB 调用 FAR 的 SUBC.过程(返回的段地址为 A310H，偏移地址为 0400 H)

(4)从 SUBC:返回 SUBB

(5) SUBB 调用 NEAR 的 SUSUBD 过程(返回的偏移地址为 0C00H)。

(6)从 SUBD 返回 SUBB

(7)从 SUBB 返回 SUBA

(8)从 SUBA 返回 MAIN

解:堆栈状态示意图如图 2.5 所示。

2.41 设 x.y 为无符号字节数。试编写一个程序段完成下列数学表达式的计算。

$$y=\begin{cases} x-20, & x\geqslant 20 \\ 3x, & x<20 \end{cases}$$

解:对条件分支程序要往意各分支的出口,以避免执行完某个分支后又执行另一个分支.程序段如下:

```
        MOV     AL, x
        CMP     AL, 20
        JNC     NEXT
        MOV     BL, AL
        SHL     AL,1
        ADD     AL BL
        JMP     EXIT
NEXT:   SUB     AL,20
EXIT: HLT
```

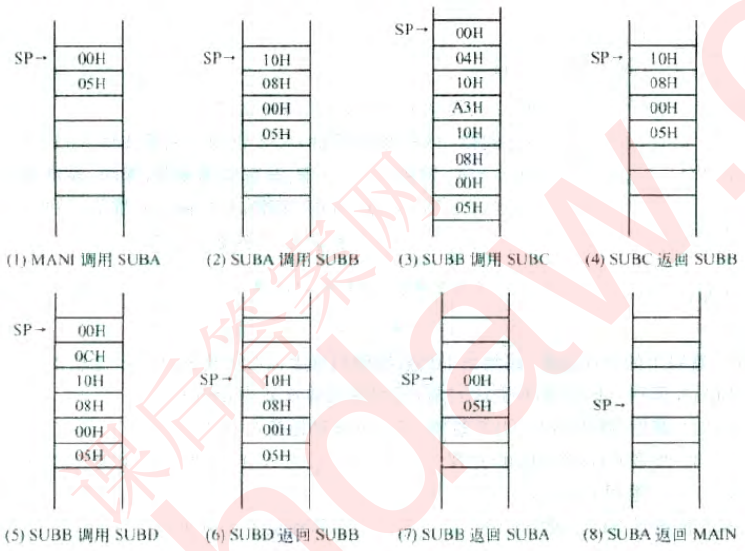


图 2.5 例 2.40 图

2.42 写出能代替下列重复串操作指令完成同样功能的指令序列。

(1)REP MOVSW （2）REPZ CMPSB （3）REPNZ SCASB

(4)REP LODSW （5）REP STOSB

解:(1)AGAIN: MOV AX, [SI]

MOV ES: [DI], AX

ADD SI, 2

ADD DI, 2

LOOP AGAIN

（2）AGAIN: MOV AL, [SI]

INC SI

INC DI

CMP AL, ES: [DI-1]

LOOPZ AGAIN

（3）AGAIN: MOV AX, [SI]

CMP AL, ES: [DI-1]

LOOPZ AGAIN

（4）AGAIN: MOV AX, [SI]

```
            ADD SI，2
            LOOP AGAIN
（5）AGAIN：  MOV ES：[DI]，AL
          INC DI
          LOOP AGAIN
```

2.43 假定在数据段中已知字符串和未知字符串的定义如下:

```
    STRING1     DB    'MESSAGE AND PROCCESS'
    STR I NG2   DB    20 DUP( ? )
```

试用串操作指令编写完成下列功能的程序段(设 ES 和 DS 重叠)。

（1）从左到右把 STRINGI 中字符串搬到 STR ING2

(2)从右到左把 STRINGI 中字符串握到 STR1NG2。

（3)搜索 STRING1 字符串中是否有空格。如有，记下第一个空格的地址.并放人 BX 中。

(4)比较 STRING1 和 STRING2 字符串是否相同。

解:(1)

```
LEA   SI，STRING1
LEA   DI,STRING2
MOV   CX,20
CLD
REP   MOVSB
```

(2)

```
LEA   SI,STRING1[19]      ;从高地址(右)开始
LEA   DI，STRING2[19 ]
MOV    CX，20
STD；地址递减
REP   MOVSB
```

(3)

```
LEA    DI，STRING1
MOV    CX，20
MOV    AL，20H
MOV    BX，0
    CLD
        REPNE SCASB
        JNE   ND_MATCH
MATCH:DEC        DI
        MOV    BX，DI
NO__MATCH:HLT
```

(4)

```
LEA   SI，STRING1
LEA   DI，  STRING2
MOV   AL，0          ;不匹配，(人 L)二 U
MOV   CX，20
CLD
REPE   CMPSB
JNE    NOT_EQU
MOV    AL，1         ;匹配，( AL)=1
NOT_EQU:H LT
```

2.44 已知内存中起始地址为 BLOCK 的数据块中的字节数据有正有负。要求编写一个程

序。将其中的正、负数分开，分别送至同一段中的两个缓冲区，设正、负数级冲区的首地址分别:PLUS_ BATA 和 MINUS_ DATA.字节数总数为 COUNT 个。

解:正数和负救的分检可用 TEST 指令侧试符号位实现，而数据块的存取，可用 MOV 指令存取.亦可用申指令存取。假定用串指令，程序段如下:

```
        LEA     SI,BLOCK          ;取数组首地址
        LEA     DI,PLUS_  DATA    ;取正数缓冲区首地址
        LEA     BX，MINUS  DATA   ;取负数缓冲区首地址
        MOV     CX,COUNT          ;取数组长度
        PUSH    DS
        POP     ES                ; ES 指向 D6 段
        CLD                       ;地址递增
AGAIN :  LODSB                    ;取一个数组元素,SI 自动加 I
        TEST    AL, 80H           ;教组元素大于等于 0?
        JNZ    MINUS              ;负数，转负数存储处理
        STOSB:保存正数，Ili 自动加 i
          JMP     NEXT
MINUS:   XCHG    DL,BX            ;DI 指向负数存储区
        STOSB                     ;保存负数，DI 自动加 1
        XCHG    DI, BX            ;恢复 DI ,BX 分别指向正、负数缓冲区
NEXT:    LOOP    AGAIN
        HLT
```

2.45 请逐条注释下列两个程序段的每个指令行.并说明它们完成的功能。

```
( 1 )       LEA      BX , ARRAY      ;取数组 ARRAY 的起始地址
            LEA DI, RESULT           ;取数组.RESULT 的起始地址
            MOV      CL,4            ;里循环计数初值
AGAIN:      MOV      AL,[BX]         ;取数组 ARRAY 元素
            PEST     AL,80H          ;测试数组元素是否为正数
            JZ       NEXT            ;为正，转 NEXT
            NEG      AL              ;求负数的绝对值
NEXT:       MOV      [DI],AI.        ;保存数组元家绝对值至 RESULT 数组
            INC      BX              ; BX 指向下一 ARRAY 数组元素
            INC      DI              ; DI 指向下一 RESULT 数组元素
            DEG      CL              ;循环计数器减 1
            JNZ      AGAIN           ;未处理完，转 AGA IN 继续
(2)         MOV      AL,0            ;( AL)“U
            MOV      SI，-1          ;数组下标(5I)=一 1
            MOV      CX,100          ;循环计数器 (CX) = 100
NONZERO:    INC      SI              ;指向下一数组元素
            MOV    AL，ARRAY1[SI]    ;取数组 ARRiAY1 第(SI)个元素
            ADD     AL,ARRAY2[SI]    ;与数组 ARRAY 第(SI]个元素相加
            MOV     SUM[SI],AL       ;和存入数组 SUM 的第[SI]个元素
            LOOPNZ   NONZERO
            JZ       ORENTRY         ;若和为零转 DRENTRY 处理
```

```
ZERO:     RET                          ;返回主程序
ORENTRY:  INC  CX                      ;[CX]+1 送 CX
          MOV WORD PTR NO, CX          ;CX 值存人 NO 单元
          JMP    ZERO                  ;转 ZERO 返回主程序
```

解:(1)程序功能是将数组 ARRAY 中 4 个字节数求绝对值后送数组 RESULT}

(2)程序功能是将长度为 100 的字节数组 ARRAY1 和 ARRAY 的对应元素相加，结果存入数组 SUM 的对应元素中，并判断两个数组元素的和是否为零，若为零，则后续数组元素不求，未求和元素个数(含当前和为零的元素)送 NO 单元保存。

3.1 假设 VAR1 和 VAR2 为字变最，LAB 为程序中的一个标号、试指出下列指令的错误之处:

(1)ADD VAR1，VAR2 （2）SUB AL，VAR1 （3）JNZ VAR1 （4）JMP LAB[SI]

(5) JMP NEAR LAB

解:此题相关知识点是变量与标号的区别-

（1)错误，不允许存储器变鸯间直接传数;

（2）错误，源与目的操作数类型不一致;

（3）错误，变量不能用作条件转移指令的操作数;

(4)错误，标号不能用作变址寻址的位移世;

(5)错误。缺 PTR 运算符。

3.2 已知程序中有如下数据定义:

```
DATX    DB     10，  ’ ABCD’，-1
        DW     -3，1 00H  ’AB’
        DD     1   1223344H
```

试将经汇编后对应于 DATX 数据区的数码填人下表中:

+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F

解：经汇编后表中数据如下：

+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0AH	41H	42H	43H	44H	FFH	FDH	FEH	00H	01H	42H	41H	44H	33H	22H	11H

/

3.3 给定如下数据定义:

```
A1 DW   1,2,3，’AB’,’ C’
A2  DB        6 DUP（?)
A3  DB        0
R1  EQU     A3-A1
```

（1）画出变量的内存分配图;

(2)常最符号 R1 的值为多少:

解:内存分配图如图 3.5 所示。

R1 的值=变最 A3 的偏移地址-变量 A1 的偏移地址

=数据区 A1 和 A2 所占的总字节数

=16

3.4 从 FIRST 开始的 1fH)个单元中存放着一个字符串.结束符为$。编写一个程序.统计该字符串中字母’A’的个数。

解:对固定长度的字符串扫描或比较，常用带重复前缀的申扫描或比较指令，而当字符串长度不确定时，则需对字符译进行逐个扫描或比较，并用字符串结束标志(判断条件)来控制扫描或比较结束、此越通过与结束符’$’比较来控制循环扫描结束.程序如下:

```
DATA    SEGMENT
        FIRST   DB   ’AGHI K...$’      ;定义字符串，结束符为$
        NUMBER  DB   ?                 ;结果存放单元
DATA    ENDS
```

```
CODE    SEGMENT
        ASSUME   CS: CODE, DS: DATA
START:      MOV    AX,DATA          ;建立数据段的可寻址性
            MOV    DS, AX
            MOV    BL，0            ;BL 用作统计计数器
            LEA    DI,FIRST         ;取字符串首址
NEXT:       MOV    AL, [ DI]        ;取一个字符
            INC    DI               ;指向下一个字符
            CMP    AL,'$'           ;与结束符’$’比较
            JE     EXIT             ;等于，搜索完结束
            CMP    AL , 'A          ;与字符‘A’比较
            JNE    NEXT             ;不等于。不统计
            INC    BL               ;找到，计数
            IMP    NEXT             ;继续
EXIT:       MOV NUMBER, BL
            MOV AH, 4CH
            INT 21H
            CODE ENDS
            END START
```

3.5 从 BLOCK 开始，存放着 256 个字节的带符号数，编写程序从这些数中找出绝对值最大的数，将其存人 MAX 单元。

解:此题是查找绝对值最大的数，因此，对数据块逐个扫描时要用数的绝对值进行比较，所以此题关键是要清楚绝对值的求法:正数的绝对值为其补码本身，负数的绝对值由其补码求补所得。用简化段定义，程序如下:

```
            . MODEL SMALL                          MOV    MAX, AL
            . STAGK;定义堆栈段
            .DATA;定义数据段
BLOCK   DB    10H,83H,0F4H, …
MAX       DB?
      ·CODE
START:  MOV    AX, @ DATA              ;定义代码段
        MOV    DS, AX                  ; 建立数据段的可寻址性
        MOV    CX,256
        MOV    MAX, 0
        MOV    BL,0                    ;存放绝对值最大的数，初值为。
        LEA    DI, BLOCK               ;BL 存放绝对值最大数的绝对值
NEXT:   MOV    AL, [DI]                ;取数据块首址
        TEST   AL,80H                  ;取一字节数据
        JE POSIT                       ;渊试是否正数
        NEG  AL                        ;是，跳过
POSIT:  CMP    BL,AL                   ;求负数绝对值
        JNC   DONE                     ;绝对值比较
        MOV     BL,AL                  ;大于等于，跳过
```

;当前数作为绝对值最大的数

.

```
                MOV     AL, [DI]
DONE:           INC DI                                  ;指向下一字节数据
                LOOP NEXT
                MOV AH, 4CH
                INT 21H
                END START
```

3.6 编写一个程序，将变量 ABC 中的 16 位无符号数用“连续除 10 取余”的方法转换成十‘进制数，要求结果用压缩 BCD 码保存在 RESULT 开始的单元中。

解:设 X 为任意无符号十进制数，则 X 可表示为:

$X=a_{n-1}\times10^{n-1}+\cdots+a_1\times10+a_0$

用 10 除 X，则:

商$=a_{n-1}\times10^{n-2}+\cdots+a_2\times10+a_1$

余数= a_0即为等值十进制数的个位数

由此可得将一个无符号二进制数用“连续除 10 取余”转换成十进制数的算法如下:

(1)用 10 除无符号二进制数，所得余数为所求十进制数的个位数;

(2)用 10 除无符号二进制数用 10 除后所得的商，求得十进制数的十位数;

(3)用与(2)相同的方法，依次求得百位数、千位数、…，直到除 10 后商为 0。

此算法循环次数是未知的，要用条件判断来结束循环，即判断商是否为零以控制循环结束。而要保持结果为压缩 BCD 码，有不同处理方法:一是先转换成非压缩 BCD 码，再组合;另一种方法是每一次循环转换二位 BCD 数，并直接组合。用后一种方法组合 BCD 码，程序如下:

```
                .MODEL SMALL
                ·STACK
                .DATA
ABC             DW      23456
RESULT          DB      3 DUP(0)
                ·CODE
START :         MOV      AX ,@ DATA
                MOV      DS,AX
                MOV      BX,10
                MOV      DI，0              ;寻址结果单元
                MOV      AX,ABC             ;取 16 位无符号二进制数
NEXT:           CMP     AX ， 0              ;判被除数是否为 0
                JZ       EXIT               ;(AX)=0，转换结束
                XOR       DX ， DX          ;被除数高位 DX 清 0
                DIV       BX                ;(DX:AX) /(BX)→DX,AX
                MOV       CL， DL           ;保存余数以与高位组成压缩
BDC 码
                XOR       DX,DX             ;被除数高位 DX 清 0
                DIV       BX                ;( DX : AX) /10→DX , AX
                SHL       DL,4
                OR        DL,CL             ; 组成压缩 BDC 码
                MOV    RESTLT [DI]，DL
                INC    DI
                JMP    NEXT
```

```
EX IT:      MOV    AH,4CH
            INT    21H
            END    START
```

3.7 已知乘数和被乘数分别存放在 ECX ： EBX 与 EDX:EAX 中。编写一个程序完成 64 位无符号乘法，将 64 位积存放在 EDXECX:EBX:EAX 中。

解：(ECX:EBX)×(EDX:EAX)

$=(ECX\times 2^{32}+EBX)\times(EDX\times 2^{32}+EAX)$

$=ECX\times EDX\times 2^{64}+(ECX\times EAX+EBX\times EDX)\times 2^{32}+EBX\times EAX$

上式可用下列竖式表示：

```
                                        EBX×EAX
                                EBX×EDX             ;个位对齐 32 位
                                EBX×EDX             ;个位对齐 32 位
                        + ECX×EDX                   ;个位对齐 64 位
-------------------------------------------------------------------
ECX×EDX×2^64+(ECX×EAX+EBX×EDX)×2^32+EBX×EAX
```

即上述 64 位乘法可分解成 4 个 32 乘法和 128 位的加法，用乘法指令和多字节加法实现。下面用宏指令实现 32 位乘法，将 64 位乘法定义成过程. 程序如下:

```
;32 位乘法宏定义
        MUL32     MACRD   A1,A2          ;EAX ×A1 送 A2,A2+4 双字单元
        MUL       A1                    ;EAX × A1 送 EDX: EAX
        MOV       A2 , EAX              ;保存低 32 位
        MOV       A2, [4], EDX          ;保存高 32 位
        ENDM
DATA    SEGMENT
    DAT1      DD   12345678H,45633445H  ;定义乘数
    DAT2      DD   33335555H.66667777H  ;定义被乘数
    M U LECXED}  DD  2 DUP ( ? )        ;保存 ECX × EDX 的中间解
    MULECXEAX  DD  2 DUP (?)            ;保存 ECX × EAX 的中间解
    MULEBXEDX  DD  2 DUP(?)             ;保存 EBX× EDX 的中间解
    MULEBXEAX  DD  2 DUP (?)            ;保存 EBX ×EAX 的中间解
DATA    ENDS
CODE    SEGMENT
ASSUME  CS :CODE , DS : DATA
START:  MOV    AX,DATA                  ; 建立 DATA 段的可寻址性
        MOV    DS,AX
        MOV    EDX,DAT2+4               ; 建立被乘数 EDX ： EAX
        MOV    EAX , DAT2
        MOV    ECX,DAT1+4               ; 建立乘数 ECX:EBX
        MOV    EBX,DAT1
        CALL   MUL64                    ;调用乘法子程序
        MOV    AH,4CH
        NT     21H
MUL64   PROC
        PUSH   EAX                      ;保存 EAX,EDX
        PUSH   EDX                      ;(EDX) →EAX
```

```
        MOV     EAX,ECX
        MUL32   EDX, MULECXEDX              ;EDX ×ECX 送 MULECXEDX
        POP     EAX
        MUL32   EBX, MULEBXEDX              ;EDX × EBX 送 MULEBXEDX
        POP     EAX                         ;恢复 EAX
        PUSH    EAX                         ;保存 EAx
        MUL32   ECX, MULECXEAX              ;EAX × ECX 送 MULECXEAX
        POP     EAX
        MUL32   EBX, MULEBXEAX              ;EBX ×EAX 送 MU LEBXEAX
          ;     EDX : ECX : EBX : EAX 结果初值
        MOV     EAX, MULEBXEAX
        MOV       EBX, MULEBXEAX[4]
        MOV     EDX,0
        MOV     ECX,0
      ;加 ECX 与 EAX 的乘积，个位对齐 32 位
        ADD     EBX,  MULECXEAX
        ADC     ECX ,  MULECXEAX [ 4 ]
        JNC     NEXT1
        INC     EDX;向 96 位进位
      ;加 EBX 与 EDX 的乘积，个位对齐 32 位
NEXT1:  ADD     EBX，MULEBXEDX
        ADC     ECX，MULEBXEDX[ 4]
        JNC NEXT2
        INC EDX                             ; 向 96 位进位
NEXT2:  ADD ECX，MULECXEDX
        ADC EDX，MULECXEDX[4]
       MUL64 ENDP
       CODE  ENDP
       END     START
```

3.8 一个十进制数 ABC(用压缩 BCD 码格式存放，低位在前)定义如下:

ABC DB 78H,56H,34H,12H 编写一个程序将 ABC 转换成 ASGII 码。

解；将 ABC 转换成 ASCII 码，即将每位十进制数(BCD 码)转换成一位 ASCII 码.而 ASCII 码数与非压缩 t3CD 数的区别仅在于高 4 位不同，前者为}f }，后者为 anon。所以，将非压缩 BCD 数转换成 ASCII 码数，只将非压缩 BCD 数的高 4 位置为 0011。假定 ASCII 码的存储顺序为高位在前，低位在后，用简化段定义，转换程序如下:

```
      ·MODEL  SMALL
      . STACK
      .DATA
ABC   DB 78H，56H，34H,12H

RESULT  DS 8DUP（？）
     .  CODE
START :  MOV AX，@ DATA
```

```
MOV DS，AX
LEA SI，ABC+3              ;从高位开始转换
LEA DI，RESULT             ;结果从低地址开始存放
MOV DX，4                  ;循环计数器
NEXT；MOV AL，[SI]
MOV CL，4                  ；先转换高位
SHR AL，CL
    OR      AL,30H         ;转换成 ASCI I 码数
    MOV     [DI],AL
     INC     DI
  MOV     AL，[ SI ]
  AND     AL，0FH    ;转换低位
  OR      AL，30H    ;转换成 ASCI I 码数
  MOV     [DI],AL
      INC     DI
  DEC     SI         ;指向下一字节单元
  DEC     DX         ;循环计数器减 1
  JNZ     NEXT
  MOV     AH,4CH
  INT     21H
  END     START
```

3.9 编写一个程序，将一个 64 位二进制数转换成 ASCI I 码字符串。

解:将一个 64 位二进制数转换成 ASCI I 码字符串，即将每位二进制数转换成一个 ASCI I 码字符。假定转换所得 ASCI I 码字符串，按二进制数高位在前，低位在后的顺序存放，用移位指令实现，转换程序如下:

```
          ″486
          ·MODEL  SMALL
          .STACK
          ·DATA
   NUM      DD    12345678H,23456789H
   RESULT   DB  64 DUP(?)
·CODE
          START：MOV AX，@DATA
          MOV DS,AX
          MOV ESI,4                      ;从高位开始转换
          MOV EDI,0                      ;结果从低地址开始存放
          AGAIN MOV EAX,NUM[ESI]         ;取长 32 位二进制数
          MOV ECX,32                     ;置循环计数器初值
          L1: SHL EAX,1                  ;最高位移人进位
          JC NEXT
          MOV RESULT[EDI],30H            ;最高位为 0，转换为’0
          JMP LP
          NEXT: MOV     RESULT[EDI],31H  ;最高位为 1.转换为’1
```

```
LP: INC     EDI                 ;EDI 指向下一结果存放单元
    LOOP    L1
    SUB     ESI,4               ;指向 b4 位二进制数低双字单元
    JNC     AGAIN               ;(ESI)}0，继续
    MOV     AH,4CH
    INT     21 H
    END     START
```

3.10 编程判断输入的 ASCI I 码字符是数字还是字母，并将判断结果分别以“D”和”L”显示.

解:用宏实现字符显示。程序如下:

```
DISPLAY  MACRD  CHAR;显示字符宏定义
         MOV    DL,CHAR
         MOV    AH,02H
         INT    21H
         ENDM
CODE  SEGMENT
ASSUME  CS: CODE
START :  MOV    AH, 01H             ;输人一个 ASCI I 码字符
         INT    21 H
         CMP    AL, '0'
         JC     ERROR               ;非数字和字母，转 ERROR 处理
         CMP    AL, '9'
         JA     NEXT                ;非数字，转 NERT
         DISPLAY   'D'              ;数字。显示字母 D
         JMP    EX1T
NEXT :   CMP    AL, 'A'
         JG     ERROR
         CMP  AL, 'Z'
         JBE    DISP _ L            ;字母，转 DISP_  L
         CMP    AL,' a'
         JC     ERRUR               ;非字母，转 ERRDR
         CMP    AL.' z'             ;(AL)成' Z'
         J BE   DISP_  L            ;字母，转 DISP _ L
ERROR:   DISPLAY   'E'              ;显示字母 E,表示非字母和数字
         JMP    EXIT
DISP_ L:    DISPLAY   'L'           ;显示字母 L
EKIT:    MOV    AH,4CH
         INT    21H
CODE        ENDS
END      START
```

3.11 如果依次输人的是一个数字和一个字母，则输出"RIGHT"标记后结束，否则输出标记“ERROR”后转向原出错处重做。试编程实现该功能。

解:用宏指令显示字符串.程序如下:

```
DISPLAY   MACRD     ADDR            ;显示字符串宏定义
```

```
                MOV     DX,OFFSET ADDR
                        MOV     AH,09H
                        INT     21H
                        ENDM
        DATA SEGMENT
        DISP_RIGHT    DB  'RIGHT',0AH,4DH,' $'
        DISP_ERRDR    DB  'ERRQR',0AH,0DH,'$'
        DATA  ENDS
        CODE  SEGMENT
        ASSUME  CS: CODE, DS: DATA
        START:      MOV   AX,DATA
                    MOV   DS,AX
        AGAIN;      MOV   AH,.01H                  ;输人一个A5CI 1码字符
                    INT   21H
                    CMP   AL, '0'
                    J C   DISP_  E                 ;非数字，转显示' ERRDR'
处理
                    CMP   AL,'9'
                    JA      DISP_  E               ;非数字，转显示'ERROR'
处理
        AGAIN1:     MOV     AH,01H                 ;输人一个ASCI I码字符
                    INT 21H
                    CMP AL,'A'
                    JC DISP_E1                     ;非字母，转显示' ERROR'
处理

                    CMP AL,'Z'
                    JBE DISP_R                      ;字母，转显示' RIGHT'
处理
                    NEXT: CMP  AL,' a'
                    JC      DISP_R                 ;非字母，转显示' ERROR'
处理
                    CMP      AL.' z'
                    J BE    DISP_ R                ; 字母，转显示' RIGHT'
处理
                    DISP_E1:DISPLAY DISP_ERROR     ;显示' ERROR'
                    JMP AGAIN1                     ;重输入字母
                    DISP_E:DISPLAY DISP_ERROR      ;显示'ERROR'
                    JMP AGAIN                      ;重输入数字
                    DISP_R:DISPLAY DIAP_RIGHT      ;显示'RIGHT'
                    MOVE AH,4CH
                    INT 21H
                    CODE ENDS
                    END START
```

3.12 编写一个程序计算 X^3 十 Y^3. 要求用子程序计算数的立方。

解:此题要根据 X,Y 为有符号数还是无符号数正确选用乘法指令。

(1)用子程序调用，假定 X,Y 为有符号数，程序如下:

```
        .486
        DATA  SEGMENT
          X   DB    68H
          Y   DB    OE6H
          RESULT    DD?
        DATA  ENDS
        CODE   SEGMENT
             ASSUME   CS : CODE ,  DS :DATA
         START:    MOV    AX,DATA
                   MOV      DS,AX
                   MOV      AL,  X
                   CALL     CUBICAL                  ;调用 CUBICAL 计算 X 的立方
                  MOV    EBX . EAX
                  MOV    AL,Y
                  CALL    CUBICAL
                  ADD    EAX,EBX
                  MOV     RESULT,EAX
                  MOV     AH,4CH
                  INT     21H
        ;以下为计算立方的子程序
                CUBICAL   PROC                      ;人口参数为 AL, EAX 返回结果
                PUSH   BX                           ;保存程序中用到的寄存器
                PUSH   DX
                MOVSX   BX,AL                       ; AL 扩充为等值 16 位乘数
                IMUL   AL
                IMUL   BX
                PUSH   DX                           ;用堆栈指令实现(DX:AX)→EAX
                PUSH   AX
                POP    EAX
                POP    DX                           ;恢复现场
                POP    BX
                RET
                CUBICAL   ENDP
                CODE   ENDS
                END   START
```

3.13 编写一个程序计算 X^3 十 Y^3. 要求用宏定义计算数的立方。

解:用宏调用，似定 x,Y 为无符号数，程序如下:

```
    CUBICAL   MACRO   PRAMT
                 MOV      AL,PRAMT
                MOVZX     BX,AL                     ;AL 扩充为等值 16 位乘数
```

```
        MUL     AL
        MUL     BX                      ;乘积存放在 Dx:Ax 中
        PUSH    DX
        PUSH    AX
        POP     EAX

        ENDM
    .486
.  MODEL  SMALL
.STACK
.DATA
X    DB     80H
Y    DB     0C6H
RESULT  DD?
            ·CODE
START:    MOV      AX, @DATA
 MOV      DS,AX
 CUBICAL  X                       ;调用宏计算 X³
 PUSH   EAX                       ;保存 X³
 CUBICAL  Y                       ;调用宏计算 Y³
 POP    EBX                       ;恢复 X³
 ADD    EAX,EBX
 MOV    RESULT,EAX
 MOV    AH,4CH
 INT    21H
 END    START
```

3.14 以 DATA 为首地址的区域，是程序中开辟的一块字节型排队存储区。请设计子程序，完成图 3.7 所示的数据推移，要求每调用一次，数据推移一个单元。

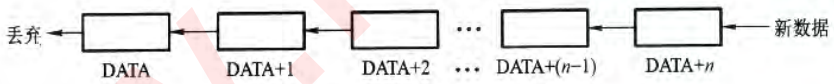


图 3.7　习题 3.14 图

解:假定主、子程序用 BL 寄存器传递新数据。满足要求的子程序如下:

```
INPUT DATA PROC
        PUSH  SI
        PUSH  CX
        MOV    SI , OFFSET  DATA          ;取数据区首址
        MOV    CX, CNT                    ;取数据区长度
              DEC    CX
AGAIN:    MOV    AL,[SI+1]                ;读取下一数据
          MOV    [ SI ] , AL              ;数据前移
          INC    SI
          LOOP   AGAIN
          MOV    [SI],BL                  ;用新数据充埃最后一
```

单元

```
                POP     CX
                POP     SI
                  RET
        INPUT_ DATA       ENDP
```

3.15 STRI 和 STR2 是两个字符串，两字符串的结束符都为’ $ ‘，试编写一个程序比较这两个字符串是否等长且相同，若是，则显示’ MATCH’ ,否则显示’ NOMATGH’ o

解:逐个字符比较.直到检侧到’ $’，若某个字符不相等，刚不匹配。

```
        ·MODEL SMALL
        .STACK
        . DATA
STR1    DB  ’WATERLOO BRIDGE IS A HEATBREAKING STORY’,‘$’
STR2    DB  ’WHICH HAPPENED DURING WORLD WAR ONE’,’ $’
MSG1    DB  ’MATCH’,0AH,0DH.‘$’
MSG2    DB  ’NO MATCH’. ’.0AH.0DH,’$’
        . CODE
START:      MOV     AX, @ DATA
            MOV     DS, AX
            MOV     S1，0
NEXT:       MOV     AL,STR1[S1]
            CMP     AL,STR2[S1]
            JNE     NOMATCH                 ;不匹配结束
            CMP     AL, ‘$ ‘                ;比较完?
            JE      MATCH                   ;匹配结束
            INC     SI
            CMP     NEXT                    ;继续比较
MATCH:      MOV     DX,OFFSET MSG1          ;两串匹配，显示’MATCH’
            JMP     DISP
NOMATCH;    MOV     DX,OFFSET MSG2          ;两串不匹配。显示’
NOMATCH’
DISP:       MOV     AH,09H                  ;用 09H 号 DOS 功能调用
显示字符串
            INT 21H
            MOV     AH,4CH
            INT     21H
            END     START
```

3.16 三个数据区 TW ,SH?C 和 YY 分别存放着 30 个学生的三门功课的成绩(百分制)，用子程序计算每个学生的三门功课的平均分，将平均分放在第四个数据区 PJ 中。

解:程序如下:

```
DATAREA  SEGMENT;定义数据段
    YW      DB 30 DUP(?)            ;存放第一门功课的成绩
    SHX     DB   30 DUP (?)         ; 存放第二门功课的成绩
    YY      DB    3D DUP(?)         ;存放第三门功课的成绩
    PJ      DB  30 DUP (?)          ;存放平均成绩
```

```
DATAREA     ENDS
 CODE   SEGMENT
 ASSUME   CS：CODE，DS ：  DATAREA
 MAIN      PROC   FAR                    ;主程序定义为远过程
 START:PUSH   DS                          ;标准序
 SUB AX，AX
 PUSH AX
 MOV AX，DATAREA                    ;建立 DATA REA 段的可寻址性
 MOV DS，AX
 MOV SI，0                           ;SI 作循环计数器和传递数组下标
 AGAIN:CALL PADD                    ;调用 PADD 求平均成绩
 INC SI                             ;指向下一元素
 CMP SI，30
 JB AGAIN                           ；未计算完，继续
 RET
 MAIN    ENDP
 PADD    PROC                       ;定义计算平均成绩的子程序
 PUSH CX                            ;保存程序中用到的通用寄存器
 PUAH AX
 MOV AL，YW[SI]                     ；取第一门功课的成绩
 AND AX，0FFH
 ADD AL，SHX[SI]                    ；与第二门功课的成绩相加
 ADC AH，0
 ADD AL，YY[SI]                     ;与第三门功裸的成绩相加
 ADC AH，0
 MOV CL，3
 DIV CL                             ；计算平均成绩
 MOV     PJ[SI] ，AL                ;保存结果
  POP     AX
  POP     CX
  RET                               ;返回调用程序
 PADD         ENDP
 COVE         ENDS
 END     START
```

3.17 编写一个程序从键盘输人 4 位十六进制数的 ASCII 码，并将其转换成 A 位十六进制数存人 DX 寄存器中。

解:无回显键盘字符翰人用 DOS 的 08H 号功能调用，输出用 02H 号功能调用，若输入为小写转换成大写显示

```
    CODE   SEGMENT
    ASSUME   CS: CODE
START:    MOV     CX,4                ;设循环次数
          MOV     BX,0
AGAIN;    MOV     AH,08H              ;输人一个 ASCII 码字符
          INT     21H
```

```
                CMP     AL，‘0’
                JC      AGAIN                   ;无效十六进制数，放弃.
重输人
                CMP     AL, ’9’
                JA      L1                      ;非数字，转 L1 大写字
母处理
                MOV     DL., AL                 ;保存 ASCI I 码，以供
显示
                AND     AL.0F                   ;转换成十六进制数 A~F
                JMP     NEXT
        L1:     CMP     AL，’A’
                JC      AGAIN                    ;无效十六进倒数.放
弃，重输人
                CMP     AL,’F’
                JA      L2                      ;转 L2 小写字母处理
                MOV      DL,AL                  ;保存 ASCI I 码 Y 供显
示
                SUB     AL,37H                  ;转换成十六进制数 A~F
                JMP     NEXT
        L2:     CMP      AL，’a’
                JC      AGAIN                   ;无效十六进制数，放弃.
重输人
                CMP      AL, ’f’
                JA      AGAIN                   ;无效十六进制数，放弃.
重输人
                SUB      AL,32H                 ;将小写转换成大写
                MOV      DL, AL                 ;保存 ASCI I 码，以供
显示
                SUB      AL,37H                 ;转换成十六进制数 A~F
        NEXT:   MOV      AH,02H
                INT      21H                     ;显示输入十六进制字
符
                PUSH     CX                     ;以下形成 4 位十六进制
数
                MOV      CL,4
                SHL      BX , 4
                OR       BL,AL
                POP      CX
                LOOP     AGAIN
                MOV      DX,BX
                MOV      AH,4CH
                INT      21H
        CODE    ENDS
                END      START
```

3.18 编写一个程序将 EAX 的内容以十六进制格式显示在计算机屏幕上。

解:此题应先将 EAX 的内容转换成十六进制 ASCI I 码数串，再用 09H 号功能调用显示。

```
        .486
        ·MODEL   SMALL
        ·STACK
        .DATA
ASC     DB     8 DUP ( ? )              ;存放 ASCI I 码数串，高位在前
        DB     OAH, ODH, ' $ '          ;定义回车、换行和字符串结束符
        .CODE
START:   MOV       AX,@DATA
         MOV       DS,AX
         LEA       DI,ASC
         MOV       CX,8
NEXT:    ROL       EAX,4                ;高 4 位移至低 4 位
         MOV       BL,AL                ;取 1 位 16 进制数
         AND       BL,0FH
         CMP       BL,0AH
         JC        CON1
         ADD       BL,37H          ;A~F→'A'~'F'
        JMP CON12
       CON1: ADD   BL,30H
       CON12: MOV  [DI],BL
       INC   DI
       LOOP   NEXT
       MOV   DX,OFFSET ASC
       MOV   AH,09H
       INT   21H                        ; 显示
       MOV   AH,4CH
       INT   21H
       ENF   START
```

3.19 输入一串以'$'结束的字符到 DAA 中，且逐个将 DAA 变量中的字符传送到 DBB 变量中，当遇到'$'传送结束，并且不将'$'传给 DBB 中。

解:带回显的字符输人用 DDS 的 UIH 号功能调用。

```
      DATA SEGMENT
        DAA   DB   258 DUP(?)
        DBB   DB   256 DUP(?)
      DATA   ENDS
      CODE   SEGMENT
          ASSUME   CS:CODE , DS :DATA
      START:    MOV          AX,DATA
                MOV          DS. AX
                MOV          ES. AX
                CLD
                LEA          DI , DAA          ; 取字符串首址
```

```
            LEA         BX . DAB
AGA I N:    MOV         AH,01H              ;输人并显示一个 ASCI I 码字
符
            INT         21H
            STOSB;AL 送 DAA 串
            CMP         AL,'$'
            JZ      EXIT                    ;等于，转 EXIT,结束
            XCHG    BX , DI                 ;DI 指向 DBB 级冲区
            STOSB                           ;AL 送 DBB 串
            XCHG    BX,DI                   :恢复 DI , BX 指向 DAA , DBB
            JMP     AGAIN
EXIT:       MOV     AH,4CH
            INT     21H
CODE        ENDS
            END     START
```

3.20 编写用递归子程序方法求前 30 个斐波那契数的程序。斐波那契数的定义如下:

$F_0=0$

$F_1=1$

$F_n=F_{n-1}+F_{n-2}$

解:设计递归子程序时，主、子程序间的参数(人口参数和返回的函数值)通常要用堆栈传递，这时要注意两个问题:

(1)主程序参数压栈在 CALL 指令之前,进人子程序后,为了保护现场还随有其他压栈操作,所以主程序压人的参数不在栈顶，而是位于程序返回地址的下面。因此，子程序对参数的引用，不能使用 POP 弹栈指令。通常用 BP 指针相对于栈顶的基址寻址，使用时要正确计算参数相对于栈顶的位移量。

(2)返回指令 RET 要带参数，以便返回的同时，废弃栈顶的参数。

下面采用堆栈参数传递编写此递归子程序。主程序要传递给子程序的参数包括斐波那契数序号和返回函数值存放单元的地址。将栈顶内容定义成结构，子程序使用结构引用来访问主程序传递的参数，程序如下:

```
Frame   STRU;子程序调用帧结构
  SAVE_   BP          DW?                 ;保存 BP 指针
  SAVE_   CS_  IP       DW   2 DUP ( ? )  ;保存 CS: IP 地址
  N                   DW?                 ;调用参数:斐波那契数序
号
  Result_   addr      DW?                 ;调用参数:返回结果单元
地址
   Frame   ENDS
DATAREA SEGMENT;定义数据段
  FBUF      DW        30 DUP ( ? )        ;存放斐波那契数
  RESULT    DW?
  DATAREA     ENDS
  CODE  SEGMENT
 ASSUME   CS:CODE,DS:DATAREA
```

```
MAIN        PROC    FAR                 ;主程序定义为远过程
START:      PUSH    DS                  ;标准序
            SUB     AX,AX
            PUSH    AX
            MOV     AX,DATAREA          ;建立 DATA REA 段的可寻址性
    MOV DS,AX
    MOV CX,0                            ;Cx 作循环什数器和斐波那契数序号
    LEA DI,FBUF                         ;DI 改指向 F0存放单元
    AGAIN:LEA SI,RESULT                 ;建立 FF( N)调用参数
    PUSH SI                             ;压返回结果单元地址
    PUSH CX                             ;压斐波那契数序号
    CALL FAR PTR FF                     ;计算斐波那契数
    MOV AX,RESULT                       ;以下保存结果
    MOV [DI],AX
    INC DI
    INC DI
    INC CX
    CMP CX,30
    JBE AGAIN                           ;未计算完，继续
     RET
    MAIN  ENDP
    FF   PROC  FAR;定义计算斐波那契数的递归子程序
    PUSH  BP
    MOV  BP,SP                          ;BP 为调用帧结构指针
    ;取人口参数
    MOV  SI,[BP].Result_addr            ;结果存放地址
    MOV  AX,[BP].N                      ;取斐波那契数序号 N
    CMP  AX,1
    JA  TESTN;
    MOV  [SI],AX
    JMP  EXIT2;结束
    TESTN:  DEC  AX                     ;建立计算 FF(n 一 1)的调用参数
    PUSH  SI                            ;结果存放地址
    PUSH  AX                            ;取斐波那契数序号 N-1
    CALL  FF                            ;递归调用 FF 计算 Fn-1
    PUSH  WORD  PTR[SI]                 ;保存 Fn-1，
    PUSH  SI                            ;结果存放地址
    MOV   AX, [BP].N
    SUB  Ax,2;N-2
    PUSH   AX                           ;压斐波那契数序号 N 一 2
    CALL   FF                           ;递归调用 FF 计算 Fn-2
    POP    AX                           ;恢复 Fn-1
```

```
    ADD     [SI],AX Fn-1+ F n-2
    EXIT2:POP     BP
          RET     4                    ;废弃栈顶参数，返回调用程序
    FF        ENDP
   CODE      ENDS
   END     START
```

3.21 若数组 ARRAY 在数据段中已做如下定义:

```
ARRAY  DW  100 DUP[?]
```

试指出下列语句中各操作符的作用，指令执行后有关寄存器的内容是多少。

```
MOV  BX,DFFSET  ARRAY
MOV  CX,  LENGTH  ARRAY
MOV  SI,0
  …
ADD  SI,TYFE  ARRAY
```

解:OFFSET 操作符是取变 t 的偏移地址;TYPE 操作符是取变 t 的类型;LENGTH 操作符是取变量的长度。指令执行后:

(BX)=教组 ARRAY 的偏移地址;

(CX）=数组 ARRAY 的长度=100;

(SI)=当前 SI 的值+2，即 SI 指向当前数组元素的下一元素。

3.22 某 8086 单板机系统共可接收 10 个键盘命令(分别为 A,B,C,..,J].实现这 10 个命令功能的程序分别为过程 PO.P1,″..,F9。试编程从键盘接收命令，并转到相应的过程去执行。要求用两种方法:

(1)用比较、转移指令实现。

(2)用跳转表实现。

解:（1）用比较、转移指令实现分支的程序如下:

```
DATA  SEGMENT
  …
DATA  ENDS
CODE    SEGMENT
ASSU ME CS: CODE , DS: DATA
START:  MOV   AX,DATA
        MOV    DS,AX
INPUT:     MOV    AH,1                ;输人键盘命令（键的 ASCI I 码)送 AL
           INT    21H
           CMP    AL，'A'             ;为 A 命令?
           JZ    PO                   ;是，转过程 PO
           CMP    AL, 'B'
           JZ     P1
           CMP    AL.'C'
           JZ     P2
           CMP    AL, 'D'
           JZ     P3
           CMP    AL，'E'
```

```
        JZ      P4
        CMP     AL, 'F'
        J Z     P5
       CMP     AL, 'G'
        JZ      P6
        CMP     AL, 'H'
        JZ      P7
       CMP     AL , ' I'
        JZ      P8
       CMP     AL. 'J'                  ;为 J 命令?
        JZ      P9                      ;是，转过程 P9
       J MP    INPUT                    ;转重输人字符
       P0: …
           …
       P1:    …
              …
       P9     …
              …
       CODE   ENDS
       END    START
```

（2）采用跳转表实现时，又可分为表内地址分支和表内指令分支、编程时要注意区别两种结构跳转表的定义方法和正确的寻址方式，表内地址分支在数据段定义跳转表.用存储器间接寻址;表内指令分支在代码段定义跳转表，用寄存器直接寻址。以下给出两种方法:

①用表内地址分支

```
            DATA   SEGMENT
BASE   DW     P0,P1,...,P9;定义跳转表
DATA       ENDS
CODE     SEGMENT
ASSUME   CS: CODE, DS: DATA
START:MOV       AX,DATA
      MOV       DS, AX
INPUT:    MOV     AH, 1                  ;输入健盘命令(健的 ASC.I I 码)送 AL
          INT       21 H
          CMP       AL, 'A'              ;小于' A'?
          J B      INPUT                 ;小于重输入
          CMP       AL, 'J'
          JA        INPUT
          SUB      AL, 'A'               ;'A' ~' J' 转换为跳转表下标 0--9
          MOVZX      BX.,AL              ;跳转表下标扩展为 16 位基址
          SHL       BX, 1                ;按字对齐跳转表项(BX)×2
          JMP      BASE[BX]              ;转处理程序
```

```
P0:   …
      …
P1:   …
      …
P9:   …
      …
CODE      ENDS
END       START
```

②用表内指令分支

```
DATA  SEGMENT
     …
DATA ENDS
CODE  SEGMENT
ASSUME  CS: CODE, DS: DATA
START:     MOV  AX , DATA
           MOV   DS,AX
INPUT:     MOV    AH，1          ;输入健盘命令(健的 ASC.ＩＩ码)送 AL
           INT      21 H
           CMP      AL, ’A’      ;小于’ A’?
           J B      INPUT        ;小于重输入
           CMP      AL, ’J’
           JA       INPUT
           SUB      AL,’A’       ;’A’~’J’转换为跳转表下标 0--9
           AND  AX,0FH           ;跳转表下标扩展为 16 位
;按 3 字节对齐跳转表项(AX)× 3
           MOV    BX,  AX
           SHL    BX,1
ADD        AX,BX
MOV        BX,OFFSET BASE        ;取跳转表地址
ADD        BX , AX               ;计算表内跳转指令地址
JMP        BX                    ;转处理程序,用寄存器间接转移
JMP        P0;定义跳转表
JMP    P1
 …
JMP        P9
CODE     ENDS
END  START
```

3.23 若已定义以下数据段:

```
DATA  SEGMENT
    BUF    DB   100 DUP(?)
      GOOD    DB?
```

```
PASS    DB?
BAD     DB?
AVRG    DB?
...
DATA    ENDS
```

若已将某年级 100 名学生徽机原理课的成绩以压缩 BCD 数的形式存人变量 BUF 中，试编写程序段统计成绩高于 80 分、低于 60 分和介于 60 分至 80 分之间的学生人数，仍以压缩 BCD 数形式将这三个数分别存人 GOOD,BAD 和 PASS 变量中(假定任一档的人数都不达到 100 人)，并计算全年级平均成绩，也以压缩 BCD 形式存人变量 AVRG 中(假定平均成绩低于 100 分，且舍去小数点后的数)。

解:此题统计计数和求总成绩均要用 BCD 加法.而对以 BCD 形式表示的十进制数，被 10^N 除的除法可用移位操作实现，即右移 n 位 BCD 数，也即对压缩 BCD 码右移 4×n 位二进制数，而对非压缩 BCD 码是右移 8× n 位二进制数。下面用宏实现 BCD 统计计数，程序如下:

```
;定义 BCD 加 1 计数宏
BCD _ ADD_1    MACRG    ADDR                ; ADDR 作 BCD 计数器
               MOV   AL,ADDR                ;取计数器数据
               ADD   AL,1                   ;计数器加 1
               DAA                          ; BCD 数调整
               MOV   ADDR , AL              ;保存计数器值
                     ENDM
          DATA  SEGMENT
          BUF   DB  100  DUP(?)
          GOOD    DB?
          PASS    DB?
          BAD     DB?
          AVRG    DB?
  DATA    ENDS
  CGDE    SEGMENT
    ASSUME  CS: CODE, DS: DATA
  START;        MOV          AX,DATA
                MOV           DS,AX
               MOV            BX,0          :求和寄存器 I3}清 0
               MOV      SI,OFFSET  BUF
               MDV      PASS,0               ;及格计数器初值为 4
               MOV      BAD,0                ;不及格计数器初值为 f1
               MOV      GOOD, 0              ; 80 分以上计数器初值为
0
               MOV      CX,100
AGAIN:         MOV      AL,[SI]              ;取成绩数据
               CMP       AL,  80H            ;大于 80 分?
               JC        NEXT                ;小于$D 分，转 NEXT
               BCD_  ADD_ 1 GOOD             ;统计 80 分以上人数
NEXT;          CMP       AL,60H              ;大于等于 6}分?
               JNC       NEXTO               :大于等于，转及格计数
```

处理

```
                    BCD_   ADD_   1   BAD                ;统计 60 分以下人数
                    JMP      NEXT1
          NEXT0:    BCD_   ADD_   1   NUM1               ;统计 60 分至 80 分之间
的人数
          NEXT1:    ADD      AL,  BL                     ;求总成绩
(BX)+(AL)--( BX)
                    DAA
                    MOV      BL, AL
                    MOV      AL, BH
                    ADC      AL ,0
                    DAA
                    MOV      BH, AL
                    INC      SI                          ;指向下一成绩数据
                    LOOP     AGAIN
                    MOV      AVRG, BH                    ;BH 为平板均成绩
                    MOV      AH, 4CH
                    INT      21H
          CODE      ENDS
                    END      START
```

3.24 编制一个程序,其功能是将一个字数组 ARRAY 的正数和负数分开存放于从 PLUS 和 MINUS 开始单元中，并在屏幕上显示出正数和负数的个数。设该教组长度放在数组的第一个字单元中。

解:正数和负数可通过符号位侧试分检，而数据显示，必须将数据转成 ASCII 码字符或串后，用相应的 DOS 功能调用物出。下面用子程序完成数据转换和显示，程序如下:

```
          DATA  SEGMENT
              ARRAY     DW   100, 78H. OFFB4H, OFBH, 52H. ...
              COUNT     EQU  $-ARRAY
              PLUS      DW   COUNT/2  DUP(?)
              MINUS     DW    COUNT/2  DUPt?)
              PLUS _ CT    DW   0
               MINUS_  CT   DW   0
             ;定义显示数据
              DISP  PLUS  DB  'PLUS=',4 DUP(?),ODH,OAH,$'
              LISP  MINUS DB   'M INUS= ',4 DUP(?),ODH,OAH,' $'
              DATA ENDS
    STACK  SEGMENT  STACK 'STACK
    DB  100  DUP(?)
    STACK  ENDS
    CODE  SEGMENT  PARA  'CODE'
ASSUME  CS:CODE,SS:STACK,DS:DATA,ES:DATA
    START   FROC  FAR;主程序定义成远过程
              PUSH  DS                    ;定义标准序
             SUB   AX,AX
```

```
                PUSH    AX
                MOV     AX, DATA                ;建立数据段、附加数据段的可寻址
性
                MOV     DS, AX
                MOV     ES ,AX
                LEA     SI,ARRAY                ;取数组首地址
                LEA     DI , PLUS               ;取正数缓冲区首地址
                LEA     BX. MINUS               ;取负数缓冲区首地址
                MOV     CX, [SI]                ;取数组长度
                ADD     SI,2                    ;SI 指向数组第一个元素
                MOV     PLUS_ CT,0              ;统计正数计数器情。
                MOV     MINUS_ CT,0             ;统计负数计数器清 A
                CLD                             ;清方向标志
L I :   LODSW                                   ;取一个数组元素
                TEST   AX , 8000H               ;数组元素≥0?
              JNZ    L2                         ;负数，转负数存储处理
                STOSW                           ;保存正数
              INC    PLUS_ CT                   ;正数计数器加 1
                  JMP    AGAIN
L2;     XCHG    BX,DI                           ;DI 指向负数缓冲区
                STOSW                           ;保存负数
              INC     MINUS_ CT                 ;负数计数器加 1
                XCHG BX, DI                     ;恢复 DI , BX 指向正、负数缓冲
区
AGAIN:LOOP    L1
              ;将正教个数转换成 ASC I I 码串送显示缓冲区
MOV  AX,PLUS_CT
LEA    DI, DISP _ PLUS +5
CALL   TURN
:将负数个数转换成 ASCI I 码串送显示缓冲区
START
TURN
MOV      AX,MINUS_CT
LEA      DI , D ISP_MINUS+6
CALL TURN
LEA      DX,DISP _ PLUS
CALL     DISP                                   ;显示正数个数
LEA      Dx , DISP_MINUS
CALL DISP                                       ;显示负数个数
RET
 START   ENDP
TURN  PROC                                      ;AX 转换成 ASCI I 码送 DI 所指单
PUSH  CX                                        ;保存子程序中用到的寄存器
PUSH  BX
```

```
    MOV   CH,4                          ;置循环计数初值
    MOV     CL.4
    NEXT;    ROL     AX ,CL             ;高 4 位移至低 4 位
    MOV     BL,  AL                     ;取 1 位十六进制数
    AND     BL.  0FH
    CMP     BL,  0AH
    JC      CON1
    ADD    BL,  37H
    JMP     CON12
    CON1:  ADD    BL,  30H
    CON12:  MOV  [DI ] ,  BL
    INC    DI
    DEC.    CH
    JNZ     NEXT
    POP     BX                          ;恢复子程序中用到的寄存器
    POP    CX
    RET
    TURN  ENDP
    DISP    PROC                        ;显示 DX 所指字符串
   MOV      AH,09H
    INT      21H;显示
    RET
  DISP         ENDP
  CODE         ENDS
  END      START
```

3.25 试编写一程序求级数 12+ 22+ 32+…的前几项和刚大于 1 000 的项数 N.

解:此题未知循环次数，要用条件“和大于 1 000”来控制循环结束。程序如下:

```
            DATA SEGMENT
                N  DB?
            DATA  ENDS
            CODE SEGMENT
              ASSUME  CS : CODE , DS : DATA
START:      MOV       AX,DATA
            MOV       DS,  AX
            MOV       BX,0              ;BX 存放级数前几项和，初值 0
            MOV       CL,0              ;存放项数 N，初值 0
AGAIN :     INC     CL                  ;项数 N 加 1
              MOV       AL,  CL
          MUL       CL                  ;计算第 N 项:Nz->AX
          ADD       BX,AX               ;求级数和
          CMP       BX,1000             ;和大于 1 000?
          JBE       AGAIN               ;不大于继续
          MOV       N,CL                ;保存项数 N
          MOV       AH,4CH
```

```
            INT     21H
CODE        ENDS
                END     START
```

3.26 设数据段中有三个变量单元 A,B 和 C 中存放有三个数，若三个数都不为 0，则求出此三数之和存人 SUM 单元;若有一个为 0，则将其他两个单元也清 0。请编写此程序。

解:用条件判断指令实现，程序如下:

```
        DATA        SEGMENT
            A     DW?
            B     DW?
            C     DW?
        SUM   DW   2 DUP(?)
DATA    ENDS
CODE    SEGMENT
    ASSUME   CS: CODE, DS:DATA
START:      MOV     AX,DATA
            MOV     DS,AX
            MOV     BX,0                ;CX:BX 存放三数之和，初值 0
            MOV     CX,0
            CMP     A, 0                ;A 等于 0?
            JE      CLEAR0              ;转 B,C 清 0 处理
            ADD     BX,A:求和
            ADC     CX, 0
            CMP     B, 0
            J E     CLEAR1
            ADD     BX,B
            ADC   CX ,0
            CMP     C,0
            J E     CLEAR2
            ADD     BX , C
            ADC     CX,0
            MOV     SUM [ 0 ] , BX      ;保存三数之和
            MOV     SUM[2],CX
EXIT:       MOV     AH,4CH              ;返回 DOS
            INT     21H
CLEAR0:     MOV     B,0
            MOV     C, 0
            JMP     EXIT
CLEAR1:MOV    A,0
            MOV     C,0
            JMP     EXIT.
CLEAR2:MOV    B, 0
            MOV     A,0
            JMP     EXIT
CODE      ENDS
```

```
    END     START
```

3.27 假设已编制好五个乐曲程序，它们的人口地址(含段首地址和偏移地址)存放在数据段中的跳转表 MUSICTAB 中。试编写一个点歌管理程序，其功能是:根据键盘输人的乐曲编号 0 ~ 4 转到所点乐曲的人口，执行此乐曲程序。

解:用回车键控制点歌管理程序结束。

```
                .486
        DATA SEGMENT
        DISP  DB  'INPUT ERROR!',0AH,0DH,'$'
        ;定义跳转表，地址为 32 位远指针(16 位段基址和 32 位偏移地址)
          MUSICTAB  DF  MUSIC0，MUSIC1，MUSIC2，MUSIC3 ，MUSIC4
        DATA  ENDS
        CODE SEGMENT
             ASSUME  CS:CODE,DS:DATA
        START :     MOV     AX , DATA
                    MOV     DS,AX
      INPUT:        MOV     AH,1                    ;输人乐曲编号送 AL
                    INT     21H
                    CMP     AL,0DH                  ;若输人回车符结束
                        JZ      EXIT
                    CMP     AL , 30H                ;小于' 0'?
                    JB      ERR                     ;小于转错误处理
                    CMP     AL,34H                  ;大于' 4'?
                    JA      ERR                     ;大于转错误处理
                    AND     EAX,0FH                 ;编号转换为跳转表下
标 0 一 4
                    SHL     EAX，1                   ;(EAX) × 2→EAX
                    MOV     EBX,EAX
                    SHL     EAX，1                   ;(EAX) × 4→EAX
                    ADD     EAX,EBX                 ;(EAX ) × 6→EAX
                    JMP     MUSICTAB[EAX]             ;转处理程序
        ERR:MOV     DX, OFFSET DISP
                    MOV     AH,09H
                    INT     21H                       ;显示错误信息
                    JMP     INPUT                     ;重输人
        MUSIC0:     ...
        MUSIC1:      ...
        MUSIG2:     ...
        MUSIC3:          ...
        MLISIC4:         ...
      EXIT:       MOV     AH,4CH
                  INT     21H
        CODE      ENDS
        END       START
```

3.28 编制一个能循环显示 4 条新闻标题的控制程序。每条新闻标题 NEW I ，

NEW2 ,NEW3,NEW4 及其人口地址表均存放在数据区中。人口地址表设为:

NEWTAB DW NEW1 ,NEW2, NEW3, NEW4

解:用变址寻址实现。并假定有任意键按下，程序停止显示。

```
DATA  SEGMENT
  NEW1      DB'.....', 0DH,0AH,' $'              ;定义新闻标题 NEW1
  NEW2      DB '.....'  0DH,0AH,' $''            ;定义新闻标题 NEW2
  NEW3      DB' ......' }0DH,0AH. ' $'           ;定义新闻标题 NEW3
  NEW4      DB' .. ....'0DH,0AH, ' $'            ;定义新闻标题 NEW4
  NEWTAB  DW    NEW 1, NEW2 , NEW3 . NEW4        ;定义人口地址表
DATA  ENDS
CODE SEGMENT
ASSUME  CS: CODE,DS: DATA
START:   MOV    AX,DATA
         MOV    DS, AX
AGAIN:   MOV    SI,0
NEXT:    MOV    AH,1                  ;扫描键盘
         INT    16H
         JNZ    RETURN                ;任愈键按下，退出
         MOV    DX, NEWTAB[ SI]       ;取新闻标题人口地址
         MOV    AH,09H                ;用 09H 号功能调用显示新
闻标题
         INT    21H
         INC    SI                    ; S1 指向下一新闻标题
         INC    SI
         CMP  SI,8                    ;显示完?
         JNZ  NEXT                    ;未完循环显示
         JMP  AGAIN                   ;显示完,从第一条重新开始
         RETURN:  MOV  AH,4CH
         INT  21H
         CODE  ENDS
                 END  START
```

4.4 试根据 EPROM 的接口特性，设计一个 EPROM 写人编程器的硬件电路和软件流程。

解:EPROM 写人编程时，需要加专门的编程电源(+25V)和编程脉冲，其编程脉冲宽度比 CPU 的读/写脉冲要宽的多，所以 EPROM 写人编程时要通过接口电路相连，以产生编程所需要的信号，接口电路可用简单的锁存器，亦可用可编程接口芯片。图 4.2 给出了一个用 8255A 作为 8086 CPU 和 2764 EPRDM 存储器编程的接口电路。2764 是 8K×8 位 EPROM 存储器芯片，其存取时间为 250 ns，其引脚含义为: $\overline{CE}$—片选线; $\overline{PGM}$—编程脉冲输人;V_{PP},—编程电源。图中.8255A 的 B 口用于输出编程数据，A 口和 C 口的 PC_C ~ PC_0 输出片内地址，P C_5, PC_6 用于产生 2754 的片选和编程脉冲。其软件流程如图 4.3 所示。

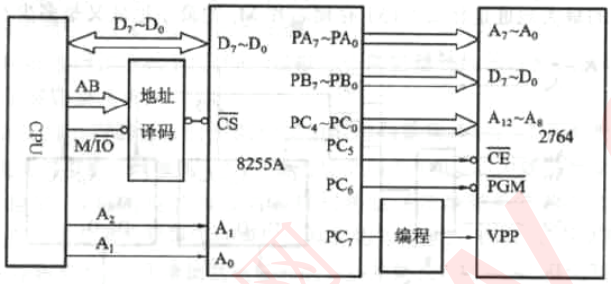


图 4.2 EPROM 与人编程器接口电路

4.8 在有 16 根地址总线的徽机系统中画出下列情况下存储器的地址译码和连接图。

(1) 采用 8K×8 位存储芯片，要形成 64 KB 存储器。

(2)采用 4K×4 位芯片，要形成 32 KB 存储器。

(3)采用 4K×1 位芯片，要形成 16 KB 存储器。

(4)若要设计一个 256 KB 的存储器系统，应怎么办?

解:(1)要构成 64 KB 存储器，需 8 个 8K×8 位存储芯片。先列出各存储芯片的存储空间分配位表，如表 4.2 所示。

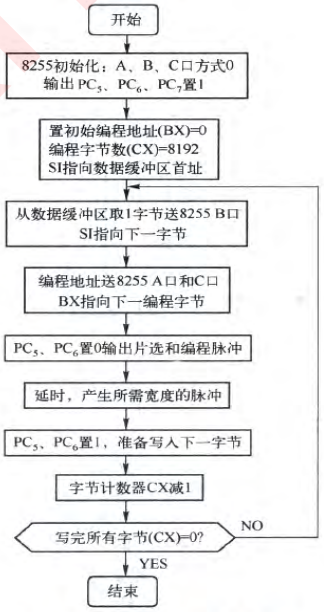


图 4.3 EPROM 编程程序流程

芯片号	A_{15} A_{14} A_{13}	$A_{12}A_{11}A_{10}A_9A_8A_7A_6$

0	0	0	0	0000H~1FFFH
1	0	0	1	0000H~1FFFH
2	0	1	0	0000H~1FFFH
3	0	1	1	0000H~1FFFH
4	1	0	0	0000H~1FFFH
5	1	0	1	0000H~1FFFH
6	1	1	0	0000H~1FFFH
7	1	1	1	0000H~1FFFH

于是，由表 4.2 可以确定存储器字扩展的译码方案:用 3 线一 8 线译码器(74L5138)对高位地址线 A_{15}~A_{13} 译码来产生 8 个存储芯片的片选信号。由此可画出存储器连接图，如图 4.4 所示。

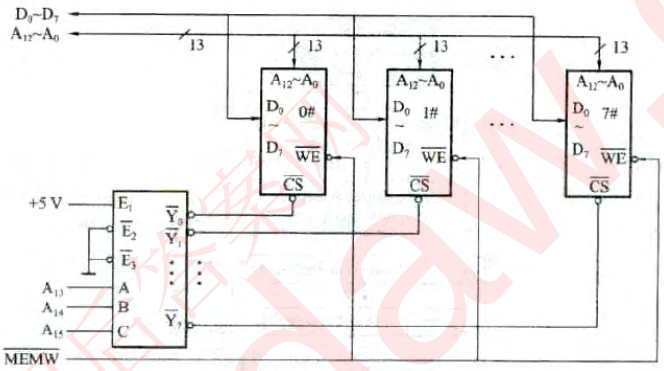


图 4.4　8 K×8 位芯片组成的 64 KB 存储器

(1) 采用 4K×4 位存储芯片时，由于字长不足名位，所以首先要进行位扩展.即以每两个芯片为一组构成 8 位字长的存储器，与 CPU 连接时，同一组芯片中，各芯片的片选线和地址线并联，而数据线则分别与 C.PU 的不同数据线连接。若要构成 32 KB 存储器.还需进行字扩展，需 8 组存储芯片。同样，先列出各存储芯片组的存储空间分配位表.如表 4.3 所示。

表 4.3 4K×4 位存储芯片构成的 32KB 存储器的存储空间分配位表

芯片号	A_{15} A_{14} A_{13} A_{12}	$A_{12}A_{11}A_{10}A_9A_8A_7$
0	0 0 0 0	0000H~1FFFH
1	0 0 0 1	0000H~1FFFH
2	0 0 1 0	0000H~1FFFH
3	0 0 1 1	0000H~1FFFH
4	0 1 0 0	0000H~1FFFH
5	0 1 0 1	0000H~1FFFH
6	0 1 1 0	0000H~1FFFH
7	0 1 1 1	0000H~1FFFH

译码方案可选择用 3 线一 8 线译码器（74LS138）对高位地址线 A_{14}~A_{12} 译码来产生 8 个存储芯片组的片选信号，而地址线 A_{15}，则用做译码器的使能控制线。按此方案设计的存储器连接图如图 4.5 所示。

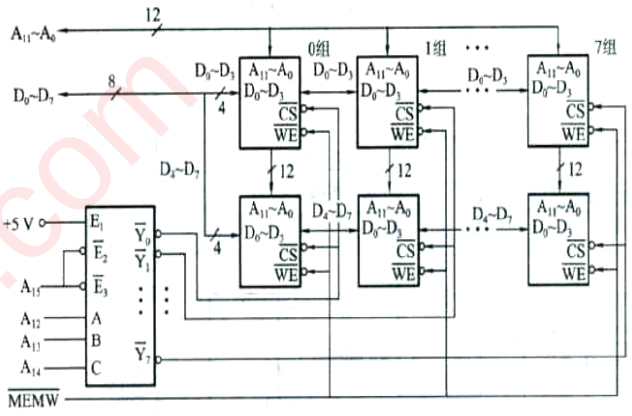


图 4.5　4 K×4 位芯片组成的 32 KB 存储器

(2) 采用 4Kx1 位芯片时，要 8 个芯片为 1 组构成 8 位字节存储器。要形成 16 KB 存储器需 4 组存储芯片。此时，可选择与(2)相同的译码方案，但只需 4 个片选信号。按此方案设计的存储器连接图如图 4.6 所示。

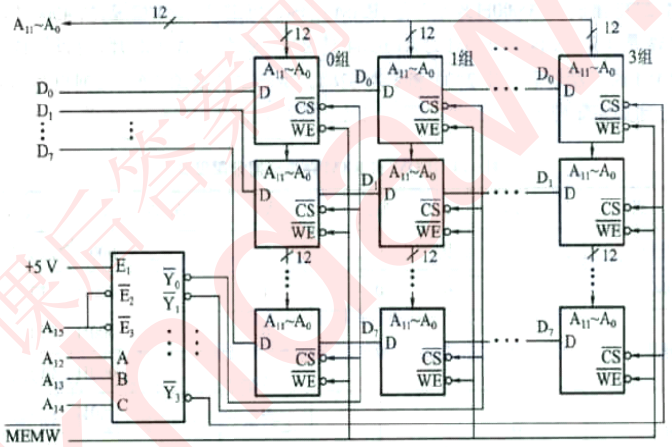


图 4.6　4 K×1 位芯片组成的 16 KB 存储器

(4]在只有 15 根地址线的徽机系统中，256 KB 的存储器已超出了徽机系统所能寻址的范围。此时。要设计 256 KB 的存储器系统需采用存储器扩充寻址，即将存储器划分成 4 个 64 KB 的存储模块，每个存储模块内部的寻址信号仍由 16 位地址总线控制，而存储器接口电路则要增加相应的块选控制逻辑来选择存储器模块。若以图 4.4 的 RAM 阵列作为 64KB 的存储器模块，则可用块选控制逻辑产生的块选信号作为译码器的译码使能信号.如图 4.7 所示。这时.只有块选控制逻辑选中时译码器才工作。

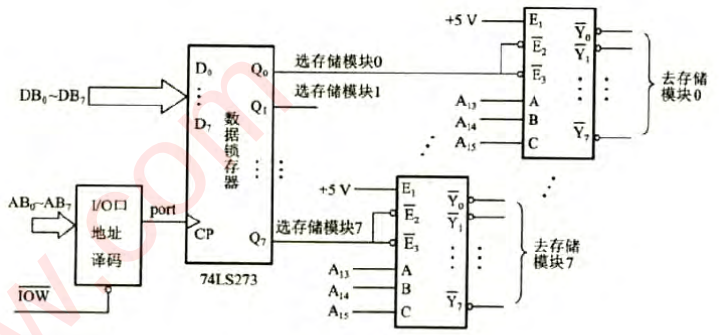


图 4.7　存储器扩充寻址示意图

4.9 试为某 8 位微机系统设计一个具有 8 KB ROM 和 40KB RAM 的存储器。ROM 用 EPRQM 芯片 2732 组成，从 0000H 地址开始。RAM 用 SRAM 芯片 6264 组成，从 4000H 地址开始。

解:2732 是.4 K×8 位 EPROM 芯片、6264 是.8 K×8 位 SRAM 芯片，所以构成 8 KB ROM 和 40 KB RAM 的存储器需 2 片 2732 和 5 片 6264。由给定的地址空间，可列出各芯片的存储空间分配位表，如表 4.4 所示。

表 4. 4　ROM 和 RAM 存储空间分配位表

芯片组	A_{15} A_{14} A_{13}	A_{12}	$A_{11}A_{10}A_9A_8A_7A_6A_5A_4A_3A_2A_1A_0$
EPROM0	0 0 0	0	000H~FFFH
EPROM1	0 0 0	1	000H~FFFH
RAM0	0 1 0	0	000H~FFFH
		1	000H~FFFH
RAM1	0 1 1	0	000H~FFFH
		1	000H~FFFH
RAM2	0 0 0	0	000H~FFFH
		1	000H~FFFH
RAM3	1 0	0	000H~FFFH
	1	1	000H~FFFH
RAM4	1 0	0	000H~FFFH
	0	1	000H~FFFH

由表可知，地址线设置可用 A_{15}~A_{13} 参与片选译码，A_{12}~A_0 用于选择片内存储单元。对 EPROM 而言，两个芯片占用同一片选信号，所以，此方案还需对 A_{12} 进行二次译码，产生 EPRQM 芯片的片选信号。

假定用 3 线一 8 线译码器对 A_{15}~A_{13} 译码，用或门对 A_{12} 进行二次译码，存储器连接如图 4.8 所示。

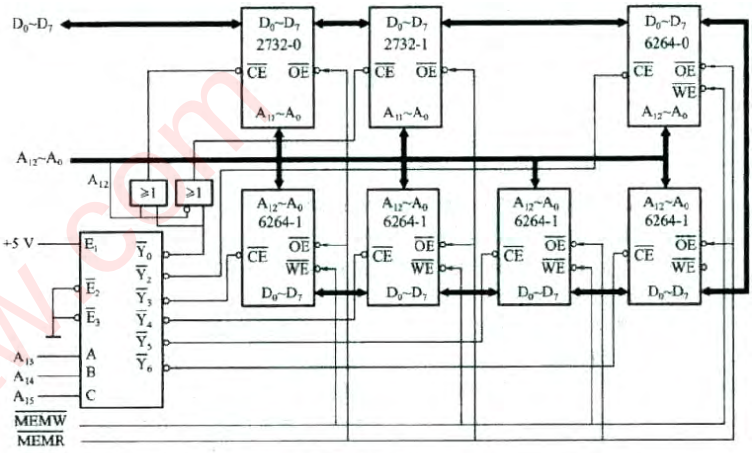


图 4.8 存储器连接图

4.10 图 4.9 所示为某 8086 徽机存储器的部分电路接线田。试分析:

(1)该存储器的总容量为多少字节?

(2} M_0和 M_1的地址范围分别为多少?

解:此存储器系统采用双体存储器结构，其中:A_0用于选偶存储体，$\overline{BHE}$ 选奇存储体。存储器芯片 M_0 ， M_1的片选逻辑表达式为:

$$\overline{CS_{M0}}=\overline{\overline{A_0}\cdot M/\overline{IO}\cdot A_{19}\cdot A_{18}\cdot A_{17}}=\overline{A_0}\cdot M/\overline{IO}\cdot A_{19}\cdot A_{18}\cdot A_{17}$$

$$\overline{CS_{M1}}=\overline{\overline{BHE}\cdot M/\overline{IO}\cdot A_{19}\cdot A_{18}\cdot A_{17}}=\overline{BHE}\cdot M/\overline{IO}\cdot A_{19}\cdot A_{18}\cdot A_{17}$$

即当： $A_{19}A_{18}A_{17}A_0\overline{BHE}M/\overline{IO}$=110011 时，$\overline{CS_{M0}}$=0，选中 M_0

$A_{19}A_{18}A_{17}A_0\overline{BHE}M/\overline{IO}$=110101 时，$\overline{CS_{M1}}$=0，选中 M_1

$A_{19}A_{18}A_{17}A_0\overline{BHE}M/\overline{IO}$=110001 时，$\overline{CS_{M2}}$=0，选中 M_0、 M_1

由图中地址总线 A_{16}~A_1用于片内地址选择可知，存储芯片 M_0和 M_1的容量均为 2^{16}B。由此可得:

(1)存储总容量=M_0和 M_1的容量之和=2×2^{16}B =128KB

(2) M_0的地址范围为:C0000H~DFFFFH 的偶地址。

M_1的地址范围为:C0000H~UFFFFH 的奇地址。

4.11 已知某微机系统的存储器如图 4.10 所示.试问:

(1)图中所用存储器芯片是哪一类存储器?其存储容量为多少?

(2)该存储器的总容量为多少?

(3)整个存储器的地址范围为多少?其中。0#~3#芯片的地址范围分别为多少?

解:(1]图中所用存储器芯片是既有读允许线，又有写允许线的 SRAM 芯片或 F lash 存储器芯片。该芯片有 14 根片内地址线，8 根数据线，存储容量为:16K× 8 位。

(2) 此题虽然各存储器芯片的容量为 16 K × 8 位，但地址线 A 多用于选择片内存储单元.又参与高段地址译码，译码电路产生的片选地址范围为 0000H~1FFFH.即寻址空间为 8KB 所以，尽管 0#和 3#芯片的容盘为 16KB,但他们的寻址空间都只在 8KB。由此可计算出该存储器的总容最为:

存储器的总容量=0#~3#芯片的寻址空间之和= 8 KB + 16 KB十16 KB十8KB= 48KB

(3)根据上述存储器连接图可列出各存储芯片的存储空间分配位表，如表4.5所示。由此可得:整个存储器的地址范围为 B2000H~B3FFFH; 0#芯片的地址范围为 B2000H ~ B3FFFH ;1#芯片的地址范围为 B4000H~BBFFFH ； 2#芯片的地址范围为 B8000H~ BBFFFH; 3#芯片的地址范围为 BC000H ~BDFFFH。

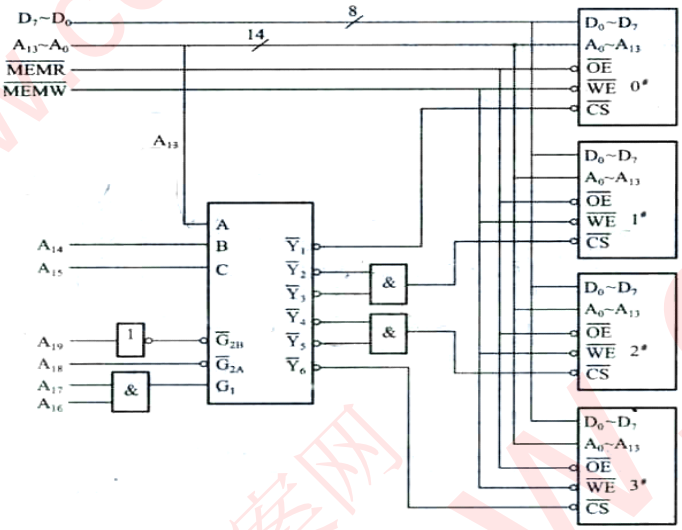


图 4.10 某微机存储器电路

表 4.5 习题 4.11 存储空间分配位表

芯 片	$A_{19}A_{18}A_{17}A_{16}A_{15}A_{14}$	A_{13}	$A_{12}A_{11}A_{10}A_9A_8A_7A_6A_5A_4A_3A_2A_1A_0$
0#	1 0 1 1 0 0	1	0000H~1FFFH
1#	1 0 1 1 0 1	0	0000H~1FFFH
		1	0000H~1FFFH
2#	1 0 1 1 1 0	0	0000H~1FFFH
		1	0000H~1FFFH
0#	1 0 1 1 0 0	1	0000H~1FFFH

4.12 已知某 8086 单片机系统的 SRAM 如图 4.ll 所示。

(1)试分析芯片(1)、芯片(2)和整个存储器的地址范围分别为多少?

(2)若要将芯片(l)、芯片(2)分别作为偶数存储体和奇数存储体。联合构成一个 64Kx16 位的存储器，要求总的地址范围不变.应如何改变图中的接法?请画出新的连线图。这时芯片(1)、芯片(2)的地址范围分别为多少?

解:(1)根据图 4.11 列出存储芯片((l)和芯片(2)的存储空间分配位表，如表 4.6 所示。据此可得:芯片(1)的地址范围为 2000OH--2FFFFH;芯片(z)的地址范围为 30000H - 3FFFFH;整个存储器的地址范围为 20000H -- 3FFFFH .

(2）这时要用 A_0 和 $\overline{BHP}$ 分别做偶存储体(芯片(1))和奇存储体(芯片(2))的体选控制信号，参与高端译码。 $A_1 \sim A_{16}$ 用做体内存储单元选择信号，芯片(1)和芯片(2)的数据线分别与数据总线的 $D_0 \sim D_7$、$D_7 \sim D_{15}$ 相连。修改后的电路如图 4.12 所示。

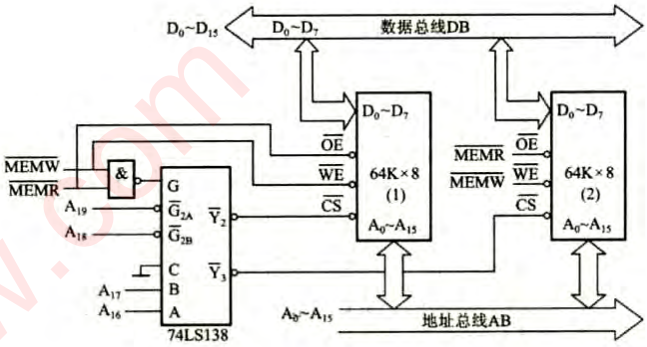


图 4.11　某 8086 单片机系统的 SRAM

表 4.6　习题 4.12 存储空间分配位表

芯　片	$A_{19}A_{18}A_{17}A_{16}$	$A_{15}A_{14}A_{13}A_{12}A_{11}A_{10}A_9A_8A_7A_6A_5A_4A_3A_2A_1A_0$
(1)	0 0 1 0	0000H～FFFFH
(2)	0 0 1 1	0000H～FFFFH

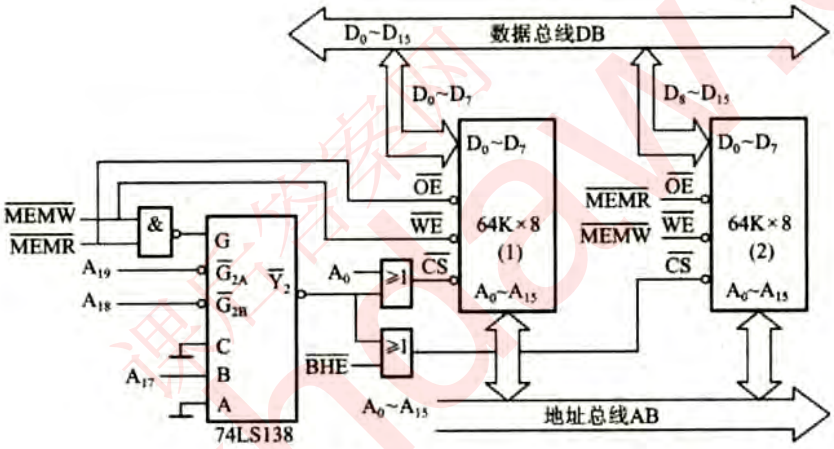


图 4.12　修改后的 SRAM 电路图

4.13 已知某 8088 系统的 EPROM 用 2754 构成，如图 4.13 所示。试问其存储地址范围为多少?若要将地址范围变为 C4000H- C5FFFH，请修改其连接。

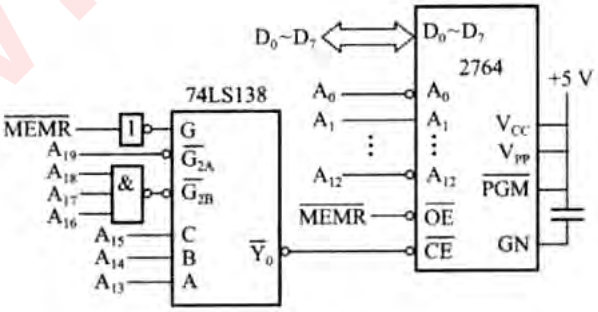


图 4.13　某 8088 系统的 EPROM

解:当　$A_{19}A_{18}A_{17}A_{16}A_{15}A_{14}A_{13}=0111000$ 时.译码器输出 $\overline{Y_0}$ 有效.由此可得，存储地址范围为 70000 H 一 71FFFH。

要将地址范围变为 C 4000H~5FFFH。必须使 $A_{19}A_{18}A_{17}A_{16}A_{15}A_{14}A_{13}=1100010$ 时，译

码器输出的片选信号选中 2764 芯片。此时.可按图 4.14 修改存储器连接。

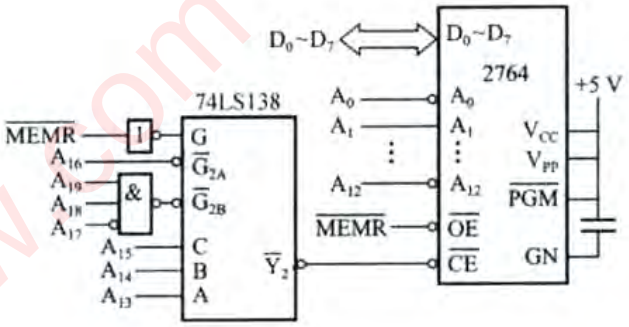


图 4.14　修改后的 EPROM 电路图

4.20 虚拟存储器和高速缓冲存储器在原理上有没有相似的地方?如有请说明其相似之处和主要区别。

解:高速缓存与内存的映射关系和虚拟存储器中内存与外存的映射关系在原理上有很大的相似性，它们都是基于程序局部性原理，其共同之处是;两者都把程序中最近最常使用的部分驻留在高速存储器中，而把高速存储器中不常用的部分送回到低速存储器中;这种调人/调出是由硬件或操作系统自动完成的，对用户是透明的;两者都力图使存储器系统的速度接近高速存储器，而价格却接近低速存储器。不同之处在于高速缓存技术用于解决 CPU 与常规主存储器在处理速度上的不匹配问题，而虚拟存储器技术则用于增大用户可利用空间。前者在 CPU 与主存间使用，后者用于内存与外存之间。

5.10 为什么输人接口的数据缓冲寄存器必须有三态输出功能，而输出接口却不需要?

解:输人接口的数据缓冲寄存器的谕出端是直接接在数据总线上的，如果数据寄存器没有三态输出功能，则无论数据寄存器被寻中还是未被寻中，其数据都会被送上数据总线，若此时总线上真正要传送的数据与该输人缓存器的内容不一致时，就会发生总线冲突。所以输人接口的数据缓冲寄存器必须有三态输出功能，以便接口未被寻中时，其输出端处于高阻态而与总线隔离。

对于输出接口来说，其输人端与数据总线相连，而输出端与外设相连，因此其输出不影响总线状态;并且外设一般只与一个输出数据缓存器相连，所以输出接口的数据缓存器无需三态输出功能。

5.11 已知 PC 机系统中某接口板的 I/O 端口译码电路如图 5.4 所示，试分析出各 I/O 端口和 I/O 芯片的端口地址或地址范围。

解:74LS138 使能信号 G_1、$\overline{G_{2B}}$ 的逻辑表达式为:

$$G_1 = A_9$$

$$\overline{G_{2B}} = \overline{A_7 A_6 A_5 A_4}$$

即仅当 $A_9A_7A_5A_4$ = 11111 时，使能信号 G_1= 1，$\overline{G_{2B}}$ = 0 有效。此时，若 $A_3A_2A_8$ = 000，$\overline{Y_0}$ =0；$A_3A_2A_8$ =001，$\overline{Y_1}$ =0；$A_3A_2A_8$ =111，$\overline{Y_7}$ = 0;所以，各 I/O 芯片的地址范围为:

I/O 芯片 1:1111110000B—11111100113B 即 3F0H 一 3F3H;
I/O 芯片 2: l011110100B--1011110111.即 2F4H 一 2F7H;
I/O 芯片 3:1111111100B 一 11111111113,即 3FCH 一 3FFH

$\overline{Y_0}$ 译出的地址范围 2F0H~2F3H 再经一级 74LS139 译码器对 A_1 和 A_0 进行二次译码。分别得到 4 个读端口和 4 个写端口。各 I/O 端口的地址为:

输出口 1 和输人口 1 为 2F0H ;
输出口 2 和输入口 2 为 2F1H;
输出门 3 和输入口 3 为 2F2H;
输出口 4 和输入口 4 为 2F3H

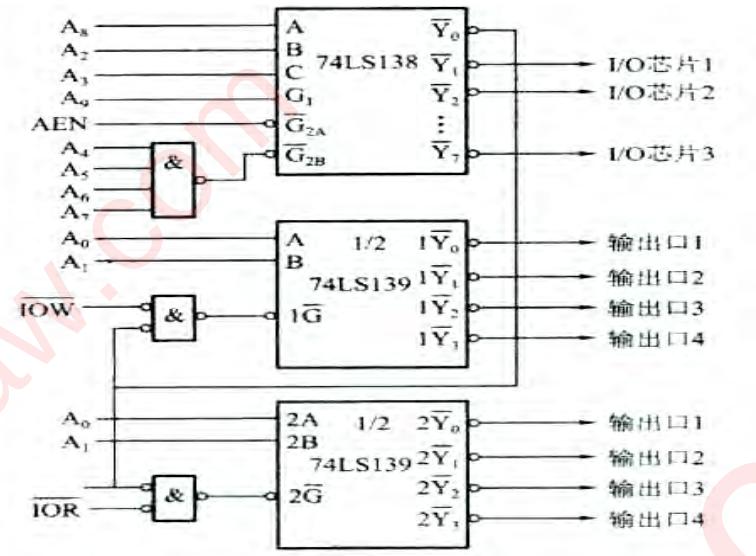


图 5.4　某接口板的 I/O 端口译码电路

5.12 已知 PC.机系统中某接口板的板地址译码电路如图 5.5 所示。现希望该板的地址范围为 028H 一 0287H.请确定 DIP 开关各位的状态(打开或闭合)。•

解:用二进制写出地址范围为 1010000000B~1010000111B。即要使该板的地址范围为 0280H~0287H，必须使: $A_9A_8A_7A_6A_5A_4A_3$=1010000B.

此题采用比较器译码，仅当比较器的对应输人一致时，比较器的输出为 0,即板选信号有效。所以，DIP 开关各位的状态应为:

DIP_0	DIP_1	DIP_2	DIP_3	DIP_4	DIP_5
闭合	闭合	闭合	闭合	断开	闭合

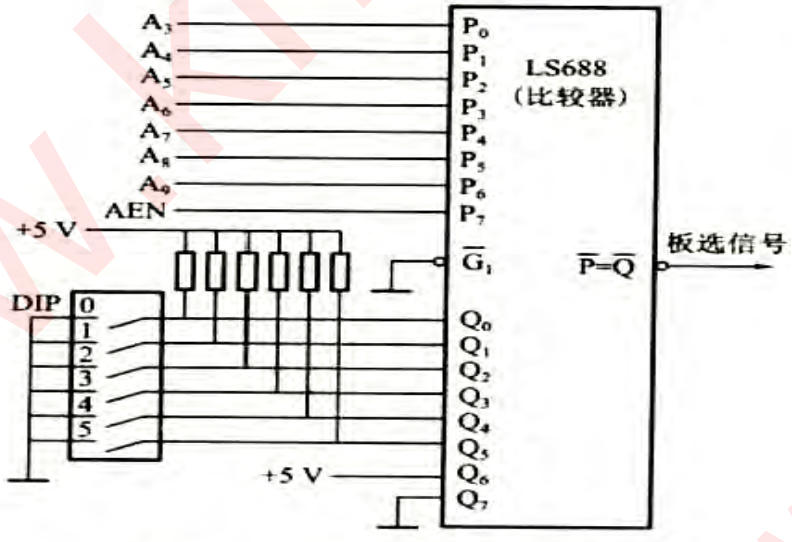


图 5.5　某接口板的板地址译码电路

5.13 试用 80486 汇编语言以查询输人方式编写一个子程序，从 CRT 终端愉人 128 个字节并存人首地址为 BUFFER1 的内存缓冲区。CRT 终端的数据口地址为 0008H，状态口地址为 0020H。状态口的 D_0 位为读状态位， D_0=1 表示输入数据有效。

解:程序如下:

```
        MOV     DI,0
        MOV     CX，128
AGAIN:  MOV     DX,  0020H
WAIT$:  IN      AL.DX                   ;读接口状态
        TEST    AL,00000001 B           ;准备就绪?
        JZ      WAIT $                  ;继续读状态
        MOV     DX, 0008H
        IN      AL, DX                  ;输人数据
        MOV     BUFFER1[DI].AL          ;保存输人数据
        INC     DI
        LOOP    AGAIN                   ;未完继续输人
        HLT
```

5.14 在上题基础上，假设状态口的 D_7 位为写状态位，D_7=0 表示缓冲存储器空闲，试用查询法编写一个子程序。把内存中的 BUFFED 开始的 64 个字节的字符串送至 CRT 终端。

解:程序如下;

```
        MOV     DI,  0                  ;指向 BUFFER2 第一字节
        MOV     CX,64
AGAIN:MOV   DX,  0020H                  ;取状态端口地址
WAIT$:IN    AL,DX                       ;读接口状态
        TEST    AL,10000000B            ;输出缓存器空?
        JNZ     WAIT$                   ;非空，继续读状态
        MOV     DX,  0008H              ;取数据端口地址
        MOV     AL, BUFFER2[ DI ]       ;取输出数据
        INC     DI
        OUT DX,AL                       ;输出数据
        LOOP  AGA1N                     ;未完继续输出
        HLT
```

6.6 什么叫“总线冲突”?总线冲突的后果因驱动器是 OC 门和三态门有什么不同?为了避免总线冲突，必须对驱动器使能信号进行控制，图 6.1 是对三态双向驱动器加了控制逻辑的设计实例。试写出图中两个三态使能端信号 G_1、G_2 的逻辑表达式。并据此分析出存储器 M 和 I/O 端口在驱动器前后的地址空间范围。(提示:对驱动器后的 M 和 I/O 的访问必须以 G_1 (对读操作)或 G_2 (对写操作)有效为前提;驱动器前、后的 M 与 I/O 空间之和分别等于 16 位与 8 位地址线的总寻址空间。)

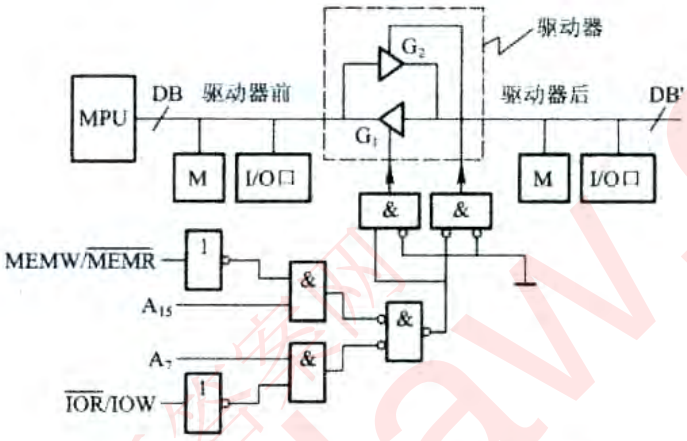


图 6.1 增加了控制逻辑的三态双向驱动器

解:总线冲突是指在总线上同时有两个或两个以上的模块要传送相互矛盾的信息时引起的冲突。冲突的表现形式后果因驱动总线的逻辑器件类型而异。若是 OC 门则不能传送高电平“1”;若是三态门，则高电平“1”与低电平“0”接到一起相当于短路，可能烧坏器件。图 6.1 中、使能端信号 G_1、G_2 的逻辑表达式为:

$$G_1 = \overline{MEMW / \overline{MEMR}} * A_{15} + \overline{\overline{IOR} / IOW} * A_{17}$$

$$G_2 = (MEMW / \overline{MEMR} + \overline{A_{15}}) * (\overline{IOR} / IOW + \overline{A_7})$$

由图可知，系统使用不同的读/写控制信号访问存储器 M 和 I/O 口，即存储器和 I/O 端口是分开编址的，假定存储器和 I/O 端口的地址线分别为 $A_{15} \sim A_0$、$A_7 \sim A_0$.若要读驱动器后的 M 和 I/O 必须使 G_1 有效，由 G_1 表达式可知:

当 $MEMW / \overline{MEMR} = 0, A_{15} = 1$ 时.允许读驱动器后的 M

当时，$\overline{IOR} / IOW = 0, A_7 = 1$ 事，允许读驱动器后的 I/O:,

据此可知，驱动器后的 M 空间为 8000H~0FFFFH; I/O 空间为 80H~0FFH。
驱动器前、后的 M 和 I./O 空间之和分别等于 16 位与 8 位地址线的总寻址空间。所以，驱动器前的 M 和 I./O 空间分别为 0000H~7FFFH 和 00H~7FH。

6.7 在三线菊花链仲裁中，主控器 C_i 获得总线占用权的必要条件之一是检测到 $BGIN_i$ 由无效变有效的边沿，为什么要这样规定?如果把该必要条件变成只需检测到 $BGIN_i$ 有效即可，行不行?为什么?试结合波形图予以说明.

解:规定该必要条件的目的是避免总线冲突。如果把该必要条件变成只需检测到 $BGIN_i$ 有效即可是不行的。下面结合图 6.2 予以说明。

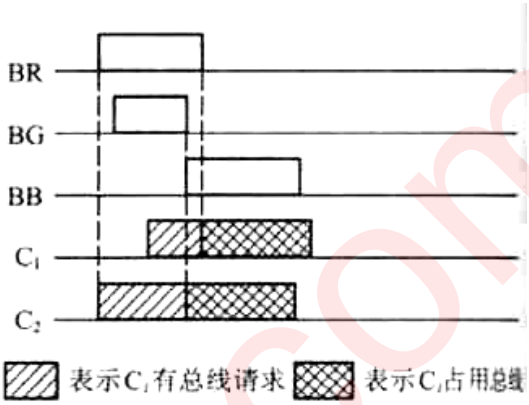


图 6.2 总线仲裁时序图

假定当前没有主控器占用总线.而此时 C_2 提出总线请求. 但 C_1 没有请求，仲裁器将发出总线允许信号（BG 有效)，因 C_1 没有提出总线请求.在收到 BG 时不接管信号总线，而是将 BG 信号向后传递，当 C_2 接收到 BG 信号时，开始接管总线;若在 BG 信号已过 C_1，但还未撤销时，C_1 也提出了总线请求，而这时 $BR_1=0$、$BGIN_1=0$，表明 C_1 也接管总线的条件，C_1 也接管总线，从而出现 C_1、C_2 同时占用总线的情况，产生总线冲突。.

7.7　8259A 只有两个端口地址，但可读/写寄存器数远远多于两个，如何保证正确读/写?

解:8259A 中使用了如下几种方法来实现同一地址寻址多个内部寄存器:

(1)利用命令字 OCW:事先指定读 IRR 或 ISR;

(2)利用命令字中位 4 和位 3 的状态来决定写 ICW1,OCW:还是写 OCW3 ;

(3)根据顺序来决定同一接口地址下的命令字(ICWZ,ICW3 ,ICW4,OCW1) 。

7.11 某 80X86 微机的中断系统有 5 个外部中断源，接在 8259A 的 $IR_0 \sim IR_7$ 端，中断类型码为 5BH,5CH,5DH,5EH 和 5FH , 8259A 的端口地址为 B0H,B1H。允许它们以全嵌套工作方式工作，中断请求采用电平触发方式。试编写 8259A 的初始化程序。

解:先确定要写哪些控制字，以及每个控制字对应位的取值，再编程。此题，要写的控制字及位取值如下:

(1) ICW_1= x x xllxll，即:中断请求为电平触发、单片 8259A、写 ICW4 。

(2) ICW_2=01011x x x，即:此片 8259A 中断类型码高 5 位为:01011

(3) ICW3 无需写。

(4) ICW4 = 00000 x 01，即:一般嵌套、非缓冲、正常 EOI ,8086 /8088 模式。

其中 x 表示取值为 0 或 1 均可。假定控制字中的 x 固定为 0，初始化程序如下:

```
MOV     DX,0B0H          ;指向端口 0
MOV     AL,1BH           ;中断请求为边沿触发、单片 8259A、写 ICW4
OUT     DX , AL          ;写 ICWI
INC     DX               ;指向端口 1
MOV     AL, 58H          ;中断类型码高 5 位为 01011
OUT     DX , AL          ;写 ICW2
MOV     AL,01H           ;一般嵌套、非缓冲、正常 EOI , 8086 /8088 模式
OUT     DX , AL          ;写 ICW4
```

7.12 某 80X86 系统中设置三片 8259A 级联使用，一片为主，两片为从，从片分别接人主片的 IR_2 和 IR_4 。若已知当前主 8259A 和从 8259A 的 IR_5 上各接有一个外部中断源，其中断向量号分别为 75H,85H,95H。假设它们的中断入口地址均在同一段中，段基址为 4310H,偏移地址分别为 1230H,2340H,3450H;所有中断都采用边沿触发方式、全嵌套方式、正常 EOI 结束方式。

(1)试画出该系统的硬件连线图;

(2)试编写全部初始化程序。

解:(1)此题硬件连线要考虑各 8259A 与 CPU 的连接，还要考虑三片 8259A 间的级联连接。8259A 与 CPU 连接的方法是:

①端口选择线与一般与 MPU 低位地址线 A_0 直接相连，也可与其他 A_i 相连。

②数据线 $D_0 \sim D_7$ 与 MPU 数据线 $D_0 \sim D_7$ 直接相连。

③片选线 $\overline{CS}$ 与余下 MPU 高位地址线经译码后产生的片选信号相连。

④ $\overline{RD}$、$\overline{WR}$、$\overline{INTR}$ 分别与控制总线组合形成的 $\overline{IOR}$、$\overline{IOW}$ 和中断响应信号 $\overline{INTR}$ 相

连。

⑤主片的 INT 与 MPU 的 INTR 直接相连。

8259A 级联连接方法是:

①从片的 INT 与主片的某个 IR_i 直接相连。

②主片的 $CAS_2 \sim CAS_0$ 与各从片的 $CAS_2 \sim CAS_0$ 直接相连。

③ $\overline{SP}/\overline{EN}$ 则取决于是采用缓冲还是非缓冲方式。

本题采用非缓冲方式，主片 $\overline{SP}/\overline{EN}$ 接+5V，从片 $\overline{SP}/\overline{EN}$ 接地。由此画出硬件连线图，如图 7.3 所示。

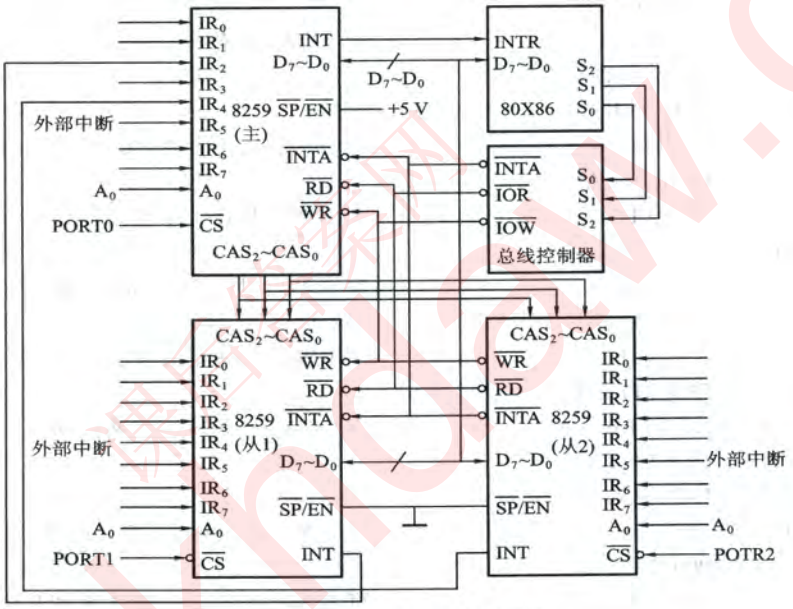


图 7.3　8259A 构成的中断逻辑

(2)初始化程序应包括两部分内容，即主、从片的初始化和填写相关中断向量表。对主片编程时要注意。在多片级联方式下，系统要工作在完全嵌套方式，主片必须设置为特殊全嵌套方式，以便同级中断能打断同级中断。因为在多片级联方式下，从片的 8 个中断源是共用主片的一个中断级，而实际上从片的 8 个中断源也有优先级区别。假定当前正在服务的中断级是从片的某个中断级，若当 CPU 正在为该中断源服务时，该从片又向主片提出中断请求，此时对主片而言是同一级中断请求，但对从片来说却是出现了一个高优先级中断请求，应该优先处理，

假定主片、从片 1、从片 2 地址分别为:PORT00. PORT01 , PORT10, PORT11 , PORT20, PORT21 ,程序段如下:

```
:初始化 8259A 主片
MOV     AL,11H              ;写 ICW1 边沿触发、多片级联、写 ICW4
OUT     PORT00, AL
MOV     AL,70H              ;写 ICW2，主片中断向断号为 70H---78H
OUT     PORT01 . AL
MOV     AL.,000l0l00B       ;写 ICW3, IR2,IR4,接有从片
OUT     PORT01 . AL
```

```
MOV     AL,00010001B        ;写 ICW4 特殊全嵌套、非缓冲、正常 EOI
OUT     PORT01 , AL
MOV     AL,OFFH             ;用 OCW1 屏蔽所有中断请求
OUT     PORT01 , AL
;初始化 8259A 从片 1
MOV     AL,11 H             ;写 ICW1,边沿触发、多片级联、写 ICW4
OUT     PORT11,AL
MOV     AL.,01.H            ;写 ICW2，从片 1 中断向量号为 80H-87H
OUT     PORT11,AL
MOV     AL,02H              ;写 ICW3.对应主片 IRi 的编码为 010
OUT     PORT11. ,AL
MOV     AL,01H              ;写 ICW4，一般全嵌套、非缓冲、正常 EOI
OUT     PORT11 ,AL
MOV     AL,0FFH             ;用 OCW1 屏蔽所有中断请求
OUT     PORT11 . AL
;初始化 8259A 从片 2
MUV     AL,11H              ;写 ICW 1。边沿触发、多片级联、写 ICW4
OUT     PORT20,AL
MOV     AL.90H              ;写 ICW2。从片 2 中断向量号为 90H～97H
OUT     PORT21.AL
MOV     AL,04H
DUT     PORT21, AL
MOV     A.L,01 H
OUT     PORT21, AL
MOV     AL.0FFH
OUT     PORT21, AL
;填写 T5H,85H 和 95H 号中断向址
SUB     DI、DI              ；ES 指向中断向量表段基址
MOV     ES, DI
CLD
MOV     DI，75H*4           ;75H 号中断向量地址
MOV     AX，1230H           ;中断处理程序偏移地址
STOSW
MOV     Ax,4310H            ;中断处理程序段基址
STOSW
MOV     DI.,85H*4           ；初始化 85H 号中断向量
MOV     AX.2340H
STOWS
MOV     Ax .4310H
STOSW
MOV     DI,95H *4           ；初始化 95H 号中断向量

MOV     AX,3450H
STOW S
```

```
MOV       Ax.4310H
STOSW
MOV       AL.00H                ;用 OCW1 开放主片所有中断请求
OUT       PORT01 H . AL
OUT       PORT 11 H , AL        ;用 OCW1 开放从片所有中断请求
OUT   PORT21H,AL
```

7.13 某 8085/8088 系统中，若 8259A 处于单片、全嵌套工作方式.且采用非特殊屏蔽和非特殊结束方式，中断请求信号是边沿触发。IR_0 的中断类型码为 58H。试对 8259A 初始化编程。

解:此题无需写 ICW3。其他控制字及位取值如下:

（1）ICW_1=xxx10x11，即:中断请求为边沿触发、单片 8259A ,写 ICW_4

(2) ICW_2=01011 x x x ,即:此片 8259A 中断类型码高 5 位为 01011

(3) ICW4 = 00000 x O1，即:一般嵌套、非缓冲、正常 EOI,8086/8088 模式。

其中 x 表示取值为 0 或 1 均可。假定控制字中的 x 固定为 0,初始化程序如下:

```
MOV       DX, x x x x x x x 0B    ;指向端口 0
MOV       AL,13H                  ;中断请求为边沿触发、单片 8259A、写 ICW4
OUT       DX,AL                   ;写 ICW 1
INC       DX                      ;指向端口 1
MOV       AL,58H                  ;中断类型码高 5 位为 01011
OUT       DX , AL                 ;写 ICW2
MOV       AL,O1H                  ;一般嵌套、非缓冲、正常 EOI,8086 模式
OUT       DX , AL                 ;写 ICW4
```

7.14 分别列出下列情况下应向 8259A 提供的操作命令字:

(1)读中断请求寄存器 IRR;

(2)读中断屏蔽寄存器 IMR;

(3)读中断服务寄存器 ISR;

(4)向 8259A 发中断结束命令 EOIo。

解:(1)应先对 0 端口写 OCW3(读 IRR 命令字)，再通过读 0 端口获得 IRR 值，读 IRR 命令字为 00001010B，指令序列为:

```
MOV       AL,   00001010B       ;读 IRR 命令字
OUT       x x x x x x x 0B,AL   ;写 OCW3(读 IRR 命令字)
IN        AL, x x x x x x x 0B  ;读 IRR
```

(2)无需发操作命令字，直接对 1 端口读操作:

```
IN        AL, x x x x x x x 1B  ;读 IMR
```

(3)应先对 0 端口写 OCW3(读 ISR 命令字)，再通过读。端口获得 ISR 值，读 ISR 命令字为 00001011B，指令序列为:

```
MOV       AL, 00001011B         ;读 ISR 命令字
OUT       x x x x x x x x0B,AL  ;写 OCW3(读 ISR 命令字)
IN        AL, x x x x x x x 0B  ;读 ISR
```

(4)有四种中断结束命令 EOI，由写 OCW:设置。

非特殊 EOI 命令:00100000B;

特殊 EOI 命令:$01100L_2L_1L_0B$;

自动循环 EOI 命令:10100000B;

特殊循环 EOI 命令: $11100L_2L_1L_0B$

7.16 若一 PC/AT 应用系统中有 5 个故障源 A,B,C,D,E，当任一个源变为低电平时，则有故障要处理。它们的优先权顺序是从 A 到 E 依次降低，处理程序的人口地址分别为 8000H, S100H,8200H,8300H 和 8400H。现只有 IRQ_2 可用，试按查询式中断识别与判优方案设计该中断系统，包括硬件设计和软件编程。

解:按查询式中断识别与判优方案设计的中断系统如图 7.4 所示。

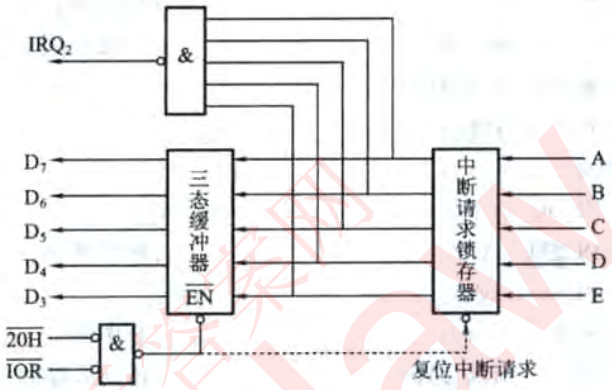


图 7.4 查询式中断识别与判优电路

5 个故障源 A, B,C,D,E 通过与非门与 IRQ_2 相连。即任一个源变为低电平时通过 IRQ_2 向 CPU 发中断请求，在中断服务程序中通过读取三态缓冲器来识别故障源，并通过读端口控制信号复位中断谓求.程序如下:

```
DATA    SEGMENT
BASE    DW    8000H ，0000H          ;故障源 A 处理程序的人口地址 CS: IP
DW    8100H,0000H                    ;故障源 B 处理程序的入口地址 CS:IP
DW     8200H,0000H                   ;故障源 C 处理程序的人口地址 CS:I P
DW     8300H,0000H:                   ;故障源 D 处理程序的人 A 地址 CS:IP
DW     8400H . 0000H                 ; 故障源 E 处理程序的人口地址 CS:I P
DATA      ENDS
CODE     SEHMENT
ASSUME    CS:CODE，DS: DATA
START :       MOV    Ax , DATA
              MOV    DS. Ax
              CLI                    ;关中断
              MOV    DI,0AH *.4      ;取 IRQ2 的中断向量地址
              CLD
             XOR    Ax,Ax
             MOV ES, Ax              ;ES 指向中断向量表段基址
             MOV Ax ,OFFSET INT_ROUT ；填写中断向量的偏移地址
                STOSW
```

```
        MOV    Ax,SEG INT_ROUT          ;填写中断向量的段基址
          STOSW
          STI                           ;开中断
                                        ；其他处理
           MOV    AH,4CH                ;中断服务程序
           INT    21 H
           INT_ROUT PROC:
           PUSH    Ax                   ;保护现场
           PUSH    Bx
           STI                          ;开中断
           LEA     Bx.BASE              ;指向故障源 A 中断处理程序地址
           IN      AL,20H               ;读状态寄存器
           NOT     AL                   ;取反.将低电平变为高电平
           AND     AL,0F8H
WAITl:     TEST    AL,OFFH              ;有末处理中断?
           JZ     RETURN                ;无中断返回
           SHL     AL. , 1              ;有故障
           JNC     NEXT                 ;无，跳过
           CALL.    DWORD   PRT[BX]     ;转相应故障源中断处理
NEXT:     ADD      Bx,4                ;指向下一故降源中断处理程序地址
           JMP     WAIT1
RETURN: CLI                             ;关中断
           POP      BX                  :恢复现场
           POP      Ax
           STI                          ;开中断
           I RET
INT_ROUT ENDP
 CODE     ENDS
          END     START
```

7.17 试编写一个基于查询式中断识别与判优方案的中断程序，该程序控制 8 台设备，假定状态寄存器地址为 044DH，其最低位优先级最高，最高位优先级最低:某位置"1"表示相应的设备有服务请求。同时假定已定义一个地址数组 ADDR _ TAB，数组中的第 i 个元素提供第 i 台设备的中断处理程序人口地址的偏移量

解:基于查询式中断识别与判优的中断程序如下:

```
 INT_ PROC     PRDC
               PUSH     AX              ;保护现场
               PUSH     Bx
               STI                      ;开中断
               MOV      BX,0            ;指向第 0 台设备中断处理程序地址
               IN      AL,40H           ;状态寄存器
  WAIT $ :     TEST     AL.0FFH         ;有未处理中断?
               JZ       RETURN          ;无中断返回
              SHL       AL ,1           ;第 i 台设备有中断?
               JNC      NEXT            ;无，跳过
```

```
                CALL      ADDR_TAB[.BX]   ;转第 i 台设备中断处理
NEXT:           ADD     BX,2              ;指向下一台设备中断处理程序地址
                JMP       WAIT$
RETURN :        CLI                       ;关中断
                POP       BX              ;恢复现场
                POP       AX
                STI                       ;开中断
                IRET
INT_PROC        ENDP
```

7. 18 假设起始地址的标号为 I NT_ RDUT 的 9 型中断例程与主程序处于相同的源模块中.试为主程序编写装填中断向量表的程序段。

解:装填中断向量表可用 DOS 的 25H 号功能调用，亦可用 MOV 指令或串操作指令直接对中断向量表进行写操实现。用 DOS 的 25H 号功能调用。程序段如下:

```
PUSH        DS
MOV         DX . OFFSET INT_ROUT        ;取中断向量的偏移地址送 DX
MOV         AX . SEG INT_ROUT           ;取中断向盘的段基址送 DS
MOV         DS , AX
MOV         AH,25H                      ;取 DOS 的功能调用号送 AH
MOV         AL, 09H                     ;取中断类型码送 AL
INT         2IH                         ;调 25H 号功能调用填写中断向量
POP         DS
```

用串操作指令填写.程序段如下:

```
PUSH        ES
CLI                                     ;关中断
MOV         DI.09H * 4                  :取 9 型中断的中断向量地址
CLD
XOR     Ax,Ax
MOV     ES,Ax                           ;ES 指向中断向量表段基址 0000H
MOV     AX,     OFFSET INT_ROUT         ;填写中断向里的偏移地址
    STOSW
MTV     Ax , SEG INT_ROUT               ;填写中断向量的段基址
    STOSW
ST I                                    ;开中断
    POP     ES
```

7.19 假设某 80486 系统工作在实地址方式。已知(SP)=0100H,(5) = 0300H, (FLAGS) = 0240H,00020DH 至 00023H 单元的内容分别是 40H .00H .00H .01 H。同时还已知 INT_8 的偏移量 00A0H 在段基址为 0900 H 的 CS 段内。试指出在执行 IN 几指令并进人该指令相应的中断服务程序时.SP,SS,IP,CS,FLAGS 和堆栈最顶端三个字的内容。

解:由题意可知:INT8 指令的地址为 0900H:00A0H; IN 几中断服务程序的人口地址为 0100H:0040 H 。

CPU 执行 INT 8 指令时。先保存 FLAGS , CS:IP 值至堆栈，并清除 FLAGS 中的 1F 和 TF 位，进而改变 CS: IP 值转入中断服务程序。由此可得 CPU 在执行 INT_8 指令并进人该指

令相应的中断服务程序时各寄存器和存储器单元值如下:

(SP)=(SP)-6=0100H-6=00FAH, (SS)=0300H;

(CS:IP)为中断服务程序人口地址:(IP)=0040H. (CS) = 0100H;

FLAGS 中 IF=TF=0,即 FLAGS9 = FLAGS8 = 0。所以:

(FLAGS)=0040H:

堆栈最顶端三个字的内容为:(00FEH)=0240H ; FLAGS 值

(00FCH) = 0900H ;断点 CS 值

(00FA00H)=00A0H :断点 IP 值

7.20 讨论在 80X86 系统的调试应用中.INT3 指令和单步自陷所产生的中断差别，指出其应用场合。

解:INT3 指令产生的是断点中断，即程序运行遇到断点(INT3 指令)时才暂停.转去执行断点处理程序，主要用于在程序连续运行方式下设置断点调试程序;而单步自陷则使程序进人单步运行方式。即每执行一条指令，程序运行都会暂停，转去执行单步处理程序，用于单步运行调试程序。

7.21 若 80X86 系统正以单步方式运行某用户程序(该程序已开放外部中断，即 IF=1)的过程中执行一条除法指令时，1NTR 线上出现了可屏蔽中断请求，与此同时.这条除法指令也产生了除法出错中断。试以流程图表示 CPU 处理这三种同时出现的中断的过程。

解:中断过程如图 7.5 所示。CPU 先响应除法出错中断.再响应 INTR 中断，处理完毕返回单步运行的用户程序。

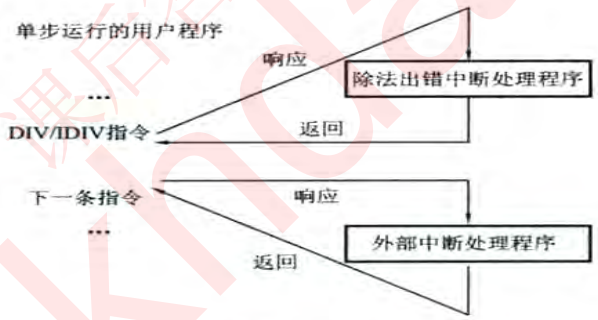


图 7.5 中断过程

7.22 试编写出只有一片 8259A 的 8088 系统中 8259A 的初始化程序. 8259A 的地址为 02C0H 和 02C1 H，要求:

(1)中断请求物人采用电平触发;

(2) IR_0 请求的中断类型码为 16

(3)采用缓冲方式;

(4)采用普通的 EOI 命令。

解:初始化程序如下:

```
MOV    DX,02C0H            ;端口 0
MOV    AL,00011011B        ;中断请求为电平触发、单片 8259A、写 ICW4
OUT     DX,AL              ;写 ICW1
INC     DX                 ;指向端口 1
MOV     AL,0001000B        ;中断类型码高 5 位为:00010
OUT     DX,AL              ;写 ICW2
MOV     AL.,00001101B      ;一般嵌套、级冲、正常 EOI ,8086 模式
```

OUT DX,AL ;写 ICW4

7.23 试编写一段使 8259A 的优先级顺序从高到低依次为 $IR_6 \rightarrow IR_7 \rightarrow IR_0 \rightarrow IR_1 \rightarrow IR_2 \rightarrow IR_3 \rightarrow IR_4 \rightarrow IR_5$ 的程序。假定 CPU 为 8288,8259A 的偶地址为 20 H.奇地址为 21H，且当前优先级为;(1) IR_0 ;(2) IR_3 。

解:用特殊循环优先级设置命令指定最低优先级，设置方法与当前优先级无关。所以(1)和（2)均用下述指令完成:

```
MOV   AL,11000101B        : IR5为最低优先级
OUT   20H,AL              ;写特殊循环优先级设置命令
```

7.24 试编写一段将 8259A 中的 IRR, ISR 和 IMR 的内容传送至存储器中从 REG_ ARR 开始的数组中去的程序，假定 CPU 为 80X86, 8259A 的偶地址为 50 H，奇地址为 51H。

解:程序如下:

```
MOV     Dx,50H              ;指向端口 0
MOV     AL.AL,00001010B     :读 IRR 命令字
OUT     Dx.AL               ;写 OCW3(读 IRR 命令字)
IN      AL.Dx               ;读 IRR
MOV     REG_ARR,AL
MOV     AL , 00001011 B     ;读 ISR 命令字
OUT     Dx . AL             ;写 OCW3(读 ISR 命令字)
IN      AL,DX               :读 ISR
MOV     REG_ARR[1],AL.
INC:    DX                  ;指向端口 1
IN      AL.Dx               ;读 IMR
MOV     REG_ARR[2],AL
```

8.5 表 8.2 给出了对 8254 依次发出的 8 条读回命令。试将各条命令的作用及其执行后结果填入表中

解：如表 8.2 所示。

表 8.2

读回命令									命令作用	执行结果
次序	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0		
1	1	1	0	1	0	1	0	0	读回通道 1 的计数值	锁存通道 1 的计数值
2	1	1	0	0	1	0	0	0	读回通道 2 的计数值和状态	锁存通道 2 的计数值和状态
3	1	1	1	0	1	0	1	0	读回通道 0、2 的状态	锁存通道 0 的状态
4	1	1	1	0	1	1	0	0	读回通道 1、2 的状态	锁存通道 1 的状态，对通道 2 无效
5	1	1	0	0	0	1	1	0	读回通道 0、1 的计数值和状态	锁存通道 0 的计数值，但对通道 0 状态和通道 1 无效
6	1	1	0	1	0	0	1	0	读回通道 0 的计数值	命令无效
7	1	1	0	0	1	1	1	0	读回通道 0、1、2 的计数值和状态	命令无效
8	1	1	1	1	1	0	1	0	无效命令	命令无效

8.6 欲使 8254 的通道 2 产生一周期为 1 ms 的脉冲序列，试编写初始化程序。设已有基准时钟频率为 4MHz。

解：由题意可知：时钟周期=l /(4 MHz)=0.25us

通道 2 应工作在方式 2，计数初值=脉冲周期/基准时钟周期=(1ms)/(0.25 us)=4000,假定控制寄存器和通道 2 地址为 PORT_CTR 和 PORT_CH2,初始化程序如下：

```
MOV     AL,0B4H                    ;通道 2 方式 2、二进制计数
MOV     DX,PORT_CTR
OUT     DX. AL
MOV     AX, 4000
MOV     DX, PORT_CH2
OUT      DX,.AL                    ;写低 8 位
MOV     AL, AH
OUT      DX .AL
```

8.7 用'8254 组成一个实时时钟系统通道 0 作为秒信号产生器。通道 1 和 2 分别用做分和时的计时。画出硬件电路。并编写主程序和中断服务程序(设系统已提供的基准时钟颇率为 50kHz)。

解:硬件电路如图 8.10 所示。8254 各端口地址为 80H~83H，通道 0、通道 1 和通道 2 均工作在方式 2 分别输出周期为 1 s,1 min 和 1h 的定时中断信号。各通道计数初值为:

定时时间通道 0 计数初值=定时时间/时钟周期=1s/（1/(50KHz)）=50000

通道 1 计数初值=60 s/1=60

通道 2 计数初值=(1 h)/(1 min)=60

实时时钟系统主程序和中断服务程序如下:

```
DATA    SEGMENT
HOUR      DB    0
MINUTE     DB    0
SECOND     DB    0
DATA       ENDS
CODE       SEGMENT
```

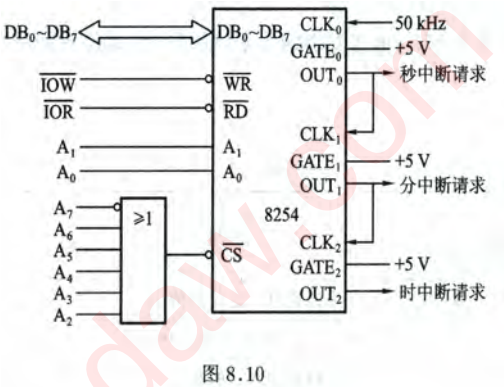


图 8.10

```
                ASSUME    CS : CODE , DS : DATA
START:     MOV          AX,DATA                    ;主程序
           MOV          DS,AX
           MOV           AL,34H                    ;通道 0 方式 2
           OUT           83H,AL                    ;写人 8254 通道 0 方式控制字
           MOV           AX, 50000                 ;写人计数器 0 的初值
           OUT           80H,AL
           MOV           AL,AH
           OUT           SOH,AL
           MOV           AL,54H                    ;通道 1 方式 2，写低字节
           OUT           83H,AL                    ;写人 8254 通道 1 方式控制字
           MOV AL，60                              :写人计数器 1 的初值
           OUTS     81H，AL
           MOV AL，94H                             通道 2 方式 2，写低字节
           OUT 83H，AL                              写人 8254 通道 2 方式控制字
           MOV AL，60                              写人计数器 2 的初值
           OUT 82H，AL
              CLI                                   ;关中断

…..                                                :填写中断向量(略)
              STI                                   ;开中断
….                                                 ;其他处理
              MOV AH，4CH
              INT     21H
INI_ HOUR    PROC                                  ;时中断处理
              INC    HOUR
              CMP    HOUR，24
            JB NEXT
              MOV HOUR，0
NEXT:    IRET
INI HOUR ENDP
INI MINUTE PROC                                    ;分中断处理
            INC MINUTE
            CMPM 生 NUTE .60
            JB NEXT
              MOV MINUTE，0
NEXT:    IRET
INI MINUTE ENDP
INI SECOND PROC                                    ;秒中断处理

              INC SECOND
              CMP SECOND，60
              JB NEXT
              MOV SECOND，0
```

NEXT:IR 五 T
INI SECOND ENDP
CODE ENDS
END START

8.8 试简要说明 8254 应用于 8 位 8088,16 位 8086/80286 和 32 位 80386/80486 等不同字长的 PC 机系统时。与地址总线的接口有什么不同?在 16 位和 32 位系统中，数据线的连接对地址总线接口有影响吗?若有。如何影响?

解: 8086/80286 与 8 位的可编程接口芯片(如 8253/8254)连接时，因其数据线是 16 位的.此时，要使能正常工作，与地址总线的接口通常有两种方法:

(1)按 16 位端口编址，使接口中的端口只具有偶地址，即让系统地址线 A0 参加高位地址译码，并固定为"0",8253/8254 端口地址线 A1,A0 通常与系统地址线 A2,A1 相连(也可与 A0 以外的其他地址线相连)。

(2)按 8 位端口编址，使接口中的端口地址仍是连续的，即有偶有奇，此时，8253/8254 端口地址线 A1,A0 与与系统地址线 A1,A0 与相连。

8088 同 8253/3254 连接时，因其数据线是 8 位的.所以只能采用 8 位端口编址。

当 80386/80486 与 8 位的可编程接口芯片连接时 I/O 接口更灵活，即可用 8 位 16 位端口编址.还可用 32 位端口编址，此时，要让系统地址线 A0,A1 参加高位地址译码，并固定为 00, 8253/8254 端口地址线 A1,A0 彻通常与系统地址线 A3,A2 相连(也可与 A0,A1 以外的其他地址线相连)。

在 16 位和 32 位系统中，8253/8254 与数据总线的连接对地址总线接口有影响。当 8253/8254 的数据线与系统总线的 $D_7 \sim D_0$ 相连时.与地址总线的接口可按 8 位、l6 位或 32 位端口编址，若与系统总线的 $D_{15} \sim D_8$ 相连时，可按 16 位或 32 位端口编址;若与 D_{16} 以上的数据总线相连时，只能按 32 位端口编址。

8.9 某系统中 8254 芯片的通道 0~通道 2 和控制字端口号分别为 FFF0~FFF3H。定义通道。工作在方式 2 . CLK_0 = 5 MHZ,要求输出 OUT_0 =1 kHZ 方波;定义通道 1 工作在方式 4，用 OUT_0 作计数脉冲，计数值为 l000，计数到 0 向 CPU 发中断请求.CPU 响应这一中断后继续写人计数值 1000 ,重新开始计数，保持 1s 向 CPU 发出一次中断请求。请画出硬件连接图，并编写初始化程序。

解:硬件连接如图 8.11 所示。通道 a0 计数初值为:

通道 0 计数初值= (5 MHZ)/(1 kHZ) = 5000

初始化程序如下:

```
MOV     AL,36H;通道 a 方式 3、二进制计数
MOV     DX,0FFF3H
OUT     DX,AL
MOV     AX,5000
MOV     DX,0FFF0H
        ;写通道 A 计数初值
OUT     Dx.AL    ;写低 8 位
MOV     AL.AH
OUT     DX,AL            ;写高 8 位
```

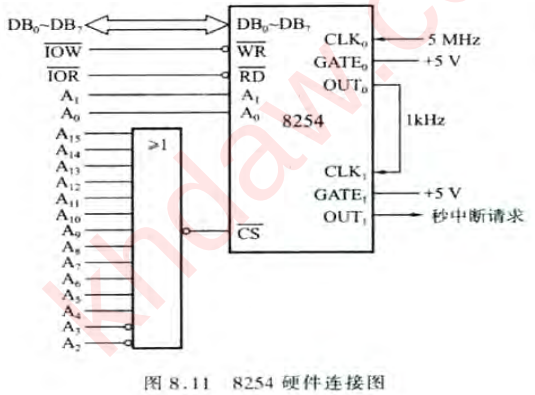


图 8.11 8254 硬件连接图

```
    MOV     AL,78H              ;通道 1 方式 4、二进制计数
    MOV     DX,0FFF3H
    OUT     DX, AL
    MOV     AX，1000
    MOV     DX.0FFF0H           ;写通道 1 计数初值
    OUT     DX,AL               ;写低 8 位
    MOV     AL,AH
    OUT     DX,AL               ;写高 8 位
```

8.10 若使用 8254 对外部脉冲进行计数.计数时间持续期由另一外来信号控制.计致过程结束时向.8259A 发中断请求信号。试编写 8254 的初始化程序.并说明一片 8254 的最大计数值为多少，如何考虑的?(编程时 8254 的地址自定。)

解:假定用通道 a 对外部脉冲进行计数，工作于方式 0，初值为 0;通道 2 工作于方式之，对一频率为 4 MHz 的时钟脉冲进行计数，用于产生 1ms 的定时信号，于是:

时钟周期=l/4 MHz = 0.25 us

通道 2 应工作在方式 2，计数初值“脉冲周期/基准时钟周期=(1 ms)/(0.25 us) =4000

假定控制寄存器、通道 0 和通道 2 地址为 PORT_ CTR,PORT_ CH0 和 PORT_ CH2.初始化程序如下：

```
MOV       AL, 0B4H                  ;通道 2 方式 2、二进制计数
MOV       DX, PORT_CTR
OUT       DX, AL
MOV       AX , 4000
MOV       DX;PORT_CH2
OUT       DX,AL                     ;写低 8 位
MOV       AL, AH
OUT       DX,AL                     ;写高 8 位
MOV       AL, 30H                   ;通道 0 方式 0、二进制计数
MOV       DX, PORT _CTR
OUT       DX,AL
MOV       AL, 0
MOV       DX, PORT _CH0
OUT       DX,AL                     ;写低 8 位
OUT       DX , AL                   ;写高_8 位
```

一片 8254 的最大计数值为 1000000000000H，此时，要将三个计数通道申行连接，即通道 0 的输出 OUTa 接通道 1 的计数辘人 CLKI，通遣 1 的物出 OUT，接通道 Z 的计数翰人 CLK2 0

8.11 能否利用上题思路设计一个智能化频率计?请说明原理，并画出原理框图。

解:可以，原理如图 8.12 所示。通道。工作于方式 2 产生定时(检侧)信号，通道 I 工作于方式。用于对外部脉冲计数，计数初值为 0。当定时时间到时，读出通道 1 计数值 N，则脉冲频率为:

脉冲频率=(65 536-N)/定时时间

8.12 假设某 PC 机 I/O 卡上的 8254 连接至一个，1 kHz 的时钟，用该 8254 以 BCD 码格式保持一天中的时间，精度为秒。在 HOURS, MINUTES, SECOND, AM, PM(时、分、秒、上午、下午)等字节装人当前时间后，启动 8254 开始计

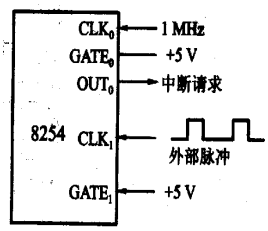


图 8.12 智能频率计原理图

时。又设 8254 的地址为 02C0~02C3H, 8254 产生的中断请求信号连接到 PC 总线的 IRQ2。试编写 8254 的初始化程序和每秒结束时修改时间的中断服务程序。

解: $T_{CLK}=1/(1kHz)=1\text{ms}$

假定用 8254 通道 0 做 1s 定时，则计数初值为:(1 s)/(1 ms)=1000 程序如下:

```
        DATA SEGMENT‘-
          HOURS     DB     0                    ;定义时计数器-
          MINUTES   DB   ‘:0’                   ;定义分计数器
          SECOND    DB     0                    ;定义秒计数器
        AM          DB     0                    ;上午标志(0 表示上午)
        PM          DB     OFFH                 ;下午标志(0 表示下午)
          DIS_BUF   DB   'INPUT TIME(00:00:00):$’    ;定义输人提示信息
       BUF          DB   9,   ?                        ;定义输人缓冲区
      TIME          DB     ’00:00:00’ , 0DH,’$’
     SAVE_IP      DW     ?
     SAVE _CS     Dw     ?
     I RQ2        EQU    0AH
DATA   ENDS
CODE   SEGMEZTT
ASSUME   CS: CODE, DS: DATA
START:   MOV      AX,DATA
         MOV      DS, AX
         MOV      DX,   OFFSE T   DIS_BUF          ;显示输人提示
         MOV     AH，9
          INT      21H
          MOV      DX,   OFFSET   BUF
          MOV     AH，OAH
            INT      2IH
…..                                               :转换成 BCD 码〔略)
          MOV      DX,2C3H
          MOV      AL, 34H                        ;通道 0 方式 2.二进制计数
          OUT     DX.AL
          MOV      DX , 2C0H
          MOV      AX,1000                         写 1s 计数初值
          OUT      DX,AL
          MOV      AL,AH
          OUT      DX,AL
          CLI                                     ;关中断
          MOV      DI ,I RQ2*4
          CLD
          XOR    AX, AX
          MOV    ES, AX                           :ES 指向中断向量表基址
          MOV      AX，ES:[DI]                    :保存系统中断向量
          MOV      SAVE_IP ， Ax
```

```
                MOV         AX，ES:[DI+2]
                MOV         SAVE_CS, Ax
                MOV         AX,OFSET INT_ROUT          ;填写中断向量的偏移地址
                STOSW
                MOV        AX, SEG INT_ROUT
                STOSW
                  STI
AGAIN:       MOV        AH,O1H
                INT         16H
                JZ          AGAIN
                CLI
                MOV         DI, IRQ2‘4
                CLD
               XOR AX,AX
              MOV         ES,AX
              MOV         AX, SAVE    IP             ;填写中断向量的偏移地址
              STOSW
              MOV         AX，SAVE    CS             ;填写中断向量的段基址
              STOSW
              STI                                    ;开中断
              MOV         AH,4CH
              INT         21 H
INT_ROUT      PROC                                   ;1 s 中断处理程序
              PUSH     AX
                STI
              MOV       AL, SECOND
              AD            D     AL,1
              DAA
              MOV     SECOND, AL
              CMP       AL , 60H
              JB        NEXT’
              MOV       SECOND, 0
              MOV       AL, MINUTES
              ADD               AL,1
              DAA
              MOV       MINUTES,AL
              CMP       4AL,60H
            JB       NExT
            MOV       MINUTES,0
            MOV       AL, HDURS
            ADD       AL, 1
            DAA
            MOV     HOURS, AL
            CMP       AL,12H
```

```
        JB      NEXT
        JZ      AM     $
        MOV     HOURS, 1                          ;时计数器为 1(13 时)
        JMP   NExT
        AM$:    NOT   AM                          ;修改上、下午标志
        NOT     PM:
        NEXT:   CLI
        POP   AX
        STI
        IRET
        INT_ROUT   ENDP
        CODE   ENDS
        END   START
```

8 .1.3 试利用 PC/AT 机系统板上的 8254 通道 0 产生年、月。日、时、分、秒的时间记录并.显示于 CRT 屏幕上。在键盘上输人当前的年、月、日、时、分、秒后启动 8254 计时操作。

解:PC/AT 机系统板上 8254 通道 0 的定时时何为 54.925 ms(十六进制约 36.ECCCH ms) , 用户的定时服务处理程序使用 INT　1CH 软中断调用。以下程序仅进行时、分、秒处理，年、月、日处理类同。为便于显示，时、分、秒均用 ASCII 码数表示。程序如下:

```
            DATA   SEGMEMT_
YEAR          DW      2002
MONTH     DB   01
DAY     DB   01
TIME54ms   DW   0,0
DIS_BUF   DB   'INPUT TIME(00:00:00) :$'
BUF     DB   9,?
TIME     DB '00:00:00',0DH,'$'
SAVE_IP     DW   ?
SAVE_CS   DW   ?
DATA ENDS
  CODE   SEGMENT
            ASSUME   CS :CODE , DS :DATA
  START: MOV            AX, DATA
            MOV           DS,AX
            MOV           DX,OFFSET DIS   BUF                ;显示输人提示
            MOV           AH, 9
              INT          21H
            MOV           DX, OFFSET BUF                     ;输人时间初值
            MOV AH, OAH                                      ;此处未进行有效性检测
            INT           21H
            CLI                                              ;关中断
            MOV           DI，1CH*4
            CLD
            XOR           AX,AX
```

```
        MOV         ES,AX                       ;ES 指向中断向量表基址 0000H
       MOV         AXES: [DI]                   ;保存系统中断向量
        MOV         SAVE_IP, AX
        MOV         AXES: [DI+2]
        MOV         SAVE_CS, AX
        MOV         AX,OFFSET INT _ROUT         ;填写中断向量的偏移地址
        STOSW
        MOV         AX; SEG 1NT . ROUT
        STOSW
         STI
AGAIN:MOV          AH,O1H                       ;有按健?(用按键控制程序结束)
          INT          16H
      JZ           AGAIN                       ;无按键转 AGAIN
          CLI                                   ;关中断，以恢复系统中断向量
      MOV          DI,1CH * 4                   ;取 INT 1CH 的中断向量地址
      CLD
      XOR          AX , AX
      MOV          ES, AX                       ;ES 指向中断向量表基址 0000H
      MOV          AX,SAVE   IP                 ;填写中断向量的偏移地址
      STOSW
      MOV    AX,SAVE_CS                         ;填写中断向量的段基址
      STOSW
      ST I                                      ;开中断
      MOV      AH , 4CH
      INT    21H
      I NT_ROUT     PROC
          PUSH     DX
      PUSH   AX
      STI
      ADD   TIME54 ms,0ECCCH
          ADC:     TIME54 ms[2].36H
          CMP      TIME54 ms[2]，1000
      JB     NEXT
          SUB      TIME54 ms[2]，1000
          I NC.      TIME[7]                    ;秒计数器个位加 1
          CMP      TIME[7」，’9’
        J B        NEXT
          MOV..    TIM1;['7]，’0’              ;秒汁数器个位清 0
      INC      TIME[6]                          ;秒计数器十位加 1
      CMP    TIME[6]   ,’6’
      JB    NEXT
      MOV   TIME[6]   ,’0’                      ;:秒计数器十位清 0
      INC      TIME[4]                          ;分计数器个位加 1
      CMP    TIME[ 4].'9’
```

```
        J BE       NExT
        MOV        TIME[4]，'0’
        INC        TIME[3]
        CMP        TIME[3]，'6 ‘
        JB       NExT
        MOV        TIME[3]，’0’
        INC    . TIME[1]
        CMP      TIME[ 1]，’9’
        JBE     NEXT 1
        MOV     TIME[1].’0’
              INC       TIME                              ;时计数器十位加 1
        NEXTl:CMP     WORD PTR TIME，'42’
                    JNZ       NEXT
                      MOV       WORD PTR TIME, '00’
        NEXT:MOV       AH, 09H                           ;用 DOS9 号功能调用显示时间
                      MOV      DX,OFFSET TIME
                    INT      21H
                      CLI
                      POP       AX
                      POP       DX
                      STI
                      IRET
        INT    ROUT       ENDP
        CODE         RNDS
                      END       START
```

8 .14 试编写“程序，使，PC/AT 系统板上的发声电路发出 200 Hz 至 900Hz 的警报声。

解:PC/AT 系统板上 8254 通道 2 用于产生音频信号，时钟频率为 1.1931816MHz，用 16 进制表示为 1234DDH(Hz),要发出 200 Hz 至 900Hz 频率连续变化的报警声，即 OUT2 要输出 200 Hz 至 900 Hz 频率连续变化的方波。程序如下:

```
          DATA    SEGMENT
              P8254CTR      EQU         43H               ;系统板 8254 控制口地址
              P8254PORT2      EQU      40H               ;系统板 8254 通道 2 地址
              P8255PB      EQU      61H                  ;系统板 8255B 口地址
            CLK                DW       34DDH,12H         ;定义时钟频率
          DATA    ENDS
          CODE    SEGMENT
              ASSUME      CS:CODE , DS : DATA
        START:    MOV           AX,DATA
                  MOV         DS, AX
                  MOV         AL,03H                ;8255 已由系统初始化，此处不做
                  OUT         P8255PB,AL            ;PBD,PBl 置 1、使 GATE2 -1，并
                                                    ;开放 8254 OU'T2 输出
                  MOV      BX    200
      AGAIN:      MOV AX    CLK                       ;取 CLK2 时钟频率
```

```
        MOV     DX,CLK[2]
        DIV     Bx                          ;计算 8254 通道 2 计数初值
        CALL    SOUND                       ;调用发声程序
        ADD     Bx ,10                      ;调整方波输出频率
        CMP     Bx,900                      ;发声完?
        JBE     AGAIN
        MOV     AL, DDH
        OUT     P8255PB,AL                  ;PED,PB1 置 D 关闭发声
        MOV     AH.4CH
        INT 21H
;以下为发声子程序。人口参数:Ax 为计数初值
SOUND   PROC
            PUSH    Bx
            PUSH    Cx
            PUSH    Ax                      ;保存计数初值(人口参数)
            MOV     AL,0B6H                 ;8254 通道 2 方式 3 二进制计数
            OUT     P8254C TR,AL
            POP     AX
            OUT     P8254PORT2,AL           ;写计数初值
            MOV     AL,AH
            OUT     P8254 PORT2 , AL
            MOV     Bx,40                   ;延时发声一段时间
DELAY:MOV   Cx.2801
D110MS:     LOOP    D110MS
            DEC     Bx
            JNZ     DELAY
            POP     Cx                      ;恢复现场
            POP     Bx
            RET
SOUND       ENDP
CODE        ENDS
            END     START'
```

8.16 试利用 8254 设计一个多波群发生器，该发生器周期性地物出 500 kHz , 200 kHz . L00 kHz,50kHz,20kHz, 10kHz,5 kHz,2 kHz,l kHz 的正弦波，每种拯率的信号的持续期都为 10ms。假定可提供给 8254 的时钟颇率为 5 MHz. 8254 的地址为 05C8H ~05CBH。试完成硬件和软件设计。

解:用 8254 通道 0 输出方波中断，通过动态改变计数初值改变方波频率; 通道 1 用于产生 10ms 中断，以修改方波频率，计数初值固定为:10 ms/(1 /(5 000 kHz))=50000
硬件连接如图 8.13 所示。程序如下:

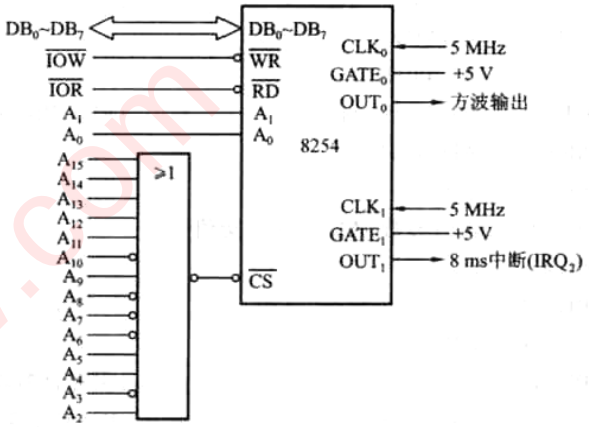


图 8.13　多波群发生器电路

方波输出

图 8.13 多波群发生器电路

```
  DATA      SEGMENT
    BASE    DW          500 , 200，100,60,20        ;定义频率表(kHz) ,0 结束
            DW          10,5,2，1,0
    CNT     DW          0
    1RQ2    EQU         0AH                         ;IRQ2 中断向量号
    CLK     EQU         5000                        ;时钟频率 5 000 kHz
    PCTR    EQU         05CBH                         ;8264 控制口地址
    PORT0   EQU         O5C8H                         ;8264 通道 0 口地址
    PORTl   EQU         05C9H
DATA   ENDS
CODE   SEGMENT
    ASSUME   CS :CODE , DS :DATA
START:      MOV      AX,DATA
            MOV      DS,DATA
            MOV     DX ,PCTR
            MOV      AL，0B4H
            OUT      DX,AL                          ;通道 1 方式 2，二进制计数


  MOV       Ax,50000                                :通道 1 计数初值
            MOV      Dx，PDRT1
            OUT      Dx，AL
            MOV     AL , AH
            OUT     DX , AL
            CLI                                     ;关中断
            MOV      CNT,0                          ;CNT 指向第一个频率
            MOV      DI，IRQ2*4                     ;取 IRQ2 的中断向量地址
            CLD
            XOR     AX , AX
```

```
                MOV     ES , AX                          ;ES 指向中断向量表段基址
MOV           AK,OFFSET INT_ROUT                         ;填写中断向量的偏移地址
                STOSW
                MOV       AX, S5EG INT_ROUT              填写中断向量的段基址
                STOSW
                STI                                      ;开中断
AGAIN:      MOV           AH,01H                         ;有按键?
                INT 16H
                JZ            AGAIN                      无按键转 AGAIN
                MOV           AH,4CH
                INT           21H
INT_ROUT    PROC                                         ;10 ms 中断处理程序
                  PUSH    BX
                  PUSH    Dx
                  PUSH    AX
                  PUSH    SI
                    STI
                  MOV     AL .36H                      8254 通道 0 方式 3 二进制计数
                  MOV       Dx . PCTR
                  OUT     DX , AL
                  MOV       SI,CNT                     ;SI 指向当前输出频率
                  ADD       CNT,2                      ;CNT 指向下一输出频率
                  MOV       BX., DASE[ SI ]              ;取当前输出频率
                  CMP       BX,0                       ;频率为 0?
                  JNZ       NEXT                       ;非 0 跳过
                  MOV       CNT, 0                       ;CNT 指向第一个输出频率
                MOV    BX,BASE                          ;取第一个输出频率
  NEXT:           MOV       AX，CLK                      ;取输人时钟频率
                  XOR       DX , DX
                  DIV       BX                           ;计算 8254 通道 0 计数初值
                  MOV       DX, PORT0
                  OUT       DX , AL                      ;写计数初值
                  MOV       AL,AH
                  OUT       DX,AL
                    CLI                                  ;恢复现场
                  POP       SI
                  POP       AX
                  POP       DX
                  POP       BX
                  STI
                  IRET
INT_ROUT    ENDP
CODE          ENDS
                  ‘                    END    START
```

8 .17 试说明如何利用 8254 测量从同一信号线送来的两个脉冲的时间间隔，测量的最大时间间隔为 1h，读时精度为 1 ms。假定时钟频率为 5 MHz,8254 的地址为 05C8H~05CBH。

解:测量方法如图 8..14 所示。8254 的通道 0 工作于方式 3 用于产生周期为 1 ms 的测量脉冲，计数初值为:

计数初值=1 ms/(1 /5 MHz)=5 000

通道 1 工作于方式 0 用于测量脉冲间隔，方法是将 OUT0 输出的测量脉冲接到 CLK1 端，被测量信号经一级非门倒相后接到 8254 的 GATE1 端（即将侧鱼脉冲间隔转换为测量脉冲宽度).保证通道 1 仅在脉冲间隔间对测量脉冲计数，计数初值设为 0000H 当测量完时.读出通道 1 当前值 N,可计算出所测脉冲的间隔:

脉冲间隔=（65 536-N)* T_{CLK_1} = (65 535-N)*1 ms =65 536-N(ms)

被测量信号同时用作控制启动和停止测量的中断请求信号。方法是用第一个被测脉冲的上升沿产生启动测量中断请求信号，在中断程序中写人通道 1 计数初值 0 启动通道 1 测量;用第二步被侧信号产生结束测量中断请求信号，在中断服务程序中完成上述读通道 1 当前值 N 和计算脉冲间隔的任务。

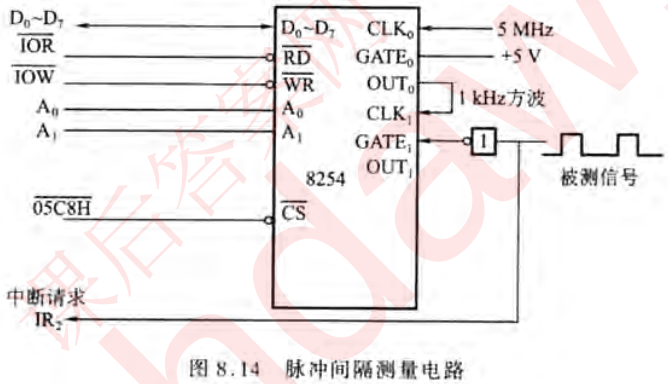


图 8.14 脉冲间隔测量电路

按此原理编制的测量程序段如下:

```
…..
STATS   DB          0                ;定义启停状态，0 为启动，1 为停止
TIME    DB          ?                :保存脉冲间隔
…..
INI8253:MOV    AL.36H                ;通道 0 方式 3
        MOV    DX,05CBH
        OUT    DX，AL                ;写入 8254 通道 0 方式控制字
        MOV    Ax, 5000H             :写人计数器 0 的初值
        MOV    DX , 05C8H
        OUT    DX,AL
        MOV    AL,AH
        OUT     DX,AL
        MOV    AL,70H                 通道 1 方式 0
        MOV    DX,05CBH
        OUT   DX , AL                ;写入 8254 通道 1 方式控制字
        MOV   AX，0H                 ;写人计数器 1 的初值
        MOV    DX , 05C9H
        OU T   DX,AL
```

```
                OU T     DX,AL.
……
INTR:PUSH       DX                          ;测量中断程序
                PUSH    CX
                PUSH    AX
                CMP     STAT,0              :启动测量?
              JNZ   READ                    ;STATS!=0,计算脉冲间隔
              MOV     AL,0                  ;写通道 1 计数初值 0，开始测量
              MOV     DX,05C9H
              OUT      DX,AL
              OUT      DX,AL
              MOV     STATS , 1             ;置状态为停止测量
              JMP      EXIT
              READ:      MOV     AL,40H     ;计数器 1 的锁存命令
              MOV      DX,05CBH
              OUT       DX,AL
              MOV         DX,05C9H
              IN     AL,DX                  ;读通道 1 当前计数值
              MOV    CH,AL
              IN     AL,DX                  ;读高位
              MOV     CL, AL
              IN       AL, DX
              MOV     CH,AL
              NEG      CX                  ;计算被测脉冲宽度
              MOV     TIME，CX             ;保存测量结果
              …..                          ;屏蔽 IR2 中断
               EXIT:         POP     AX
               POP     CX
               POP     DX
               STI
               IRET
```

9.4 写出下列两种情况下 8,}55A 的工作方式控制字(包括 I/O 方式控制字和必要的按位置位/复位控制字)。

(1) 8255A 用做键盘和终端地址接口，如图 9 ..4 所示。.

(2)8255A 用做基本软盘接日，如图 9.5 所示。

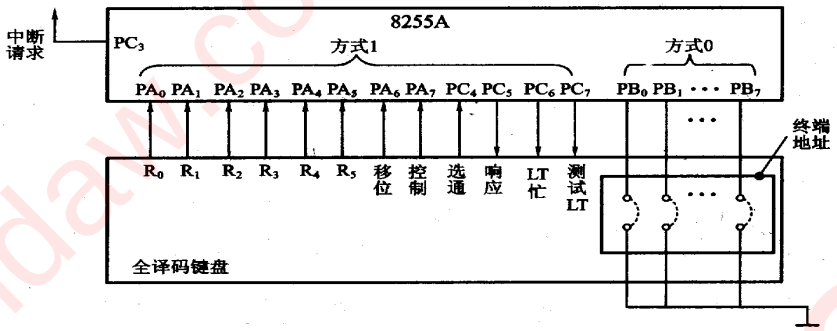


图 9.4 8255A 作为键盘和终端地址接口

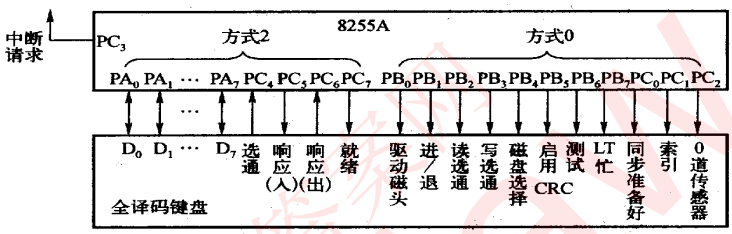


图 9.5 8255A 作为基本软盘接口

解:(1)由图 9.4 可知:A 口工作在方式 1 输人，采用中断读键盘，C 口的 PC4 , PC5 为 A 口方式 1 输人提供固定的握手联络信号，而 PC6,PC7 用于输出“LT 忙”和“测试 LT"，所以 C 口高 4 位工作在方式。输出，B 口用于输人终端地址，所以 B 口应工作在方式。输人。由此分析可知，8255A 的初始化包括设置工作方式和开中断操作，其控制字为:

工作方式控制字:1011001 x B

按位置位/复位控制字(开放中断 INTEA=1，即 PC4 置位):00001001B

(2) A 口工作在方式 2 中断方式输人/输出，B 口和 C 口低 4 位工作在方式 0 输出，所以 8255A 的初始化也包括设置工作方式和开中断操作，其控制字为:

工作方式控制字:11 x x x 000B

开放输人中断按位置位/复位控制字，即 PC4 置位:0000l001B

开放输出中断按位置位/复位控制字，即 PC6 置位:00001101B

9.5 设 8255A 的端口 A,B,C 和控制寄存器的地址为 F4H,F5H,F6H,F7H，要使 A 口工作于方式 0 输出，B 口工作于方式 1 输人.C 口上半部输人，下半部输出，且要求初始化时使 PC6=0.试设计 82SSA 与 PC 系列机的接 A 电路，并编写初始化程序。

解:82SSA 与 FC 系列机的接口电路如图 9.5 所示。初始化程序如下: •

```
MO V    A L , 10001110F3          ;方式字
  OUT      0F7H, AL
MOV      AL,00000110B              ;PC6=0
  OUT      0F7H, AL
MOV      AL,00000101               ;开中断
  OUT      0F7H,AL
```

9.6 在 PC 系列微机系统中，用 8255A 做某快速启停电容式纸带机接口的硬件连接如图 9.7

所示,试用汇编语言编写 8255A 的初始化程序。

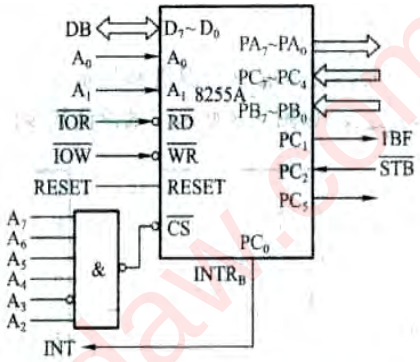


图 9.6 8255A 与 PC 系列机接口电路

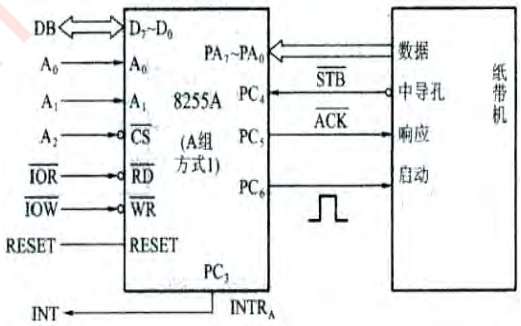


图 9.7 8255A 作为纸带机接口的连接电路

解:由图 9.7 可知，A 口工作于方式 1 中断输入.PC6 用于输出正脉冲启动纸带机，所以 C 口上半部工作于方式 0 输出。A 口、B 口、C 口和控制寄存器的地址为 x x x x x 000B、 x x xx x 001B, x x x x x 0l0B, x x x x 011B，可任选其中的一个地址、编制初始化程序如下:

```
MOV     AL,10110x x xB          ;A 口方式 1 输人，C 口上半部愉出卜.
OUT     03H;AL                  ;写方式控制字
MOV     AL, 00001100B           ; Pq 输出正脉冲启动纸带机
OUT     03H,AL
MOV     AL,00001101B
OUT     03H,AL
MOV     AL,00001100B;
OUT     03H,AL
MOV     AL,00001001B.           ;PC4 置位开放中断
OUT     03H,AL，
```

9.7 用 8255A 做某 PC 系列微机系统中的键盘和显示器的接口，A 口与一全译码键盘连接，B 口与一自扫描视频显示器连接，如图 9.8 所示。试设计其接口驱动程序。

解:A 口工作于方式 1 中断输人，B 口工作于方式 1 中断输出，C 口上半部工作于方式 0 输出。A 口、B 口、C 口和控制寄存器的地址为 x x x x x 000B, x x x x x 001B, x x x x x 010B,xxxxx011B，可任选其中的一个地址。接口驱动程序包括 8255A 的初始化程序和键盘输人及显示输出程序。

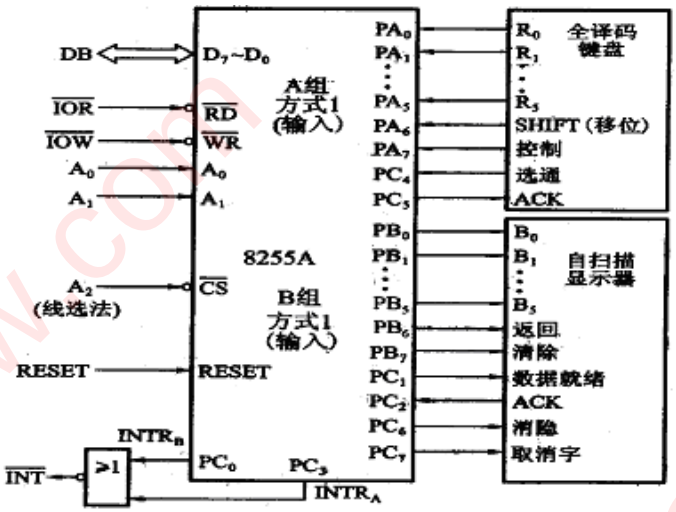


图 9.8　8255A 作为键盘和显示器的接口电路

8255A 初始化程序如下:

```
INI8255:MOV     AL,0B4H
        OUT     03H,AL
        MOV     AL,09H
        OUT     03H,AL
        MOV     AL,05H
        OUT     03H,AL
```

输人/输出中断处理程序如下:

```
INT_IO: PUSHA
        IN      AL,02H
        TEST    AL , 08H
        JNZ     KEY
        TEST    AL,O1H
        JNZ     DISP
EXIT:   POPA
        STI
        IRET
KEY:    ….
        JMP     EXIT
DISP:   …….
        JMP     EXIT
```

9.8 用 PC 系列微机实现对注塑机的时间顺序控制。注塑机生产一个工件的工艺流程为:合模(1s)—注射(2 s)—延时(3 s)—开模(1 s)—产伸(1 s)—产退(1 s) 。假若用 8255A 的 B 口 PB。~PB5 每根线控制一个执行机构动作，用 PA7 和 PA6 作为掉电和低温警告监视输人。如果正常，各执行机构按工艺流程顺序周而复始地切换;一旦出现异常，则通过 PC0 控制一红色 LED 发亮，作为故障报警，并设置 6s 故障处理时间，时间到，若故障已排除，则系统又继续运行，否则停止生产。

(1)请设计出硬件连接电路;

(2)按控制要求编制程序。

解:硬件连接如图 9.9 所示，8255A 各端口地址为 0F4H~ 0F7H, A 组、B 组均工作在方式 0。采用软件延时(用 INT 15H 的 86 号功能调用)，控制程序如下:

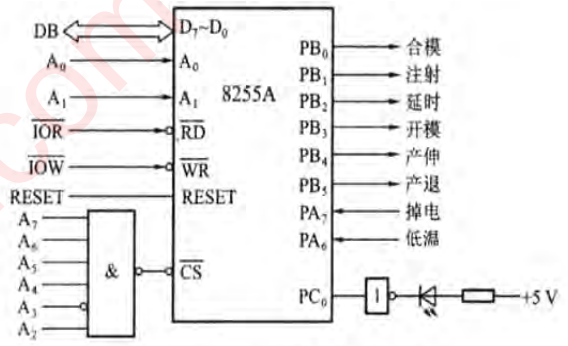


图 9.9　8255A 控制注塑机的连接电路

```
        DATA SEGMENT
             DELAY   TIME   DB    1,2,3,1,1,1        ;定义延时秒值
……..
DATA    ENDS
CODE     SEGMENT
        ASSUME GS: CODE, DS: DATA
START:      MOV     AX,DATA
            MOV      DS,Ax
            MOV      AL, l 0010000B                  : A 口方式 0 输人、B 和 C 口方式 0 输出
            OUT      0UF7H.AL
            MOV     AL , 00000000B    PC0=0          .发光二极管灭，系统正常
            OUT     0F7H.AL
BEIGIN:      MOV     DL,01H                          ；PB 口输出控制字初值，从合模开
            MOV      BX,0                           ;BX 指向对应延时秒值存放单元
REPEAT:    IN        AL.0F4H                        ;读 A 口
            TEST     AL, 03H
            JNZ      ERROR                          ;转故障处理
            MOV      AL,DL                          ;取 PB 口输出控制字
            OUT      0F5H,AL                        ;控制执行机构
            SHL      DL, 1                          ;输出控制字左移 1 位。指向下一工艺
            MOV     CL, DELAY_TIME[BX]              ;取延时秒值
WAIT$:CALL     IIELAY1S                            ;延时 1s
              DEC       CL.
              JNZ       WAIT$
            INC     BX                              : BX 指向下一道工艺延时秒遗
            CMP      BX,6                           ;6 个工艺处理完?
            JNZ      REPEAT                         :未完.继续下一道工艺
            JMP      BEIGIN                         ;重开始下一工艺流程
ERROR:    MOV     AL,01H
                OUT     0F7H ,AL
                MOV     CX, 6
WAIT1:CALL      DELAY1S
```

```
                LOOP    WAIT1
                IN      AL ,    0F4H
                TEST    AL , 03H
              JNZ    EXIT
                MOV     AL ,00H
                OUT    0F7H
              JMP    REPEAT
EXIT:MOV     AH, 4CH
                INT    21H
;延时 1s 子程序，用 INT 15H 的 86H 号功能调用
;1s=1 000 000 us=F4240H us
DELAYIS     PROC
                PUSH    AX                          ;保护程序中用到的寄存器
                PUSH    CX
                PUSH    DX
                MOV    AH, 86H                      ;INT 15H 调用的功能号
                MOVCX ,0FH                          ;延时时间的高 16 位
                MOV    DX ,4240H                     ;延时时间的低 16 位
                INT    15H
                POP    DX                           ;恢复保存的寄存器值
                POP    CX
                POP    AX
                  RET
DELAYIS ENDP
CODE                ENDS
                  END       START
```

9.9 拟用 8255A 和 AT 机系统板上 8254 通道 1 设计一个十字路口交通灯管理系统。要求:东西、南北方向各装有红、黄、绿指示灯;东西向和南北向通行时间分别为 30 s 和 20 s;由绿灯变为红灯前的 3s 内，绿灯灭而黄灯亮;时间由两位 7 段 LED 显示器倒计时显示;当某方向显示红灯时，允许人工干预强行改变两个方向的指示灯颜色，以让救护车、警车通行，但发出改变通行命令后，应有 3s 时间使显示绿灯的方向变为显示黄灯且以每秒 10 次的频率闪烁，以示警告。试完成硬件和软件设计(仍假定 8255A 和 8254 的地址分别为 03C0H~ 03C3H 和 04C0H~04C3H)。

解:用 8255A 设计的十字路口交通灯管理系统如图 9.10 所示。A 口、B 口方式 0 输出，分别控制东西方向和南北方向 LED 显示器显示定时时间，C 口方式 0 输出控制交通灯，其中 PC7~PC5 分别控制东西方向红、黄、绿灯，PC3~PC1 分别控制南北方向红、黄、绿灯。8254 用于定时.通道 0 工作于方式 3 输出 1 kHz 方波.计数初值为:

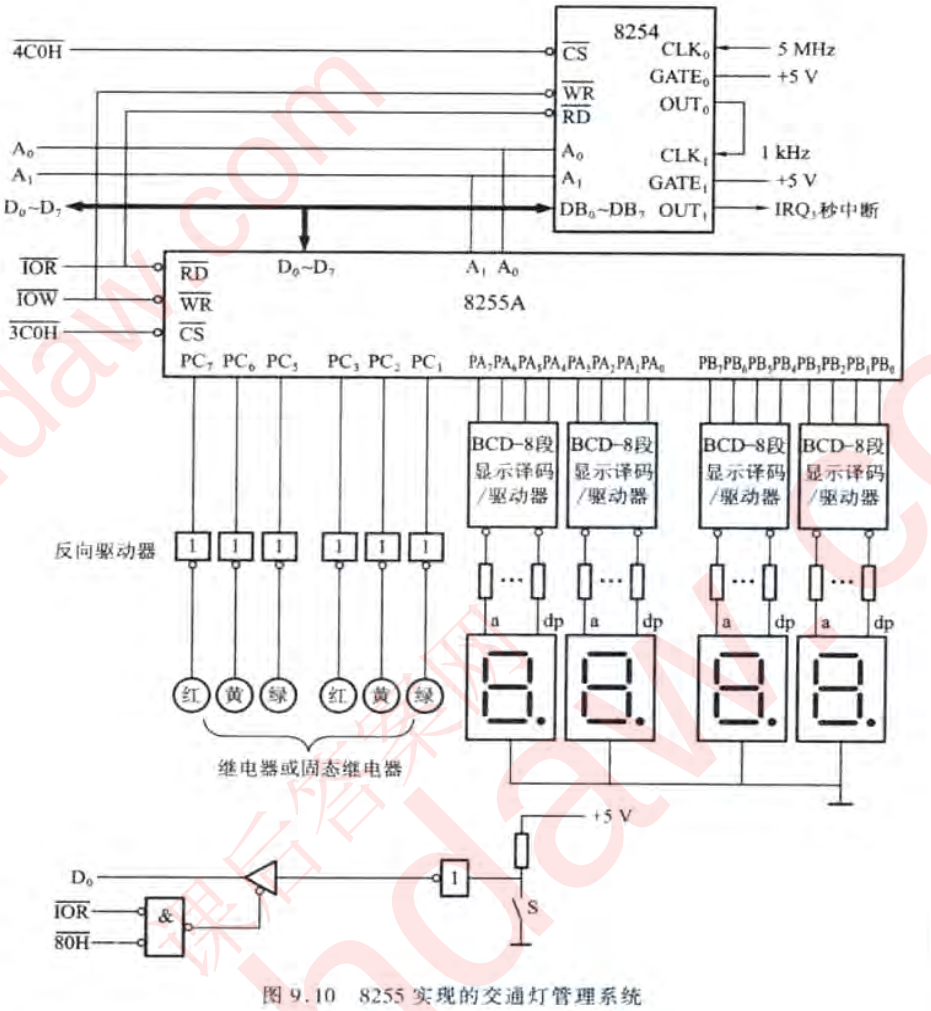


图 9.10　8255 实现的交通灯管理系统

通道 0 计数初值=(5 MHz) /(1 kHz) = 5 000

通道 1 工作于方式 2 输出周期为 1s 的负脉冲作为秒定时中断请求信号，接系统板 IRQ3，计数初值为:

通道 1 计数初值=1 s/(1 /(1 kHz)) =1 000

在秒中断处理程序中，完成开关检测和交通灯控制，处理流程如图 9.11 所示。程序如下:

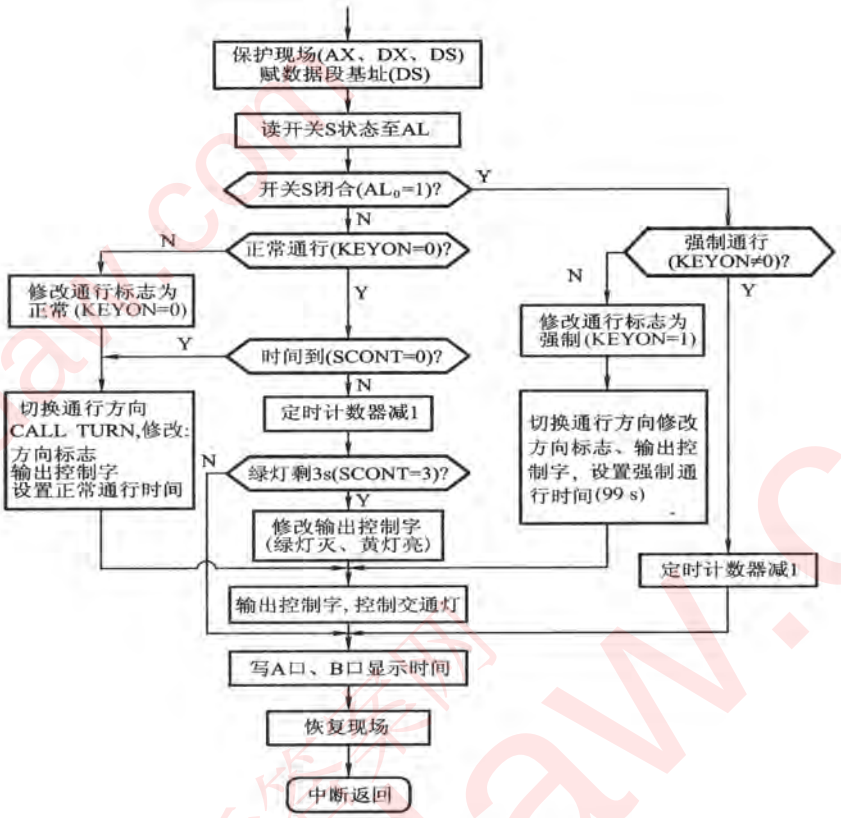


图 9.11　中断处理程序流程

```
    • MODEL     SMALL
    • STACK
    • DATA
FM      DB    1                          ;方向标志 FM = 1 东西向通，为 0 南北向通
SCONT     DB      30H                    ;秒计数器，初值 30 s(BCD 数)
KONG      DB      00101000B              ;红黄绿灯控制字，东西向绿灯亮南北向红灯亮
KEYON     DB      0                       ;强制改变方向标志，初始为正常
SAVECS     DW      ?
SAVEIP      DW      ?
KEEP      DB       ?
MODE      EQU        10000000B
 • CODE
START:MOV AX，@DATA
                MOV DS,AX
                CLI
                MOV AH，35H
                MOV AL,OBH
                INT    21H                  ;保存系统 IRQ3 原中断向量
                MOV SAVEIP,BX
                MOV SAVECS, ES
                PUSH DS
                MOV AX,SEG    TIMEIS        ;计数中断服务程序人口地址
```

```
MOV DS,AX
MOV DX,OFFSET    TIMEIS
MOV AH,25H
MOV AL,0BH
   INT    21H
   POP    DS
MOV AL,MODE                    ;写 8255 方式控制字
MOV DX,3C3H
OUT    DX , AL
MOV DX,4C3H                    ;初始化 8253 通道 0 工作方式
MOV AL, 36H
OUT    DX,AL
MOV DX,4C0H
MOV AX,5000                    ;通道 0 计数值
OUT    DX,AL
MOV AL , AH
OUT       DX,AL
MOV       DX,4C3H                    ;初始化 8253 通道 1 工作
方式
MOV       AL , 74H
OUT        DX,AL
MOV       DX,4C1H
MOV       AL,1000                    ;通道 1 计数值
OUT       DX,AL
IN       AL,21H
MOV       KEEP, AL                   ;保存系统中断屏蔽字
AND       AL, OF7H
OUT       21H,AL                     ;清 IRQ3 中断屏蔽位
STI                                  ;开中断
WAIT$:    MOV       AH,1             ;有键按下?
INT     16H
JZ    WAIT$                          ;无键按下，等待
MOV     AL, KEEP                     ;退出
OUT       21H,AL                     ;恢复系统中断屏蔽字
MOV       DX, SAVEIP                 ;恢复系统 IRQ3 中断向量
MOV       AX, SAVECS
MOV       DS , AX
MOV      AH, 25H
MOV      AL, 0BH
INT      21H
MOV       AH, 4CH
INT      21H
;计数中断服务程序
TIM1S           PROC FAR
```

```
 PUSH AX
PUSH DX
PUSH DS
MOV AX，@ DATA
MOV DS,AX
IN      AL,80H                          ;输人开关 S 状态
TEST AL,01H                             ;检测开关 S 闭合?
JNZ   LL                                ;闭合转 LL，判是否从断开到闭合
CMP   KEYON , 0                         ;判断是否从闭合到断开?
JNZ   ONTOFF                            ;从闭合到断开，转 ONTOFF
J MP   STR
ONTOFF:      MOV KEYON , 0
CALL TURN
JMP OUTPUT
LL:      CMP KEYON，0                          ;从断开到闭合?
JZ       OFFTON                                ;是，转强制切换处理
CALL STIME                                     ;强制状态，修改秒计数
值

JMP   EXIT                                     ;返回
OFFTON:      MO V   KEYON，1
NOT FM
MOV SCON,99H
CMP   FM,OFFH
JZ       QDFM                                  ;是，转 QDFM
MOV KONG,10000010B                             ;修改控制字，南北向绿灯亮
JMP OUTPUT
QDFM:        MOV KONG,00101000B                ;修改控制字，东西向绿灯亮
JMP OUTPUT
STR:       MOV AL, SCONT
CMP     AL,OOH
JNZ     NEXT
CALL TURN
JMP OUTPUT
NEXT:     CALL STIME
CMP   AL,03H
JNZ     EXIT
;剩 3s，绿灯灭，黄灯亮处理
CMP   FM,OFFH
JZ       DXFM
MOV AL,00000l00B
AND     KONG, OFOH
JMP     FDON
DXFM:       MOV AL,01000000B
AND   KONG,   0FH
```

```
FDON:   OR     KONG, AL                    ;修改输出控制字
JMP   OUTPUT                               ;转控制字输出
DFM:     MOV SCON, ,30H                    ;修改定时时间。东西向通
30 s
MOV      KONG ,   00101000B                ;修改控制字，东西向绿灯
亮
OUTPUT:      MOV     AL, KNG               ；从 C 口输出控制字
MOV   DX ,   3C2H
OUT   DX, AL
EXIT:     MOV AL,SCONT
MOV AX,3COH
OUT   DX,AL
INC   DX
OUT   DX, AL
MOV AL,20H
OUT     20H . AL
POP   DS
POP     DX
POP   AX
IRET
TIME1S     ENDP
STIME:   MOV   AL, SCONT                   ;修改秒什数值子程序
SUB   AL, 1                                ;秒计数值减 1
DAS                                        ;保持为 BCD 数
MOV   SCONT , AL                           ;保存秒计数值
RET
TIME1S      ENDP
STIME:     MOV     AL,    SCONT
SUB   AL ,1
DAS
MOV     SCONT ,AL
RET
TURN     PROC                              ;东西向和南北向切换子程
序
NOT     FM                                 ;东西向和南北向转换
CMP   FM, 0FFH                             ;东西向?
JZ   NEXT                                  ;转东西向处理
MOV     SCONT ,20H                         ;修改定时时间，南北向通
20 s
MOV     KONG, 10000010B
RET
NEXT:   MOV     SCONG ,30H
MOV      KONG ,00101000B
RET
```

```
TURN    ENDP
ENDP    START
```

9.10 用 8255A 和 8254 编程，使扬声器发出 600 Hz 的可听频率，击任一键停止(假设提供的主时钟为 1.193 181 6 MHz)。

解:假定用 8ox86 系统板上的 8255A 和 8254 构成的发声系统编程，即用 8254 通道 2 拍出 600 Hz 的方波，8255A 的 PB0 作为 $GATE_2$ 门控信号，PB_1 控制 OUT_2 的输出能否送至扬声器。系统板上 8255A 的端口地址为 6d0H~63H.8254 的端口地址为 40H-43H.

8254 通道 2 计数初值为:1.988

```
·STACK
    • CODE
START:  MOV     AL,101110110B          ;通道 2 方式 3，二进制计数
        OUT     43H,AL                 :写通道 2 方式控制字
        MOV     AX.1988                ;取通道 2 计数初值
        OUT     42H,AL                 ;写低字节
          MOV     AL, AH
        OUT     42H,AL                 ;写高字节
        IN      AL,61H                 ;读 8255 的 PB 口输出值(引脚状态值)
        PUSH    AX
        OR      AL,A3H                 :PB0，PBl 置 1,其他位保持原值
        OUT     61H,AL
WA1T $: MOV     AH,0                    ;调键盘功能调用
        INT     16H
        JZ      WAIT $                 ;无按健，继续发声
        POP     AX                     ;恢复保存的 PB 口输出值
        OUT     61H,AL                 ;PB 口恢复原输出值
        MOV     AH,4CH                 ;返回 DOS
        INT     21H
        END     START
```

9.11 某 486 微机系统中用并行接口 8255A 来监视两个设备，每个设备有 8 位状态信息输出(假定输出信号与 TT L 电平兼容)。其中 $1^{\#}$ 设备各位物出“1"表示正常.“0”表示有故障，要调用计算机内的故障程序 STRUB 进行报警；其中 2#设备各位输出“0’代表正常，输出“1"则点亮一红色 LED 表示有故障。试设计该接口电路，并编写出满足要求的程序。

解:接口电路如图 9.12 所示。8255A 各端口地址为 0F4H--0F7H, A 组、B 组均工作在方式 0,其中 A 口用于输人 $1^{\#}$ 设备的 8 位状态.B 口用于抢人 2#设备的 8 位状态，PC_0 控制一红色 LED 显示是否有故障。相应的检测程序如下:

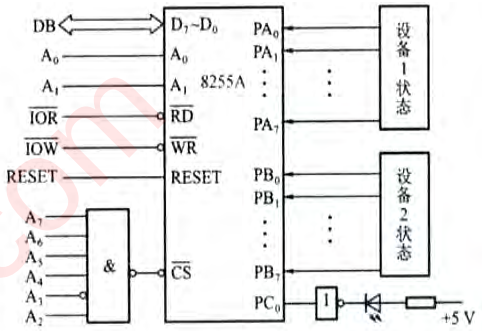


图 9.12 8255A 构成的设备监测电路

```
START:    MOV    AL, 10010010B                ;A 口、B 口方式 0 输人.C 口输出
OUT    0F7H ,AL
MOV   AL ,00000000B
OUT    0F7H   ,AL
REPEAT:    IN    AL ,0F40H                    ;输人 1#设备状态
CMP   AL, 0FFH                                ;设备有故障?
JZ   NEXT
CALL   STRUB
NEXT:    IN    AL,0F5H                       ;输人 2#设备状态
CMP    AL ,0H                                 :设备有故障?
JNZ    LEDON
MOV   AL, 00000000B
OUT    0F7H ,AL
JMP   REPEAT
LEDON:   MOV    AL ,00000001B
OUT    0F7H ,AL
JMP   REPEAT
LEDON:   MOV   AL,   00000001B
 OUT    0F7H, AL
JMP   REPEAT                                  ; 重复检侧
 HLT
```

9.12 试用两个 8255A 设计一个并行接口电路.把两个 CPU 为 80486 的 ISA 总线系统(一主一从)连接在一起。实现主机向从机的单向通信。假设用中断驱动式实现传送同步，试画出硬件连接图，并编写主机和从机的工作程序。

解:利用 8255A 实现主机向从机单向通信的硬件连接图如图 9.13 所示。

主、从机通过两个 8255A 接口芯片相连，8255A -- 1 工作在方式 1 输出，用于主机检出数据;8255A - 2 工作在方式 1 输人，用于从机输人数据。工作原理如下:

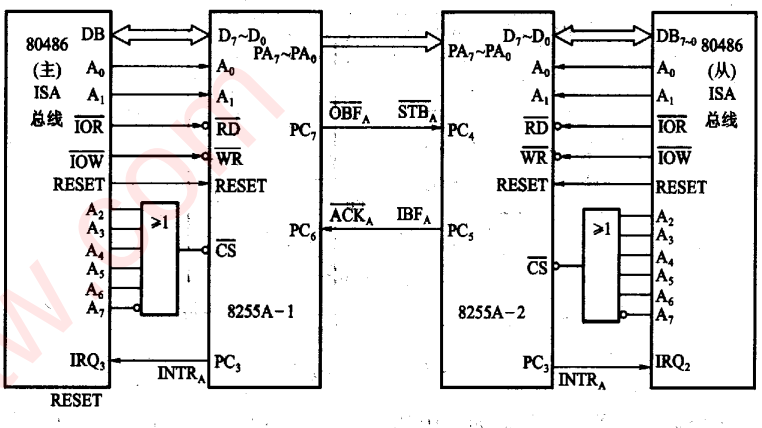


图 9.13　用 8255 实现的双机通信

(1)主机向 8255A -1 的 A 口写人一个数，8255A -1 的 $\overline{OBF_A}$ 变有效，表示主机的数据已锁存于 8255A-1 的 A 口，该信号用做 8255A - 2 的数据选通信号($\overline{STB_A}$)，使来自主机的数据锁存于 8255A - 2 的 A 端口，并使 IBF_A 变高，表示 8255A- 2 的愉人缓冲器满。同时，该 IBF_A 信号又用做对 8255A - 1 $\overline{OBF_A}$ 的应答信号，表示外设(8255A - 2)正在取数据。若 8255A 一 2 是中断允许的($INTE_A$ =1)，IBF_A 变高将使 8255A - 2 的 $INTR_A$ 变高向从 CPU 发出中断请求。

(2)从机通过响应中断，读取 8255A - 2A 口中的数据后，将清除 8255A - 2 的 IBFA，使 8255A -1 的 $\overline{ACK}$ 变有效，表示数据已取走，8255A 一 1 在接收到 $\overline{ACK}$ 信号后，将清除 $\overline{OBF_A}$，表示 8255A -1 A 口的输出缓冲器已空。若 8255A-1 是中断允许的($INTE_A$ = 1)，则在 $\overline{OBF_A}$ 和 $\overline{ACK}$ 变高后，将使 8255A -1 的 INTRA 变高向主机发出中断请求。

(3)主机通过响应中断开始输出下一数据，又重复上述过程。

据此原理，主机和从机的工作程序都应包括用于 8255A 和中断向量表的初始化程序、以及用于输出/输人的中断处理程序。下面分别给出此程序:-

(1)主机的工作程序

假定输出数据存放在 MESSG 开始的缓冲区中，数据块长度保存在 COUNT 单元中。则程序如下:

```
        DATA SEGMENT
        MESSG   DB     256 DUP( ? )        ;定义输出缓冲区
        COUNT DB?                          ;定义数据块长度
        MSGUT DW   0                       ;缓冲区当前指针，初始指向第 1 个字节
        IRQ2       EQU     0AH             ;中断请求级
    DATA   ENDS
CODE   SEGMENT
  ASSUME CS: CODE, DS: DATA
START:   MOV        AX, DATA
```

```
            MOV     DS,AX
              CLI
1NI8255:MOV     AL,10100000B
            OUT     83H,AL
            MOV     AL,00001101B
              OUT     83H,AL
…
STI
…
    IUTPUT: PUSH     Ax                    ;从机的中断输出程序
            PUSH     BX
            MOV    BX, MSGUT               ;取数据在缓冲区中的位里 MSGUT
            MOV     AL ,MESSG[BX]
            OUT     80H, AL               ;从 A 口输出数据
            INC     MSGUT
            MOV      Ax . COUNT
            CMP     Ax，MSGUT
            JNE     RETURN
              MOV     AL, 00001100B
              OUT     83H.AL
    RETURN:…
                    POP     BX
                    POP      AX
                    STI
                    IRET
    CODE          ENDS
                    END      START
```

(2)从机工作程序

假定输人数据存放在 BUF 开始的缓冲区中。COUNT 指示实际翰人的数据块长度。程序如下：

```
DATA      SEGMENT
  BUF     DB     256DUPE(?)                ;定义输人缓冲区
  COUNT     DB     0
    MSGUT     DW       0                  ;缓冲区当前指针，初始指向第一个字节
  IRQ2     EQU    0AH                     ;中断请求级
DATA    ENDS
CODE    SEGMENT
   .ASSUME    CS : CODE . DS: DATA
START:       MOV        AX .DATA
             MOV       DS, Ax
             CLI                          ;关中断
INI8255:MQV        :'AL,10110000B         ;A 口方式 1 输人
             OUT        83H,AL            ;写方式控制字
             MOV       AL,00001001B        :PC4 置位开放中断
```

```
                OUT       83H,AL
                STI
…                                          ;中断向量表和其他初始化
STI                                        ;开中断
…                                          ;其他工作
  INPUT:    PUSH      AX
                PUSH      Bx
                MoV     BX，MSGUT
                IN        AL.80H
                MOV      BUF[BX] , AL
                INC      MSGUT
                INC      COUNT
                CMP      AL,0DH
                JNE      RETURN
                  MOV       AL ,00001000B
OUT    83H ,AL
RETURN:….
                POP      BX
                POP      AX
                STI
                IRET
CODE              ENDS
END       START
```

9.13 一个异步串行发送器发送具有 8 位数据位的字符，在系统中使用 I 个奇偶校验位和 2 个停止位。若每秒发送 I00 个字符，则其波特率、位周期和传输效率各为多少?

解:由题惫可知，串行数据格式为:1 个起始位，8 位数据位，1 个奇偶校脸位和 2 个停止位。所以.宇符长度=1+8+1+2=12(位)。

波特率=每秒发送的字符数*字符长度=100*12=1200 bps

位周期=1/波特率=1/1 200= 0 . 833 rns

传输效率=字符中数据位数/字符长度=8/12=67%

9 .231NS 8250 中有多少个可访问的寄存器和多少个端口地址? 请写出它们的对应关系。从已学过的各种可编程接口芯片中，试总结出一般可编程接口芯片中是如何解决寄存器多、端口地址少的矛盾的。

解:INS 8250 中有 10 个 8 位寄存器，但只提供了 8 个端口地址。其对应关系见主教材表 9.4.在各种可编程接口芯片中，解决内部读/写寄存器多、端 A 地址线少的矛盾的常用方法有:

(1)利用命令字中的某一位(或某些位)的状态作为标志，在同一地址下。通过不同的状态寻址不同的内部寄存器。

(2)在同一端口地址下，利用读写的顺序来决定写人或读出不同的内部寄存器。

(3)一个只读寄存器和一个只写寄存器可以共用同一个地址，然后利用读命令(产生读信号)和写命令(产生写信号)来区分访问的是只读寄存器还是只写寄存器。

(4)用一个专门的指示器，在同一端口地址下，按指示器的不同状态，来访问不同的内部寄存器。

(5)利用某个命令字事先加以指定，而后寻址同一地址可访间不同的寄存器。

9.24 使用 8250 作串行接口时，若要求以 1200 的波特率发送一个字符。字符格式为:7

个数据位,1 个停止位，1 个奇校验位。试编写 8250 的初始化程序(设 8250 的基地址为 2FBH).初始化程序如下:

```
MOV     AL，10000000B
MOV     Dx , 2FBH
OUT      Dx . AL
MOV      AL, 60H
MOV      Dx.2F8H
OUT     Dx，AL
MOV      AL,   0
INC      Dx
OUT     Dx. AL
MOV     AL ,00001010B
MOV     DX . 2FBH
OUT    Dx,AL                          ;写 LCR，设置数据帧格式
 MOV    DX , 2FCH                    ;指向 MODEM 控制寄存器
 MOV    AL,00000011B                 ;数据终端就绪、请求发送
 OUT    DX. AL                       ;写 MODEM 控制字
```

9.25 使用 8250 芯片作异步申行数据传送接口，若传送的波特率为 2 400.则发送器(或接收器)的时钟频率为多少?

解: $f_{工作时钟}$ =波特率*16=2 400* 16 = 38 400 Hz = 38 . 4 kHz

9.26 某远程数据测量站，它的侧量数据以 300 , 500 . 1200 ,2 400 ,4 800 及 9 .600 波特中的一种串行输出、在什算机中用 8250 接收这个串行数据，试设计它的硬件连接和初始化程序。

解:8254 与系统总线的连接如图 9.14 所示。图中使用 MC1488 , MCI489 进行 TTL 电平与 EIA 电平间的转换，以满足 RS-232-C 接口标准。CS0 , CS1, . $\overline{ADS}$ 固定接有效电平，而 DISTR,DOSTR 固定接无效电平，这是常用的接法。

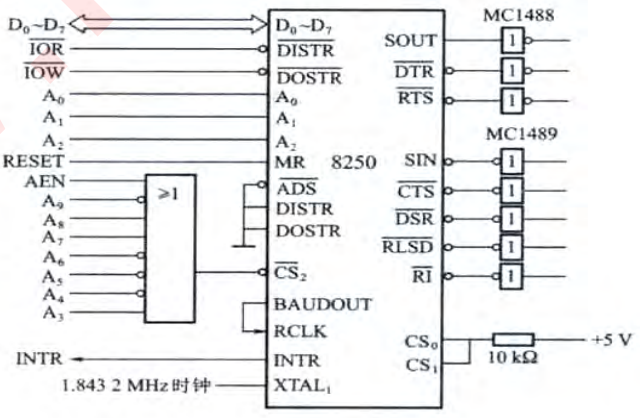


图 9.14　8250 与系统总线的连接

假定传输波特率为 9600 ,则初始化程序如下:

```
INI8250: MOV    DX,273H
         MOV    AL,80H
         OUT    Dx,AL
```

```
        MOV    DX, 270H
        MOV    AL,OCH
        OUT    DX,AL
        MOV    AL,OOH
        INC    DX
        OUT    Dx , AL                    ;写人除数寄存器高字节端口
        MOV    DX, 273H
        MOV    AL,00011110B               ;7 位数据位、偶校脸,2 位停止位
        OUT    Dx . AL                    ;写 LCR,设置数据核格式
        MOV    Dx , 274H                  ;指向 MODEM 控制寄存器
        MOV    AL, 00000011B              ;数据终端就绪、请求发送
        OUT    Dx.AL                      ;写 MODEM 控制字
        MOV    Dx,271H
        MOV    AL,43H                     ;允许发送保持器空、接收缓冲器满中断
        OUT      DX , AL                  ;写中断允许字
```

9.27 要用 8250A 以 300 波特的速率发送汉字编码信息，试为它编写合适的初始化程序(地址自定)。

解:发送汉字编码，数据位应为 8 位。假定 8250A 的基地址为 BASE.则初始化程序如下:

```
          BASE    EQU 270H
INI8250: MOV    Dx , BASE+3              ;设置线路控制寄存器地址
        MOV    AL.80H
        OUT          Dx.AL              ; DLAB= 1，允许访间除数寄存器
        MOV          DX,BASE
        MOV           AL.80H            产生 340 波特率的除数低字节
        OUT           DX,AL             ;写人除数寄存器低字节端口
        MOV           AL.01H            产生 300 波特率的除数高字节
        INC         Dx
        OUT           Dx,AL             ;写人除数寄存器高字节端口
        MOV          DX. BASE+3
        MOV           AL,00001111       ;8 位数据位、奇校验,2 位停止位
        OUT           Dx,.AL            ;写 LCR，设置数据帧格式
        MOV          Dx,13ASE + 4       ;指向 MODEM 控制寄存器
        MOV          AL , 00000011 B    ;数据终端就绪、请求发送
        OUT           DX,AL             ;写 MODEM 控制字
```

9.28 若某微机系统要求以} C}OU 的波特率进行异步通信，每字符的数据位 8 位。停止位 2 位，采用奇校验位，允许发送和接收中断，请设计其驱动程序(包括 8250 的初始化程序和通信中断服务程序)。

解:假定以如图 9.l4 所示的硬件连接编程，则初始化程序为:

```
            BASE      EQU      27OH
INI8250: MOV    DX. BASE + 3                ;设置线路控制寄存器地址
        MOV         AL,BAH
        OUT         DX,AL                  ; DLAB= 1。允许访问除数寄存器
        MOV         DX ,BASE
```

```
MOV      AL,OCH          ;里产生 9600 波特率的除数低字节
OUT      Dx,AL           ;写人除数寄存器低字节端口
MOV      AL, 0CH         ;置产生 9600 波特率的除数高字节
INC      DX
OUT      DX,AL           ;写人除数寄存器高字节端口
MOV      DX,BASE+3
MOV      AL,  00001111B  ;8 位数据位、奇校验,2 位停止位
OUT      DX,AL           ;写 LCR，设置数据顿格式
MOV      DX , BASE+4     ;指向 MOUEM 控制寄存器
MOV      AL ,00000011B   ;数据终端就绪、请求发送
OUT      DX , AL         ;写 MODEM 控制字
MOV      DX, BASE+1
MOV      AL. 03H         ;允许发送保持器空、接收缓冲器满中断
OUT      DX,AL           :写中断允许字
```

通信中断服务程序如下.其中 R_BUFFER 为接收缓冲区， R_CNT 为接收字符数.R_IH 为接收缓冲区当前指针；S_BU FFER 为发送缓冲区，S_CNT 为发送字符数，S_OUT 为发送级缓冲区当前指针。

```
INTERRUPT  PROC
           PUSH    Dx          ;保护现场
           PUSH    5I
           PUSH    Ax
           STI
AGA1N:     MOV     DX, BASE+2  :设置中断标识寄存器地址
           IN      AL,DX       ;读 IIR
           TEST    AL.01H      ;有未处理中断?
           JZ      RETURN      ;无，返回
           AND     AL, 06H
           CMP     AL,04H      ;接收中断?
           JZ      RECEIVE     ;转接收处理
           CMP     AL,02H      ;发送中断
           JZ      SENTI
           ERROR:  …
           JMP     AGAIN
           SEND:   MOV     DX,270H
           MOV     SI，S_OUT
           MOV     AL, S_BUFFER[SI]
           OUT     DX,  AL
           I NC    S_OUT
           DEC     S_ CNT
           JNZ     AGA 1N
           MOV     DX,BASE+1
           IN      AL, DX
           AND     AL , 00001101 B
           OUT     DX,AL
```

```
        JNZ     AGA1N
RECEIVE:    MOV     DX ,270
        MOV   SI ,R_IN
        IN      AL, DX
        MOV     R_BUFFER[SI] , AL
        INC     R_IN
        DEC     R_CNT
        JNZ     AGAIN
        MOV     DX ,BASE+1
        IN      AL, DX
        AND     AL, 00001110B
        OUT     DX,AL
        JNZ     AGAIN
RETURN:     CLI
        POP     AX
        POP     SI
        POP     DX
        STI
        IRET
INTRRUPT    ENDP
```

9.29 要在 PC 系列机上做一个自发自收的异步串行通信实验，要求通信波特率为 4800 ,每个字符由 1 位起始位、7 位数据位，1 位偶校验位,2 位停止位组成。实验前先要检查 0 号异步适配板的存在性。并填入板的基地址区。试完成对 8250 的初始化编程。

解:0 号适配器板的存在性可通过检测中断标识寄存器 IIR 的高 5 位是否为 0 来判断。假定 0 号板的墓地址区首地址为 RS232_ BASE。实现上述要求的程序段如下:

```
MOV         DX,3FAH                 ;0 号板的 IIR 的地址
IN          AL.DX                   :读此 IIR
TEST    AL，0F8H                                ;测试高 5 位
JNZ     EXIT                                    ;此板不存在，结束
MOV     RS232   EASE[ 0 ] .3F8H
MOV     AL, 80H
MOV   DX, 3FBH
OUT   DX,AL
MOV   AL,00H
MOV   AL, 00011110B
OUT   DX, AL
MOV   AL, 00H
INC   DX
OUT     DX, AL
MOV   DX ,3FBH
MOV     DX, 3FBH
OUT     DX, AL
MOV     AL, 00H
INC   DX
```

```
        OUT     DX, AL
        MOV    DX, 3FBH
        MOV    AL, 00011110B
        OUT     DX, AL
        MOV    DX, 3FCH
        MOV     AL, 00010011B                ;环路检侧、数据终端就绪、请求发送
        OUT    DX , AL                      ;写 MODEM 控制字
        MOV    DX, 3F9H
        MOV     AL, 03H                      ;允许发送保持器空、接收缓冲器清中断
        OUT    DX, AL                       ;写中断允许字
EXIT: ……
```