

实验一 汇编语言源程序的输入

一、实验目的

1. 通过实验了解和熟悉微机系统的配置。
2. 学习在 DEBUG 状态下输入汇编源程序的方法。
3. 初步掌握调试(在 DEBUG 状态下)的过程。

二、实验原理

1. 本实验要求在 DEBUG 状态下输入汇编源程序,并用 DEBUG 命令进行调试。用单步跟踪的方法验证指令的功能。

2. 以下是给定的参考程序,并在实验时在每条指令的“; ”符号右边按要求填写指令的执行结果。

注:(1)微机进入 DEBUG 状态下之后,一切立即数和地址数据均被默认为十六进制数,在输入时数的后面不加后缀“H”;

(2)在 DEBUG 状态下执行程序时,“INT 20H”指令可使系统执行完该指令前的程序时返回到“-”提示符状态,并且恢复 CS 和 IP 寄存器原来的值。

三、实验步骤

1. 开机后进入 DOS 系统,

C > DEBUG ✓ (✓回车符) - (为 DEBUG 提示符)

当显示器出现提示符“-”时,说明已进入 DEBUG 状态,这时,可用 DEBUG 命令进行操作。

2. 用 DEBUG 的 Register 命令检查所有寄存器内容,并作记录。命令格式:

R [寄存器名]

该命令的功能是显示寄存器的内容,或修改某一指定寄存器内容,若[寄存器名]缺省,则显示所有寄存器内容。例如:

-R

3. 用 DEBUG 的 Assemble 命令输入汇编源程序。格式:

A [内存地址]

注:用“[]”符号括起来的部分表示可以省略。

该命令的功能是从指定的内存地址开始(括号不要输入)逐条输入汇编语言源程序并汇编成机器码存入内存。若地址缺省,则接上一个 A 命令最后一条指令之后输入汇编语句,

若没有用过 A 命令，则从 CS: 0100H 地址开始输入。例如：

—A

OCD3: 0100—

在输入 A 命令之后，或每输入一条指令之后，显示器的左端给出了内存的段地址和偏移地址。

每条指令均用回车(↵)结束。若输入的指令有语法错误，DEBUG 拒绝接收，并给出提示，此时可以重新输入。程序的最后一条指令输入完之后，再按一次回车键(↵)，即可结束汇编命令，回到 DEBUG 提示符“—”状态。

4. 用 DEBUG 的 Unassemble 命令反汇编。命令格式：

U [起始地址[终止地址]]

该命令的功能是从起始地址到终止地址反汇编目标码，缺省值是接上一个 U 命令或从 CS: 0100H 地址开始。例如：

—U

显示器上将显示程序的内存地址、指令机器码的汇编源程序三列对照清单。

5. 用 DEBUG 的 Trace 命令单步跟踪程序。命令格式：

T [=起始地址] [指令条数]

该命令的功能是从指定的起始地址开始逐条执行指令，每执行完一条指令，屏幕显示所有寄存器内容和下一条指令地址和指令。若[=起始地址]缺省，则 T 命令从 CS: IP 地址开始执行指令。

例如：

—T↵

重复这一过程，即可看到每条指令执行后，所有寄存器和标志寄存器的标志位内容。此时，要检查内存单元的数据，可用 DEBUG 的 D 命令。

6. 用 DEBUG 的 Dump 命令显示存储器单元的内容。命令格式：

D[起始地址[终止地址]]

该命令的功能是从起始地址到终止地址，连续显示存储器单元的内容。若地址缺省，则接上一个 D 命令或从 DS: 0100H 地址开始显示。例如：

—D↵

四、参考程序和实验结果

| | | | |
|------|-------------------|------------|------------------------------------|
| MOV | AX, 2000 | | ; AL=00H |
| MOV | DS, AX | | ; DS=2000H |
| NOT | AX | | ; AX=DFFFH |
| XOR | AX, AX | | ; AX=0000H |
| DEC | AX | | ; AX=FFFFH |
| INC | AX | | ; AX=0000H |
| MOV | BX, 2030 | | ; BH=20H |
| MOV | SI, BX | | ; SI=2030H |
| MOV | [SI], BL | | ; [2030H]=30H |
| MOV | WORD PTR[SI], 10F | | ; [2030H]= 0FH [2031H]=01H |
| MOV | DI, SI | | ; DI=2030H |
| MOV | [DI+50], BH | | ; [DI+50H]=20H |
| MOV | BH, [SI] | | ; BH=0F |
| MOV | BL, [DI+50] | | ; BL=20F |
| MOV | SP, 5000 | | |
| PUSH | AX | ; AX=0000H | [SS: 4FFEh]=00H [SS: 4FFFh]=00H |
| PUSH | BX | ; BX=0F20H | [SS: 4FFCh]=20H [SS: 4FFDh]=0FH |
| POP | AX | | ; AX=0F20H |
| POPF | | | ; F=00000000 |
| NEG | BX | | ; BX=F0E0H |
| XCHG | BX, AX | | ; BX=0F20H |
| STD | | | ; F=01010001 |
| STI | | | ; F=01110001 |
| CLD | | | ; F=00110001 |
| CLI | | | ; F=00010001 |
| ADC | DI, 2050 | | ; DI=4081H F=00000010 |
| ADC | SP, DI | | ; SP=9081H F=10010010 |
| ADC | AX, 1500 | | ; AX=05E0H F= 00000001 |
| SUB | AX, BX | | ; AX=F6C0H BX=0F20H |

| | | | |
|-----|----------|-------------|----------|
| SHL | AH, 1 | ; AH=ECH | |
| RCL | AX, 1 | ; AX=D981H | |
| SHR | BH, 1 | ; BH=07H | |
| RCR | BL, 1 | ; BL=90H | |
| MOV | CL, 4 | | |
| MOV | DX, 80F0 | | |
| ROL | DX, CL | ; DX= 0F08H | CL=04H |
| INT | 20 | ; CS= 00A7H | IP=1072H |

实验二 数据的建立与传送程序

一、实验目的

1. 继续学习 DEBUG 命令。
2. 验证指令的功能。

二、实验内容

在 DEBUG 状态下, 分别输入下面各程序段, 每输入完一个程序段, 用 G 命令进行连续方式执行程序, 在连续执行时, 要记录程序的执行结果。

1. 在内存 10000H 单元开始, 建立 00H~0FH~00H 31 个数, 要求 00H~0FH 数据逐渐增大, 0FH~00H 逐渐减小。该程序从内存 CS:0100H 地址开始输入。

(1) 源程序:

```
MOV     AX, 1000H
MOV     DS, AX
MOV     SI, 0
MOV     CL, 0FH
XOR     AX, AX
PPE1:   MOV     [SI], AL
        INC     SI
        INC     AL
        DEC     CL
        JNZ     PPE1
        MOV     CX, 10H
PPE2:   MOV     [SI], AL
        INC     SI
        DEC     AL
        LOOP    PPE2
        INT     20H
```

注: 转移指令的符号地址直接用绝对偏移地址, 该地址在用 A 命令汇编输入时, 可以看到程序全部运行完之后, 可用 DEBUG 的 Dump 命令查看建立的数据块内容。例如:

-D1000: 00 1E

(2) 执行结果:

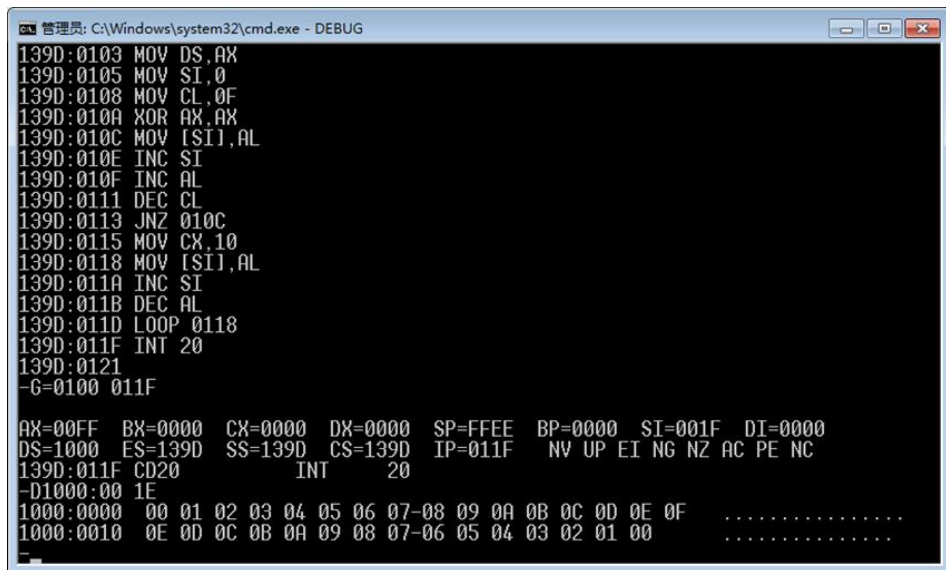


图 2-1

2. 把上一个程序的执行结果(建立的 31 个字节数据块,其首地址在 10000H),分几种方式传送到以下指定的区域。

(a) 该程序从内存 CS:0150H 开始输入。把数据块传送到 15050H 开始的存贮区域中。

检查内存数据块的传送情况,可用“D”命令。

(1) 源程序:

```
MOV      AX, 1000H
MOV      DS, AX
MOV      SI, 0
MOV      DI, 5050H
MOV      CX, 1FH          ; 数据块长度是 31
PPEA:    MOV     AL, [SI]
MOV      [DI], AL
INC      SI
INC      DI
LOOP     PPEA
```

(2) 执行结果:



(b)用串传送指令 MOVSB,把数据块传送到 15150H 开始的区域,该程序从内存 CS:0200H 开始输入。

检查程序最后的执行结果，可用“D”命令，例如：-D1000:5150✓

(1) 源程序:

```
MOV     AX, 1000H
```

MOV DS, AX

MOV ES, AX

MOV SI, 0

```
MOV     DI, 5150H
```

```
MOV      CX, 1FH
```

CLD

```
PPEA:  MOV  SB
```

INC DI

LOOP PPEA

INT 20H

(2) 执行结果:

```

-A 0200
139D:0200 MOV AX,1000
139D:0203 MOV DS,AX
139D:0205 MOV ES,AX
139D:0207 MOV SI,0
139D:020A MOV DI,5150
139D:020D MOV CX,1F
139D:0210 CLD
139D:0211 MOVSB
139D:0212 LOOP 0211
139D:0214 INT 20
139D:0216
-G=0200 0214

AX=1000 BX=0000 CX=0000 DX=0000 SP=FFEE BP=0000 SI=001F DI=516F
DS=1000 ES=1000 SS=139D CS=139D IP=0214 NV UP EI PL NZ NA PE NC
139D:0214 CD20          INT      20

```

图 2-3

(c)用重复串操作指令“REP MOVSB”把数据块传送到 15250H 开始的区域。该程序从 CS: 250H 地址开始输入。

检查程序的最后执行结果时，可用：—D1000: 5250H

(1) 源程序：

```

MOV     AX, 1000H

MOV     DS, AX

MOV     ES, AX

MOV     SI, 0

MOV     DI, 5250H

MOV     CX, 1FH

CLD

REPZ

MOVSB

INT     20H

```

(2) 执行结果：


```

-A 250
139D:0250 MOV AX,1000
139D:0253 MOV DS,AX
139D:0255 MOV ES,AX
139D:0257 MOV SI,0
139D:025A MOV DI,5250
139D:025D MOV CX,1F
139D:0260 CLD
139D:0261 REP MOVSB
139D:0263 INT 20
139D:0265
-G=0250 0263

AX=1000 BX=0000 CX=0000 DX=0000 SP=FFEE BP=0000 SI=001F DI=526F
DS=1000 ES=1000 SS=139D CS=139D IP=0263 NV UP EI PL NZ NA PE NC
139D:0263 CD20          INT    20

```

图 2-4

(d) 用串操作的减量工作方式，把数据块传送到 25050H 开始的区域。该程序从 CS:0300H 开始输入。

(1) 源程序：

```

MOV     AX, 1000H

MOV     DS, AX

MOV     AX, AX

MOV     ES, AX

MOV     SI, 1E

MOV     DI, 506EH

MOV     CX, 1FH

STD

REP

REP     MOVSB

INT     20H

```

检查程序的最后执行结果，用 D 命令：-D2000: 5050 ✓

(2) 执行结果：

```

-A 0300
139D:0300 MOV AX,1000
139D:0303 MOV DS,AX
139D:0305 ADD AX,AX
139D:0307 MOV ES,AX
139D:0309 MOV SI,1E
139D:030C MOV DI,506E
139D:030F MOV CX,1F
139D:0312 STD
139D:0313 REP MOVSB
139D:0315 INT 20
139D:0317
-G=0300 0315

AX=2000 BX=0000 CX=0000 DX=0000 SP=FFEE BP=0000 SI=FFFF DI=504F
DS=1000 ES=2000 SS=139D CS=139D IP=0315 NV DN EI PL NZ NA PE NC
139D:0315 CD20          INT      20

```

图 2-5

```

-D2000:5050
2000:5050 00 01 02 03 04 05 06 07-08 09 0A 0B 0C 0D 0E 0F .....
2000:5060 0E 0D 0C 0B 0A 09 08 07-06 05 04 03 02 01 00 00 .....
2000:5070 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
2000:5080 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
2000:5090 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
2000:50A0 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
2000:50B0 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
2000:50C0 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....

```

图 2-6

实验三 分支程序设计

一、实验目的

- 1.练习分支程序的编写方法。
- 2.练习汇编语言程序的上机过程。

二、实验原理

- 1.通过分支程序设计调试和运行，进一步熟悉掌握汇编程序执行的软件环境。
- 2.通过分支程序的执行过程，熟悉 EDIT 的使用，建立 OBJ 文件 EXE 文件的方法。

三、实验内容

1. 将一个字符串变量 string 中的小写字母转换成大写字母并显示出来。
2. 给出三个有符号数，编写一个比较相等关系的程序：
 - (1) 如果这三个数都不相等，则显示 0；
 - (2) 如果这三个数中有两个数相等，则显示 1；
 - (3) 如果这三个数都相等，则显示 2；

四：实验结果

1. 将一个字符串变量 string 中的小写字母转换成大写字母并显示出来。

(1) 源程序：

```
DATAS SEGMENT
    STRS DB 'hello, my name is li meng hao', 10, '$'
    LEN EQU $ - STRS
DATAS ENDS

STACKS SEGMENT
    DB 100 DUP(0)
STACKS ENDS

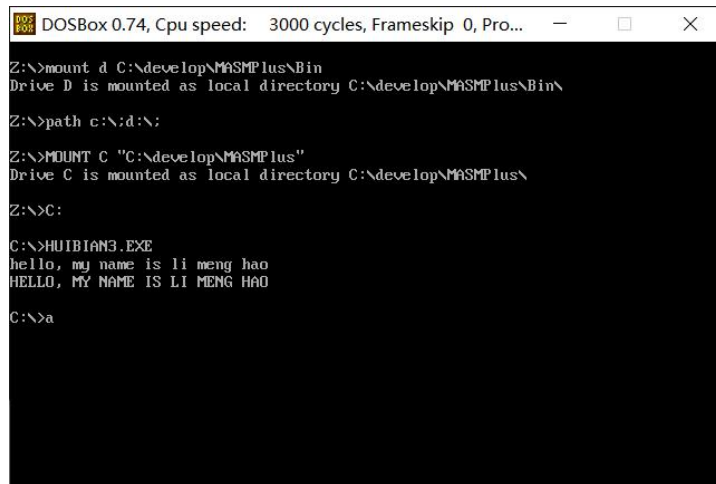
CODES SEGMENT
    ASSUME CS:CODES, DS:DATAS, SS:STACKS
START:
    MOV AX, DATAS
    MOV DS, AX
```

```

        LEA DX, STRS
        MOV AH, 9
        INT 21H
        MOV CX, LEN
        DEC CX
        LEA SI, STRS
LP1:
        MOV BL, [SI]
        CMP BL, 7AH
        JNS J1
        CMP BL, 61H
        JS J1
        SUB BL, 20H
        MOV [SI], BL
J1:
        INC SI
        LOOP LP1
        INT 21H
        MOV AH, 4CH
        INT 21H
CODES ENDS
END START

```

(2) 运行结果:



```
DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Pro...
Z:\>mount d C:\develop\MASMPlus\Bin
Drive D is mounted as local directory C:\develop\MASMPlus\Bin\
Z:\>path c:\d:\:
Z:\>MOUNT C "C:\develop\MASMPlus"
Drive C is mounted as local directory C:\develop\MASMPlus\
Z:\>C:
C:\>HUIBIAN3.EXE
hello, my name is li meng hao
HELLO, MY NAME IS LI MENG HAO
C:\>a
```

图 3-1 实验 3.1 结果

2. 给出三个有符号数，编写一个比较相等关系的程序：

- ①如果这三个数都不相等，则显示 0；
- ②如果这三个数中有两个数相等，则显示 1；
- ③如果这三个数都相等，则显示 2；

(1) 源程序：

```
DATA SEGMENT
DATA ENDS
STACKS SEGMENT STACK
    DB 100 DUP(0)
STACKS ENDS
CODE SEGMENT 'CODE'
    ASSUME CS :CODE, DS :DATA, SS:STACKS
START:
    MOV AX, DATA
    MOV DS, AX
    MOV AH, 01H
    INT 21H
    MOV BH, AL
```

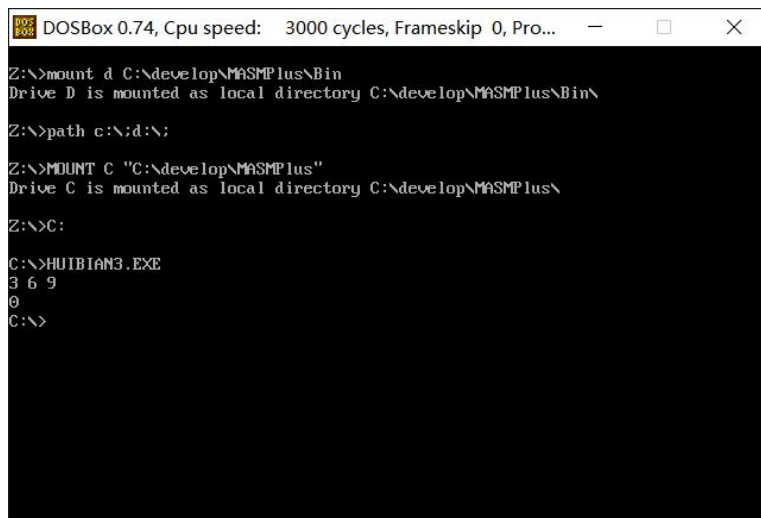
```
MOV DL, 20H
MOV AH, 02H
INT 21H
MOV AH, 01H
INT 21H
MOV BL, AL
MOV DL, 20H
MOV AH, 02H
INT 21H
MOV AH, 01H
INT 21H
MOV DH, AL
MOV DL, 0AH
MOV AH, 02H
INT 21H
MOV DL, 30H
CMP BH, BL
JNZ NEXT1
INC DL
NEXT1:
    CMP BH, DH
    JNZ NEXT2
    INC DL
NEXT2:
    CMP BL, DH
    JNZ NEXT3
    INC DL
NEXT3:
    CMP DL, 33H
```

```
JB NEXT4
MOV DL, 32H
NEXT4:
MOV AH, 02H
INT 21H

MOV AX, 4C00H
INT 21H
CODE ENDS
END START
```

(2) 运行结果

1、三个数不相等，显示 0:



```
DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Pro...
Z:\>mount d C:\develop\MASMPPlus\Bin
Drive D is mounted as local directory C:\develop\MASMPPlus\Bin\
Z:\>path c:\d:\;
Z:\>MOUNT C "C:\develop\MASMPPlus"
Drive C is mounted as local directory C:\develop\MASMPPlus\
Z:\>C:
C:\>HUIBIAN3.EXE
3 6 9
0
C:\>
```

图 3-2 实验 3.2 结果

2、三个数中有两个相等，显示 1:

```
DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Pro...
Z:\>mount d C:\develop\MASMPPlus\Bin
Drive D is mounted as local directory C:\develop\MASMPPlus\Bin\

Z:\>path c:\d:\:

Z:\>MOUNT C "C:\develop\MASMPPlus\"
Drive C is mounted as local directory C:\develop\MASMPPlus\

Z:\>C:

C:\>HUIBIAN3.EXE
9 8 9
1
C:\>
```

图 3-3 实验 3.2 结果

3、三个数中三个相等，显示 2:

```
DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Pro...
Z:\>mount d C:\develop\MASMPPlus\Bin
Drive D is mounted as local directory C:\develop\MASMPPlus\Bin\

Z:\>path c:\d:\:

Z:\>MOUNT C "C:\develop\MASMPPlus\"
Drive C is mounted as local directory C:\develop\MASMPPlus\

Z:\>C:

C:\>HUIBIAN3.EXE
3 3 3
2
C:\>_
```

图 3-3 实验 3.2 结果

实验四 统计学生成绩程序

一、实验目的

进一步掌握分支程序和循环程序的编写方法。

二、实验内容

设有 10 个学生的成绩分别为 56、69、84、82、73、88、99、63、100 和 80 分。试编制程序分别统计低于 60 分、60~69 分、70~79 分、80~89 分、90~99 分及 100 分的人数存放到 s5、s6、s7、s8、s9 及 s10 单元中。

这一题目的算法很简单，成绩分等部分采用分支结构，统计所有成绩则用循环结构完成。程序框图如下图所示。

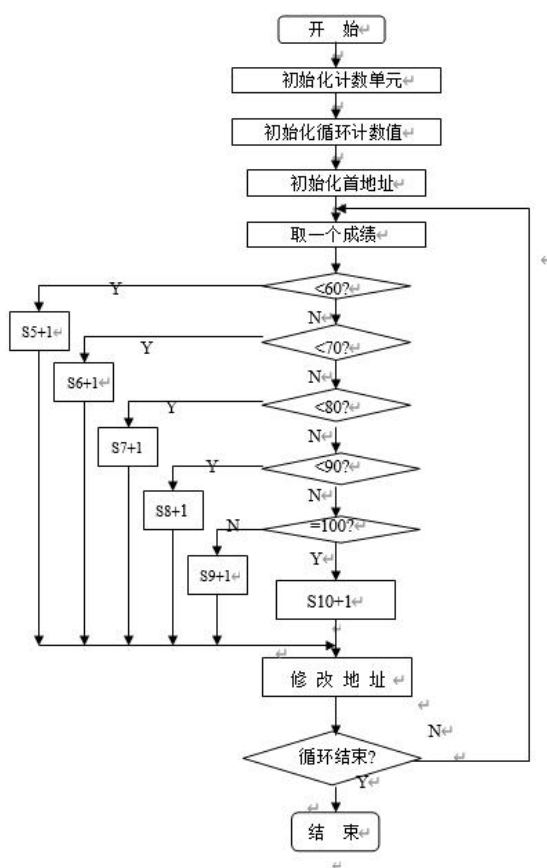


图 4-1 程序框图

三、实验结果

(1) 源程序

DATAS SEGMENT

GRADE DW 56, 69, 84, 82, 73, 88, 99, 63, 100, 80

S5 DW 0

S6 DW 0

S7 DW 0

S8 DW 0

S9 DW 0

S10 DW 0

S11 DB '\$'

DATAS ENDS

STACKS SEGMENT

STACKS ENDS

CODES SEGMENT

ASSUME CS:CODES, DS:DATAS, SS:STACKS

START:

MOV AX, DATAS

MOV DS, AX

MOV S5, '0'

MOV S6, '0'

MOV S7, '0'

MOV S8, '0'

MOV S9, '0'

MOV S10, '0'

MOV CX, 10

MOV BX, OFFSET GRADE

COMPARE: MOV AX, [BX]

CMP AX, 60

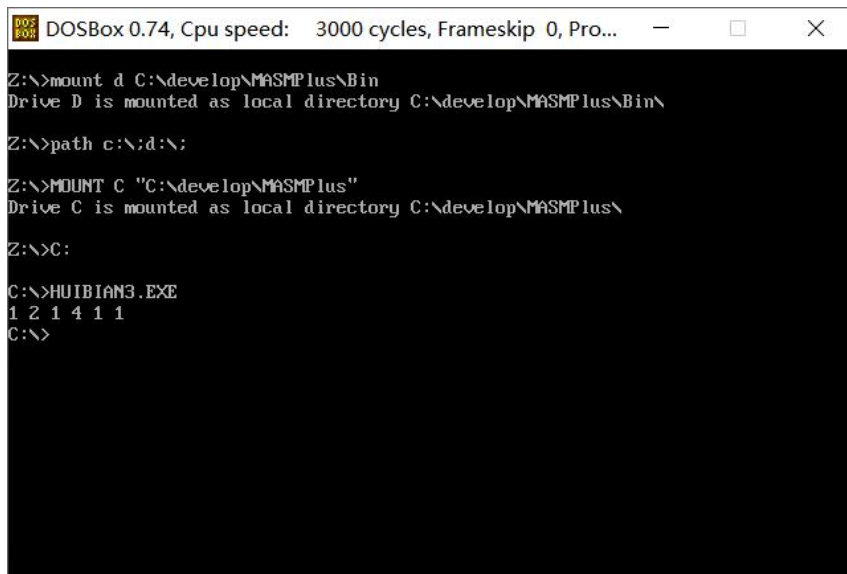
```

    JL FIVE
    CMP AX, 70
    JL SIX
    CMP AX, 80
    JL SEVEN
    CMP AX, 90
    JL EIGHT
    CMP AX, 100
    JNE NINE
    INC S10
    JMP SHORT CHANGE_ADDR
NINE: INC S9
    JMP SHORT CHANGE_ADDR
EIGHT: INC S8
    JMP SHORT CHANGE_ADDR
SEVEN: INC S7
    JMP SHORT CHANGE_ADDR
SIX: INC S6
    JMP SHORT CHANGE_ADDR
FIVE: INC S5
CHANGE_ADDR:
    ADD BX, 2
    LOOP COMPARE
    mov dx, offset S5
    mov ah, 9h
    int 21h
    MOV AH, 4CH
    INT 21H
CODES ENDS

```

END START

(2) 运行结果



```
DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Pro...  
Z:\>mount d C:\develop\MASMPPlus\Bin  
Drive D is mounted as local directory C:\develop\MASMPPlus\Bin\  
Z:\>path c:\d:\;  
Z:\>MOUNT C "C:\develop\MASMPPlus"  
Drive C is mounted as local directory C:\develop\MASMPPlus\  
Z:\>C:  
C:\>HUIBIAN3.EXE  
1 2 1 4 1 1  
C:\>
```

图 4-2 实验 4 结果

实验六 8254 定时/计数器应用实验

一、实验目的

1. 掌握 8254 的工作方式及应用编程；
2. 掌握 8254 的典型应用电路接法；
3. 学习 8254 在 PC 系统中的典型应用方法；

二、实验原理

8254 是可编程间隔定时器。为 8253 的改进型，比 8253 具有更优良的性能。8254-2 具有一些基本功能：

- (1) 有三个独立的 16 位计数器；
- (2) 每个计数器可按二进制或十进制计数；
- (3) 每个计数器可编程工作于 6 种不同工作方式；
- (4) 8254-2 每个计数器允许的最高计数频率为 10MHz；
- (5) 8254 有读回命令，除了可以读出当前计数单元的内容外，还可以读出状态寄存器的内容；
- (6) 计数脉冲可以是有规律的时钟信号，也可以是随机信号。计数初值公式如下，其中 f_{CLKi} 是输入时钟脉冲的频率， f_{OUTi} 是输出脉冲的频率。

$$n = f_{CLKi} / f_{OUTi}$$

图 6-1 是 8254 的内部结构图，它是由与 CPU 的接口、内部控制电路和三个计数器组成。8254 的工作方式如下：

- (1) 方式 0：计数到 0 结束输出正跃变信号方式。
- (2) 方式 1：硬件可重触发单稳方式。
- (3) 方式 2：频率发生器方式。
- (4) 方式 3：方波发生器。
- (5) 方式 4：软件触发选通方式。
- (6) 方式 5：硬件触发选通方式。

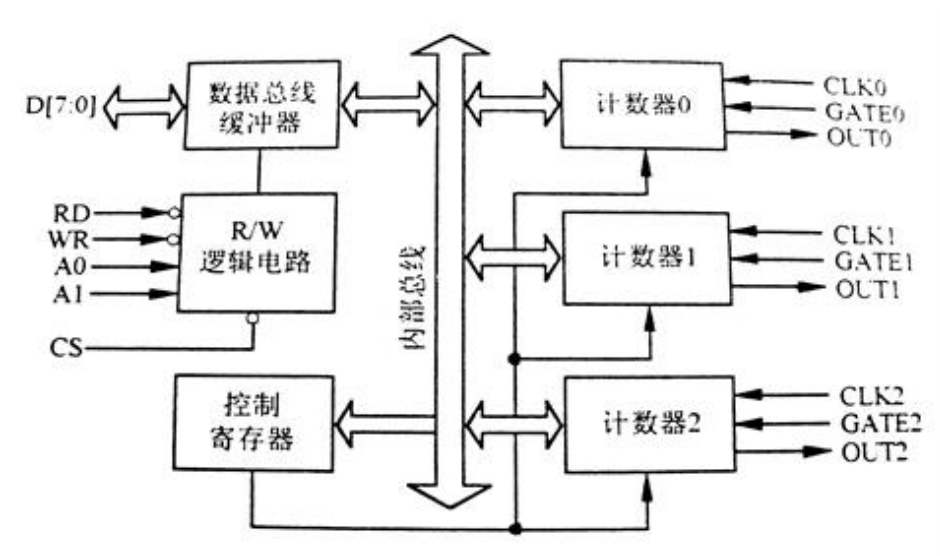


图 6-1

8254 的控制字有两个：一个用来设置计数器的工作方式，称为方式控制字；另一个用来设置读回命令，称为读回控制字。这两个控制字共用一个地址，由标识位来区分。控制字格式如下表所示。

8254 的方式控制字格式

| D7↔ | D6↔ | D5↔ | D4↔ | D3↔ | D2↔ | D1↔ | D0↔ |
|-------------|-----|---------------------|-----|-----------|-----|---------|-----|
| 计数器选择↔ | | 读写格式选择↔ | | 工作方式选择↔ | | 计数码制选择↔ | |
| 00- 计数器 0↔ | | 00-锁存计数值↔ | | 000-方式 0↔ | | 0-二进制数↔ | |
| 01- 计数器 1↔ | | 01-读/写低八位↔ | | 001-方式 1↔ | | 1-十进制数↔ | |
| 10- 计数器 2↔ | | 10-读/写高八位↔ | | 010-方式 2↔ | | | |
| 11-读出控制字标志↔ | | 11-先读/写低八位、再读/写高八位↔ | | 011-方式 3↔ | | | |
| | | | | 100-方式 4↔ | | | |
| | | | | 101-方式 5↔ | | | |

8254 读出控制字格式

| | | | | | | | |
|-----|-----|----------|-----------|----------------|-----|-----|-----|
| D7↵ | D6↵ | D5↵ | D4↵ | D3↵ | D2↵ | D1↵ | D0↵ |
| 1↵ | 1↵ | 0-锁存计数值↵ | 0-锁存状态信息↵ | 计数器选择（同方式控制字）↵ | | | 0↵ |

8254 状态字格式

| | | | | | | | |
|-----------------------------|-----------------------------|---------------------|-----|-----|-----|-----|-----|
| D7↵ | D6↵ | D5↵ | D4↵ | D3↵ | D2↵ | D1↵ | D0↵ |
| OUT 引脚现行状态↵ 1-高电平 0-低电平↵ | 计数初值是否装入↵ 1-无效计数 0-计数有效↵ | ↵ 计数器方式（同方式控制字）↵ | | | | | |

三、实验设备

微机一台、TD-PITE 实验装置一套。

四、实验内容

计数应用实验。编写程序应用 8254 的计数功能，将 8254 的计数器 0 设置为方式 0，计数值为十进制 4，用单次脉冲 KK1+作为 CLK0 时钟，OUT0 连接 MIR7，,每当按动 KK1+按动五次后，产生一次计数中断，并在屏幕上显示一个字符“M”。

五、实验步骤

- （1）按照实验线路图 6-2 连接实验电路
- （2）编写实验程序，经编译、链接无误后装入系统。
- （3）运行程序，按动 KK1+产生单次脉冲，观察实验现象
- （4）改变计数初值，验证 8254 的计数功能

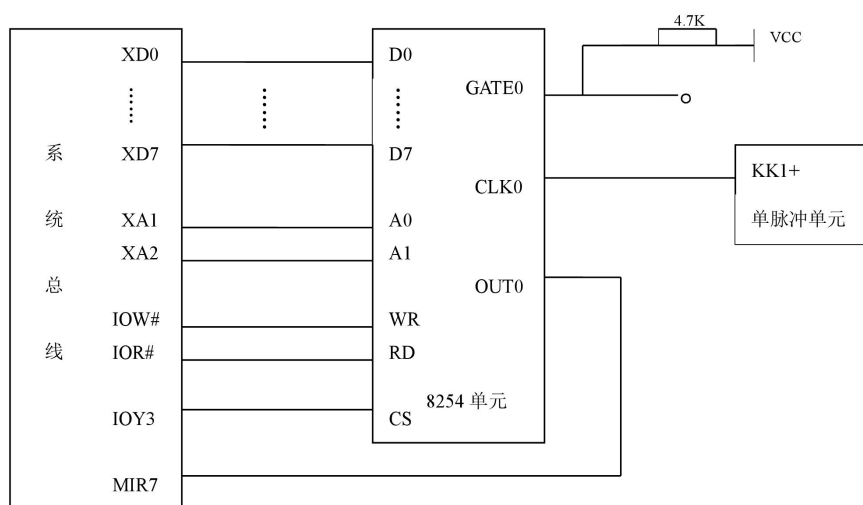


图 6-2 实验接线图

六、源程序和结果

(1) 源程序

```

A8254 EQU 06C0H
B8254 EQU 06C2H
C8254 EQU 06C4H
CON8254 EQU 06C6H
SSTACK SEGMENT
STACK DW 32 DUP(?)
SSTACK ENDS
CODE SEGMENT
ASSUME CS:CODE, SS:SSTACK
START: PUSH DS
MOV AX, 0000H
MOV DS, AX
MOV AX, OFFSET IRQ7 ;取中断入口地址
MOV SI, 003CH ;中断矢量地址
MOV [SI], AX ;填 IRQ7 的偏移矢量

```



```

MOV  AX, CS                ;段地址
MOV  SI, 003EH
MOV  [SI], AX              ;填 IRQ7 的段地址矢量
CLI
POP  DS
;初始化主片 8259
MOV  AL, 11H
OUT  20H, AL               ;ICW1
MOV  AL, 08H
OUT  21H, AL               ;ICW2
MOV  AL, 04H
OUT  21H, AL               ;ICW3
MOV  AL, 01H
OUT  21H, AL               ;ICW4
MOV  AL, 6FH               ;OCW1
OUT  21H, AL
;8254
MOV  DX, CON8254
MOV  AL, 10H               ;8254 控制字
OUT  DX, AL
MOV  DX, A8254
MOV  AL, 04H
OUT  DX, AL
STI
AA1: JMP  AA1
IRQ7: MOV  DX, A8254
MOV  AL, 04H
OUT  DX, AL
MOV  AX, 0141H             ;显示字符

```

```
INT 10H
MOV AX, 0120H
INT 10H
MOV AL, 20H
OUT 20H, AL ;中断结束命令
IRET
CODE ENDS
END START
```

(2) 运行结果

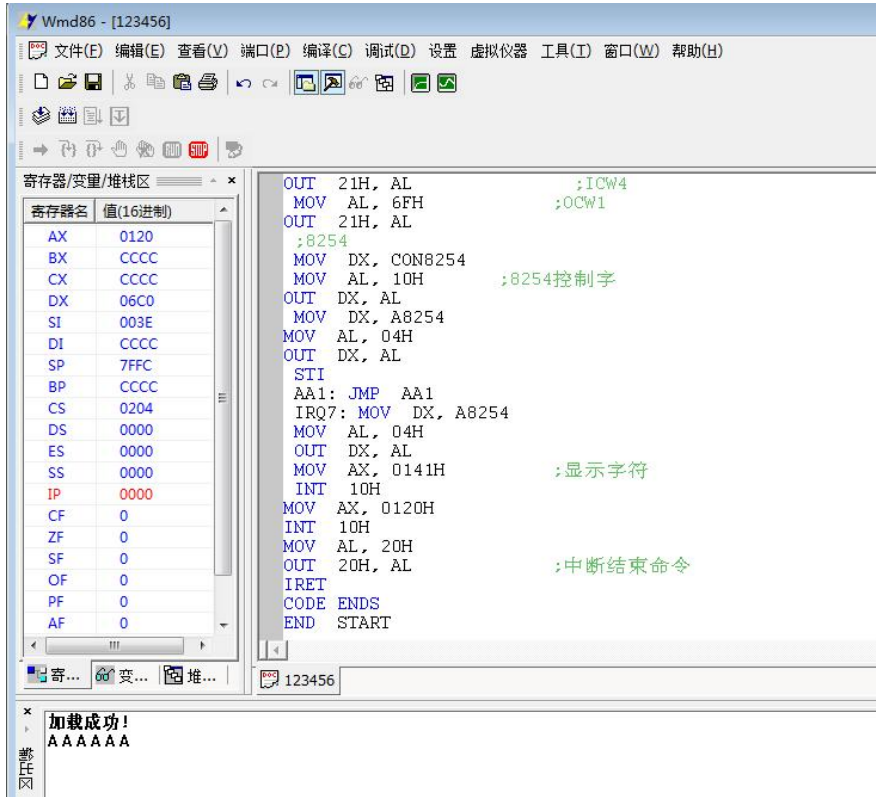


图 6-3 实验 6 结果

实验七 八 8255 并行接口

一、实验目的

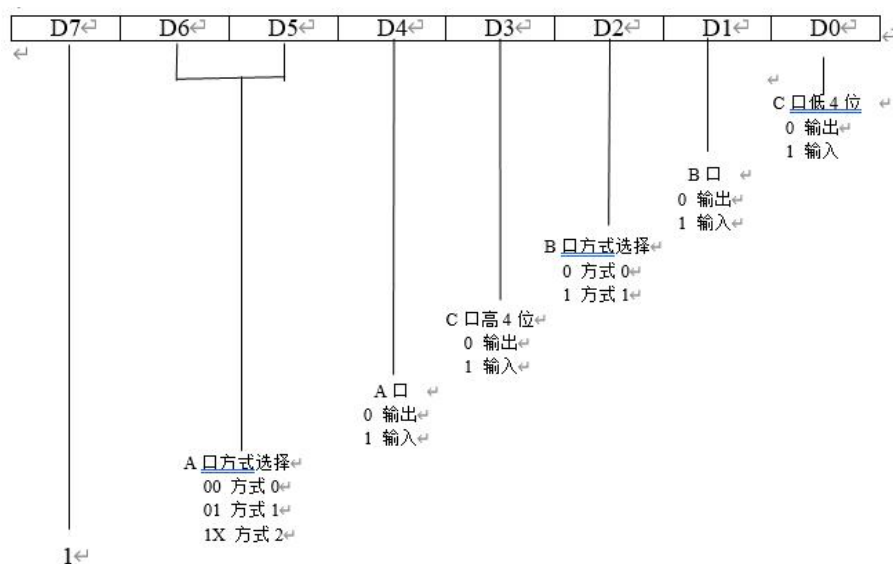
- 1、掌握 8255 的工作方式及应用编程。
- 2、掌握 8255 的典型应用电路解法。

二、实验原理

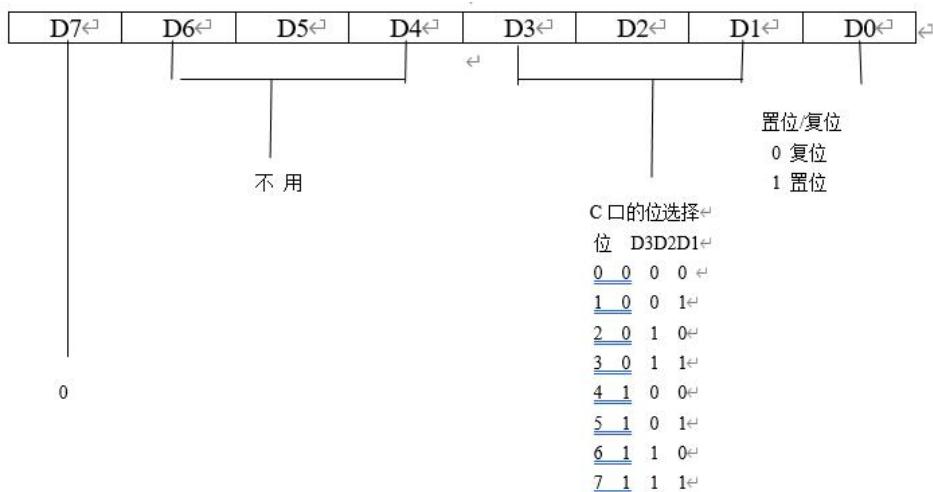
并行接口是以数据的字节为单位与 I/O 设备或被控制对象之间传递信息。CPU 与接口之间的数据传送总是并行的，即可以同时传递 8 位、16 位、32 位等。

8255 可编程外围接口芯片是通用并行接口芯片，它具有 A、B、C 三个并行接口，用+5V 单电源供电，能在以下三种方式下工作：

方式 0——基本输入/输出方式、方式 1——选通输入/输出方式、方式 2——双向选通工作方式，8255 工作方式控制字和 C 口按位置位/复位控制字格式如图所示。



(a) 工作方式控制字



(b) C口按位置位/复位控制字

三、实验设备

PC 微机一台，TD-PITE 实验装置一套。

四、实验内容

1、基本输入输出实验。编写程序，使 8255 的 A 口为输入口，端口 B 作为输出口。完成拨动开关到数据灯的数据传输。要求只要开关拨动，数据灯的显示就改变。通过对 8255 芯片编程来实现输入输出功能。

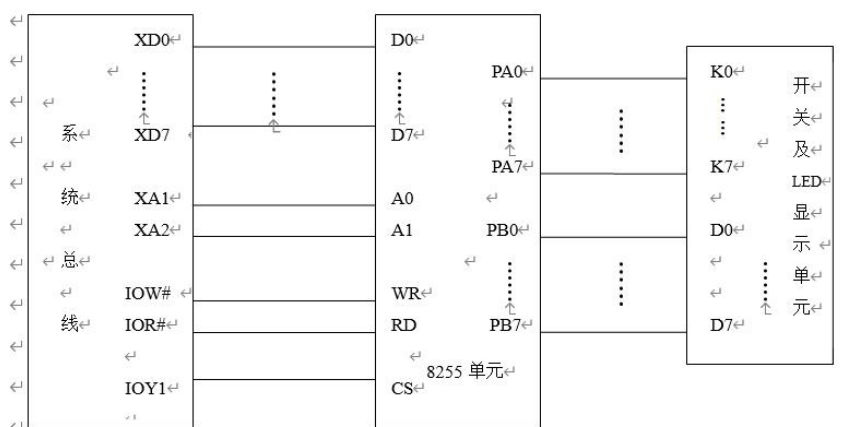
2、流水灯显示实验。编写程序，使 8255 的 A 口和 B 口均为输出，数据灯 D7~D0 由左向右，每次仅亮一个灯，循环显示，D15~D8 与 D7~D0 正相反，由右向左，每次仅点亮一个灯，循环显示。

五、实验步骤

1、基本输入输出实验。

本实验使 8255 端口 A 工作在方式 0 并作为输入口，端口 B 工作在方式 0 并作为输出口。用一组开关信号接入端口 A，端口 B 输出线接至一组数据灯上，然后通过对 8255 芯片编程来实现输入输出功能，扩展 I/O 接口信号线 IOY1 编址空间 0640H~067FH。具体实验步骤如下述：

- (1) 按如下实验线路图连接电路。
- (2) 编写程序，检查无误后汇编、连接。
- (3) 运行程序，记录拨动开关组时数据灯的显示。



8255 基本输入输出实验接线图

2、流水灯显示实验

8255 的 A 口和 B 口均为输出，，数据灯 D7~D0 由左向右，每次仅亮一个灯，循环显示，D15~D8 与 D7~D0 正相反，由右向左，每次仅点亮一个灯，循环显示。

- (1) 按如下实验线路图连接线路。
- (2) 编写程序，检查无误后汇编、连接。
- (3) 运行程序，观察 LED 灯显示，验证程序功能。
- (4) 自己改变流水灯的方式，编写程序并运行

六、源程序和运行结果

(1) 基本输入输出实验

①源程序：

```
SSTACK SEGMENT
STACK DW 32 DUP(?)
SSTACK ENDS

CODE SEGMENT
ASSUME CS:CODE

START: MOV DX, 0646H

MOV AL, 99H ;控制字
OUT DX, AL
```

```

AA1:  MOV  DX, 0640H
      IN   AL, DX
      CALL DELAY
      MOV  DX, 0642H
      OUT  DX, AL
      JMP  AA1
      DELAY: PUSH  CX
      MOV  CX, 0F00H
AA2:   PUSH  AX
      POP  AX
LOOP  AA2
      POP  CX
      RET
      CODE  ENDS
      END  START

```

（2）流水灯显示实验

①源程序：

```

SSTACK  SEGMENT
STACK   DW  32  DUP(?)
SSTACK  ENDS
CODE    SEGMENT
ASSUME  CS:CODE
START:MOV  DX, 0646H
MOV  AL, 80H  ;控制字
OUT  DX, AL
MOV  BL, 80H
MOV  BH,  01H
AA1:    ROR  BL, 1

```

```
ROL    BH, 1
CALL   DELAY
MOV    DX, 0641H
MOV    AL, BL
OUT    DX, AL
MOV    DX, 0642H
MOV    AL,  BH
OUT    DX, AL
JMP    AA1
DELAY: PUSH  CX
MOV    CX,  9F00H
AA2:   PUSH  AX
POP    AX
LOOP   AA2
POP    CX
RET
CODE   ENDS
END    START
```

②实验结果：

数据灯 D7~D0 由左向右，每次仅亮一个灯，循环显示，D15~D8 与 D7~D0 正相反，由右向左，每次仅点亮一个灯，循环显示。