华中科技大学
人工智能与自动化学院

微机原理实验一：数据传输

彭杨哲

U201914634

2021 年 11 月 6 日

1 实验目的

- 熟悉星研集成环境软件的使用方法。熟悉 Borland 公司的 TASM 编译器熟悉 8086 汇编指令，能自己编写简单的程序，掌握数据传输的方法。

2 实验内容

- 熟悉星研集成环境软件。
- 编写程序，实现数据段的传送、校验。

3 程序框图

如图1所示

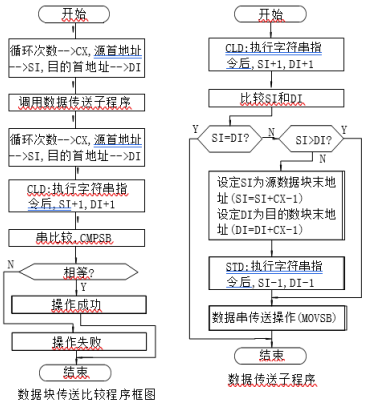

Figure 1: 程序框图

4 实验步骤

在 DS 段内 3000H ~ 30FFH 中输入数据；使用单步、断点方式调试程序，检测 DS 段内 6000H ~ 60FFH 中的内容。熟悉查看特殊功能寄存器、CS 段、DS 段的各种方法。

5 重点知识

1. 熟悉星研软件的使用方法，学会使用软件查看寄存器。

2. 熟悉几个常用的寄存器及其功能：SI，DI，CX；

3. 掌握 CMPSB 和 MOVSB 的功能。

6 参考程序

```
_STACK SEGMENT    STACK
         DW 100 DUP(?)
_STACK  ENDS
DATA SEGMENT
DATA  ENDS
CODE SEGMENT
START PROC  NEAR
         ASSUME CS:CODE, DS:DATA, SS:_STACK
         MOV    AX, DATA
         MOV    DS,AX
         MOV    ES,AX
         NOP
         MOV    CX,100H
         MOV    SI,3000H
         MOV    DI,6000H
         CALL   Move
         MOV    CX,100H
         MOV    SI,3000H
         MOV    DI,6000H
         CLD
         REPE   CMPSB
         JNE    ERROR
  TRUE:  JMP    $
  ERROR: JMP    $
Move PROC     NEAR
         CLD
         CMP    SI,DI
         JZ     Return
         JNB    Move1
         ADD    SI,CX
         DEC    SI
         ADD    DI,CX
         DEC    DI
         STD
  Move1: REP    MOVSB
  Return:RET
Move ENDP
START ENDP
CODE  ENDS
END START
```

7 实验结果

7.1 单步运行各条指令并记录相关寄存器的数值

```
1  _STACK SEGMENT    STACK;堆栈段开始
2          DW 100 DUP(?);定义100个不确定内容的字变量
3  _STACK  ENDS      ;堆栈段结束
4  DATA SEGMENT;数据段开始
5  DATA  ENDS ;数据段结束
6  CODE SEGMENT;代码段开始
7  START PROC  NEAR
8          ASSUME CS:CODE, DS:DATA, SS:_STACK
9          MOV    AX, DATA;AX=0889H,IP=0003H
10         MOV    DS,AX;IP=0005H
11         MOV    ES,AX;IP=0007H
12         NOP;IP=0008H
13         MOV    CX,100H;CX=0100H, IP=000BH
14         MOV    SI,3000H;SI=3000H,IP=000EH
15         MOV    DI,6000H;DI=6000H,IP=0011H
16         CALL   Move;SP=00C6H;IP=0026H, 下接Move子程序
17
18         MOV    CX,100H;从Move子程序跳转回来, CX=0100H, IP=0017
19         MOV    SI,3000H;SI=3000H, IP=001AH
20         MOV    DI,6000H;DI=6000H, IP=001DH
21         CLD;DF=0, IP=001EH
22         REPE   CMPSB;初始时CX=0100H, SI=3000H, DI=6000H,
                然后每一步SI,DI,CX减一, 直至CX=0, SI=3100H, DI=6100H,
                IP=0020H, ZF=1, AF=0
23         JNE    ERROR;IP=0022H
24   TRUE:  JMP    $;寄存器的值不再变化, 一直在此步循环
25   ERROR: JMP    $
26 Move PROC     NEAR
27         CLD;IP=0027H
28         CMP    SI,DI;IP=0029H, CF=1, SF=1, PF=1
29         JZ     Return;IP=002BH
30         JNB    Move1;IP=002DH
31         ADD    SI,CX;SI=3100H, IP=002FH
32         DEC    SI;SI=30FFH, IP=0030H
33         ADD    DI,CX;DI=6100H, IP=0032H
34         DEC    DI;DI=60FFH, IP=0033H
35         STD;IP=0034H, DF=1
36   Move1: REP    MOVSB;初始时CX=00FFH, SI=30FEH,DI=60FEH 每一步SI, DI,
          CX减一, 直至CX=0000H, SI=2FFFH, DI=5FFFH, IP=0036H
37   Return:RET;SP=00C8H, IP=0014H
38 Move ENDP
39 START ENDP
40 CODE  ENDS
41 END START
```

7.2 详细注释每一条指令的功能

```
_STACK SEGMENT    STACK;堆栈段开始
         DW 100 DUP(?);定义100个不确定内容的字变量
_STACK  ENDS      ;堆栈段结束
DATA SEGMENT;数据段开始
DATA  ENDS ;数据段结束
CODE SEGMENT;代码段开始
  START PROC  NEAR;START为近过程
      ASSUME CS:CODE, DS:DATA, SS:_STACK;段分配语句
      MOV    AX, DATA;将DATA的地址送入AX寄存器
      MOV    DS,AX;设置DS为DATA的地址
      MOV    ES,AX;设置ES为DATA的地址
      NOP;空操作
      MOV    CX,100H;将串的长度100H送入CX寄存器
      MOV    SI,3000H;将源首地址3000H送入SI寄存器
      MOV    DI,6000H;将目的首地址6000H送入DI寄存器
      CALL   Move;调用MOVE子程序
      MOV    CX,100H;将循环次数1000H送入CX寄存器
      MOV    SI,3000H;;将源首地址3000H送入SI寄存器
      MOV    DI,6000H;将目的首地址6000H送入DI寄存器
      CLD;设置DF=0, 使DI地址增量
      REPE   CMPSB;CX不为零且源串和目的串相等时重复比较
      JNE    ERROR;不相等则转至ERROR程序
    TRUE:  JMP    $;操作成功, 循环, 等待中断程序
    ERROR: JMP    $;操作失败, 循环, 等待中断程序
    Move PROC     NEAR;MOVE为近过程
        CLD;设置DF=0, 使DI,SI地址增量
        CMP    SI,DI;比较SI和DI的值
        JZ     Return;若上一步相等, 则转入Return子程序
        JNB    Move1;大于等于则转入到Move1子程序
        ADD    SI,CX;SI寄存器内的值加上CX的值再送入SI寄存器
        DEC    SI;设置源串末地址值
        ADD    DI,CX;DI寄存器的值加上CX的值再送入DI
        DEC    DI;设置目的串末地址
        STD;设置DF=1, 使DI,SI地址减量
      Move1: REP    MOVSB;重复传送串中的各字节, 直到CX=0为止
      Return:RET;返回
    Move ENDP;MOVE过程结束
  START ENDP;START过程结束
CODE  ENDS;代码段结束
END START ;程序结束
```

8 思考题

1. 子程序 Move 中为什么比较 SI、DI?
   答: 由于人为设置 SI,DI 时可能未考虑要传输的串的长度, 所以可能导

致源串长度过长覆盖掉目的串, 所以当 SI 等于 DI 时, 说明源串和目的串是在寄存器的同一地方, 无需传输数据, 可以直接返回; 而 SI 大于 DI 时, 从源串开始, 地址递增将数据传输至目的串位置; 但当 SI 小于 DI 时, 如果直接增量转移的话, 可能会导致源串数据的起始端传输至源串数据的末尾端, 覆盖掉了源数据, 所以从源串的末尾端开始传输便可以解决, 这样即便覆盖也是覆盖到已经传输过的源串