

2020-2021 第 2 学期
计算机接口技术实验报告

学院	计算机与通信工程学院			
专业班级	计算机科学与技术			
班级序号	180235			
学号	20188068			
姓名	孔天欣			
指导教师	张旭			
成绩				

计算机接口技术实验报告

班级：计科 1802 姓名：孔天欣 学号：20188068 实验日期：2021.4.7
学院：计算机与通信工程学院 专业：计算机科学与技术
实验顺序：17 实验名称：系统中断实验 指导教师：张旭

一. 实验目的

- (1) 掌握 PC 机中断处理系统的基本原理。
- (2) 学会编写中断服务程序。

二. 实验环境

Tpc-zk-II 集成开发环境
微机原理与接口技术实验箱

三. 实验原理

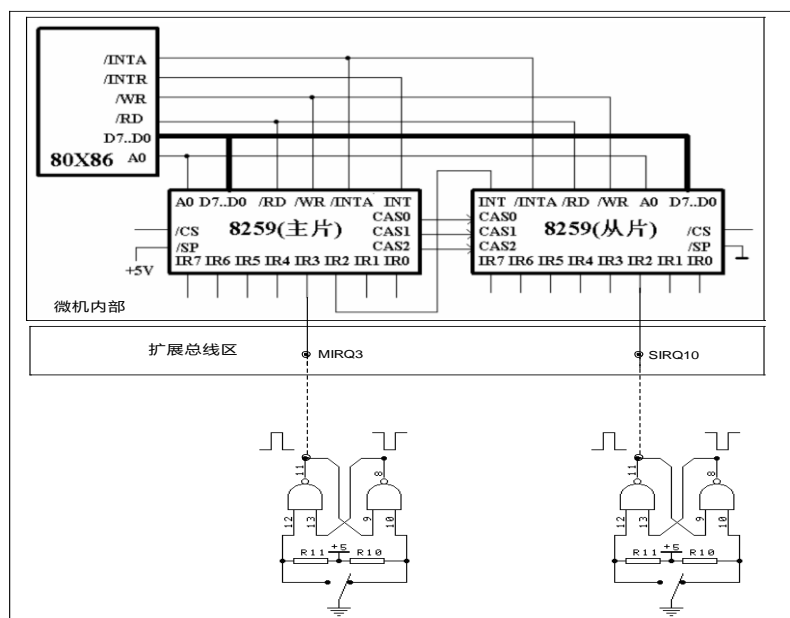


图 1.1 实验原理图

系统中断流程：

1. CPU 预先通过代码读入端口 21H 的 OCW1 值，将其和 11110111 进行与运算设置 IR3 允许中断，回写新值至 OCW1 中，以此改变中断屏蔽寄存器 IMR 的编码。
2. CPU 预先通过 8259A 主片的中断类型码 (0BH) 注册中断向量表和对应的中断子程序。
3. 按下中断脉冲，通过 20H 端口将中断信号发出，令主片引脚 IR3 接收中断信号，中断请求寄存器 IRR 变为 0000 1000 (也就是 IR3 为 1，该引脚有中断信号)。
4. 主片的中断屏蔽寄存器 IMR 判断该 IR3 的信号是否被屏蔽，若未被屏蔽，则由 INT 引脚发出中断请求信号 INTR 至 CPU。
5. CPU 处于开中断状态下则在当前指令周期完成后，响应 INTA 非至主

片的 INTA 非接口并关中断。

6. 8259A 主片接收到响应后，置中断服务寄存器 ISR 为 0000 1000，表明该中断源正在服务，同时将 IRR 复位为 0，以免中断结束后的误触发情形。

7. 到达第二个中断响应总线周期时，CPU 再次发出 INTA 非信号，此时 8259A 将它的中断类型码发送至 CPU，CPU 通过计算寻找中断向量表中该中断类型码的对应中断程序入口地址并取出去执行。

8. CPU 先保护现场，之后开中断（允许中断嵌套）执行中断子程序，输出文字至终端，最后关中断恢复现场。

9. 完成中断子程序后，CPU 发送 EOI 命令至 8259A 主片，将 ISR 复位。完成所有中断流程。

四. 实验步骤及结果分析

1. 实验代码

```
1.  DATA SEGMENT
2.  INFO DB 'kongtianxin20188068$'
3.  DATA ENDS
4.  CODE SEGMENT
5.  ASSUME CS:CODE,DS:DATA
6.  START:
7.  IN AL,21H
8.  AND AL,11110111B
9.  OUT 21H,AL
10. MOV AX,CODE
11. MOV DS,AX
12. MOV DX,OFFSET INT3
13. MOV AH,25H
14. MOV AL,0BH
15. INT 21H
16. HERE: JMP HERE
17. INT3 PROC
18.  PUSH AX
19.  PUSH DS
20.  PUSH DX
21.  STI
22.  MOV AX,DATA
23.  MOV DS,AX
24.  MOV DX,OFFSET INFO
25.  MOV AH,9
26.  INT 21H
27.  MOV DX,20H
28.  MOV AL,20H
29.  OUT DX,AL
```

```

30.  CLI
31.  POP DX
32.  POP DS
33.  POP AX
34.  IRET
35.  INT3 ENDP
36.  CODE ENDS
37.  END START
38.

```

2. 实验截图

```

Bochs BIOS, 1 cpu, $Revision: 1.103.2.2 $ $Date: 2004/02/02 22:39:22 $
ata0 master: Generic 1234 ATA-2 Hard-Disk (4 MBytes)
ata0 slave: Generic 1234 ATA-2 Hard-Disk (29 MBytes)

Booting from Hard Disk...
Starting MS-DOS...

C:\>set path=c:\dos;d:\tasm\bin
C:\>d:
D:\>cd d:\exec
D:\EXEC>deltree /Y .
Deleting .\1.exe...

D:\EXEC>..\all.com
loading file ... 1.exe
load & exec
kongtianxin20188068kongtianxin20188068_
F12 enables mouse

```

图 1.2 控制台输出

可以看到，控制台输出了相关的消息两次，第二次消息是为了确保中断写入成功及相关寄存器正确初始化。

0000FE80:	IN AL, 0X21	: E421
0000FE82:	AND AL, 0XF7	: 24F7
0000FE84:	OUT 0X21, AL	: E621
0000FE86:	MOV AX, 0XFE8	: B8E80F
0000FE89:	MOV DS, AX	: 8ED8
0000FE8B:	MOV DX, 0X16	: BA1600
0000FE8E:	MOV AH, 0X25	: B425
0000FE90:	MOV AL, 0XB	: B00B
0000FE92:	INT 0X21	: CD21
0000FE94:	JMP 0XFE94	: EBFE
0000FE96:	PUSH AX	: 50
0000FE97:	PUSH DS	: 1E
0000FE98:	PUSH DX	: 52
0000FE99:	STI	: FB
0000FE9A:	MOV AX, 0XFE6	: B8E60F
0000FE9D:	MOV DS, AX	: 8ED8

图 1.3 反汇编结果

上图可以看到，中断子程序的入口地址为 FE96H (PUSH AX)，同时，根据中断类型码为 0BH，可以算出其中断向量表的地址为 $4 \times 0BH = 2CH$ ，在下图的 2CH 地址处，可以看到 CS 为 0F8EH，IP 为 0016H，那么计算出中断子程序入口地址为 $0FE8H \times 10H + 0016H = FE96H$ ，和上图的入口地址相同。

0000:002C	16 00 E8 0F 53 FF 00 F0 53 FF 00 F0 B7 00 C3 0D
0000:003C	F4 06 70 00 E6 00 00 C0 4D F8 00 F0 41 F8 00 F0
0000:004C	74 07 70 00 39 E7 00 F0 4A 08 70 00 2E E8 00 F0
0000:005C	D2 EF 00 F0 A0 91 00 F0 FB 07 70 00 6E FE 00 F0
0000:006C	EE 06 70 00 D6 91 00 F0 53 FF 00 F0 22 05 00 00
0000:007C	0E 35 00 C0 CC 40 21 00 F8 40 21 00 BF 04 7C 0F
0000:008C	4A 01 38 0E 55 01 38 0E E5 42 21 00 6C 43 21 00
0000:009C	C4 A1 21 00 D2 40 21 00 62 07 70 00 D2 40 21 00

图 1.4 查看内存

五. 实验心得疑问建议

通过本次实验，本人成功掌握了中断处理系统的基本原理和方法，并通过编写汇编程序成功为 8259A 中断源注册中断向量表以及实现了对应的中断服务子程序，最后能够通过反汇编和内存查看等辅助工具进行程序中断相关过程的分析和计算。由此显著提高了本人对计算机中断机制本质的理解。

计算机接口技术实验报告

班级：计科 1802 姓名：孔天欣 学号：20188068 实验日期：2021. 4. 14
学院：计算机与通信工程学院 专业：计算机科学与技术
实验顺序：75 实验名称：8253 定时器计数器实验 指导教师：张旭

四. 实验目的

掌握 8253 计数特点和编程方法。掌握 8253 工作方式 3 的基本工作原理、计数特点和编程方法。

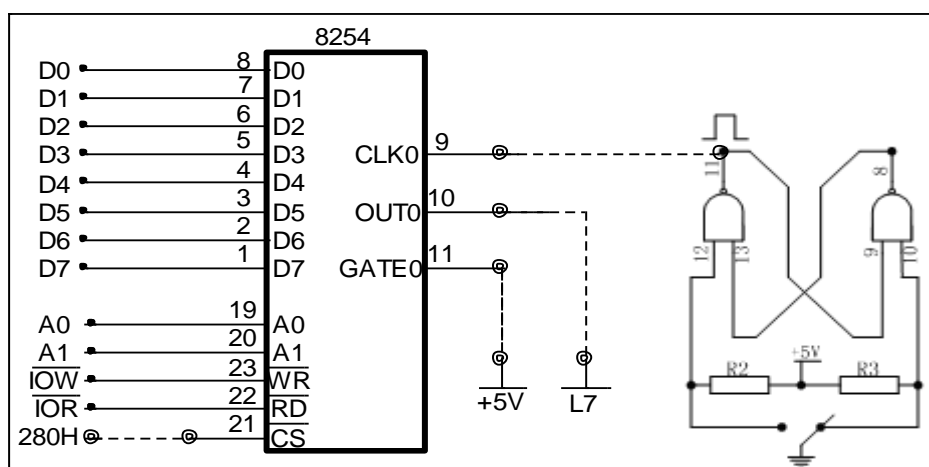
五. 实验环境

Tpc-zk-II 集成开发环境
微机原理与接口技术实验箱

六. 实验原理

1 计数实验

按图连接电路，将计数器 0 设置为方式 2，计数器初值为 N ($N \leq 0FH$)，用手逐个输入单脉冲，编程使计数值在屏幕上显示，用逻辑笔观察 OUT0 电平变化(当输入 N+1 个脉冲后 OUT0 变高电平)。并将计数过程记录下来。

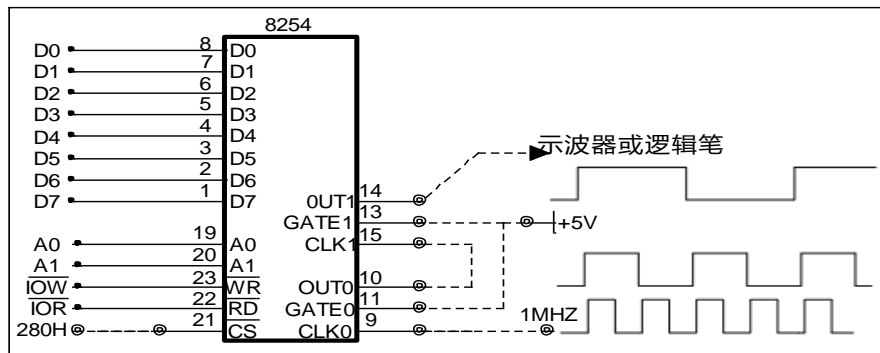


接线：

8254/CLK0	接	单脉冲/正脉冲
8254/CS	接	I/O 译码/Y0 (280H—287H)
8254/OUT0	接	(LED 显示/L7) 或者 (逻辑笔)
8254/GATE0	接	+5V

2 定时实验

按下图连接电路，将计数器 0、计数器 1 分别设置工作方式，CLK0 连接时钟 1MHZ，由 T01000 分频后变为 1000HZ，再由 T1 进行 1000 分频后得 1HZ。



接线:

8254/CLK0	接	时钟/1MHz
8254/CS	接	I/O 译码/Y0 (280H—287H)
8254/OUT0	接	8254/CLK1
8254/GATE0, GATE1	接	+5V
8254/OUT1	接	逻辑笔

五. 实验步骤及结果分析

【实验代码】

1. 计数实验

```

1. DATA SEGMENT
2. IO82530 EQU 280H
3. IO82531 EQU 281H
4. IO82532 EQU 282H
5. IO8253C EQU 283H
6. DATA ENDS
7.
8. CODE SEGMENT
9. ASSUME CS:CODE,DS:DATA
10. START:
11. MOV AX,DATA
12. MOV DS,AX
13. MOV DX,IO8253C
14. MOV AL,14H
15. OUT DX,AL
16. MOV DX,IO82530
17. MOV AL,0FH
18. OUT DX,AL
19. CNT0:IN AL,DX
20. CALL DISP
21. MOV AH,1
22. INT 16H
23. JZ CNT0
24. MOV AH,4CH

```

```

25.  INT 21H
26.
27.  DISP PROC
28.  PUSH AX
29.  PUSH DX
30.  CMP AL,9
31.  JLE NUM
32.  ADD AL,7
33.  NUM:
34.  ADD AL,30H
35.  MOV DL,AL
36.  MOV AH,2
37.  INT 21H
38.  POP DX
39.  POP AX
40.  RET
41.  DISP ENDP
42.  CODE ENDS
43.  END START

```

2. 定时实验

```

1.  DATA SEGMENT
2.  IO82530 EQU 280H
3.  IO82531 EQU 281H
4.  IO82532 EQU 282H
5.  IO8253C EQU 283H
6.  DATA ENDS
7.
8.  CODE SEGMENT
9.  ASSUME CS:CODE,DS:DATA
10. START:
11.  MOV AX,DATA
12.  MOV DS,AX
13.  MOV DX,IO8253C
14.  MOV AL,36H
15.  OUT DX,AL
16.  MOV DX,IO82530
17.  MOV AX,1000
18.  OUT DX,AL
19.  MOV AL,AH
20.  OUT DX,AL
21.  MOV DX,IO825C
22.  MOV AL,76H

```



```

23.  OUT DX,AL
24.  MOV DX,IO8253I
25.  MOV AX,1000
26.  OUT DX,AL
27.  MOV AL,AH
28.  OUT DX,AL
29.  MOV AH,4CH
30.  INT 21H
31.
32. CODE ENDS
33. END START

```

【实验截图】

1. 计数实验

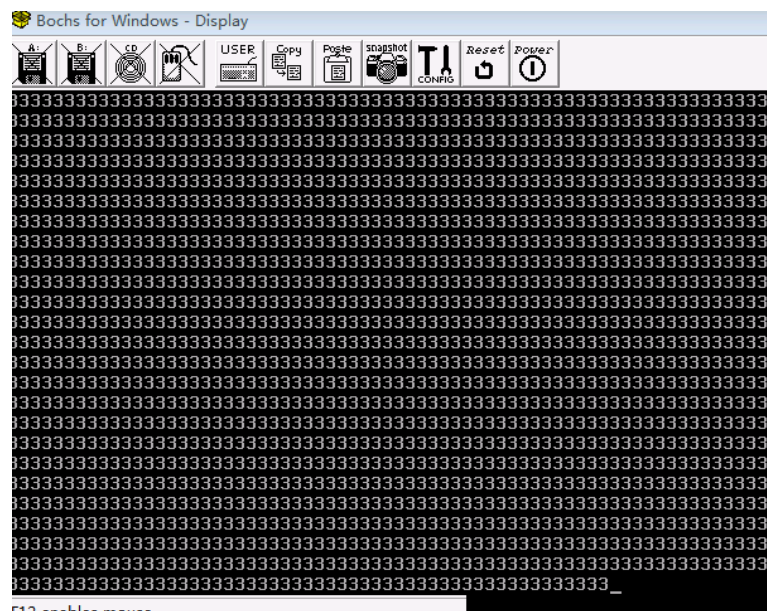


图 2.1 控制台输出

每隔约 1s 按下脉冲按钮时，输出的数字会对应的自减 1，说明计数器在正常工作。

2. 定时实验

可以看到每隔约 0.5s 逻辑笔处的红绿灯交替进行闪烁，计数灯出现自增现象。

六. 实验心得疑问建议

通过本次实验，本人成功掌握了 8253 定时器计数器的基本原理和方法，并通过编写汇编程序成功为 8253 定时器计数器设置定时任务和计数任务，最后能够通过对屏幕输出以及小灯泡闪烁等现象进行观察，进行 8253 定时器计数器相关过程的分析。由此显著提高了本人对 8253 定时器计数器本质的理解。

计算机接口技术实验报告

班级: 计科 1802 姓名: 孔天欣 学号: 20188068 实验日期: 2021. 4. 21
学院: 计算机与通信工程学院 专业: 计算机科学与技术
实验顺序: 75 实验名称: 8255 并行 I/O 输入/输出实验 指导教师: 张旭

七. 实验目的

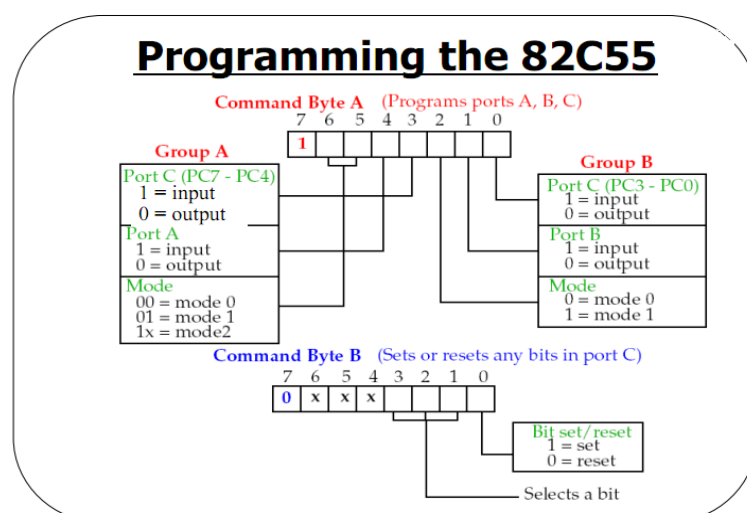
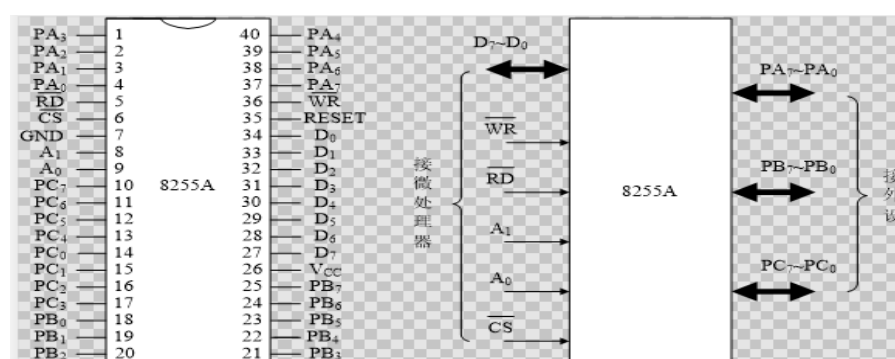
掌握 8253 计数特点和编程方法。掌握 8253 工作方式 3 的基本工作原理、计数特点和编程方法。

八. 实验环境

Tpc-zk-II 集成开发环境
微机原理与接口技术实验箱

九. 实验原理

8255 是 Intel 公司生产的可编程外围接口电路, 简称 PPI。它有 A、B、C 三个八位端口寄存器, 通过 24 位端口线与外部设备相连, 基中 C 口可分为上半部和下半部。这 24 根端口线全部为双向三态。三个端口可分二组来使用, 可分别工作于三种不同的工作方式。



8255 方式 1 是一种具有专用联络线的选通输入输出方式。只有 A 口和 B 口作为数据口，C 口的位线分别做 A 口和 B 口的联络线。C 口联络线的定义是固定的，编程者必须按照要求使用、不能改变。此方式常用于“中断”或“查询”方式进行数据传送。

方式 1 输入：

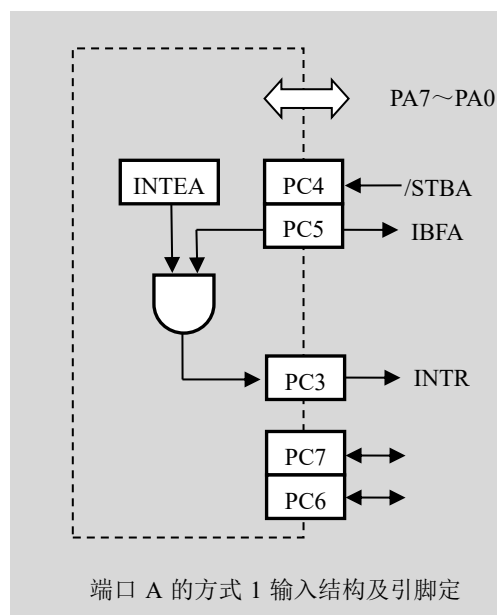
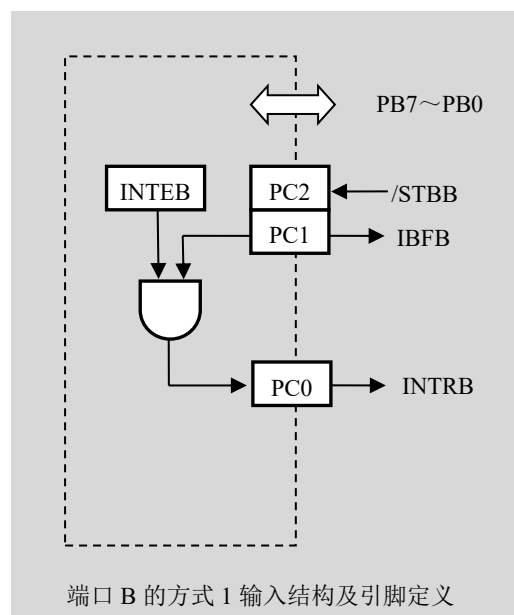
当 A 口或 B 口被设定为方式 1 输入时，两个口各指定 C 口的 3 根线作为为 8255 与外设之间的联络信号，这些信号线的定义是固定的。联络线的定义如下：

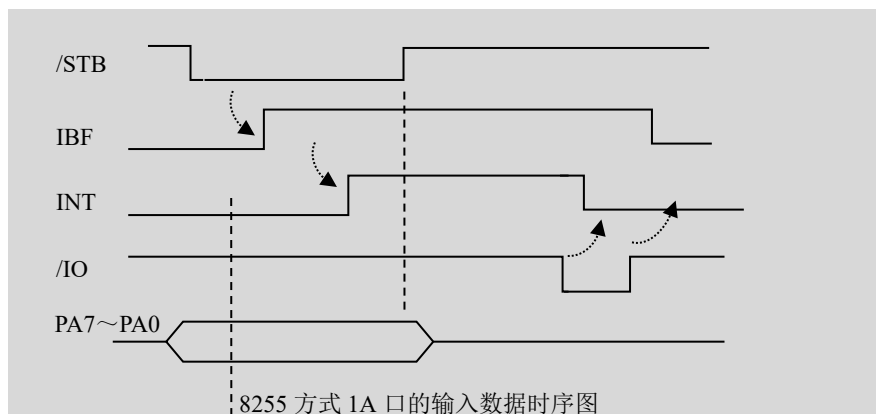
/STB --- 选通输入信号，低电平有效。当外设发来有效信号时，就把外设送来的数据锁存到端口的数据缓冲器中；

IBF --- 输入缓冲器“满”信号输出信号，高电平有效。此信号可以向外设表明一个状态：即外部的数据已经被 8255A 锁存到缓冲器中、但还没有被 CPU 取走，在这种情况下外设是不能再向 8255A 发送数据了。只有当 CPU 执行 IN AL,DX 指令读走数据后，IBF 回零，外设才可发送下一个数据；

INTR --- 中断请求输出端，高电平有效。此信号可以作为向 CPU 发出的申请信号，CPU 可以利用中断服务程序将 8255A 中的数据读走。当然 INTR 信号的产生是有条件的：8255A 中的 INTE=1（中断允许位有效）。

INTE --- 中断允许位，在 A 口或 B 口被设定为方式 1 输入时，规定了 PC4 和 PC2 做端口 A 和端口 B 的中断允许位，可事先使用对 C 口“按位置位、复位”控制字来设定 PC4 和 PC2 的电平。





方式 1 输入方式的时序图详见图。

- a) 当外设数据准备好、并检测到 IBF 为空时 (IBF=0)，将 8 位的并行数据送出，并发出“输入选通”信号 /STB，8255A 利用此信号将数据线上的并行数据进行锁存；
- b) 数据的锁存导致 IBF=1 (输入缓冲器满)，该信号可以阻止外部设备继续向 8255A 输入数据，避免数据丢失；
- c) 如果中断是允许的 (INTE=1)，8255A 则会向外部发出中断申请信号 (INTR=1)。当 CPU 检测到该信号，且利用中断服务程序将 8255A 中的数据使用 IN AL,DX 指令读走后中断申请信号 INTR=0、缓冲器满标志 IBF=0；
- d) 外部设备检测到 IBF=0 后，就可以发送下一个字节的并行数据……。

方式 1 输出：

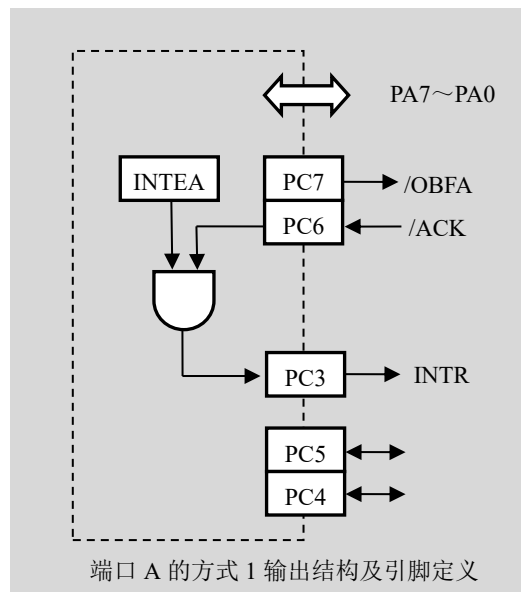
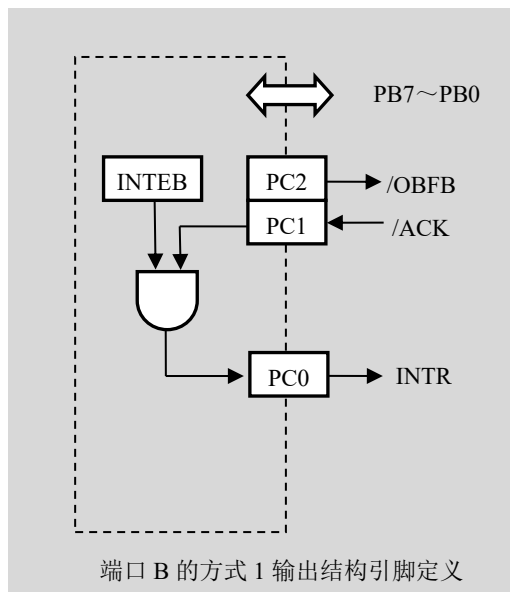
当 A 口或 B 口被设定为方式 1 输出时，两个口各指定 C 口的 3 根线作为为 8255 与外设之间的联络信号，这些信号线的定义是固定的。联络线的定义如下：

/OBF --- 缓冲器满信号，低电平有效。当 8255A 接收到 CPU 由 OUT DX,AL 指令送来的数据时，就通过该信号通知外部设备准备接收数据；

/ACK --- 外设送来的“应答”信号，低电平有效。此信号表明外设已经收到了 8255A 发出的数据信号，它是对 /OBF 的一个应答信号；

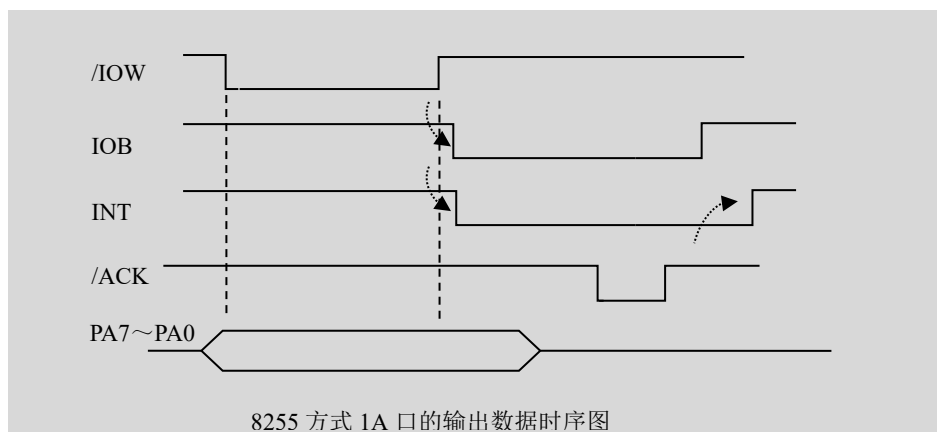
INTR --- 中断请求输出端，高电平有效。此信号可以作为向 CPU 发出的申请信号，CPU 可以利用中断服务程序向 8255A 发送下一个字节的数据。当然 INTR 信号的产生是有条件的：8255A 中的 INTE=1 (中断允许位有效)，且输出缓冲器空 (/OBF=1) 和 /ACK=1。

INTE --- 中断允许位 (设置同方式 1 输入)。



方式 1 的输出方式时序图详见下图：

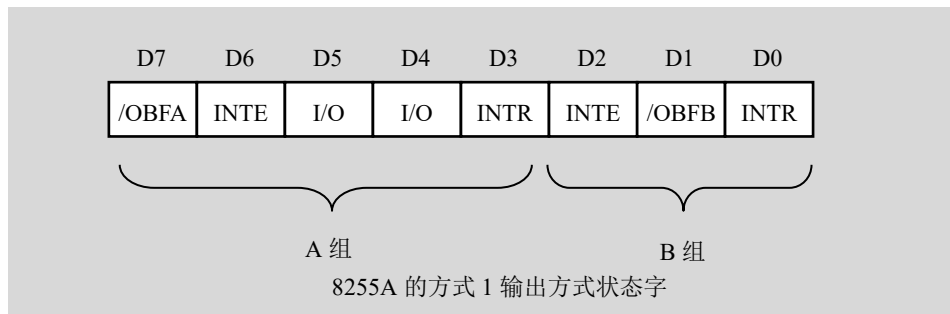
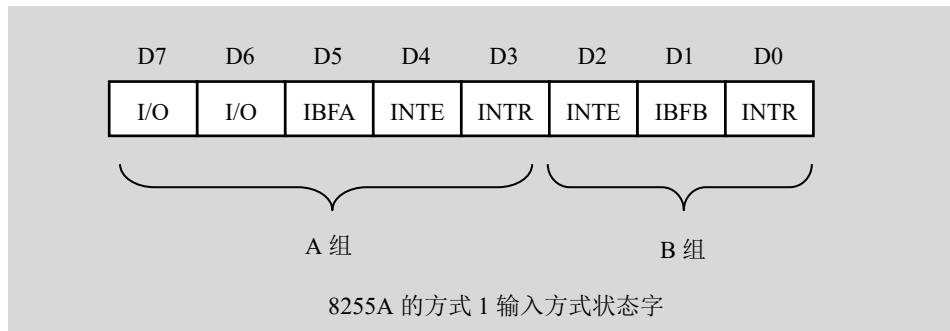
- CPU 通过执行指令 `OUT DX,AL` 将数据写入 8255A，此时指令会产生 `/IOW` 信号，在 `/IOW` 信号的上升沿时 `/OBF=1`，向外设通知 8255A 的输出缓冲期已满。在 `/IOW` 上升沿时使 `INTR` 变低、撤销中断请求；
- 8255A 的 `/OBF` 信号触发了外部设备对数据的读取，并产生一个应答信号负脉冲 (`/ACK=0`) 以表明外设已收到数据，在 `/ACK` 的上升沿时使 8255A 的中断请求信号有效 (`INTR=1`)；
- 如果中断是允许的 (`INTE=1`)，`INTR=1` 信号可以引发 CPU 的中断服务，在服务程序中 CPU 发送下一个字节到 8255A.....。



方式 1 的状态字

在上面的叙述中，采用的是“中断模式”通过 8255A 来协调 CPU 与外部设备之间的数据交换。当然也可以采用“查询状态字”的方式来实现双方的数据交换。

比如标志 `/IBF`、`/OBF` 的状态为是通过“读 C 口”实现的。



六. 实验步骤及结果分析

【实验代码】

1. 方式 0 输入输出

```

1.  data segment
2.  io8255a equ 288h
3.  io8255b equ 289h
4.  io8255c equ 28ah
5.  io8255k equ 28bh
6.  data ends
7.  code segment
8.  assume cs:code,ds:data
9.  start:
10. mov ax,data
11. mov ds,ax
12. mov dx,io8255k
13. mov al,89h
14. out dx,al
15. readc:
16. mov dx,io8255c
17. in al,dx
18. mov dx,io8255a
19. out dx,al
20. mov ah,1
21. int 16h
22. jz readc

```

```
23. mov ah,4ch
24. int 21h
25. code ends
26. end start
```

2. 方式 1 选通实验

```
1. data segment
2. io8255a equ 288h
3. io8255b equ 289h
4. io8255c equ 28ah
5. io8255k equ 28bh
6. data ends
7. code segment
8. assume cs:code,ds:data
9. start:
10. mov ax,data
11. mov ds,ax
12. mov dx,io8255k
13. mov al,0a7h
14. out dx,al
15. mov al,4
16. out dx,al
17. readb:
18. mov dx,io8255c
19. in al,dx
20. test al,00000010b
21. jz readb
22. mov dx,io8255b
23. in al,dx
24. and al,00000111b
25. mov dx,io8255a
26. out dx,al
27. mov ah,1
28. int 16h
29. jz readb
30. mov ah,4ch
31. int 21h
32. code ends
33. end start
```

3. 方式 1 中断输入实验

```
1. data segment
```

```
2.  io8255a equ 288h
3.  io8255b equ 289h
4.  io8255c equ 28ah
5.  io8255k equ 28bh
6.  data ends
7.  code segment
8.  assume cs:code,ds:data
9.  start:
10. in al,21h
11. and al,11110111b
12. out 21h,al
13. mov ax,code
14. mov ds,ax
15. mov dx,offset int3
16. mov ah,25h
17. mov al,0bh
18. int 21h
19. mov ax,data
20. mov ds,ax
21. mov dx,io8255k
22. mov al,0b8h
23. out dx,al
24. mov al,9
25. out dx,al
26. here: jmp here
27. int3 proc
28. push dx
29. push ax
30. push ds
31. mov ax,data
32. mov ds,ax
33. mov dx,io8255a
34. in al,dx
35. mov dl,al
36. mov ah,2
37. int 21h
38. mov al,20h
39. out 20h,al
40. pop ds
41. pop ax
42. pop dx
43. iret
44. int3 endp
45. code ends
```


【实验现象和截图】

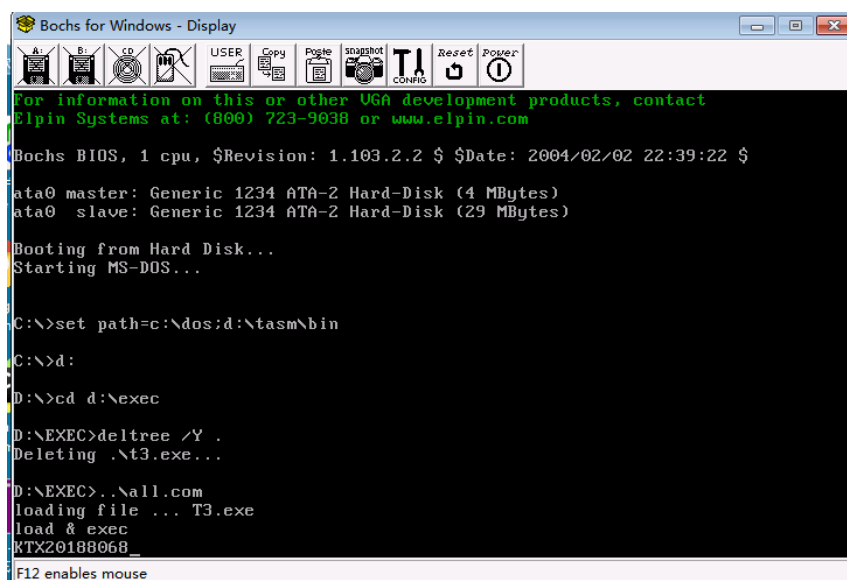
1. 方式 0 输入输出实验

可以看到，八个逻辑电平开关处有各自对应的二极管，对于每一个开关，若拨到高电平，则其对应的二极管亮；若拨到低电平，则其对应的二极管熄灭。

2. 方式 1 选通实验

可以看到 K2~K0 的开关序列对应 3 位二进制数字，共有 8 个数字，每个数字对应一个二极管，例如，K2~K0 依次为 010，则 L2 灯亮。

3. 方式 2 中断输入实验



```
Bochs for Windows - Display
For information on this or other VGA development products, contact
Elpin Systems at: (800) 723-9038 or www.elpin.com

Bochs BIOS, 1 cpu, $Revision: 1.103.2.2 $ $Date: 2004/02/02 22:39:22 $
ata0 master: Generic 1234 ATA-2 Hard-Disk (4 MBytes)
ata0 slave: Generic 1234 ATA-2 Hard-Disk (29 MBytes)

Booting from Hard Disk...
Starting MS-DOS...

C:\>set path=c:\dos;d:\tasm\bin
C:\>d:
D:\>cd d:\exec
D:\EXEC>deltree /Y .
Deleting .\t3.exe...

D:\EXEC>.. \all.com
loading file ... T3.exe
load & exec
KTX20188068
F12 enables mouse
```

图 3.1 控制台输出

8 位逻辑开关对应 ASCII 码的二进制形式，因此每设置 8 位开关各自的高低位，再按下脉冲产生中断，则会在屏幕中输出其对应的 ASCII 字符。

七. 实验心得疑问建议

通过本次实验，本人成功掌握了 8255 芯片以及并行 I/O 输入和输出的基本原理和方法，并通过编写汇编程序成功为 8255 芯片设置不同的输入输出方式，最后能够通过对屏幕输出以及逻辑开关高低电平对应的二极管亮灭小等现象进行观察，进行 8255 芯片相关过程的分析。由此显著提高了本人对 8255 芯片本质的理解。

计算机接口技术实验报告

班级：计科 1802 姓名：孔天欣 学号：20188068 实验日期：2021. 4. 28
学院：计算机与通信工程学院 专业：计算机科学与技术
实验顺序：30 实验名称：模/数转换器 0809 查询法实验 指导教师：张旭

十. 实验目的

了解模/数转换的基本原理，掌握 ADC0809 的使用方法。

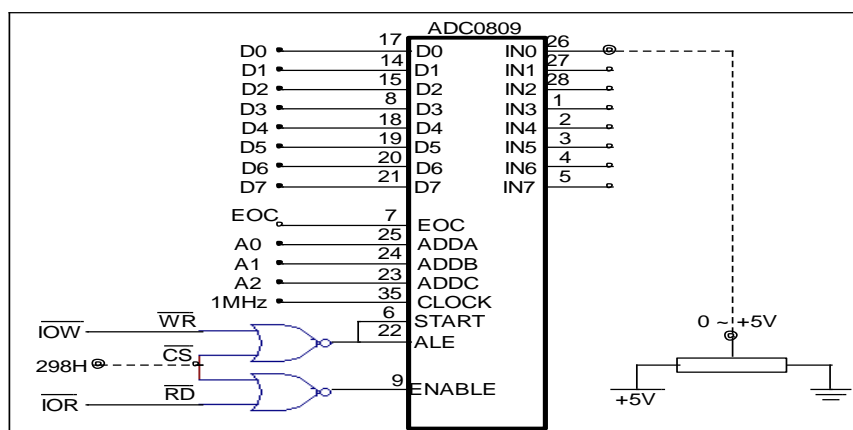
十一. 实验环境

Tpc-zk-II 集成开发环境
微机原理与接口技术实验箱

十二. 实验原理

4.1 ADC0809 延时法实验

1. 编程采集 IN0 输入的电压，在屏幕上显示出转换后的数据(用 16 进制数)。

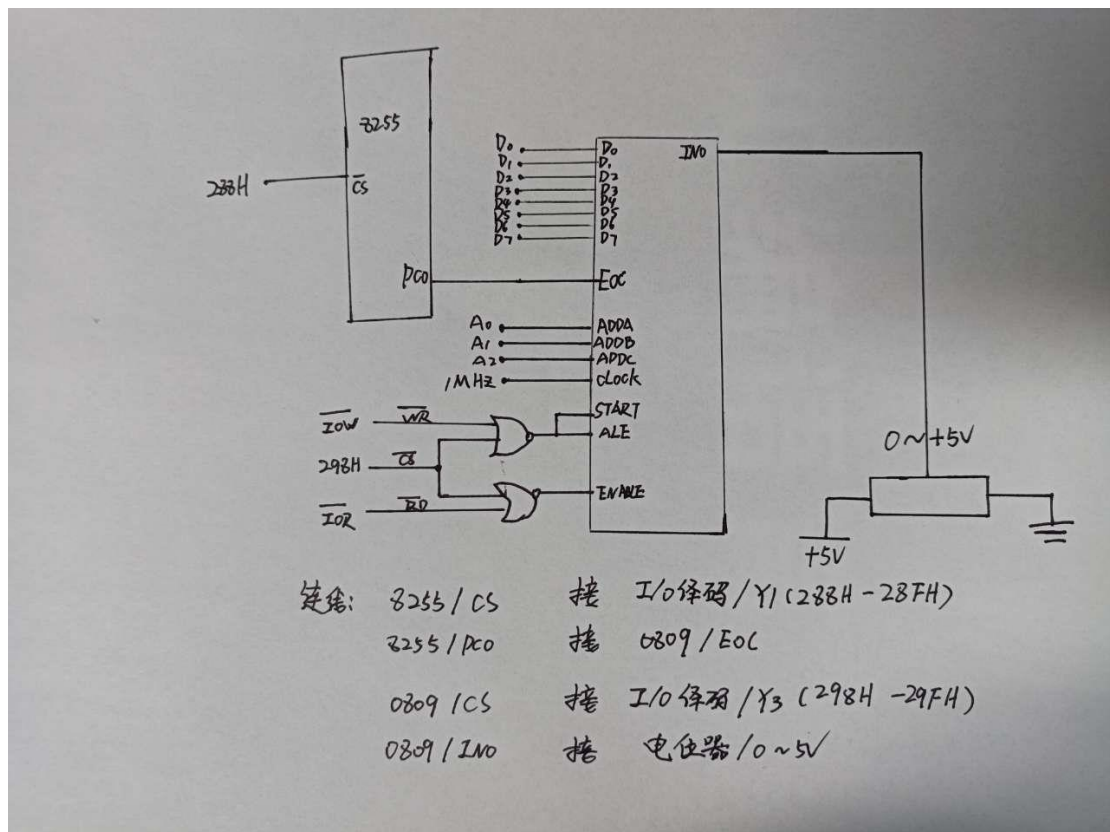


2. 接线：

0809/CS	接	I/O 地址译码/Y3 (298H---29FH)
0809/IN0	接	电位器/0~5V

4.2 ADC 查询法实验

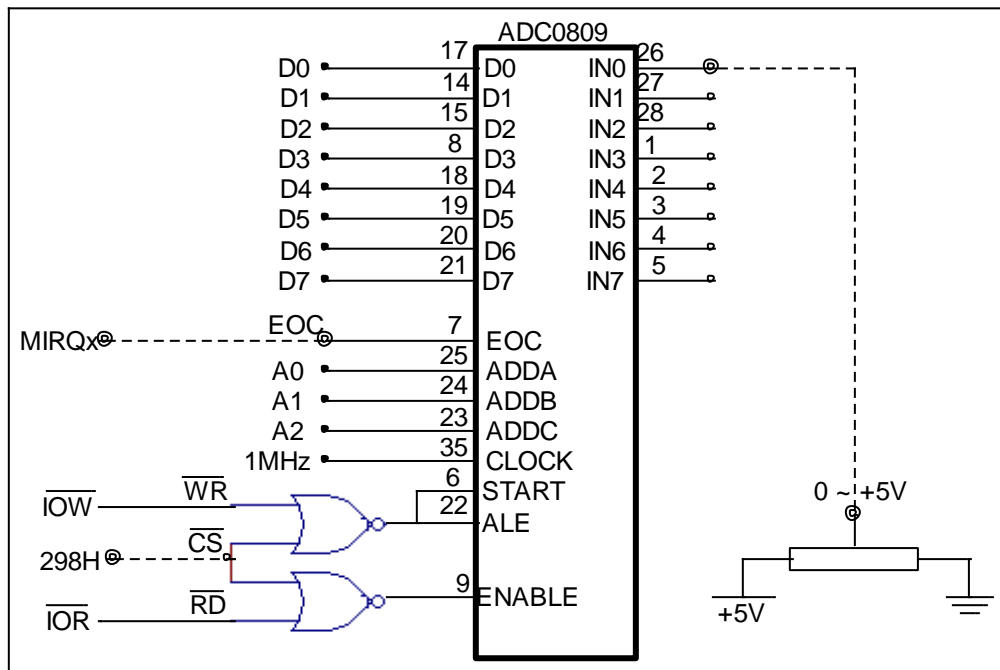
按照下图进行连线，使用查询法，检测 EOC 状态。编程采集 IN0 输入的电压，在屏幕上显示出转换后的数据(用 16 进制数)。



4.3 ADC0809 中断法实验

程序由主程序和中断服务程序组成:

- 主程序包括初始化, 等待中断; 如果有中断时在主程序中再启动一次 ADC0809, 然后等待下一次中断。主程序等待中断是借助于一个标志 (SI 寄存器);
- 中断服务程序中首先读取 ADC0809 中的数据, 然后转换的数据 (用 16 进制数) 后通过屏幕显示。



2. 接线:

0809/CS	接	I/O 地址译码/Y3 (298H---29FH)
0809/IN0	接	电位器/0~5V
0809/EOC	接	总线/MIRQx

七. 实验步骤及结果分析

【实验代码】

1. ADC0809 查询法实验

```

1.  data segment
2.  ad08090 equ 298h
3.  data ends
4.  code segment
5.  assume cs:code,ds:data
6.  start:
7.  mov ax,data
8.  mov ds,ax
9.  s:
10. mov dx,ad08090
11. out dx,al
12. mov cx,0ffh
13. s1:
14. loop s1
15. mov dx,ad08090
16. in al,dx

```

```
17. mov bl,al
18. mov cl,4
19. shr al,cl
20. and al,00001111b
21. call disp
22. mov al,bl
23. and al,00001111b
24. call disp
25. mov dl,20h
26. mov ah,2
27. int 21h
28. mov ah,1
29. int 16h
30. jz s
31. mov ah,4ch
32. int 21h
33. disp proc
34. push ax
35. push dx
36. cmp al,9
37. jle num
38. add al,7
39. num:
40. add al,30h
41. mov dl,al
42. mov ah,2
43. int 21h
44. pop dx
45. pop ax
46. ret
47. disp endp
48. code ends
49. end start
```

2. ADC 查询法实验

```
1. data segment
2. ad08090 equ 298h
3. io8255c equ 28ah
4. io8255k equ 28bh
5. data ends
6. code segment
7. assume cs:code,ds:data
8. start:
```

```
9.  mov ax,data
10. mov ds,ax
11. mov dx,io8255k
12. mov al,4h
13. s:
14. mov dx,ad08090
15. out dx,al
16. readc:
17. mov dx,io8255c
18. in al,dx
19. test al,00000001b
20. jz readc
21. mov dx,ad08090
22. in al,dx
23. mov bl,al
24. mov cl,4
25. shr al,cl
26. and al,00001111b
27. call disp
28. mov al,bl
29. and al,00001111b
30. call disp
31. mov dl,20h
32. mov ah,2
33. int 21h
34. mov ah,1
35. int 16h
36. jz s
37. mov ah,4ch
38. int 21h
39. disp proc
40. push ax
41. push dx
42. cmp al,9
43. jle num
44. add al,7
45. num:
46. add al,30h
47. mov dl,al
48. mov ah,2
49. int 21h
50. pop dx
51. pop ax
52. ret
```

```
53. disp endp
54. code ends
55. end start
```

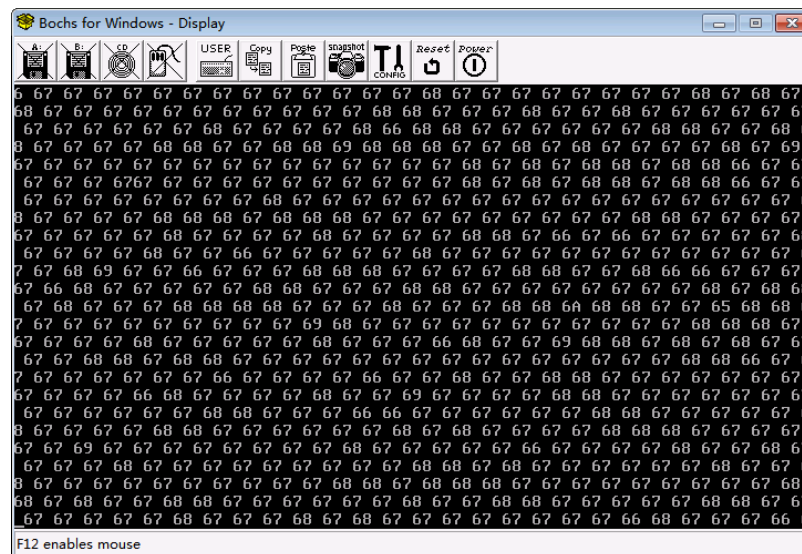
3. ADC0809 中断法实验

```
1. data segment
2. ad08090 equ 298h
3. data ends
4. code segment
5. assume cs:code,ds:data
6. start:
7. in al,21h
8. and al,11110111b
9. out 21h,al
10. mov ax,code
11. mov ds,ax
12. mov dx,offset int3
13. mov ah,25h
14. mov al,0bh
15. int 21h
16. mov ax,data
17. mov ds,ax
18. mov dx,ad08090
19. out dx,al
20. here:
21. jmp here
22. int3 proc
23. push ax
24. push bx
25. push cx
26. push dx
27. push ds
28. mov ax,data
29. mov ds,ax
30. mov dx,ad08090
31. in al,dx
32. mov bl,al
33. mov cl,4
34. shr al,cl
35. and al,00001111b
36. call disp
37. mov al,bl
38. and al,00001111b
```

```
39. call disp
40. mov dl,20h
41. mov ah,2
42. int 21h
43. mov al,20h
44. out 20h,al
45. mov dx,ad08090
46. out dx,al
47. pop ds
48. pop dx
49. pop cx
50. pop bx
51. pop ax
52. iret
53. int3 endp
54. disp proc
55. push ax
56. push dx
57. cmp al,9
58. jle num
59. add al,7
60. num:
61. add al,30h
62. mov dl,al
63. mov ah,2
64. int 21h
65. pop dx
66. pop ax
67. ret
68. disp endp
69. code ends
70. end start
```

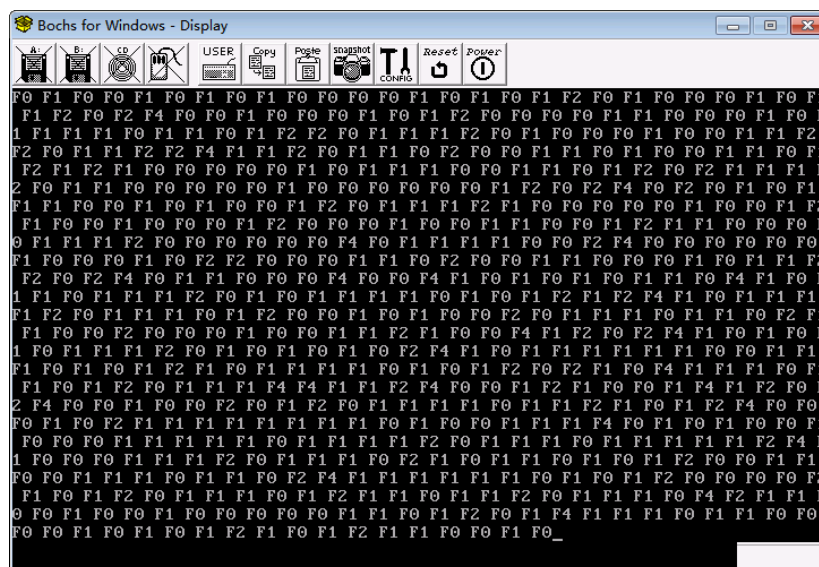
【实验现象和截图】

1. ADC0809 延时法实验

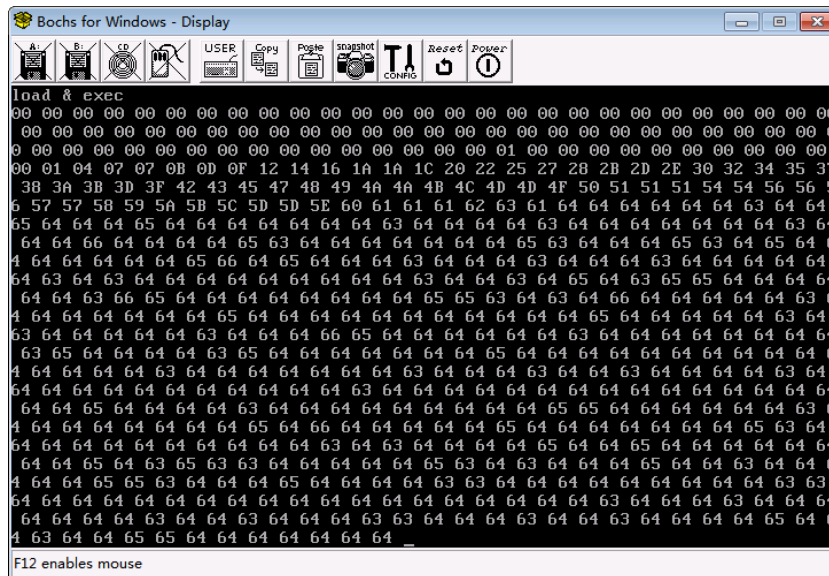


运行程序后，当旋转电压旋钮时，可以看到屏幕上不断刷新的数字逐渐发生变化，区间为 00-FF，这反映了电压的变化，说明模数转换器在正常工作。

2. ADC 查询法实验



3. ADC0809 中断法实验



八. 实验心得疑问建议

通过本次实验，本人成功掌握了模/数转换器 ADC0809 以及延时法、中断法、查询法的基本原理和方法，并通过编写汇编程序成功为 ADC0809 设置不同的模数转换方式，最后能够通过旋转电压旋钮，对屏幕输出数字变化的现象进行观察，进行 ADC0809 相关过程的分析。由此显著提高了本人对 ADC0809 本质的理解。