实验一：显示程序实验（2012年4月22日）

**1、实验目的及实验内容**

实验目的：

1.掌握在PC机上以十六进制数形式显示数据的方法

2.掌握部分DOS功能调用的使用方法

3.熟悉Tddebug调试环境和Turbo Debugger的使用

实验内容及说明：

本实验要求将指定的数据区的数据以十六进制数形式显示在屏幕上，并通过DOS功能调用完成一些提示信息的显示。

0~9的ASII码为：30H~39H，转换时，直接在原数据后面加上30H即可；A以后的数据ASII码为：41H~ 在原数据上加上7H，再加上30H即可。

实验中可使用DOS功能调用（INT21H）。

1.显示单个字符

入口：AH=02H

调用参数：DL=输出字符

2.显示字符串

入口：AH=09H

调用参数：DS：DX：串地址，’$’为结束字符

3.键盘输入并回显

入口：AH=01H

调用参数：AL=输出字符

若不回显，入口：AH=07H 或 AH=10H

4.返回DOS系统

入口：AH=4CH

调用参数：AL=返回码

**2、实验环境及实验步骤**

实验环境：PC机一台

实验步骤：

1.运行Tddebug软件，选择Edit菜单，根据实验内容的描述编写实验程序。显示部分实验流程图如下：

2.使用Compile菜单中的Compile和Link对实验程序进行汇编、连接

3.使用Run菜单中的Run运行程序，观察运行结果

4.使用Run菜单中的Debug调试程序，观察调试过程中的数据传输指令执行后各寄存器及数据区的内容

5.更改数据区中的数据，考察程序的正确性

本实验的显示部分试验流程如图所示



**3、实验过程与分析**

1. 取字节的高四位的实现

首先将一个字节读入寄存器，将该字节同0F0H相与，即可将该字节的高四位保持不变而第四位变为全0。然后将相与后的字节右移四位即得到原始字节的高四位。

1. 取字节的低四位的实现

同取字节的高四位方法类似，将字节同0FH相与，即可将该字节的低四位保持不变而高四位变为全0。这样保存原始字节的寄存器中内容变为字节的低四位。

1. 数据转换成16进制的ASCII码在屏幕显示的实现

根据转换公式，若高（低）字节数据是0~9，就直接加30H即可转化为相应的ASCII码，如对数据“A”高字节为“0100”，屏幕应该要显示“4”。故高字节“0100”需要加上30H即“0110000”才能构成4的ASCII码“0110100”即屏幕显示“4”。若高（低）字节数据是A~F，就直接加37H即可转化为相应的ASCII码，如对数据“K”的低字节为“1011”，屏幕应该要显示“B”，故低字节“1011”需要加上37H即“0110111”才能构成B的ASCII码“1000010”即屏幕显示“B“。

（注：A的16进制ASCII码为41H，K的16进制ASCII码为4BH）

1. 数据区内容输出的实现

显示单个字符即设置入口AH=02H，调用参数DL=输出字符。显示字符串即设置入口AH=01H，调用参数DS:DX:串地址，以“$”为结束符。

程序代码：

DATA SEGMENT USE16

MES DB 'Show a as hex:',0AH,0DH,'$'

SD DB 'a'

DATA ENDS

CODE SEGMENT USE16

ASSUME CS:CODE,DS:DATA

START:MOV AX,DATA

MOV DS,AX

MOV DX,OFFSET MES ;Show message

MOV AH,09H

INT 21H

MOV SI,OFFSET SD

MOV AL,DS:[SI]

AND AL,0F0H ;取高4位

SHR AL,4

CMP AL,0AH ;是否是A以上的数

JB C2

ADD AL,07H

C2:ADD AL,30H

MOV DL,AL ;show character

MOV AH,02H

INT 21H

MOV AL,DS:[SI]

AND AL,0FH

CMP AL,0AH

JB C3

ADD AL,07H

C3:ADD AL,30H

MOV DL,AL ;show character

MOV AH,02H

INT 21H

MOV AX,4C00H ;go back to dos

INT 21H

CODE ENDS

END START

将上述程序保存为.ASM文件，编译、连接，再执行。

输出结果：61（a的ASII码）

**4、实验结果总结**

根据实验结果，程序运行能正常显示出字符串和相应数据的16进制ASCII码制，达到了实验的目的。

本次实验代码有老师提供，主要是了解tddebug软件的使用，程序代码不难看懂，但其格式不甚了解，在自己编程的时候会感觉无从下手，应进一步的学习汇编语言。

实验二：描述符及描述符表实验（2012年5月13日）

**1、实验目的及实验内容**

实验目的：

1、熟悉保护模式的编程格式和描述符的声明方法

2、掌握使用选择符访问段的寻址方法

3、学习和掌握静态存储器6264的特性及扩展方法

4、熟悉通过保护模式访问扩展存储器的方法

实验内容：

32位微机保护模式下全局描述符及描述符表

要求实现在一个0级代码段中将源数据段中的一段数据复制到目标数据段中。其中所有的段描述符均放置在GDT中。

**2、实验环境及实验步骤**

实验环境：PC机一台，TD-PIT实验装置一套。

实验步骤：

1. 1、运行TDDebug软件，选择Edit菜单编写实验程序

声明各数据段

在0级代码段中实现数据传输

结束

描述GDT

1. 2、使用Compile菜单中的Compile和Link对试验程序进行汇编、连接
2. 3、使用Pmrun装入实验程序。如果装入成功屏幕上会显示“Load OK！”，否则会给出相应的错误提示信息
3. 4、若程序成功装入，可使用R命令查看调试系统为实验程序分配的系统资源
4. 5、使用GDT命令查询系统的GDT，并查看实验程序中声明的代码段、数据段描述符在GDT中的位置以及对应段的物理地址、段属性和段界限等
5. 6、使用F9运行程序，如果程序正常结束，命令显示区中将显示“Correct Running”
6. 7、使用T命令单步执行程序，在单步过程中使用D命令查询数据源数据区及目标数据区中的数据，验证执行的正确性

**3、实验过程与分析**

1. 数据结构的组织实现

在程序开始声明一个数据段“DESG”来描述这三个段的描述符，其中有代码段描述符SCODE、源数据段描述符DATAS和目标数据段描述符DATAD，将他们的选择子分别定义为SCODE\_SEL、DATAS\_SEL、DATAD\_SEL。

1. 三个描述符的定义实现

代码段描述符：SCODE DESC <CLEN，CSEG，，ATCE，，>

源数据段描述符：DATAS DESC <DLEN，DSEG1，，ATDW，，>

目标数据段描述符：DATAD DESC <BUFFLEN，DSEG2，，ATDW，，>

1. 确定装入程序的重定位信息的实现

为了给装入程序提供重定位信息，三个存储段描述符中地址的低16位用每个描述符对应的标号来填写。在程序装入内存时。调试系统会根据地址的低16位重定位该段对应的真实物理地址，并将该地址写入描述符中。

源程序：

.386P

DESC STRUC ；定义描述符结构

LIMITL DW 0

BASEL DW 0

BASEM DB 0

ATTR DB 0

LIMITH DB 0

BASEH DB 0

DESC ENDS

ATCE = 98H ；定义特殊常量

ATDR = 90H

ATDW = 92H

DSEG SEGMENT PARA USE16

；声明开始初始化GDT中的描述符

GDT LABEL BYTE

ID1 DESC <0FFFFH,0FFFFH,0FFH,0FFH,0FFH,0FFH> ；标记1

SCODE DESC <0FFFFH,CSEG,,ATCE,,> ；代码段描述符

DATAS DESC <D1LEN,DSEG1,,ATDW,,> ；源数据段描述符

DATAD DESC <BUFLEN-1,DSEG2,,ATDW,,> ；目标数据段描述符

GDTLEN = $-GDT

SCODE\_SEL = SCODE-GDT ；代码段选择子

DATAS\_SEL = DATAS-GDT ；源数据段选择子

DATAD\_SEL = DATAD-GDT ；目标数据段选择子

ID2 DESC <0FFFFH,0FFFFH,0FFH,0FFH,0FFH,0FFH > ；标记2

ID3 DESC <0FFFFH,0FFFFH,0FFH,0FFH,0FFH,0FFH > ；标记3

DSEG ENDS

；定义源数据段

DSEG1 SEGMENT USE16

HELLO DB 'Hello! everybody'

DB 'This is my first PM program!'

DB 256H DUP(0)

D1LEN =$-1

DSEG1 ENDS

；定义目标数据段

DSEG2 SEGMENT USE16

BUFLEN = 256

BUFFER DB BUFLEN DUP(0)

DSEG2 ENDS

CSEG SEGMENT USE16

ASSUME CS:CSEG,DS:DSEG

START PROC

MOV AX,DATAS\_SEL ；装入源数据段选择子

MOV DS,AX

MOV AX,DATAD\_SEL ；装入目标数据段选择子

MOV ES,AX

CLD

XOR SI,SI

XOR DI,DI

MOV CX,BUFLEN

M1: MOVSB

LOOP M1

INT 0FFH ；返回调试系统

START ENDP

CLEN =$-1

CSEG ENDS

END START

程序成功装入后，使用GDT命令查看GDT中内容，程序执行前后各项均只有TYPE值改变：数据段、代码段分别由 2、8变为 3、9，表示其分别由读/写状态、只执行状态转变为读/写，已访问状态、只执行，已访问状态，使用D0018:00查看目标数据段内容发现，程序执行前后，其内容改变，并且变得与源数据段内容一致，数据复制成功。

**4、实验结果总结**

通过这次实验我对保护模式下的汇编语言编程有了进一步的理解，也通过这次实验和第一次实验的对比，明白了汇编语言实模式与保护模式编程的异同。

在实验的具体过程中，我也更深入的理解了课堂上所讲的关于描述符的结构组成与定义等理论知识，巩固了课堂学习效果。

**实验三：8255并行接口应用实验**（2012年5月20日）

**实验目的及实验内容**

实验目的：

1、掌握8255的工作方式及应用编程

2、掌握8255的典型应用电路接法

实验内容：

流水灯显示实验。编写程序，使8255的A口和B口均为输出，实现16位数据灯的相对循环显示。

实验原理：

并行接口是以数据的字节为单位向I/O设备或被控制对象传递信息的。CPU和接口之间的数据传送总是并行的，即可以同时传递8位、16位和32位等。8255可编程外围接口芯片是Inter公司生产的通用并行I/O接口芯片，它具有A、B、C三个并行接口，因+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。

**实验环境及实验步骤**

实验环境: PC机一台，TD-PIT/TD-PIT-B实验装置一套

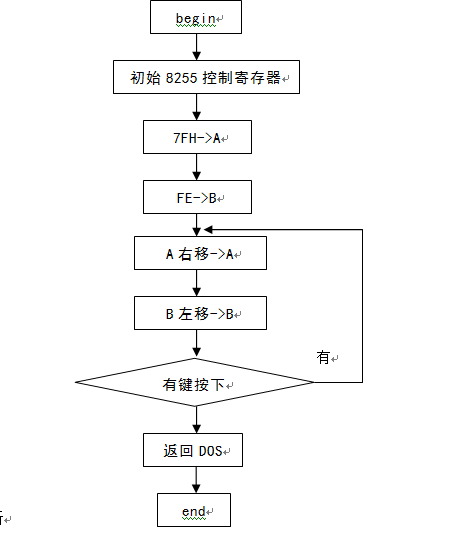
实验步骤：

首先分别向A口和B口写入7FH和FEH，然后分别将该数右移和左移一位，再送到端口上，这样循环下去，从而实现流水灯的显示。

1.设计实验线路图，并将PCI总线扩展卡上的ADD-ON总线与仿真ISA总线进行连接，完成线路连接。

2.编写实验程序，检查无误后汇编、连接。编写程序时首先分别向A口和B口写入7FH 和FEH，然后将该数右移和左移一位，再送到端口上，这样循环下去，从而实现流 水灯的显示。

3.打开试验箱电源，运行程序，观察数据灯的显示。



**实验过程与分析**

**1.**方式控制字的确定

实验要求A口、B口在工作方式0下输出，C口高四位低四位均输出，说一查表易知 方式控制字为1000000B即80H。

2.A口B口数据移位的实现

A口与B口数据移位的主要作用是利用数据的二进制位信息控制LED管的显示。如A 口初始值为7FH即01111111B，此时PA0对应的D8灯亮，而PA1~PA7对应的D9~D15灯 灭。此时A口数据循环右移移位变为10111111B，此时PA1对应的D9灯亮，而PA0PA2~PA7 对应的D8D10~D15灯灭.如此循环则可以实现流水灯的效果。数据的循环右（左）移用汇 编命令ROR（ROL）即可。

源代码：

MY8255\_A EQU 0C460H

MY8255\_B EQU 0C461H

MY8255\_C EQU 0C462H

MY8255\_MODE EQU 0C463H

DATA SEGMENT

LA DB ?

LB DB ?

DATA ENDS

CODE SEGMENT

ASSUME CS:CODE,DS:DATA

START: MOV AX,DATA

MOV DS,AX

MOV DX,MY8255\_MODE

MOV AL,80H

OUT DX,AL

MOV DX,MY8255\_A

MOV AL,7FH

OUT DX,AL

MOV LA,AL

MOV DX,MY8255\_B

MOV AL,0FEH

OUT DX,AL

MOV LB,AL

CALL DALLY

A1: MOV AL,LA

ROR AL,1

MOV LA,AL

MOV DX,MY8255\_A

OUT DX,AL

MOV AL,LB

ROL AL,1

MOV LB,AL

MOV DX,MY8255\_B

OUT DX,AL

CALL DALLY

CALL BREAK

JMP A1

DALLY: MOV CX,3FFH

D1: MOV AX,0FFFFH

D2: DEC AX

JNZ D2

LOOP D1

RET

BREAK PROC NEAR

MOV AH,06H

MOV DL,0FFH

INT 21H

JE RETURN

MOV AX,4C00H

INT 21H

RETURN: RET

BREAK ENDP

CODE ENDS

END START

**实验结果总结**

通过这次实验我了解了16550串行接口的工作方式及应用，对通讯的相关知识有了一定的了解。此外对16550芯片中的各个接口的作用以及其中各个寄存器的作用有了比较深入的理解。当然也通过流水灯实验，对外面大街上霓虹灯的实现原理有了初步的了解。

**实验四：16550串行接口应用实验**（2012年 5月27日）

**实验目的及实验内容**

实验目的：

1. 掌握16550的工作方式及应用。

2.学习有关串行同讯的知识。

实验内容：

编写程序，将一串数据发送至串口，再接受回来显示，实现串行自发自收。

实验原理：

16550是一种连接任何类型虚拟串行接口的可编程通讯接口，与Inter微处理器完全兼容地使用非常广泛的异步接受器/发送器（UART）。它内置了16字节的FIFO缓冲，最大通信速率可达115Kb/s，是现代基于微处理器设备包括PC机和许多调制解调器在内的最普通的通信接口。

1.端口地址的使用

16550的负责寄存器选择的地址线是三位A2A1A0，最大只能选择23=8个，但是16550的内部寄存器有11个。为解决这一矛盾，可在访问除数寄存器时，令DLAB位为“1”；访问接受数据寄存器、发送保持寄存器和中断允许寄存器时，令DLAB位置“0”。

2.寄存器控制字的说明

线路控制寄存器LCR，主要用于指定异步串行通信的数据格式。

线路状态寄存器LSR，主要是向处理器提供有关数据传输的状态。

3.波特率除数寄存器的值设置

除数寄存器值=基准时钟频率÷(16×波特率)

因为16550芯片输入的基准时钟频率为1.8432MHZ，查表知若波特率为9600b/s，则除数寄存器值为000CH，DLH中应填00H，DLL中应填0CH。

4.中断允许寄存器

16550共有4级中断，按优先级从高到低依次为：接收出错中断、接收缓冲器满中断、发送保持寄存器空中断和MODEM输入状态改变中断。

**实验环境及实验步骤**

实验环境：PC机一台，TD-PIT/TD-PIT-B实验装置一套。

实验步骤：

1. 编写实验程序，连续向发送寄存器写A到Z，并完成接受和显示。

2. 设计实验线路图，并将PCI总线扩展卡上的ADD-ON总线与仿真ISA总线进行连接，完成线路连接。

3. 运行程序，观察屏幕上的显示数据是否正确。

在存在变量AA中数据加1

开始

设置通信波特率

有按键按下

在TBR中写入AA

初始化LCR，设置通信数据格式

读取数据显示在屏幕上

是否到5AH

返回DOS

是

否

从RBR中读数

否

是

**实验过程与分析**

1. 寄存器选择的实现

16550的负责寄存器选择的地址线是三位A2A1A0，最大只能选择23=8个，但是16550的内部寄存器有10个。为解决这一矛盾，可在访问除数寄存器时，令DLAB位为“1”；访问接受数据寄存器、发送保持寄存器和中断允许寄存器时，令DLAB位置“0”。

1. 波特率除数寄存器的值设置

因为16550芯片输入的基准时钟频率为1.8432MHZ，查表知若波特率为9600b/s，则除数寄存器值为000CH，DLH中应填00H，DLL中应填0CH。**.**

源代码

MY16550\_0 EQU 0C480H

MY16550\_1 EQU 0C481H

MY16550\_3 EQU 0C483H

MY16550\_4 EQU 0C484H

MY16550\_5 EQU 0C485H

DATA SEGMENT

AA DB 40H

DATA ENDS

CODE SEGMENT

ASSUME CS:CODE,DS:DATA

START: MOV AX,DATA

MOV DS,AX

MOV DX,MY16550\_3

MOV AL,80H

OUT DX,AL

CALL DALLY

MOV DX,MY16550\_0

MOV AL,0CH

OUT DX,AL

CALL DALLY

MOV DX,MY16550\_1

MOV AL,00H

OUT DX,AL

CALL DALLY

MOV DX,MY16550\_3

MOV AL,1BH

OUT DX,AL

CALL DALLY

MOV DX,MY16550\_4

MOV AL,03H

OUT DX,AL

CALL DALLY

MOV DX,MY16550\_1

MOV AL,00H

OUT DX,AL

CALL DALLY

A1: INC AA

MOV DX,MY16550\_0

MOV AL,AA

OUT DX,AL

CALL DALLY

A2: CALL BREAK

MOV DX,MY16550\_5

IN AL,DX

AND AL,01H

JZ A2

MOV DX,MY16550\_0

IN AL,DX

MOV AH,0EH

INT 10H

CALL DALLY

CMP AL,5AH

JNZ A1

MOV AX,4C00H

INT 21H

DALLY: PUSH CX

PUSH AX

MOV CX,0100H

D1: MOV AX,2000H

D2: DEC AX

JNZ D2

LOOP D1

POP AX

POP CX

RET

BREAK PROC NEAR

MOV AH,06H

MOV DL,0FFH

INT 21H

JE RETURN

MOV AX,4C00H

INT 21H

RETURN:RET

BREAK ENDP

CODE ENDS

END START

实验结果

屏幕上显示A~Z 26个字母

**实验结果总结**

通过这次实验我了解了16550串行接口的工作方式及应用，对通讯的相关知识有了一定的了解。此外对16550芯片中的各个接口的作用以及其中各个寄存器的作用有了比较深入的理解。对书本教材关于寄存器控制字的设置原理有了进一步的了解，不再是停留在死记硬背的层面，对控制字每一位的控制对象和设置组合方式有了具体的概念。

**实验五：PCI设备中断应用实验**（2012年6月3日）

**实验目的及实验内容**

实验目的：

1.掌握使用PCIBIOS访问中断的方法

2.掌握使用PCI中断的方法

3.掌握操作PCI设备I/O端口的方法

实验内容：

调用PCI中断应用实验的源程序，根据当前PCI设备占用微机系统的资源情况，修改源程序中的相关内容，并按照实验要求对程序进行分析与说明，画出程序流程图，最后运行该程序完成按键中断的响应。即每产生一次按键中断，向显示器输出字符“7”。

实验原理：

1.PC机系统中8259A的应用

在硬件系统中，8259仅占用两个外设接口地址，在片选有效的情况下，利用A0来寻址不同的寄存器和命令字。

现在PC系统中包含了两片8259A中断控制器，经级联可以管理16级硬件中断，但其中部分中断源已经被系统硬件占用。两片8259A的端口地址为：主片在020H-03FH，实际使用020H和021H两个端口；从片在0A0H-0BFH范围，实际使用0A0H和0A1H两个端口。

2.中断向量表

80x86微机系统中能处理256个中断向量，其类型号为00H~0FFH，他们的中断处理方式及过程应该根据CPU的工作模式加以区分。在保护模式下，系统中有一张中断描述符表，记录所有中断/异常处理程序的入口。在实模式下微机系统采用向量中断，无论是内部还是外部中断均对应一个相应的中断向量码。每个中断向量占用4字节，分别存放中断服务程序的IP和CS。256个中断向量被放置于内存从0开始的1KB区域中，该区域称为中断向量表。对于微机系统没有使用的中断向量，可以作为中断被用户使用。使用前，只需要将软中断处理程序的程序入口填入对应中断向量即可。而微机系统已经占用的中断向量可以根据需要对其进行修改。填写或修改中断向量的方法为：

MOV [中断向量\*4],IP地址

MOV [中断向量\*4+2],CS地址

**实验环境及实验步骤**

实验环境：

PC机一台，TD-PIT实验装置一套

实验步骤：

1.利用PCI BIOS获取PCI设备 配置资源，具体操作是在TDDebug环境中，运行PCI BIOS.EXE文件（该文件保存在实验系统的G:\TDPIT目录中）

2.根据实验内容的描述调出实验参考源程序PCI\_INT.ASM（该文件保存在实验系统的G:\TDPIT目录中）

3.根据PCI设备实际占用微机系统的资源情况，修改源程序中的相关内容，确认无误后再进行编译、连接

4.设计实验线路图，并将PCI总线扩展板上的ADD-ON总线与仿真ISA总线进行连接，完成线路连接

5.用实验排线连接仿真ISA总线中IRQ信号和KK开关单元中的KKl-（下降沿）端

6.打开实验箱电源，运行实验程序，按动KKl-键，观察微机显示屏幕是否输出字符“7”

**实验过程与分析**

实验分析和说明

本实验要求使用实验装置提供的中断请求信号IRQ（下降沿有效），利用KKl-的负脉冲作为中断源，在中断处理中完成字符“7”的显示。使用试验装置提供的IRQ信号，除了要操作PC的8259外，还需要操作实验用PCI总线扩展板卡上的“INTCSR’”和“IMB4”寄存器。通过操作上述寄存器，才可以实现实验用中断的初始化、清除等。

修改之后的源代码：

;引入仿真卡中寄存器，始地址重置

INTCSR\_BYTE0 EQU 0CC38H

INTCSR\_BYTE1 EQU 0CC39H

INTCSR\_BYTE2 EQU 0CC3AH

INTCSR\_BYTE3 EQU 0CC3BH

IMB4\_BYTE3 EQU 0CC1FH

DATA SEGMENT

CSBAK DW ?

IPBAK DW ?

MKBAK DB ?

DATA ENDS

CODE SEGMENT

ASSUME CS:CODE,DS:DATA

START: CLI

MOV AX,DATA

MOV DS,AX

MOV DX,INTCSR\_BYTE0

MOV AL,00H

OUT DX,AL

MOV DX,INTCSR\_BYTE1

MOV AL,1FH

OUT DX,AL

MOV DX,INTCSR\_BYTE2

MOV AL,3FH

OUT DX,AL

MOV DX,INTCSR\_BYTE3

MOV AL,00H

OUT DX,AL

MOV AX,0000H

MOV ES,AX

;中断程序入口地址修改

MOV DI,34H ;INT 0DH,34H=0DH\*4

MOV AX,ES:[DI]

MOV IPBAK,AX ;IP

MOV AX,OFFSET MYINT

CLD

STOSW

MOV AX,ES:[DI] ;CS

MOV CSBAK,AX

MOV AX,SEG MYINT

STOSW

;OCW1 修改

IN AL,21H

MOV MKBAK,AL

AND AL,0DFH

OUT 21H,AL

STI

A1: NOP

CALL BREAK

JMP A1

MYINT: PUSH DS

PUSH AX

PUSH DX

MOV DX,IMB4\_BYTE3

IN AL,DX

MOV DX,INTCSR\_BYTE2

MOV AL,3FH

OUT DX,AL

;清除中断标志OCW2 修改

MOV AL,65H

OUT 20H,AL;

MOV AH,0EH

MOV AL,37H

INT 10H

POP DX

POP AX

POP DS

IRET

BREAK PROC NEAR

PUSH DS

MOV AH,06H

MOV DL,0FFH

INT 21H

JE RETURN

POP DS

CLI

MOV AL,MKBAK

;使用主片的奇端口

OUT 21H,AL

MOV AX,0000H

MOV ES,AX

;恢复中断入口地址

MOV DI,34H ;INT 0DH,34H=0DH\*4

MOV AX,IPBAK

CLD

STOSW

MOV AX,CSBAK

STOSW

MOV DX,INTCSR\_BYTE1

MOV AL,00H

OUT DX,AL

STI

MOV AX,4C00H

INT 21H

RETURN:POP DS

RET

BREAK ENDP

CODE ENDS

END START

实验结果

每按下一次开关，屏幕上便显示一个字符“7”

**实验结果总结**

本次实验中首先学会了DOS环境下进入根目录和复制文件的操作方法，进一步加深了自己的基础知识。本次试验主要就是在源代码的基础之上对其进行修改，使之符合本次试验所使用的实验的平台。通过对原程序的修改更进一步的理解了实模式下的中断调用，对8259的工作机制进一步加深了理解。