



# 第7章 可编程接口芯片及应用

**接口电路：**计算机和外设进行数据交互的电路。

**可编程接口芯片特点：**具有微逻辑控制功能，一般可编程芯片都有多种工作方式，可通过软件或硬件选择设定其工作方式。

- 8255并行接口
- 8253定时/计数器
- 8250 串行通讯
- ADC 0809

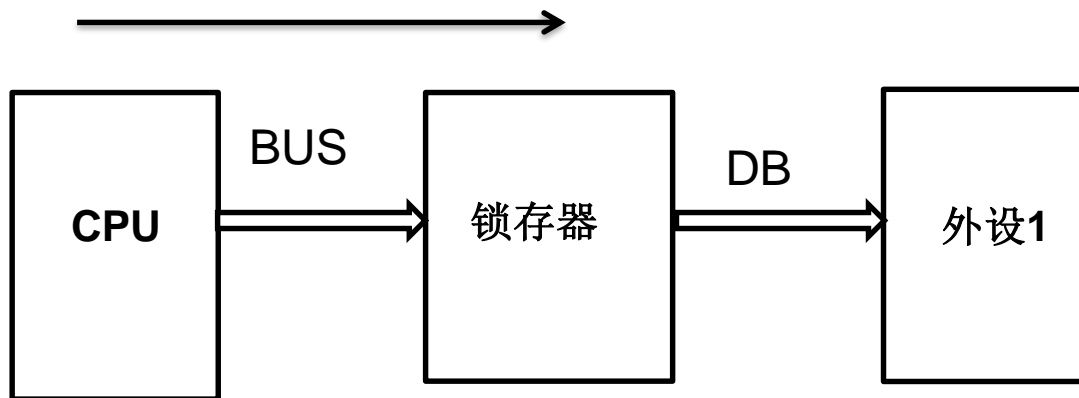
# 7.1 8255并行接口芯片

## 7.1.1 8255基本介绍

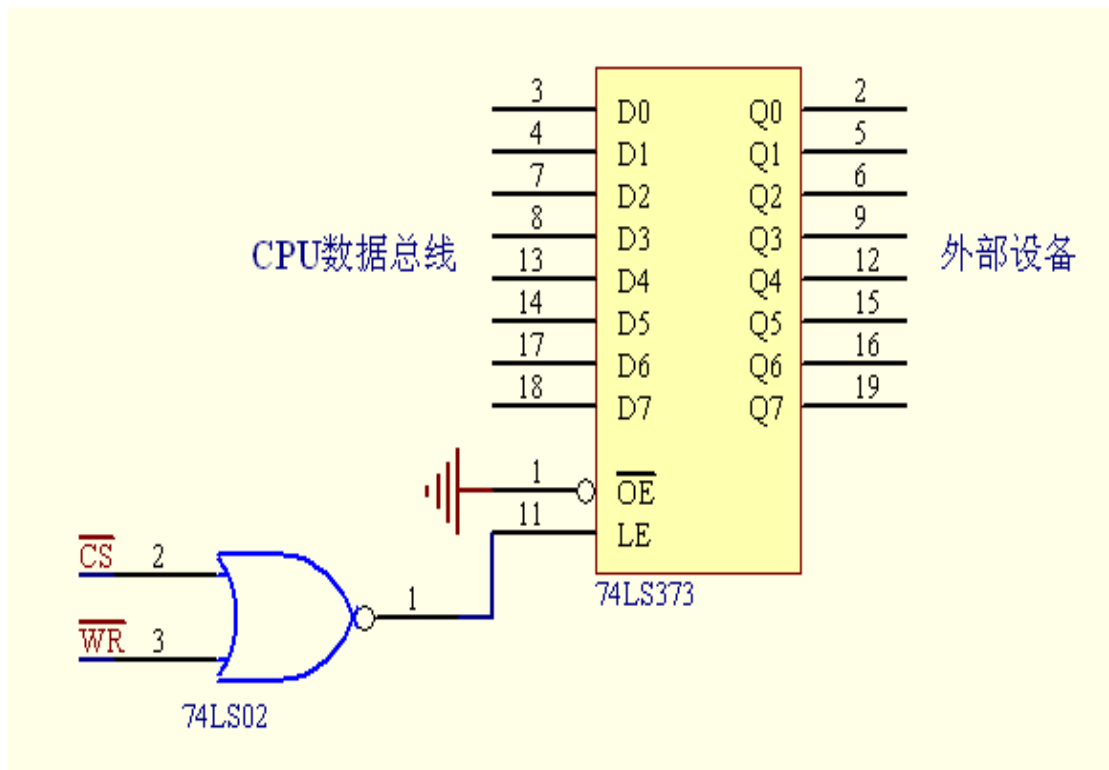
### 1.CPU和外设的数据输入输出

#### 1) CPU输出数据

OUT DX, AL



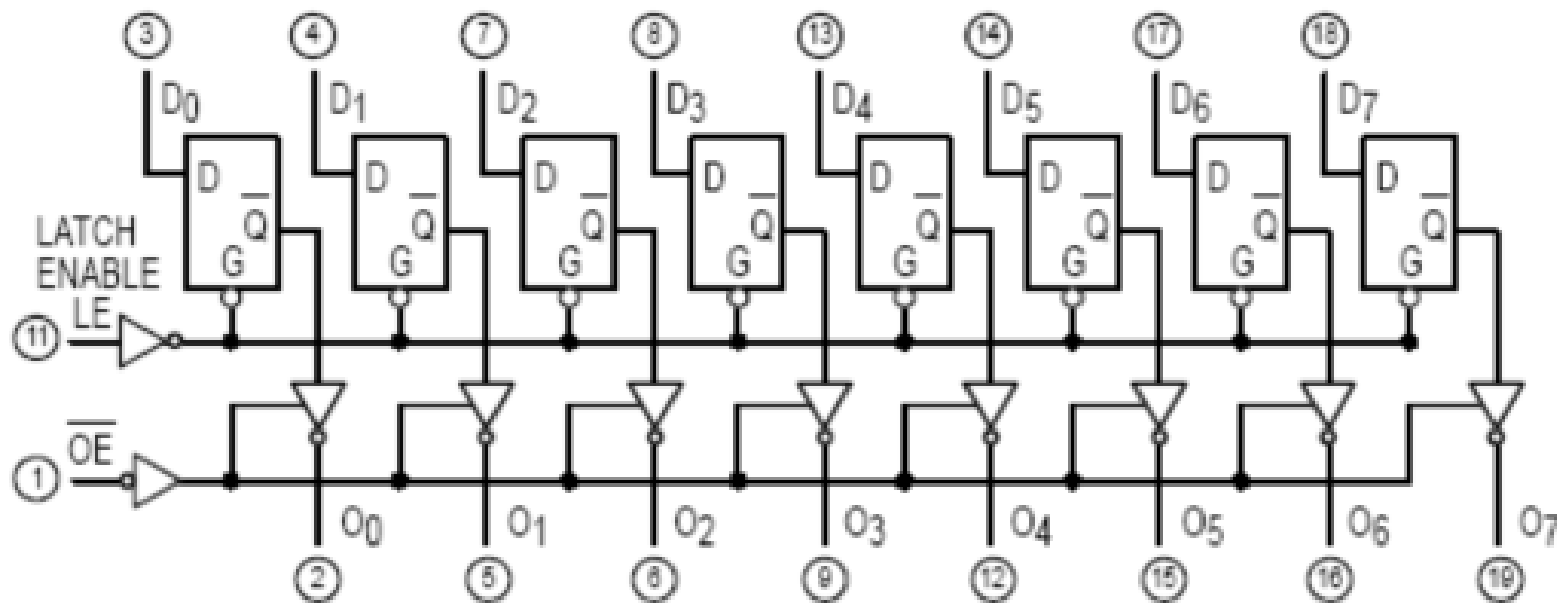
# 常用的输出锁存电路



OUT DX, AL

LE:	高电平	数据跟随
	下降沿	数据锁存
	低电平	数据不变化

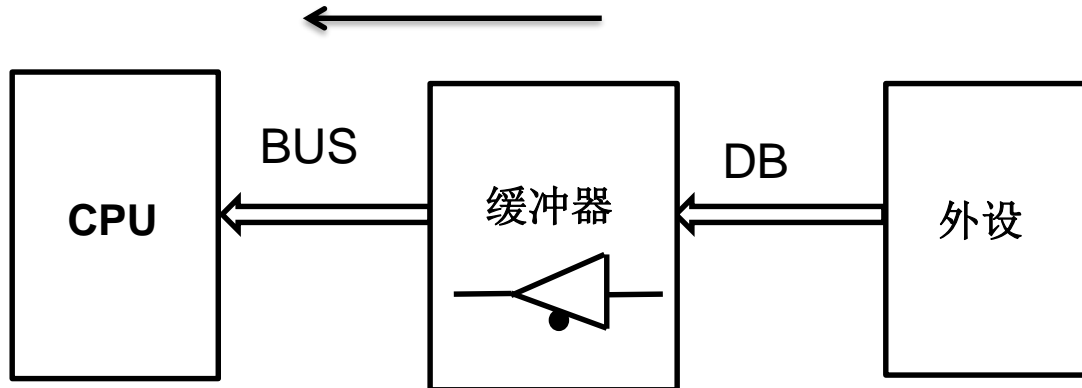
# ■ 锁存电路内部结构



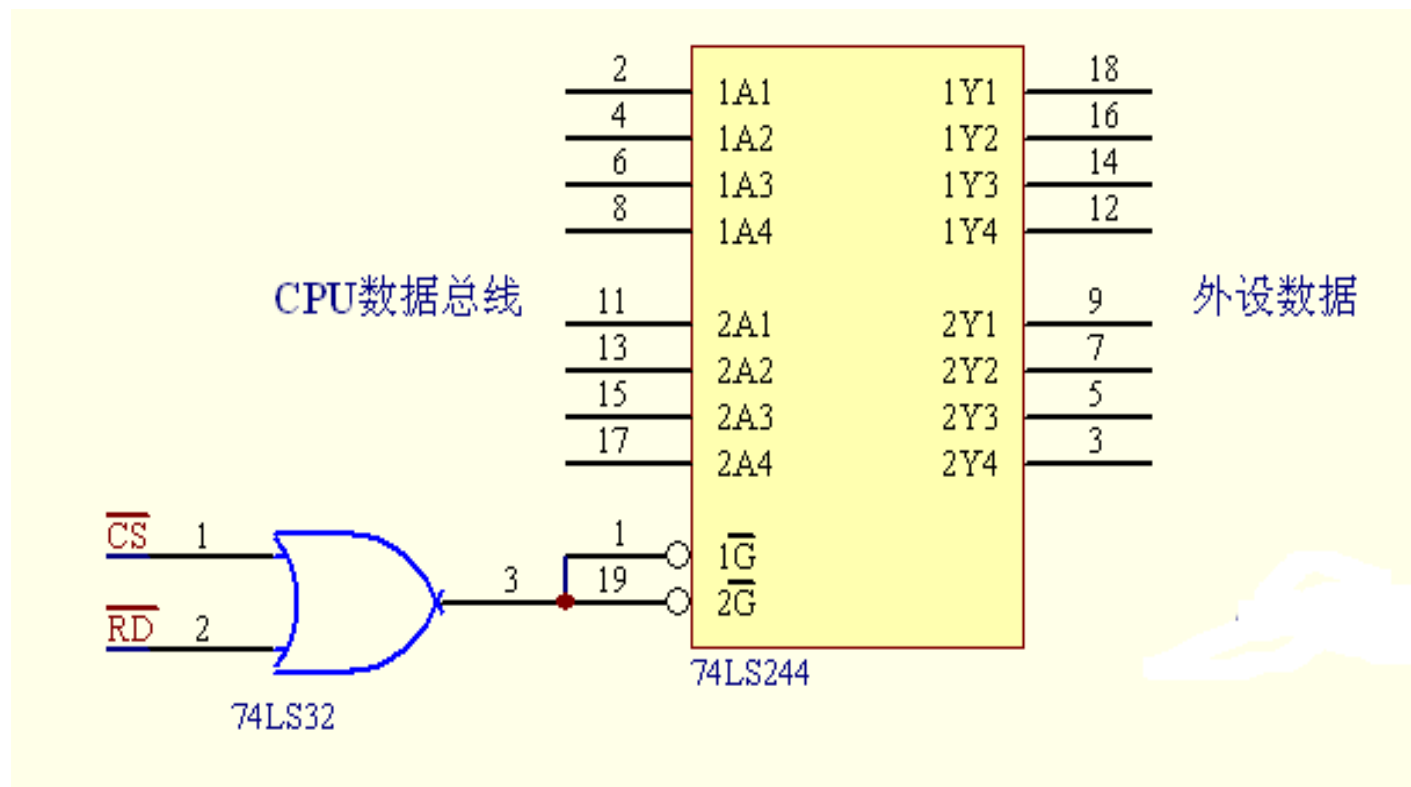
D 触发器.

## 2) CPU输入数据

IN AL, DX



## ■ 常用输入缓冲电路



IN AL, DX

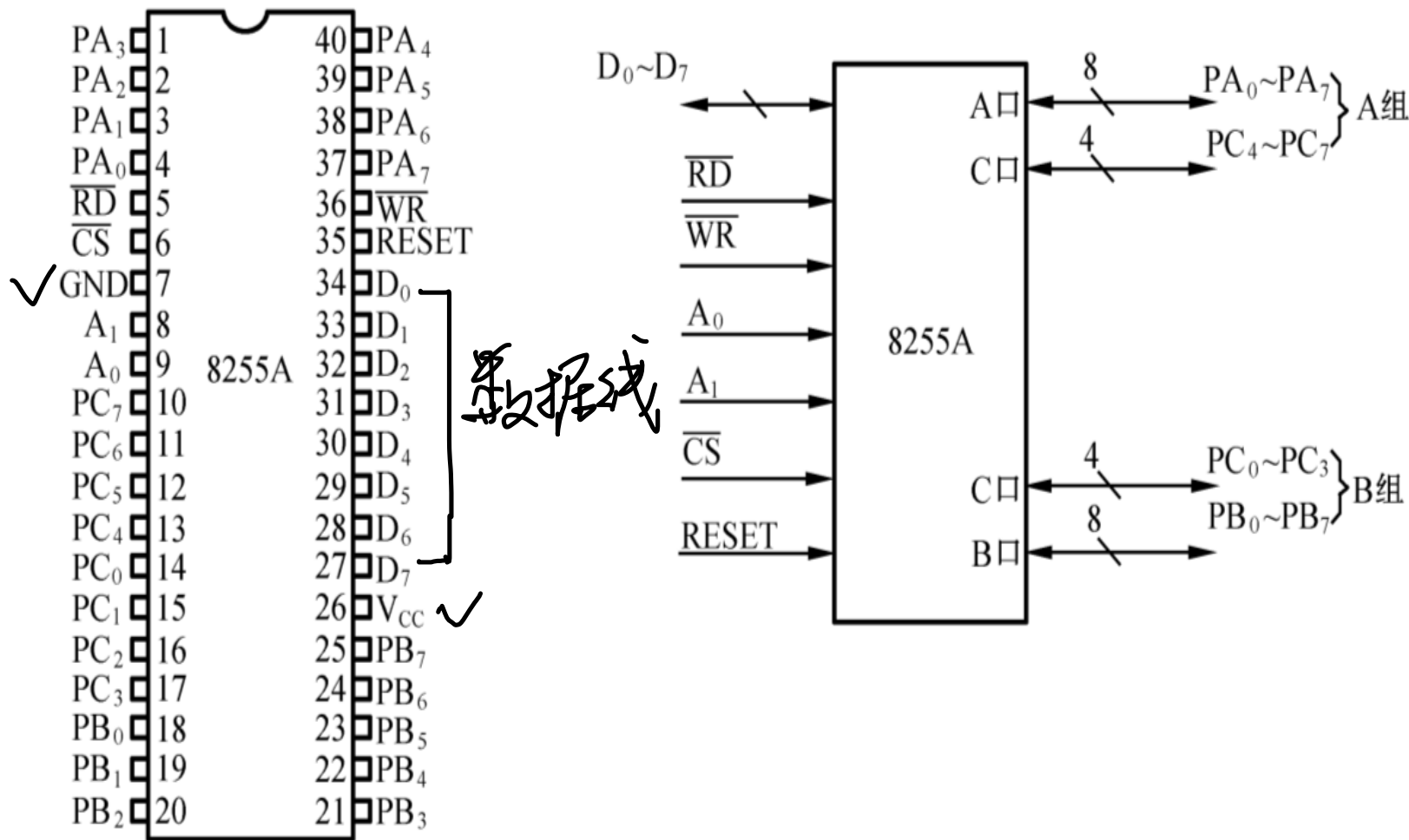
- 所有和计算机总线连接的外设，输入设备需要有输入缓冲的控制功能，输出设备需要有数据接收的锁存功能。

IN AL, PORT ; 打开对应I/O输入缓冲读入数据

OUT PORT, AL ; CPU将输出数据锁存到对应I/O

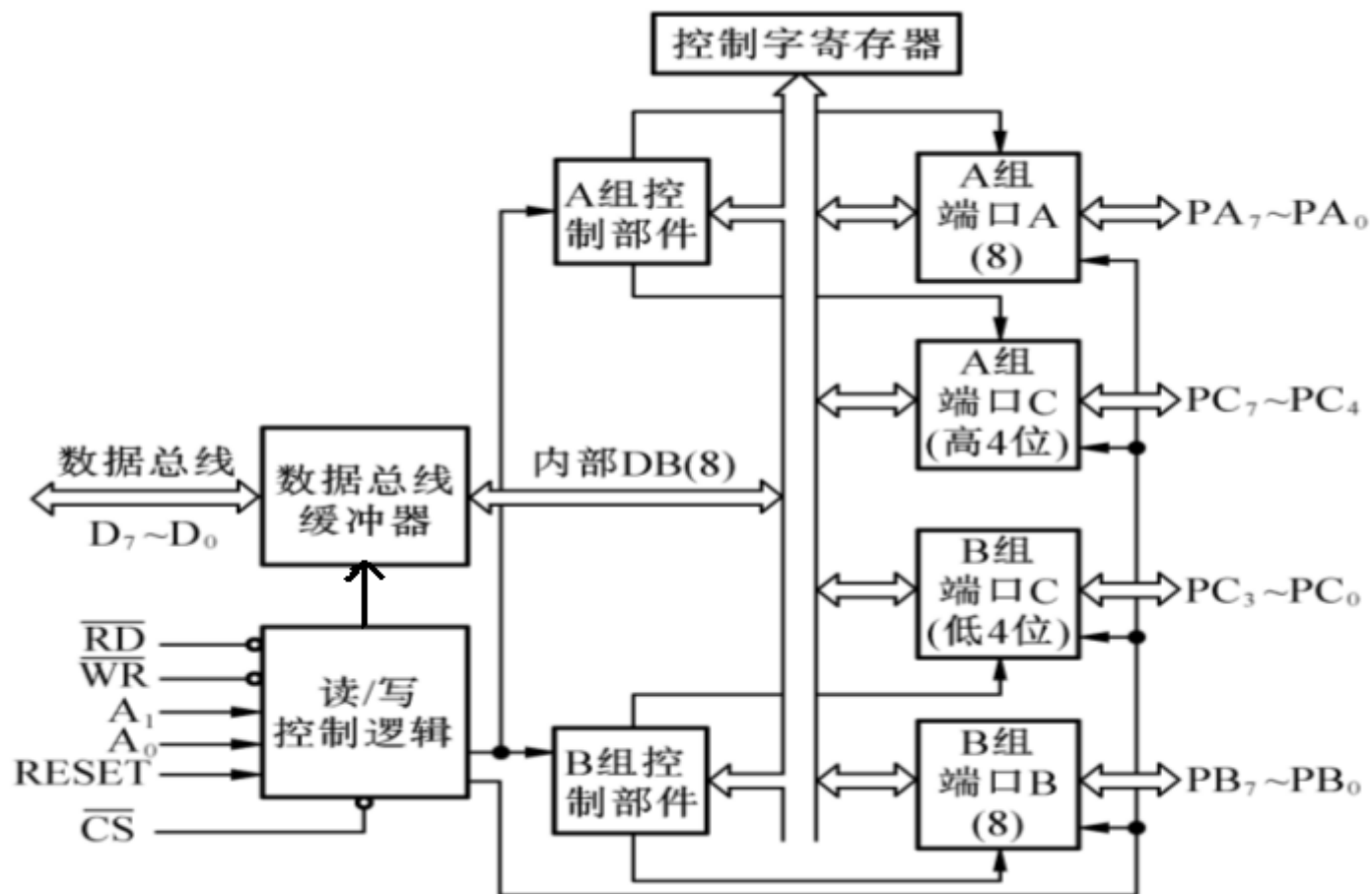
# 2. 认识8255

## 1) 管脚



A0	A1	$\overline{CS}$	操作端口
0	0	0	A口 40H
1	0	0	B口 41H
0	1	0	C口 42H
1	1	0	控制口 43H.

## 2) 内部结构



### 3) 8255的作用和现状

(1) CPU通过8255可以和外设做双向数据交互

(2) 当时INTEL8086/8088CPU组成的计算机系统，用8255来管理外设。

- 继电器，指示灯，简单开关量等 方式0 ✓
- 解码键盘，CRT显示器，打印机等 方式1 ✓
- 软磁盘 方式2

(3) 在很长一段时间内，**8255**还可以在单片机系统里用作拓展**IO**端口。

(4) **8255**已经很少使用，其功能基本被单片机的**GPIO**，以及集成芯片新技术新方法所代替。

## 4) 8255的工作方式

### 方式0 基本输入输出方式

典型代表：继电器，指示灯，输入报警开关量，数码管等

- 无条件输入输出方式
- A口 B口 C口都可以独立工作在方式0

### 方式1 选通输入输出方式

典型代表：解码键盘，CRT显示器，打印机等

- 需要握手信号线
- A口 B口可工作在方式1，需要C口配合

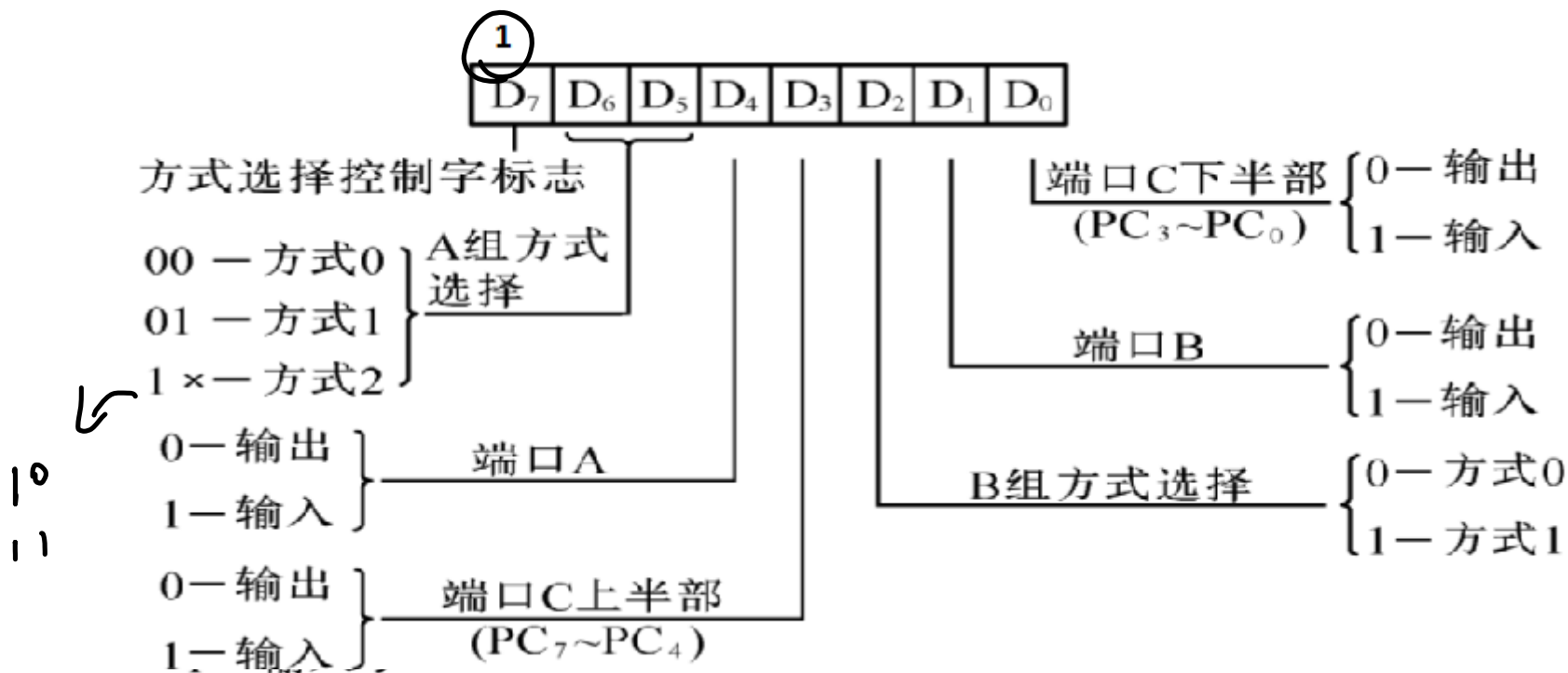
## 方式2 双向选通传送

典型代表： 软磁盘

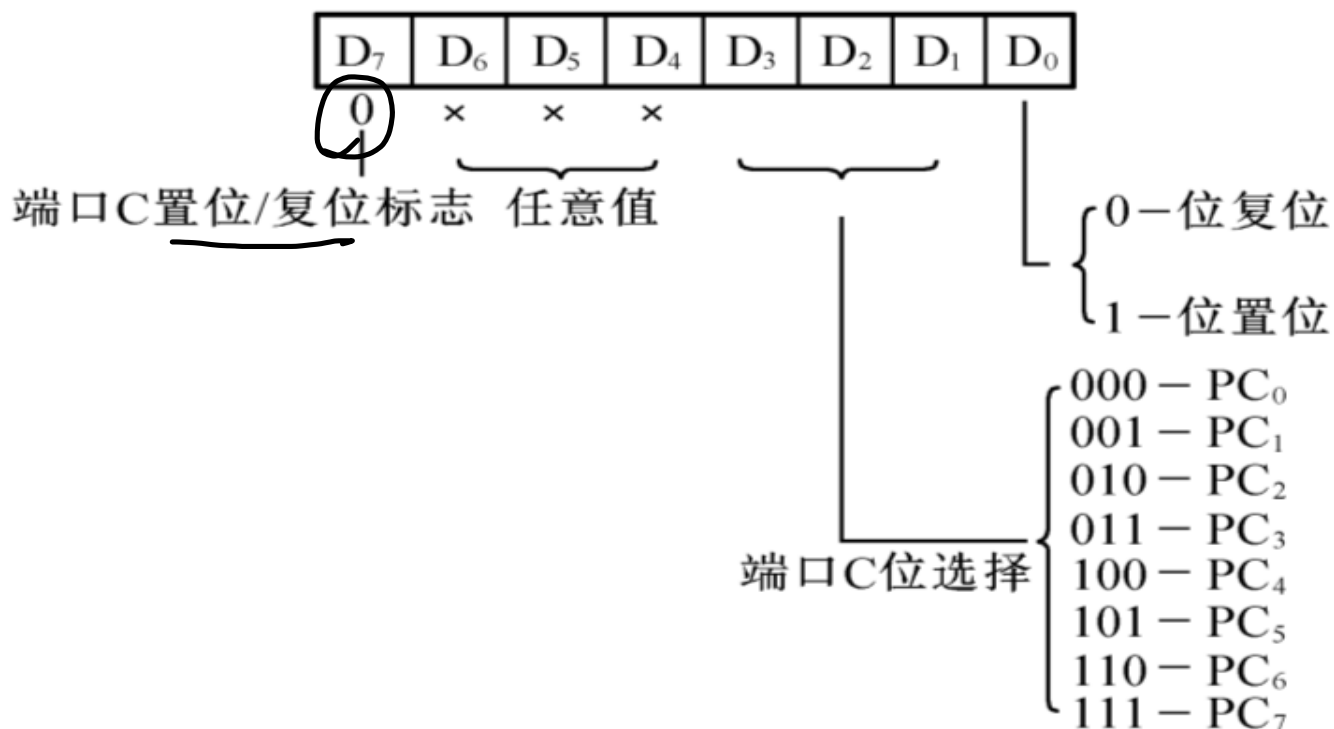
- 需要两套握手信号线
- 只有A口可以工作在方式2，需要C口配合

# 3. 8255初始化

## 1) 8255控制字格式



## 2) 端口C置位、复位（单独一个端口操作）

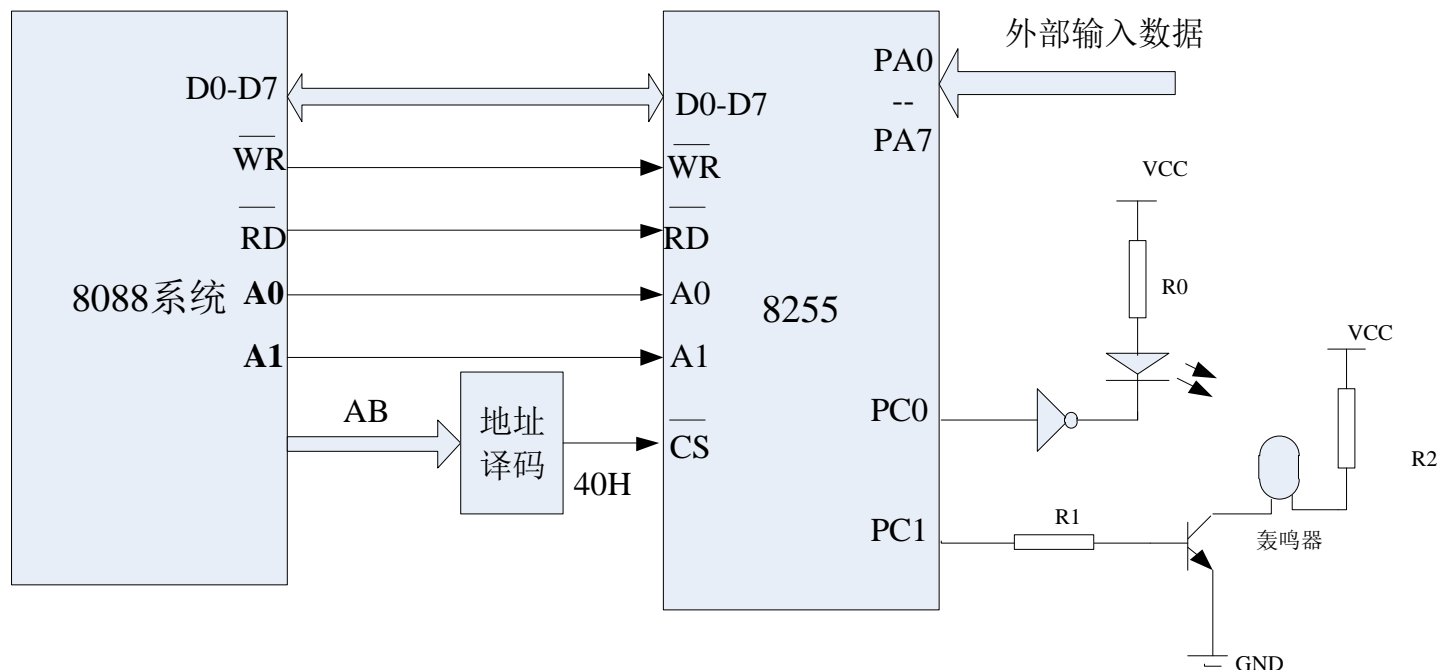


虽然是对C口的位操作但要写入控制端口

## 7.1.2. 8255应用（方式0）

### 1.一般外设的管理控制（无条件）

例1 系统通过8255的A口读入数据，如果数据 > 100, 实现声光报警，否则停止报警。编写功能程序。



- 端口地址为 40H-43H  
ABC 挖、

- 程序功能段

MOV AL, 10010000B; 工作方式设定

OUT 43H, AL  
→ 控制口.

MAIN:

IN AL, 40H ; A口读入数据

CMP AL, 100

JA ALARM\_DEAL

MOV AL, 00H ;

OUT 42H<sup>C</sup>, AL ; 使PC0=0和PC1=0

JMP OUTT

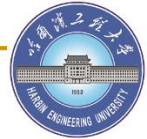
ALARM\_DEAL: ; 报警处理

MOV AL, 03H ; 使PC0=1和PC1=1

OUT 42H, AL

OUTT:

JMP MAIN ; 跳转循环监控



MOV AL, 00H 0000H.

OUT 43H ,AL ; 使PC0=0

MOV AL, 02H 0010H.

OUT 43H ,AL ; 使PC1=0

JMP OUTT

ALARM\_DEAL: ; 报警处理

MOV AL ,01H ; 使PC0=1

OUT 43H,AL

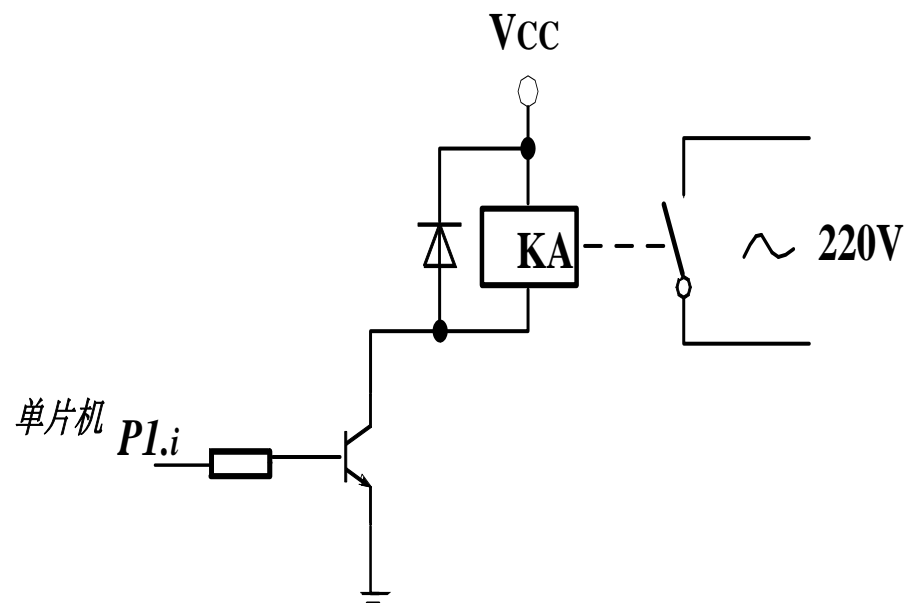
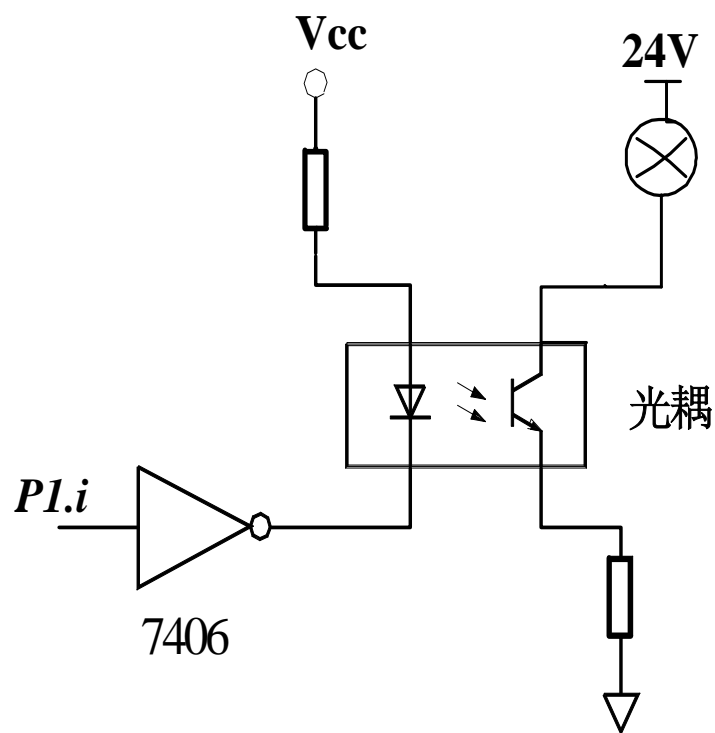
MOV AL ,03H ; 使PC1=1

OUT 43H,AL

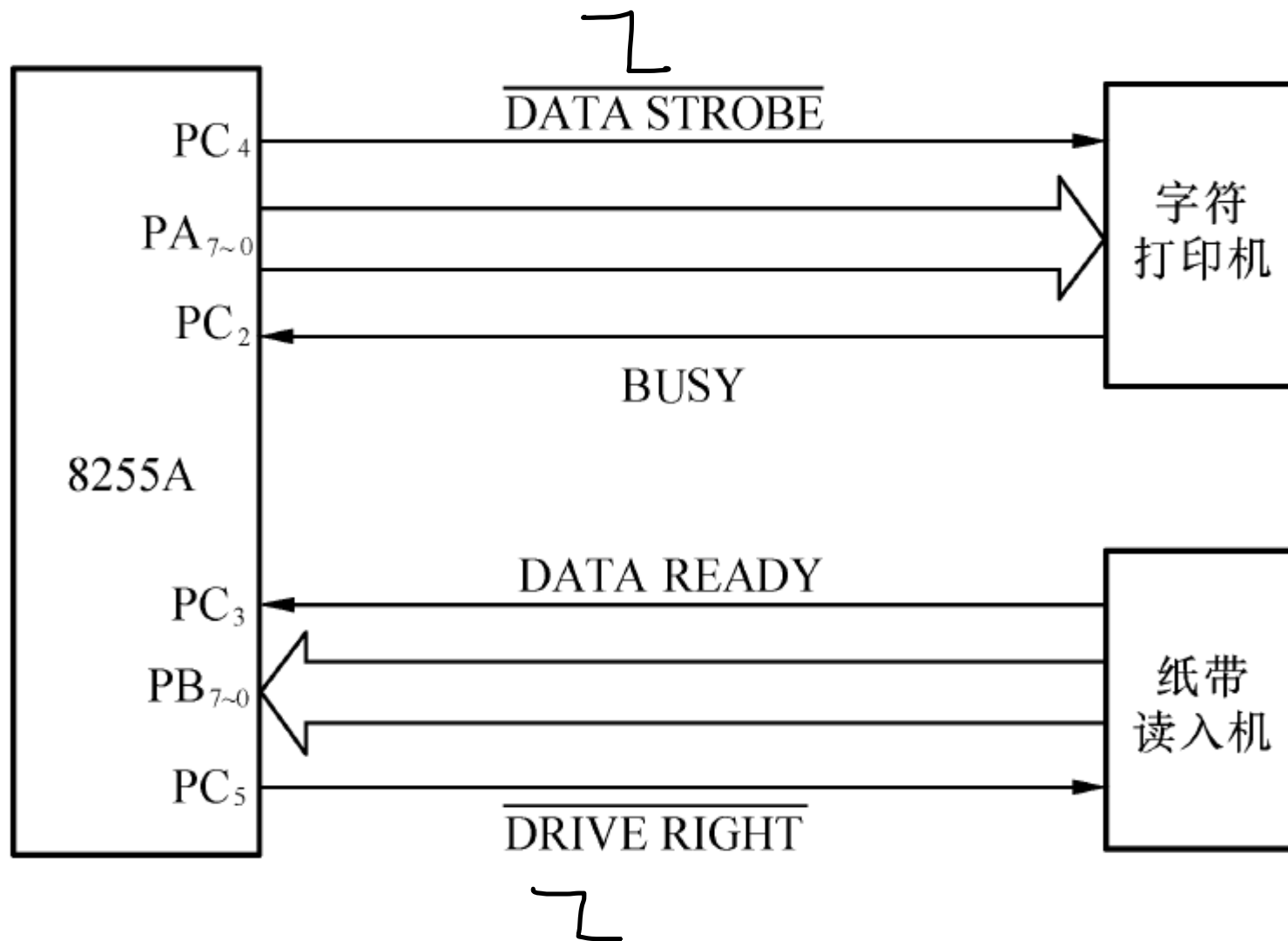
OUTT:

JMP MAIN

# 强电灯和继电器的控制




## 2.管理打印机和读入机（需要握手信号）



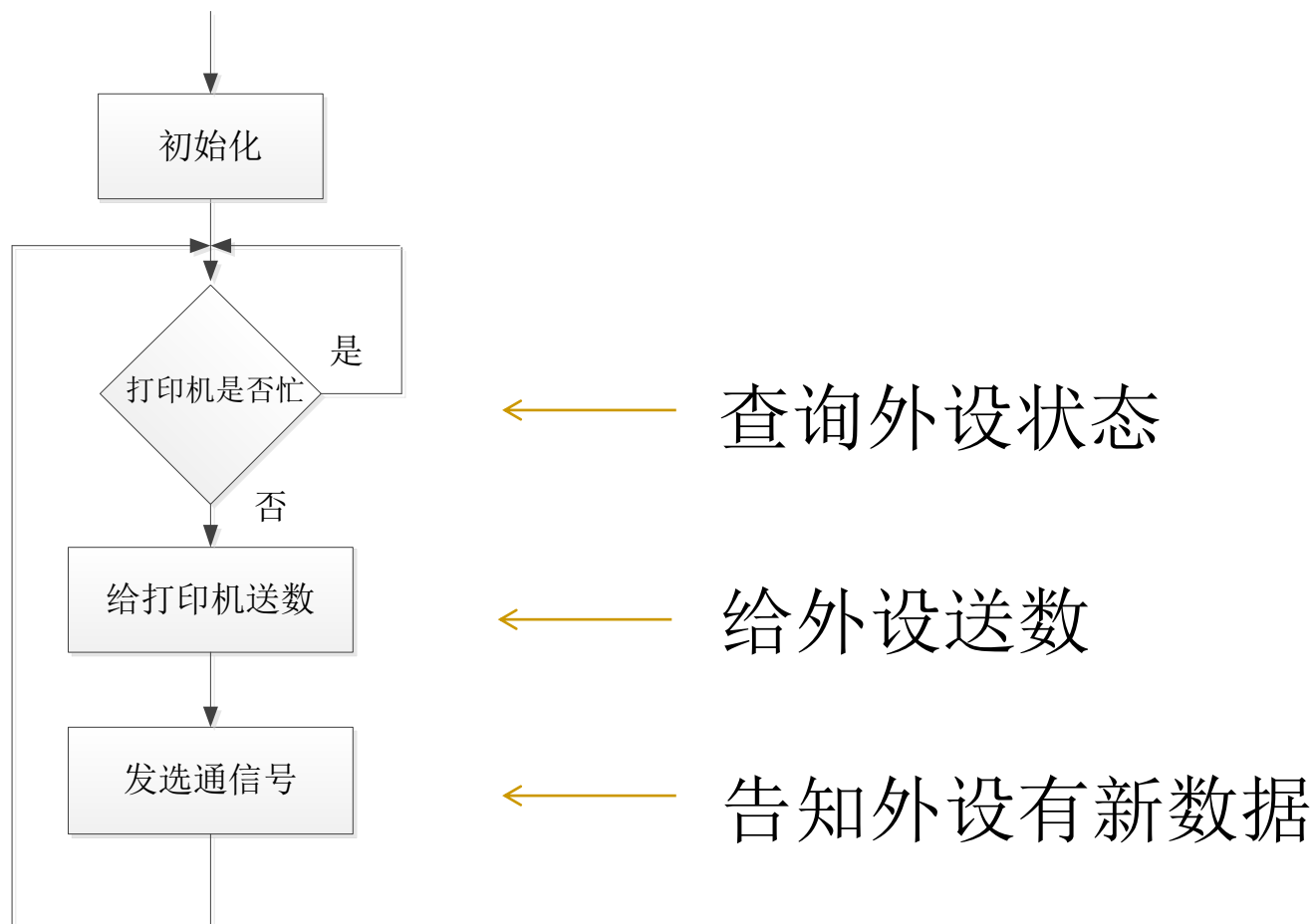
# 联络信号（握手信号）的形式

高电平 

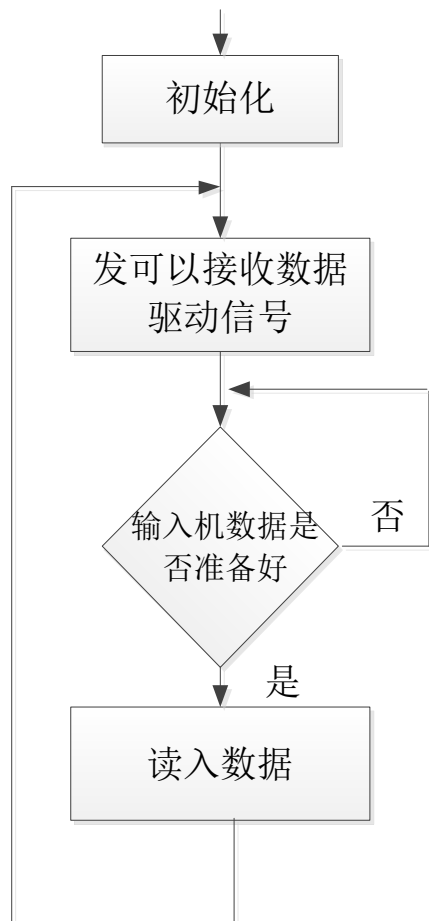
上升沿 

低电平 

下降沿 



## 计算机和输出设备（打印机）交互流程

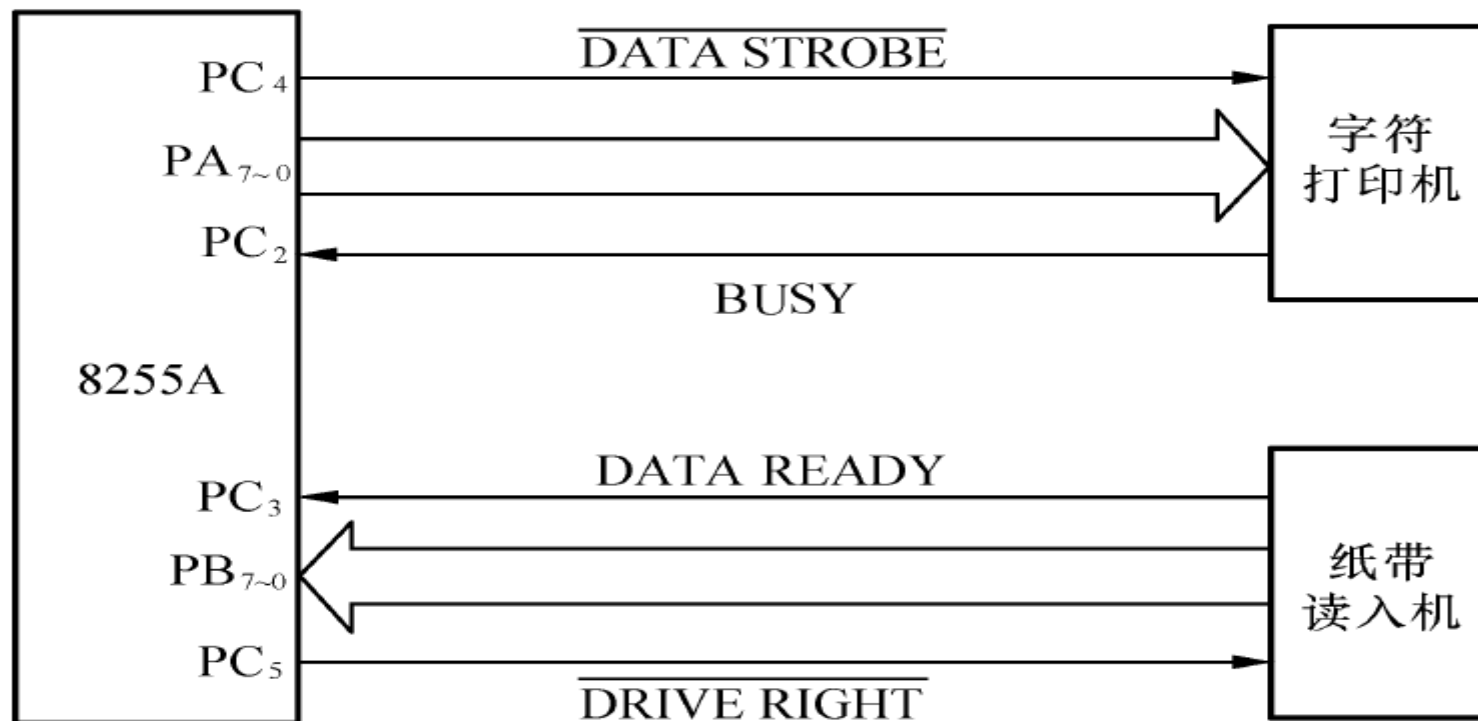


← 告知外设我可以

← 查询数据状态

← 获取新数据

## 计算机和输入设备（读入机）交互流程



信号约定:

打印机忙为高电平，空闲为低电平；

打印机位选通信号为下降沿；

读入机驱动信号为下降沿；

读入机数据准备好高电平，未准备好为低电平。



8255端口地址 E0H ,E2H ,E4H,E6H

A B C 口

初始化工作

MOV AL, 83H; A输出, B输入, C低4入  
; C口高4出

OUT 0E6H, AL

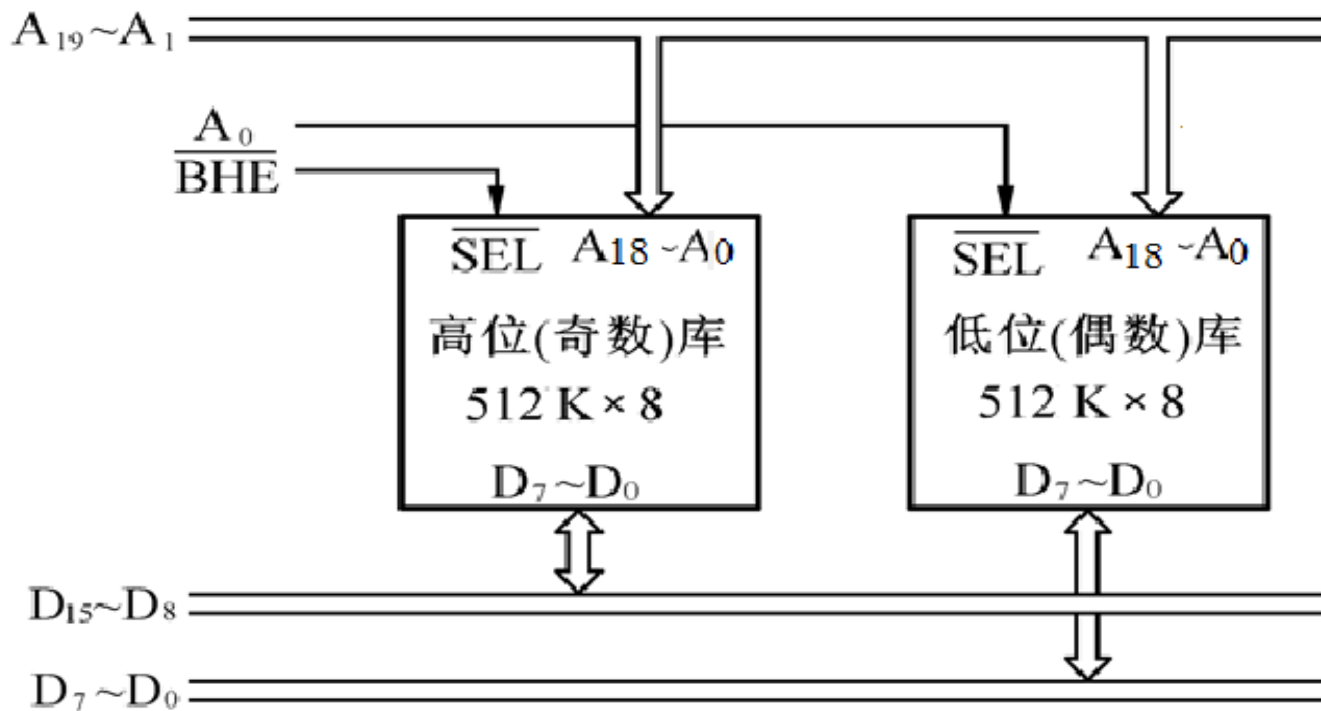
MOV AL, 09H; 选通信号PC4=1, 无效 (高电平无效)

OUT 0E6H, AL

MOV AL, 0BH; 驱动信号PC5=1, 无效

OUT 0E6H, AL

# 为什么不连续的偶地址



## I/O ADDRESSING

In the 8086, I/O operations can address up to a maximum of 64K I/O byte registers or 32K I/O word registers. The I/O address appears in the same format as the memory address on bus lines  $A_{15} \sim A_0$ . The address lines  $A_{19} \sim A_{16}$  are zero in I/O operations. The variable I/O instructions which use register DX as a pointer have full address capability while the direct I/O instructions directly address one or two of the 256 I/O byte locations in page 0 of the I/O address space.

I/O ports are addressed in the same manner as memory locations. Even addressed bytes are transferred on the  $D_7 \sim D_0$  bus lines and odd addressed bytes on  $D_{15} \sim D_8$ . Care must be taken to assure that each register within an 8-bit peripheral located on the lower portion of the bus be addressed as even.



## 打印机驱动程序段:

偶地址.

00000100 B.

LPST: IN AL, 0E4H

TEST AL, 04H; 测试PC2的BUSY状态

JNZ LPST; 忙就等待

AL  
00000100

MOV AL, PR\_DATA; 打印数据

OUT 0E0H, AL

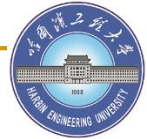
MOV AL, 08H; 选通信号PC4=0

OUT 0E6H, AL

MOV AL, 09H; 选通信号PC4=1

OUT 0E6H, AL

○ ○ ○ ○



读入机驱动程序段：

RDDV : MOV AL, 0AH

OUT 0E6H, AL; 驱动信号PC5=0

RDST: IN AL, 0E4H ; 忙就等待

TEST AL, 08H<sup>00001000</sup> ; PC3=1数据准备好?

JZ RDST → PC<sub>3</sub>=0 数据没准备好

IN AL, 0E2H

MOV REV\_DATA, AL; 保存数据

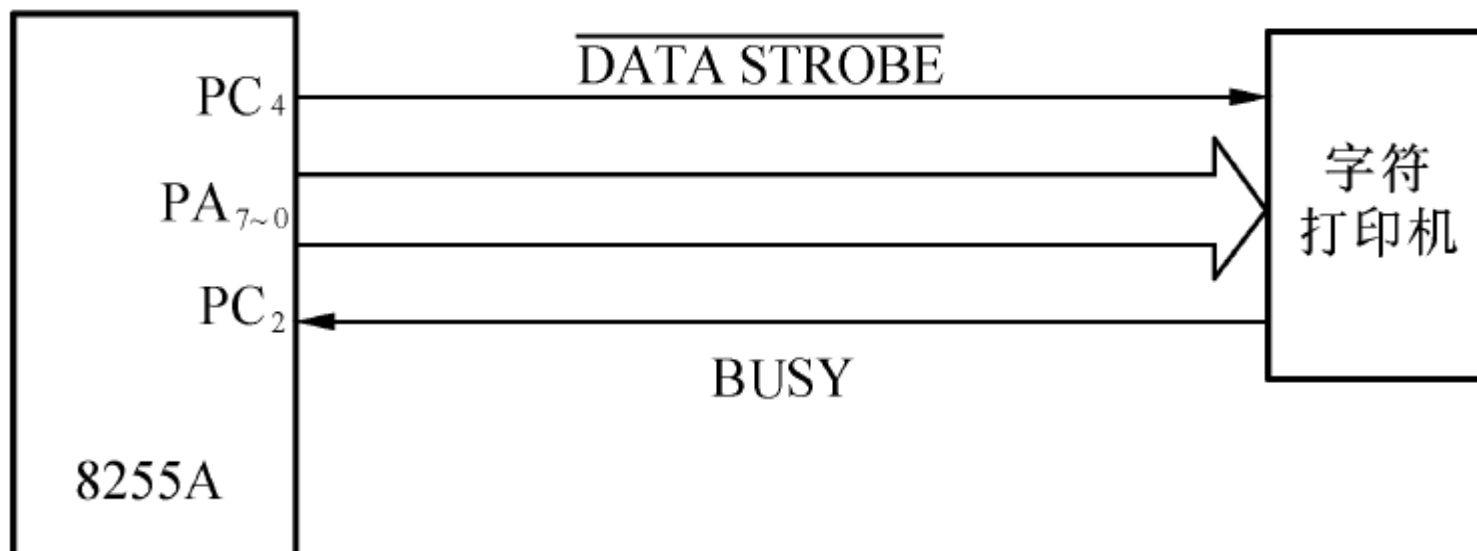
MOV AL, 0BH ; 断开驱动PC5=1

OUT 0E6H, AL

○ ○ ○

例2 内存中LP\_DATA有10个字符，它们被其他程序更新时就需要打印，否则不用打印。

（端口地址： E0H, E2H, E4H, E6）





DATA SEGMENT

LP\_DATA DB X1,...X10

CNT EQU \$-LP\_DATA

FLAG\_DTAT\_CHG DB 0 ; 数据被更新的标志位

DATA ENDS

CODE SEGMENT

ASSUME CS:CODE ,DS:DATA

GO: MOV AX , DATA

MOV DS , AX

```
MOV    AL, 81H; A输出, C低四入, C口高四出
OUT     0E6H, AL
MOV     AL, 09H ;使得选通信号为高电平
OUT     0E6H, AL
```

MAIN:

○ ○ ○

```
MOV     FLAG_DTAT_CHG, 1;更新数据
```



```
CMP FLAG_DTAT_CHG,1
```

```
JNZ MAIN  $\Rightarrow$  FLAG = 0
```

```
MOV FLAG_DTAT_CHG,0
```

```
MOV CX, CNT
```

```
MOV BX, OFFSET LP_DATA
```

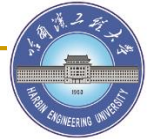
LPST :

```
IN AL, 0E4H  $\{PC2$ 
```

```
TEST AL, 04H; 测试PC2的BUSY状态
```

```
JNZ LPST ; 忙就等待
```

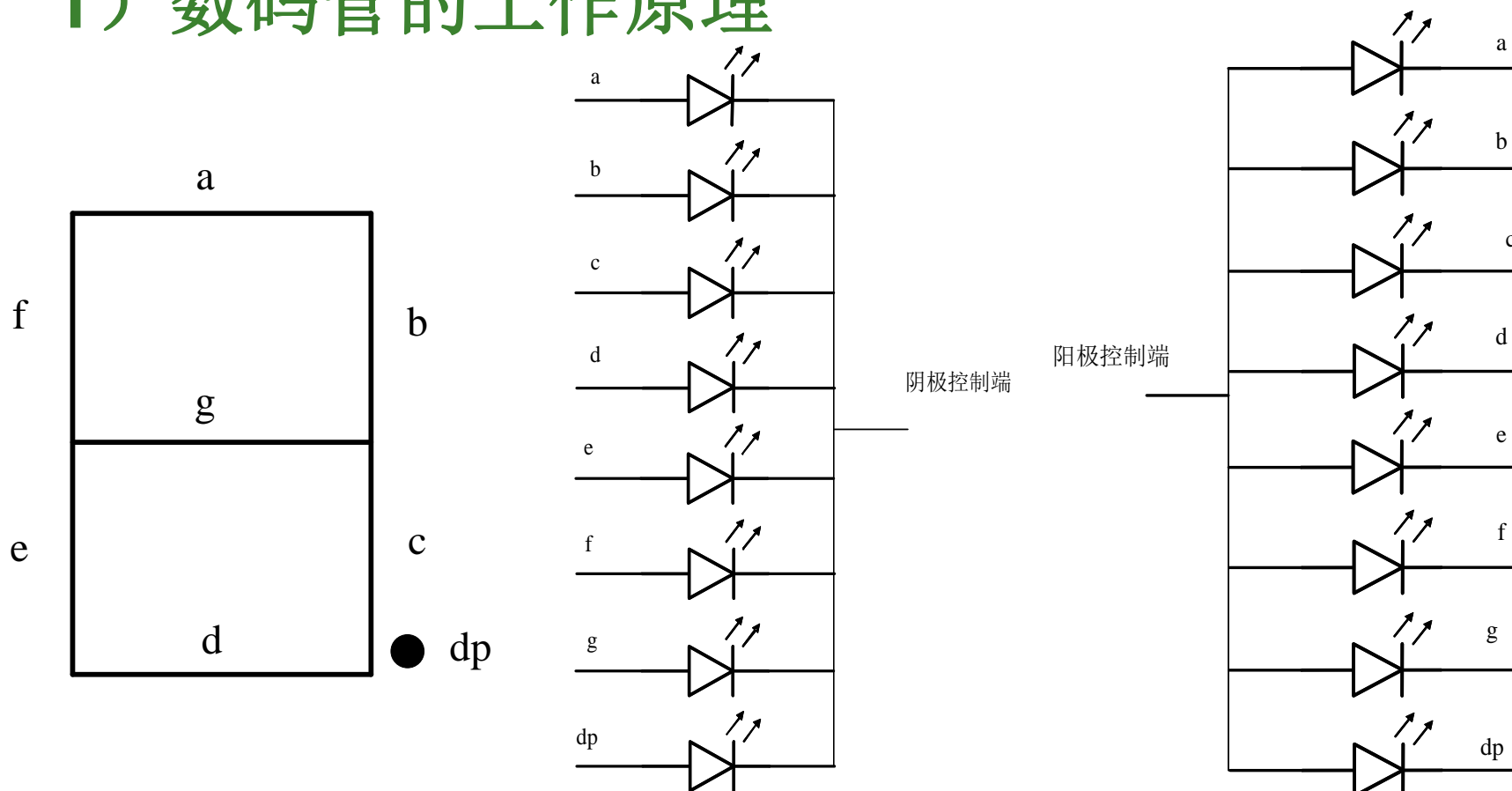
```
MOV AL, [BX]
```



```
OUT 0E0H , AL ; A口给打印机送数据
MOV AL ,08H ; 产生选通信号PC4
OUT 0E6H,AL
MOV AL ,09H
OUT 0E6H ,AL
INC BX ; 指向下一个打印数据
LOOP LPST ; 循环打印10个数据
JMP MAIN ; 跳转主监控
CODE ENDS
END GO
```

# 3.人机交互---数码管

## 1) 数码管的工作原理



数码管及内部原理图

# 显示设备数码管和 **LCD** 对比

## 数码管：

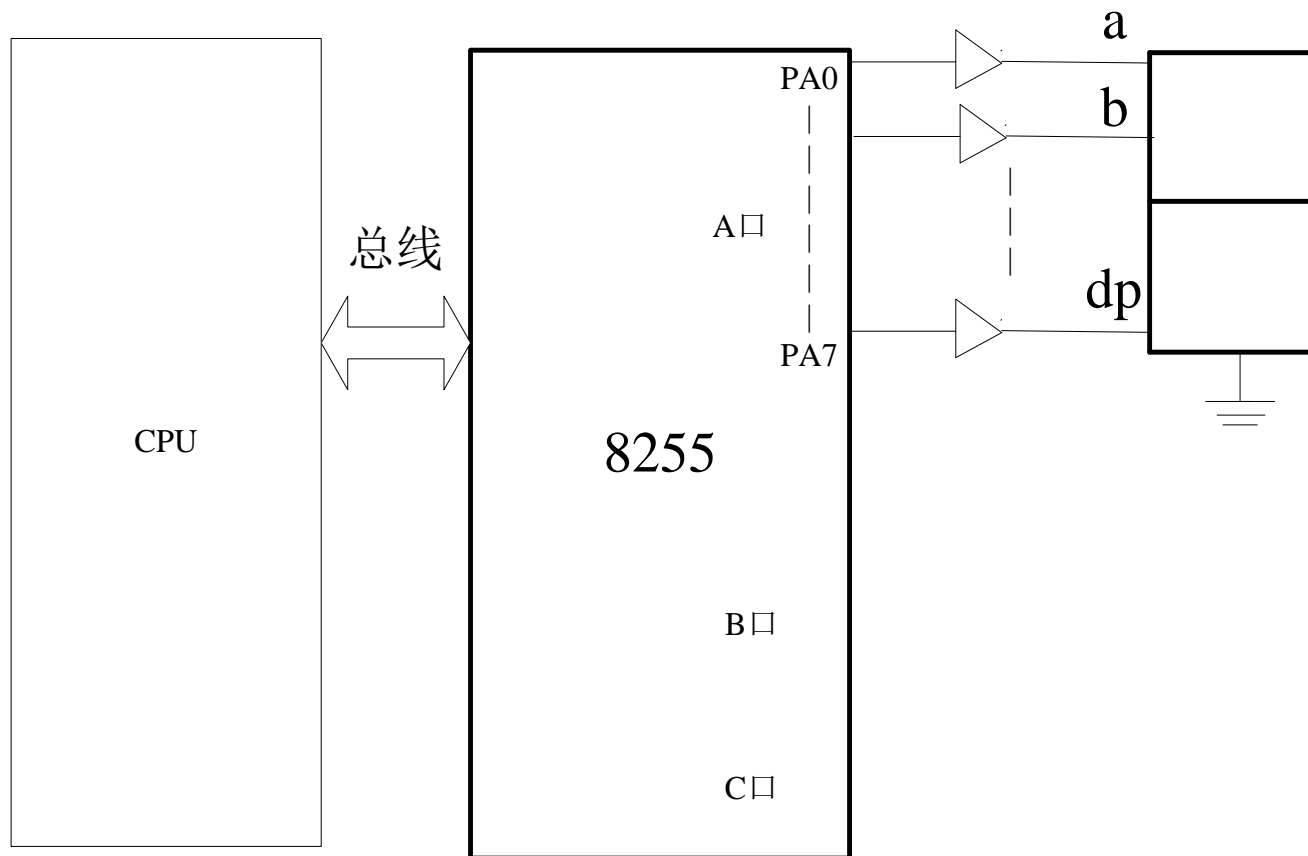
- 结构简单，抗干扰，辨识度高
- 只能显示数字，功耗比较大

## **LCD:**

- 显示内容丰富，功耗低
- 结构复杂，容易受干扰，强光下辨识低

	D7	D6	D5	D4	D3	D2	D1	D0	共阴极
	Dp	g	f	e	d	c	b	a	七段码
0	0	0	1	1	1	1	1	1	3FH
1	0	0	0	0	0	1	1	0	06H
2	0	1	0	1	1	0	1	1	5BH
3	0	1	0	0	1	1	1	1	4FH
4	0	1	1	0	0	1	1	0	66H
5	0	1	1	0	1	1	0	1	6DH
6	0	1	1	1	1	1	0	1	7DH
7	0	0	0	0	0	1	1	1	07H
8	0	1	1	1	1	1	1	1	7FH
9	0	1	1	0	1	1	1	1	6FH

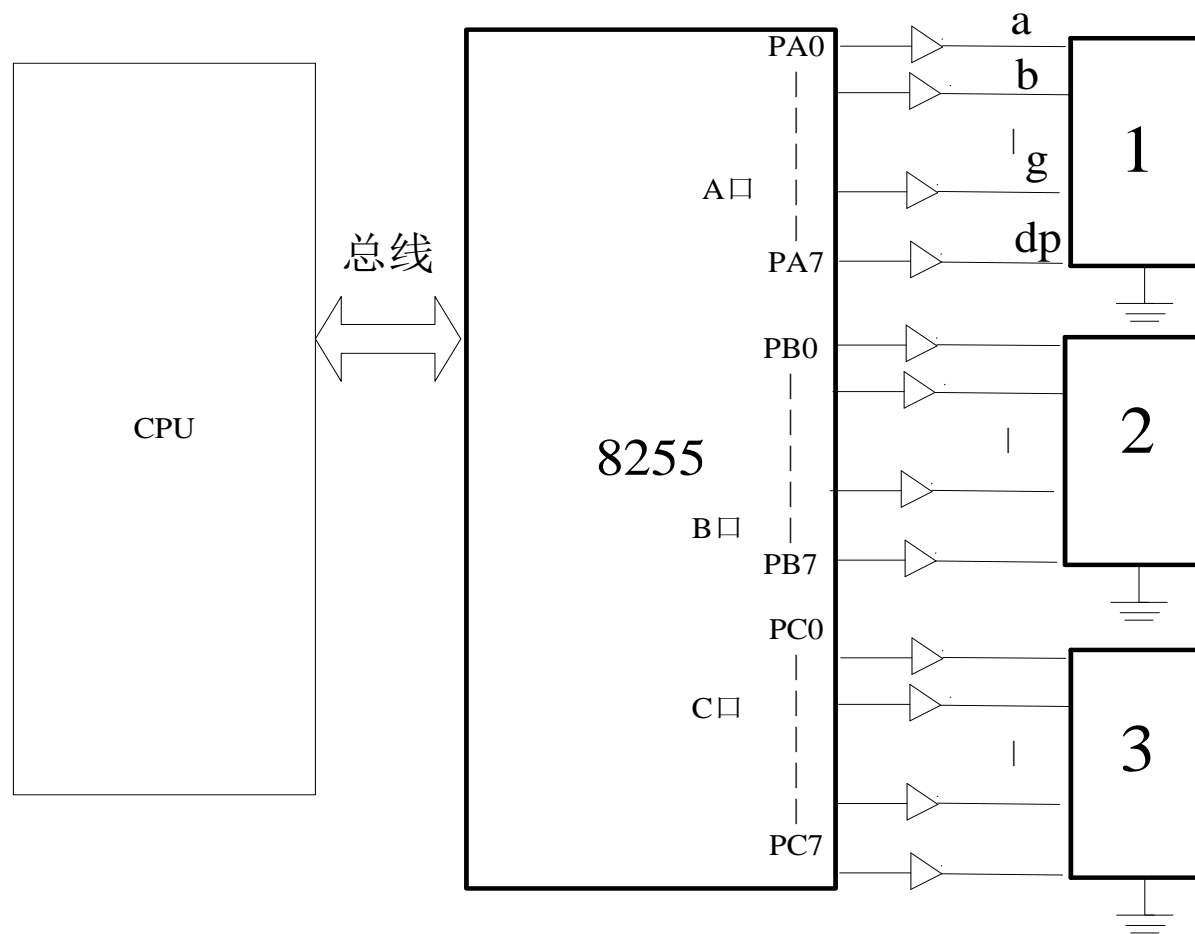
## 2) 单个数码管的管理

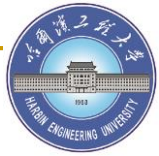


8255管理单个数码管原理图

# 3) 多个数码管管理

## (1) 多个数码管的静态显示





8255端口为:PORT\_A, PORT\_B, PORT\_C,  
PORT\_CTRL 都小于255。

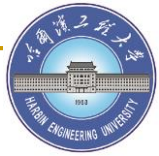
DATA SEGMENT

```
TABLE_SEG DB 3FH,06H,5BH,4FH,66H,6DH,  
            DB 7DH,07H,7FH,6FH ; 0-9七段码  
GEI DB 3 ; 最低位  
SHI DB 2  
BAI DB 1 ; 最高位
```

DATA ENDS

CODE SEGMENT

```
    ASSUME CS: CODE , DS: DATA  
GO:  MOV AX, DATA  
     MOV DS, AX
```



MOV AL, 80H ; A、B、C方式0, 输出  
OUT PORT\_CTRL, AL

MAIN: . . . ; 更新显示数据的功能段  
MOV BX, OFFSET TABLE\_SEG; 取七段码表头地址  
MOV AL, GEI  
MOV AH, 0  
ADD BX, AX  
MOV AL, [BX] ; 取最低位七段码  
OUT PORT\_A, AL  
MOV BX, OFFSET TABLE\_SEG  
MOV AL, SHI  
MOV AH, 0  
ADD BX, AX  
MOV AL, [BX]  
OUT PORT\_B, AL



MOV BX, OFFSET TABLE\_SEG; 取七段码表头地址

MOV AL, BAI

MOV AH, 0

ADD BX, AX

MOV AL, [BX] ; 取最高位七段码

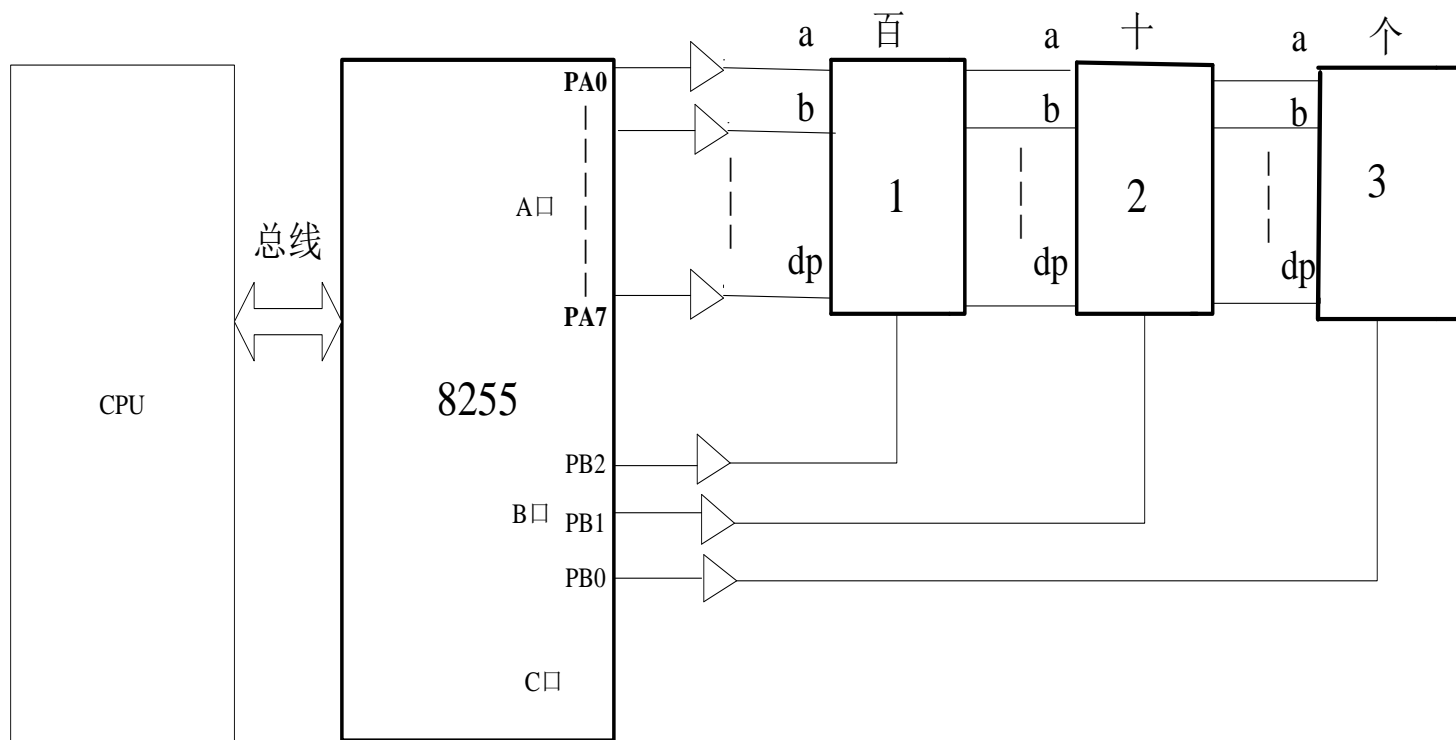
OUT PORT\_C , AL

JMP MAIN

CODE ENDS

END

## (2) 多个数码管的动态显示



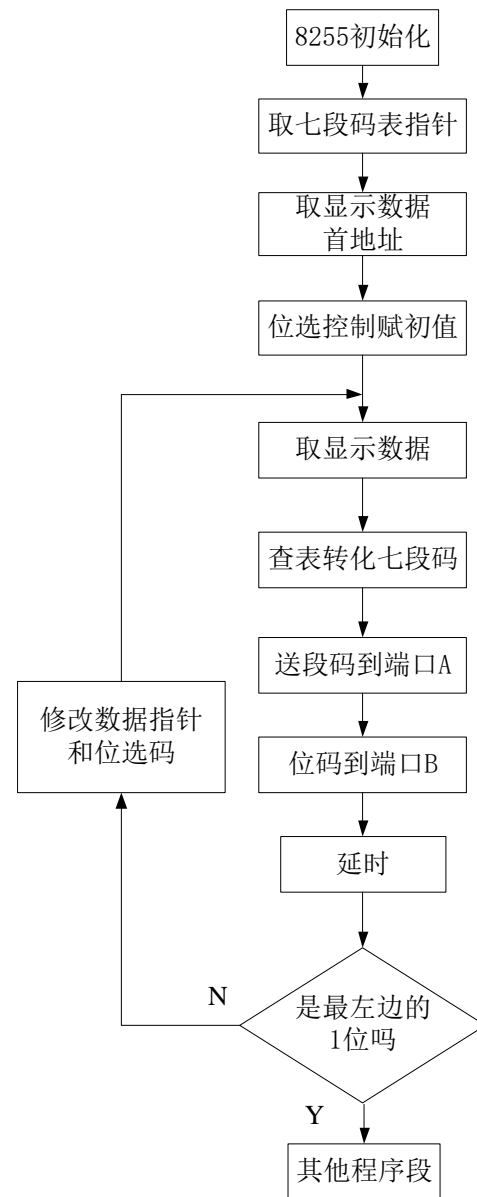
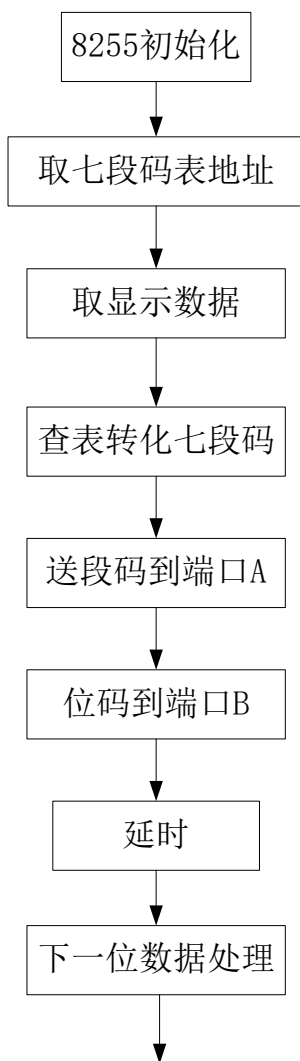
8255管理多个数码管动态显示连接示意图

## ■ 数码管动态显示原理：

一个物体在人眼滞留的时间为**0.1S-0.4S**。利用人眼对物体滞留的特点，在这个时间内快速让不同的数码管显示自己对应的数字一遍，而让人感觉数码管在连续显示各自的数字。

## ■ 数码管动态显示程序主要设计步骤：

- 1) 要显示的数据查表得到对应七段码；
- 2) 段码从**8255**的端口**A**输出；
- 3) 对应的位选为低，其他位为高；
- 4) 延时一段时间；
- 5) 下一次循环。



数码管动态刷顺序流程图

数码管动态刷新程序循环流程图



## ■ 三位数码管动态显示程序（顺序结构）

8255端口为:PORT\_A, PORT\_B, PORT\_C,  
PORT\_CTRL 都小于255。

DATA SEGMENT

```
TABLE_SEG DB 3FH,06H,5BH,4FH,66H,6DH,  
            DB 7DH,07H,7FH,6FH ; 0-9七段码  
GEI DB 3  
SHI DB 2  
BAI DB 1
```

DATA ENDS

CODE SEGMENT

```
        ASSUME CS: CODE , DS: DATA  
GO:     MOV  AX, DATA  
        MOV  DS, AX
```



```
MOV AL, 80H
OUT PORT_CTRL, AL
```

MAIN: ..... ; 更新显示数据

```
MOV BX, OFFSET TABLE_SEG; 取七段码表头地址
```

```
MOV AL, GEI
```

```
XLAT ; 查表指令, 结果在AL中
```

```
OUT PORT_A, AL
```

```
MOV AL, 7EH ; PB0=0
```

```
OUT PORT_B, AL
```

```
CALL DELAY ; 延时
```

```
MOV AL, SHI
```

```
XLAT
```

```
OUT PORT_A, AL
```

```
MOV AL, 7DH ; PB1=0
```

```
OUT PORT_B, AL
```

```
CALL DELAY
```

```
MOV AL, BAI
```

```
XLAT
OUT  PORT_A, AL
MOV  AL, 7BH    ; PB2=0
OUT  PORT_B, AL
CALL  DELAY
JMP  MAIN
```

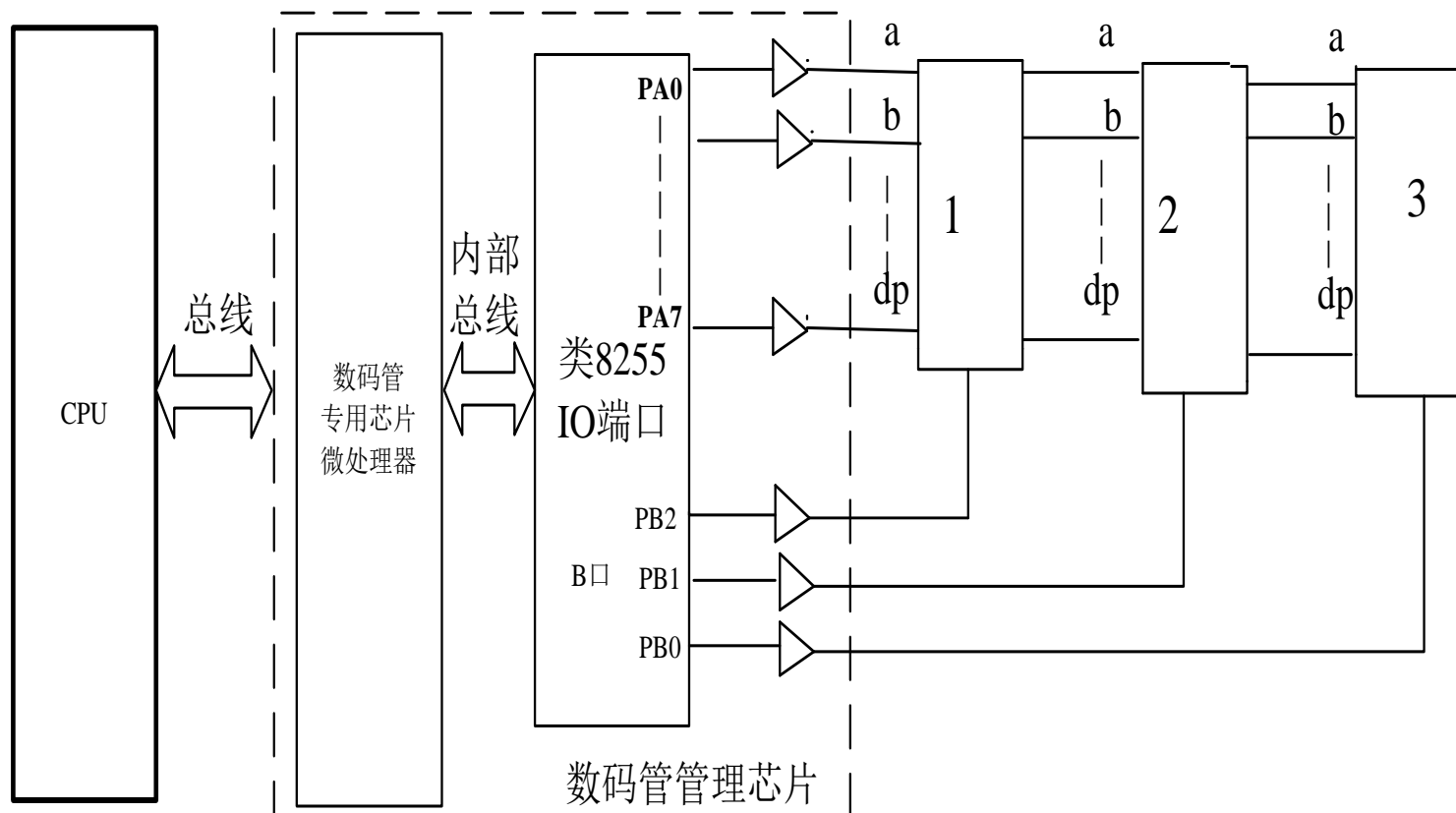
```
DELAY: PUSH  AX
      PUSH  CX
      MOV   CX, 0FFFFH
BACK1: MOV   AX, 0010H
ZERO1: DEC  AX
      JNZ   ZERO1
      LOOP  BACK1
      POP   AX
      POP   CX
      RET
```

; 延时程序约为0.34mS

1694个时钟  
主频 5MHz

```
CODE  ENDS
      END
```

# (3)专用芯片管理数码管



专用芯片管理数码管原理示意图

- 并行的数码管管理芯片**NEC8279**  
串行的数码管管理芯片**ZLG7289A**

## 三种管理数码管的方式对比

静态方式：硬件（IO端口）开销大  
占用**CPU**时间少

动态刷新：硬件（IO端口）开销少  
占用**CPU**时间多

专用芯片：硬件（IO端口）开销小  
占用**CPU**时间少



## 4. 人机交互---键盘

### ■ 编码键盘

键盘的**ASCII**码是直接由每个按键的数字电路产生的。电路复杂，成本高，重定义困难。

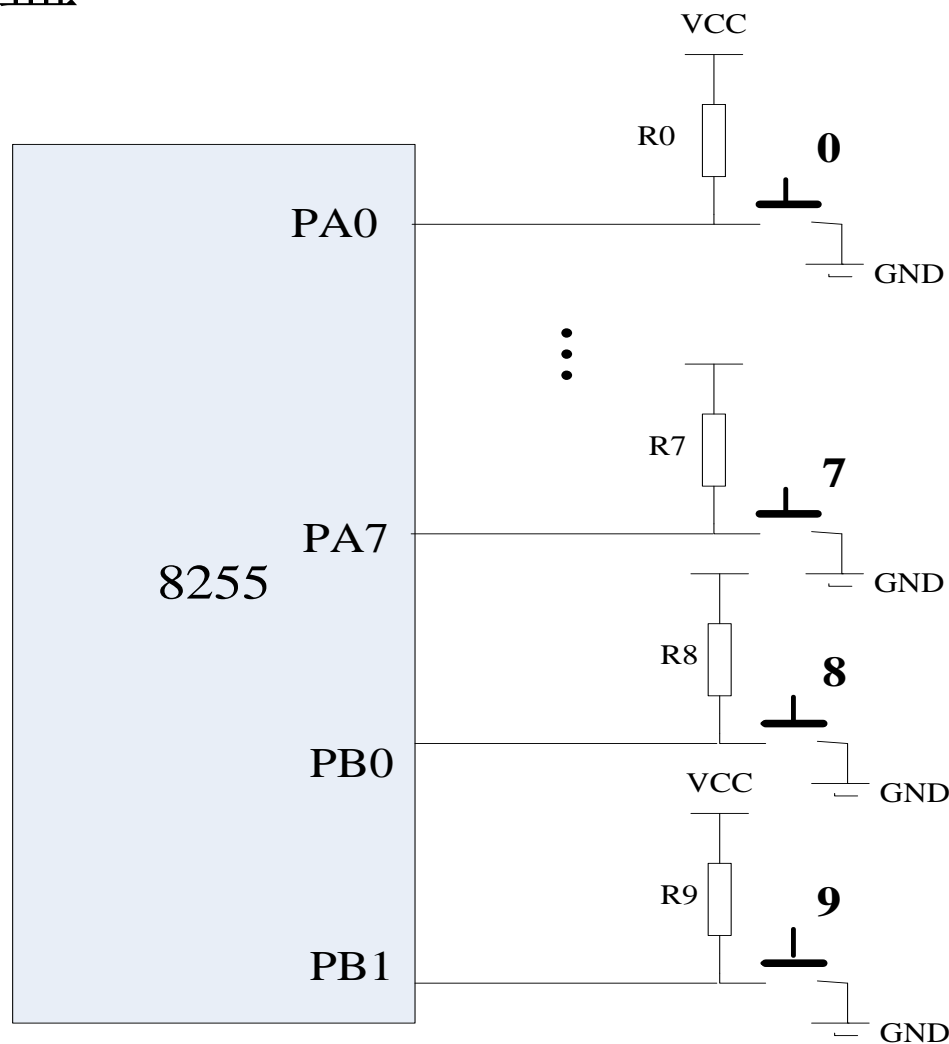
### ■ 非编码键盘

一些按键排列成行或行列式矩阵的形式，按键的接通或断开，在相应的程序配合下可产生被按下的键码。

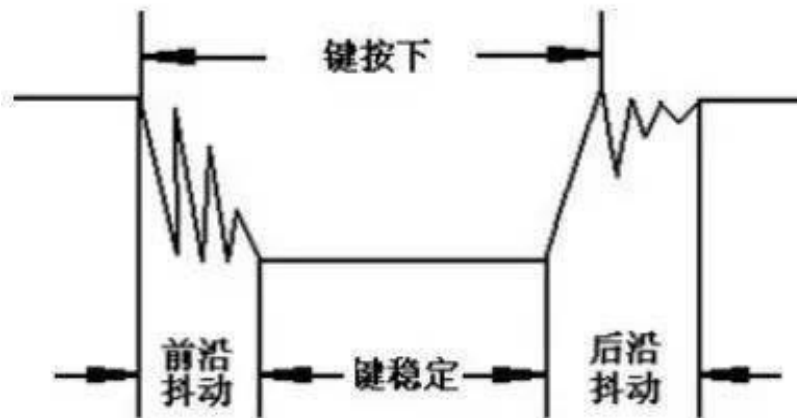
# 1) 单列式（线性）键盘

按键：

按下对应端口 为 ‘0’  
不按下对应端口为 ‘1’



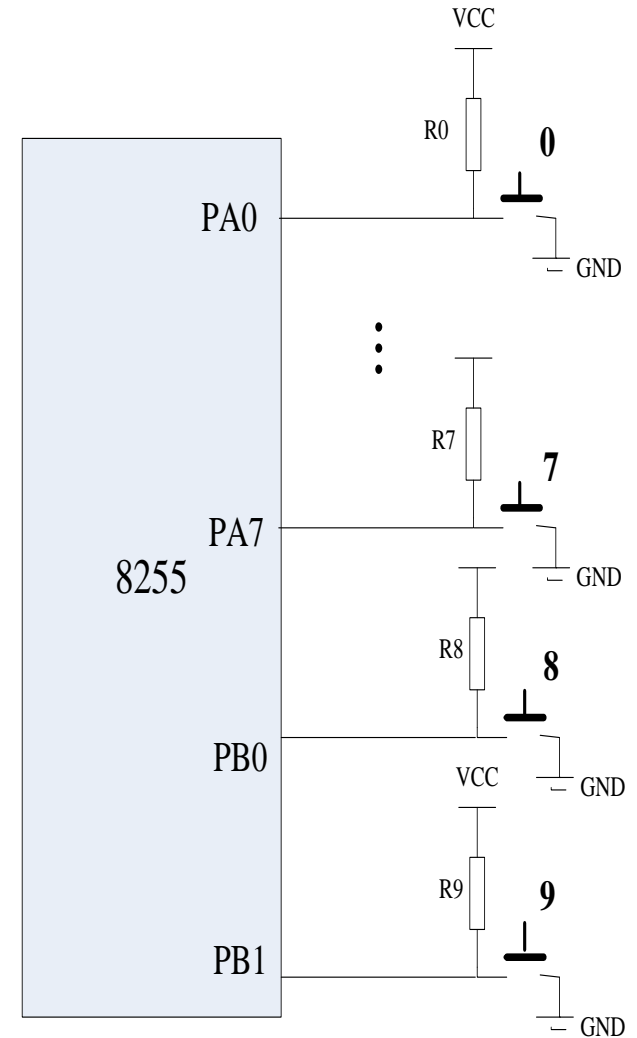
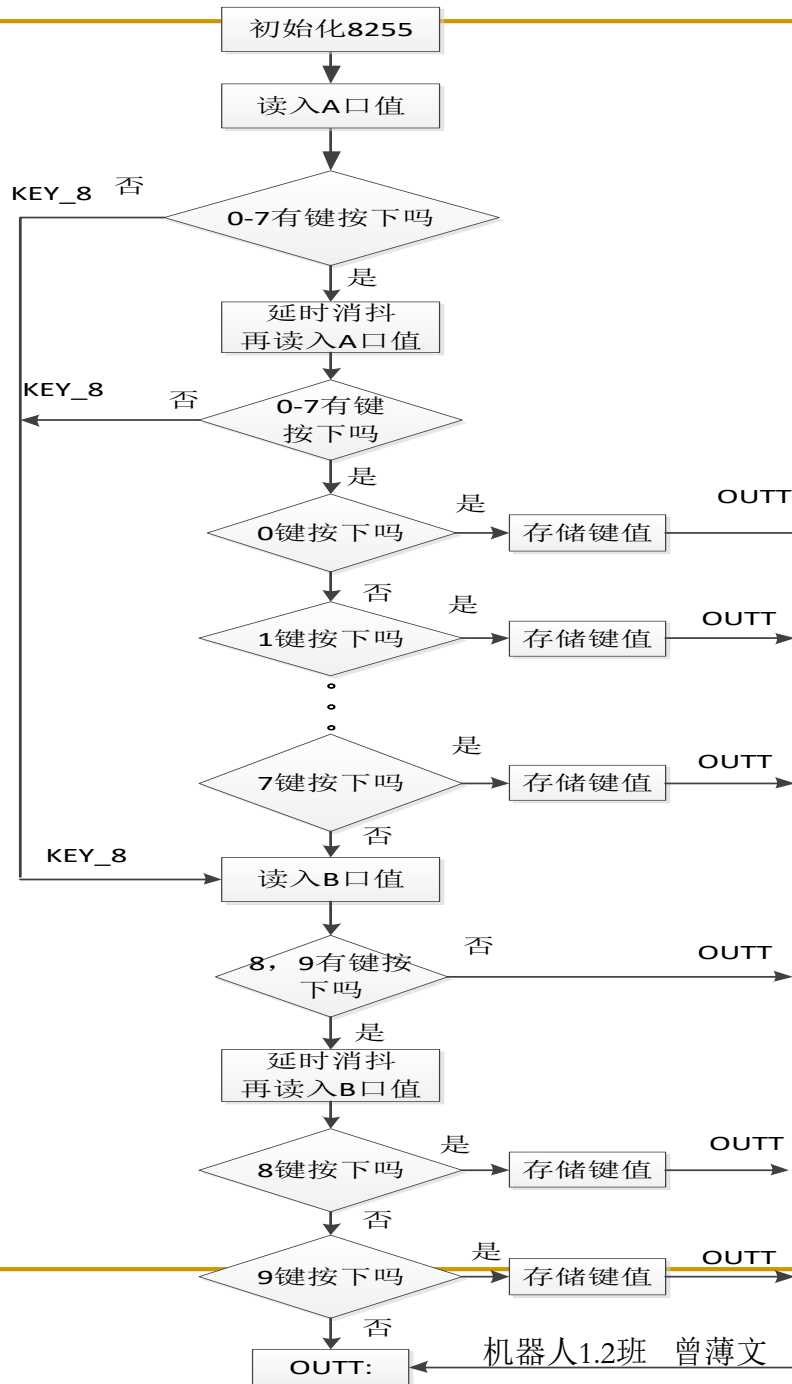
## ■ 按键抖动



一般按键时间为几百毫秒到几秒，抖动时间约为10mS左右。

## ■ 解决方法

延时 10-20mS





## ■ 单列式（线性）键盘软件

8255端口：PORTA，PORTB，PORTC，  
PORT\_CTRL，小于255

MAIN:

MOV AL, 10010010B; A口和B口方式0，输入  
OUT PORT\_CTRL, AL; 初始化8255

KEY\_SCAN:

IN AL, PORT\_A ; 读取A口状态

CMP AL, 0FFH;

JZ KEY\_8 ; 0-7没有键按下

CALL DELAY ; 有键按下，延时消抖

IN AL, PORT\_A; 再次读取A口状态

CMP AL, 0FFH;



JZ KEY\_8 ; 0-7没有键按下

KEY\_1: CMP AL , 0FEH; 判断是否为0

JNZ KEY\_1 ; 如果不等于跳转到下一按键判断

MOV KEY\_DAT,0; 按键为0

JMP OUTT

KEY\_1: CMP AL , 0FDH ; 判断是否为1

JNZ KEY\_2

MOV KEY\_DAT,1 ; 按键为1

JMP OUTT

KEY\_2: CMP AL , 0FBH ; 判断是否为2

JNZ KEY\_3

MOV KEY\_DAT,2 ; 按键为2

JMP OUTT



KEY\_3: CMP AL , 0F7H; 判断是否为3

JNZ KEY\_4

MOV KEY\_DAT,3; 按键为3

JMP OUTT

KEY\_4: CMP AL , 0EFH; 判断是否为4

JNZ KEY\_5

MOV KEY\_DAT,4; 按键为4

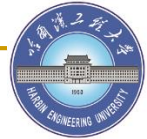
JMP OUTT

KEY\_5: CMP AL , 0DFH; 判断是否为5

JNZ KEY\_6

MOV KEY\_DAT,4; 按键为5

JMP OUTT



```
KEY_6: CMP AL , 0BFH; 判断是否为6
      JNZ KEY_7
      MOV KEY_DAT,4; 按键为6
      JMP OUTT

KEY_7: CMP AL , 7FH; 判断是否为7
      JNZ KEY_8
      MOV KEY_DAT,4; 按键为7
      JMP OUTT

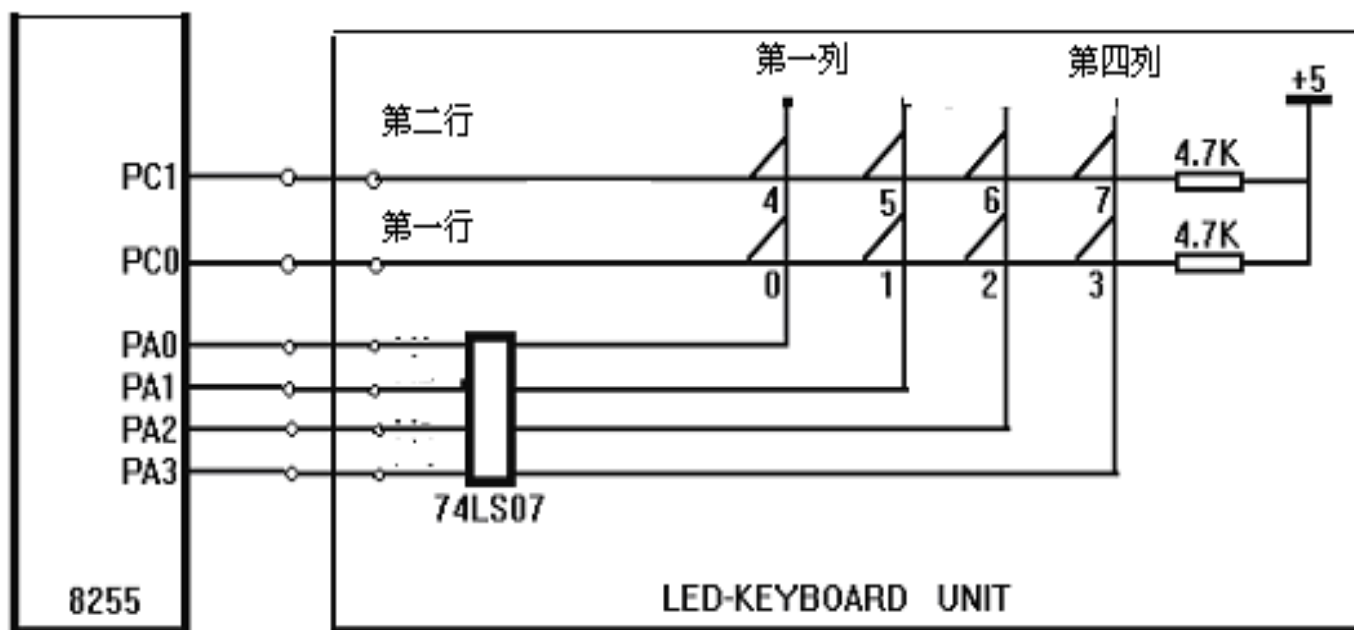
KEY_8: IN AL, PORT_B ; 读取B口状态
      AND AL, 03H ; 屏蔽其他位
      CMP AL, 03H
      JZ OUTT
      CALL DELAY
```

```
IN  AL, PORT_B ; 读取B口状态
AND AL, 03H    ; 屏蔽其他位
CMP AL, 02H    ; 判断是否为8
JNZ  KEY_9
MOV KEY_DAT,8  ; 按键为8
JMP  OUTT
KEY_9: CMP AL, 01H    ; 判断是否为9
      JNZ  OUTT
      MOV KEY_DAT,9  ; 按键为9
```

OUTT:

继续扫描键盘  
或是其他功能

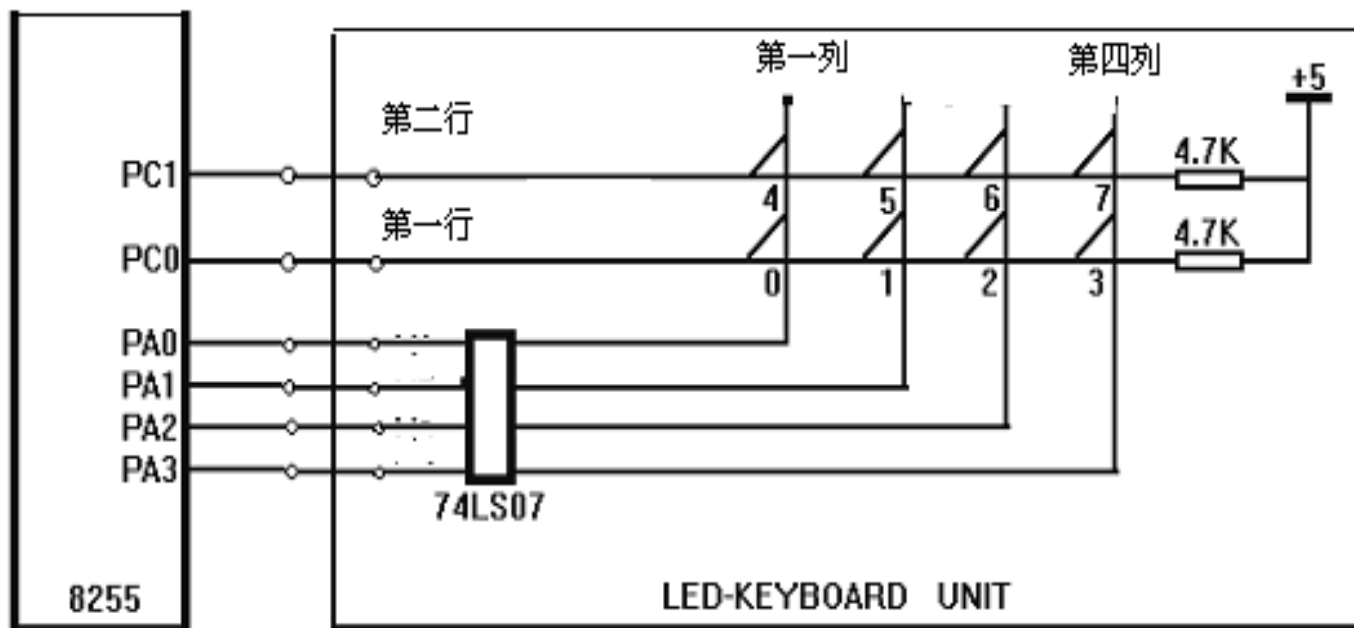
## 2) 矩阵式键盘



关键是确定按键的行列值

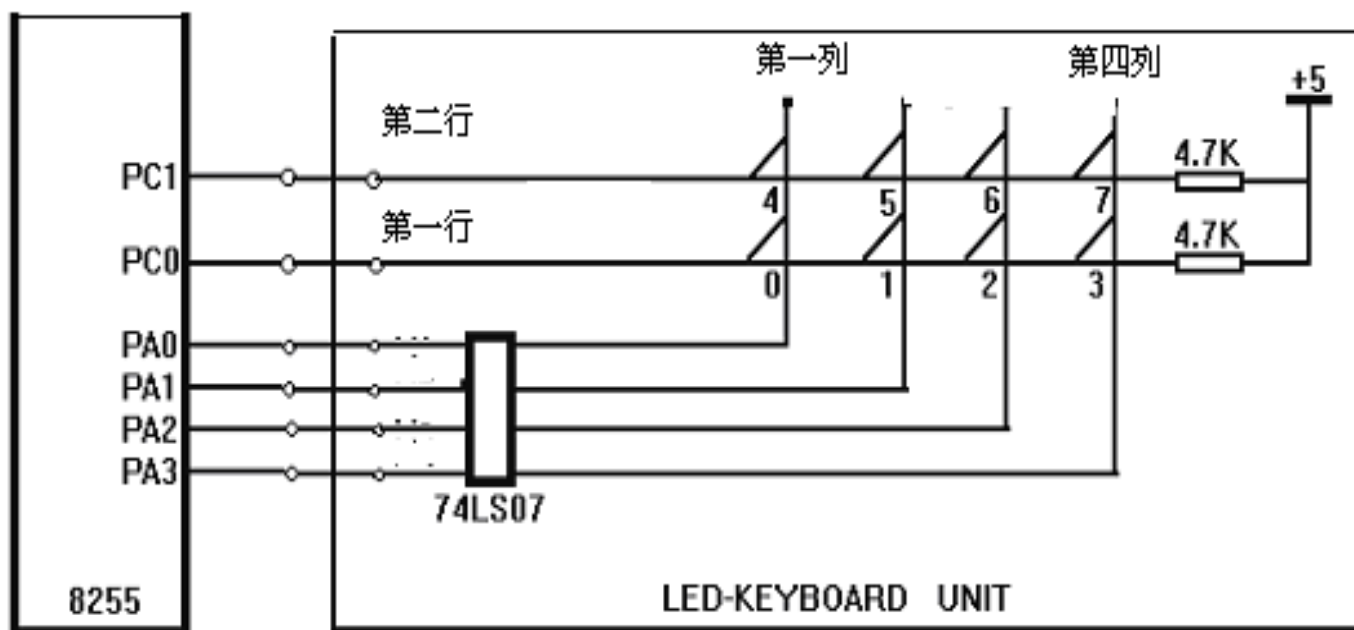
- 反转法
- 扫描法

## □ 反转法



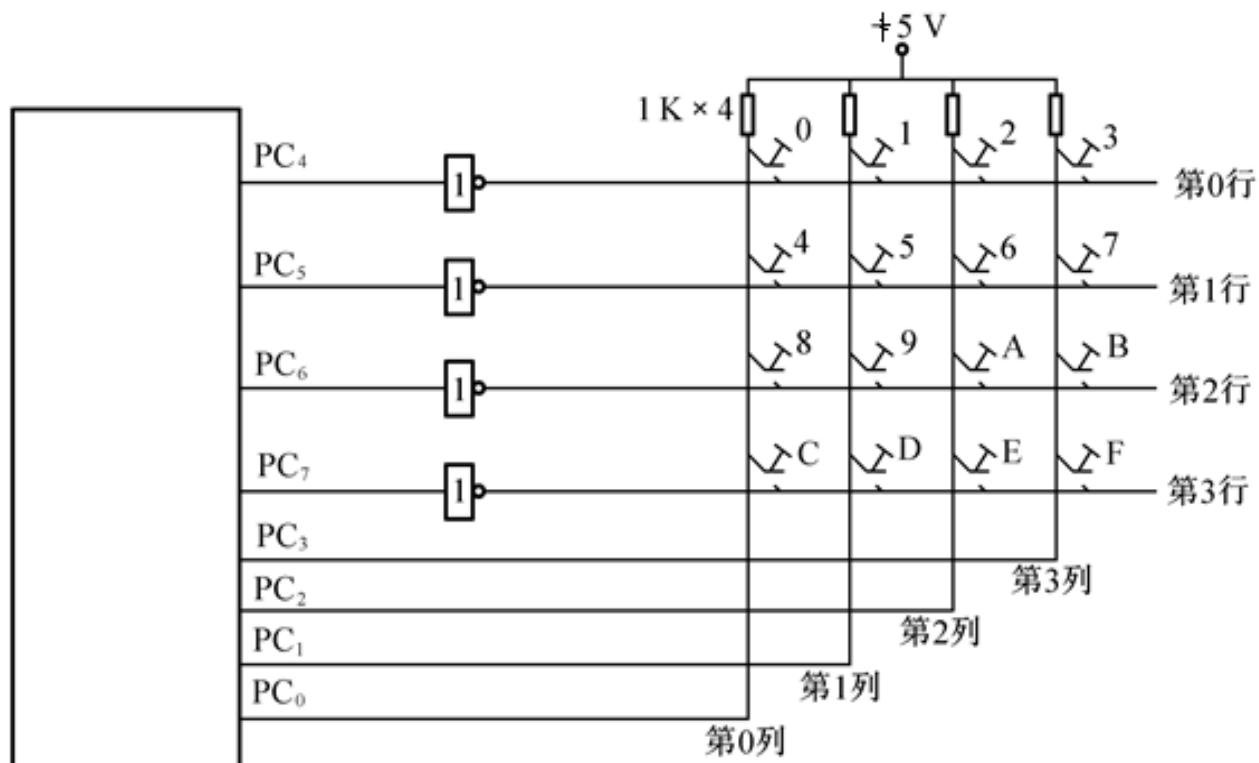
所有列输出低电平，读入行的端口电平，记录此时为低电平的行；然后所有行输出低电平，读入列的端口电平，记录此时为低电平的列，通过行列都为低电平确定按键的位置（值）

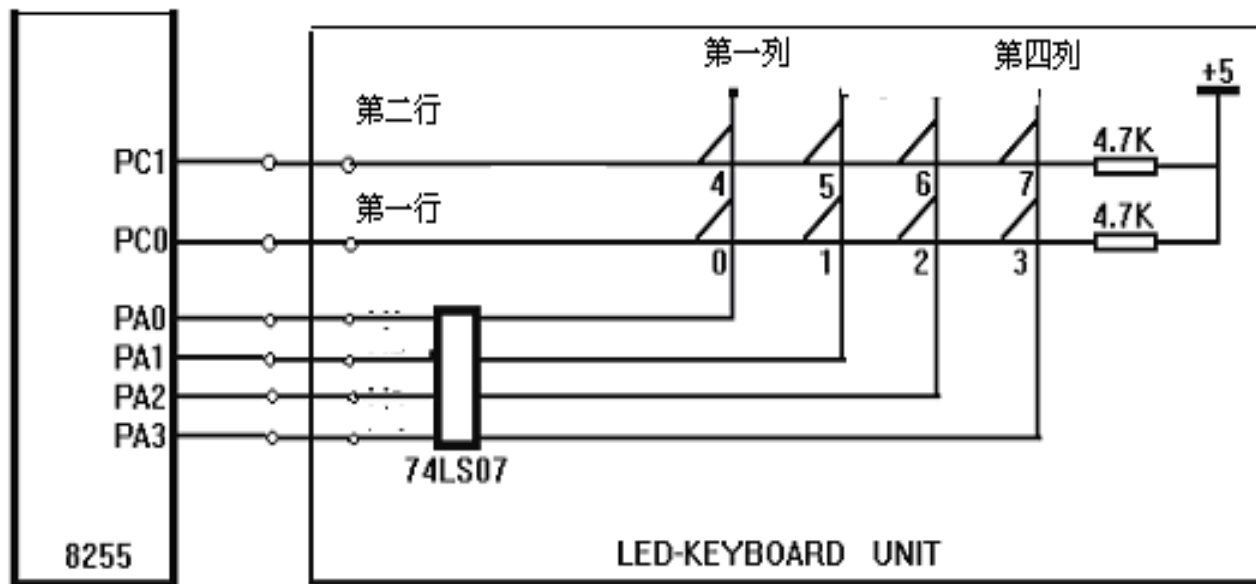
## □ 扫描法（书上）



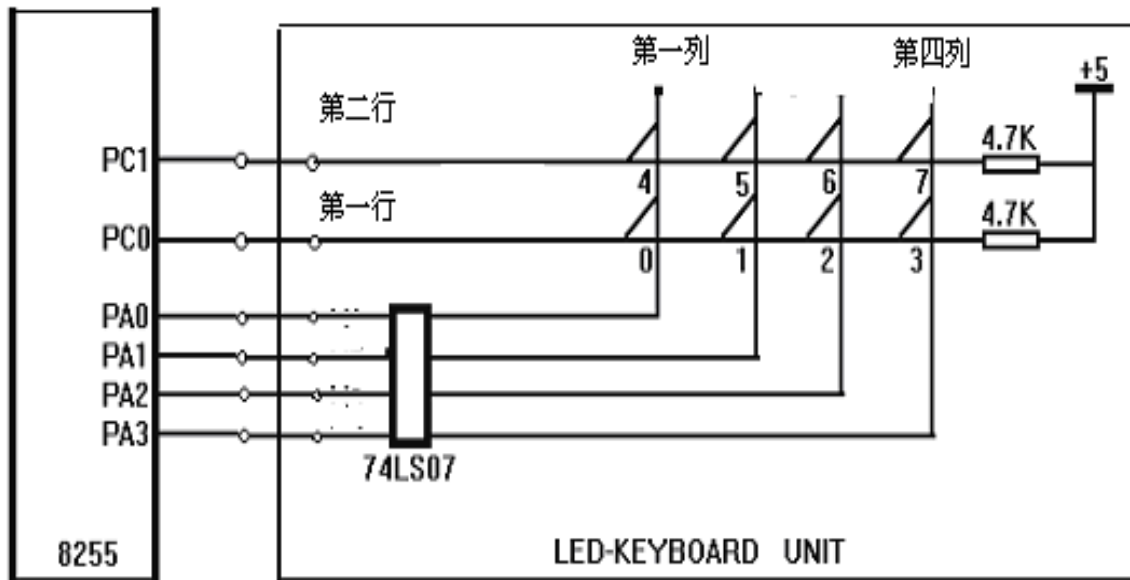
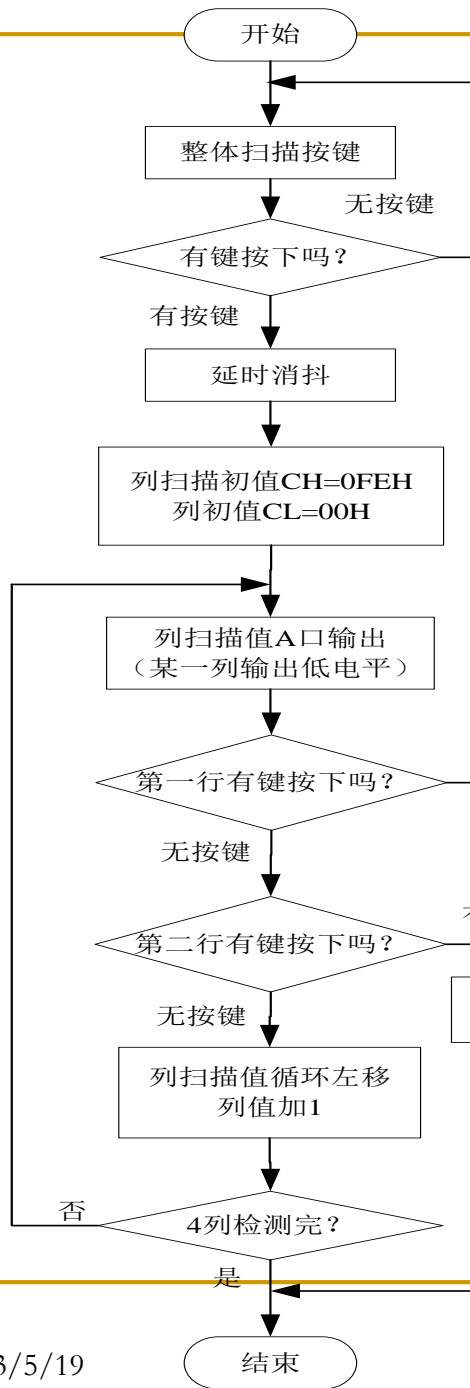
逐列输出低电平，其他列为高电平，然后行端口读入，判断哪行为低电，通过行列都为低电平确定按键的位置（值）。

# 扫描法的另一种连接形式





- 键值=行基准值+列值
- 列值从0-3
- 第一行基准值为0，第二行基准值为4





## ■ 矩阵键盘扫描法软件

8255端口: PORTA , PORTB, PORTC, PORT\_CTRL, 大于255

...

MOV AL,1000001B ;A口输出, C口低四位输入, 所有口方式0

MOV DX,PORT\_CTRL

OUT DX,AL ;送控制字

WHOLE\_SCAN: ;整体快速判断是否有键按下

MOV AL,00H

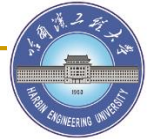
MOV DX,POTR\_A

OUT DX,AL ;A口列输出低电平

MOV DX,PORT\_C

IN AL,DX ;读入行的状态

AND AL,03H ;屏蔽其他位



CMP AL ,03H

JZ WHOLE\_SCAN ;行全为高电平（无按键），继续扫描或退出

CALL DELAY ;有键按下需要延时消除抖动

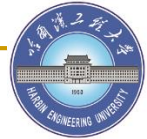
IN AL , DX ;再读入行的状态

CMP AL ,03H

JZ WHOLE\_SCAN ;行全为高电平（无按键），继续扫描或退出

MOV CH,0FEH ;列输出初值，第一列为低电平

MOV CL,0 ;列初值（列值0-3）



LINE1:

MOV AL,CH

MOV DX,PORT\_A

OUT DX,AL

;列轮流输出低电平

MOV DX,PORT\_C

IN AL,DX

;读入行的状态

TEST AL,01H

;判断是否为第一行有键按下

JNZ LINE2

;不是第一行，转第二行判断

MOV AL,00H

;第一行键初值为00

JMP KEY\_VALUE

;计算键值

LINE2:

TEST AL,02H

;判断是否为第二行有键按下

JNZ COLUMN

;跳转到调整列值输出

MOV AL,04H

;第二行键初值为04

KEY\_VALUE:

ADD AL,CL

;计算键值=行初值+列值

JMP KEY\_DEAL

;根据键盘值做相应程序处理



```
COLUMN:                ;列输出值调整程序
    INC CL              ;列值加1
    MOV AL,CH
    TEST AL,08H         ;列值输出是否为PA3（第四列）为低电平
    JZ KEY_OVER         ;结束本次键盘扫描（所有列输出低电平结束）
    ROL AL,1            ;列输出值左移(下一列输出低电平)
    MOV CH,AL
    JMP LINE1           ;列输出都要检测每行的状态
KYE_DEAL:
    ...                ; 根据键值处理任务的程序

KEY_OVER:
    ...                ; 键盘处理结束，CPU执行其他程序
```

### 3) 键盘在工程使用中的相关问题

- 采用简化键盘按键数

- 定时中断处理按键

线性键盘

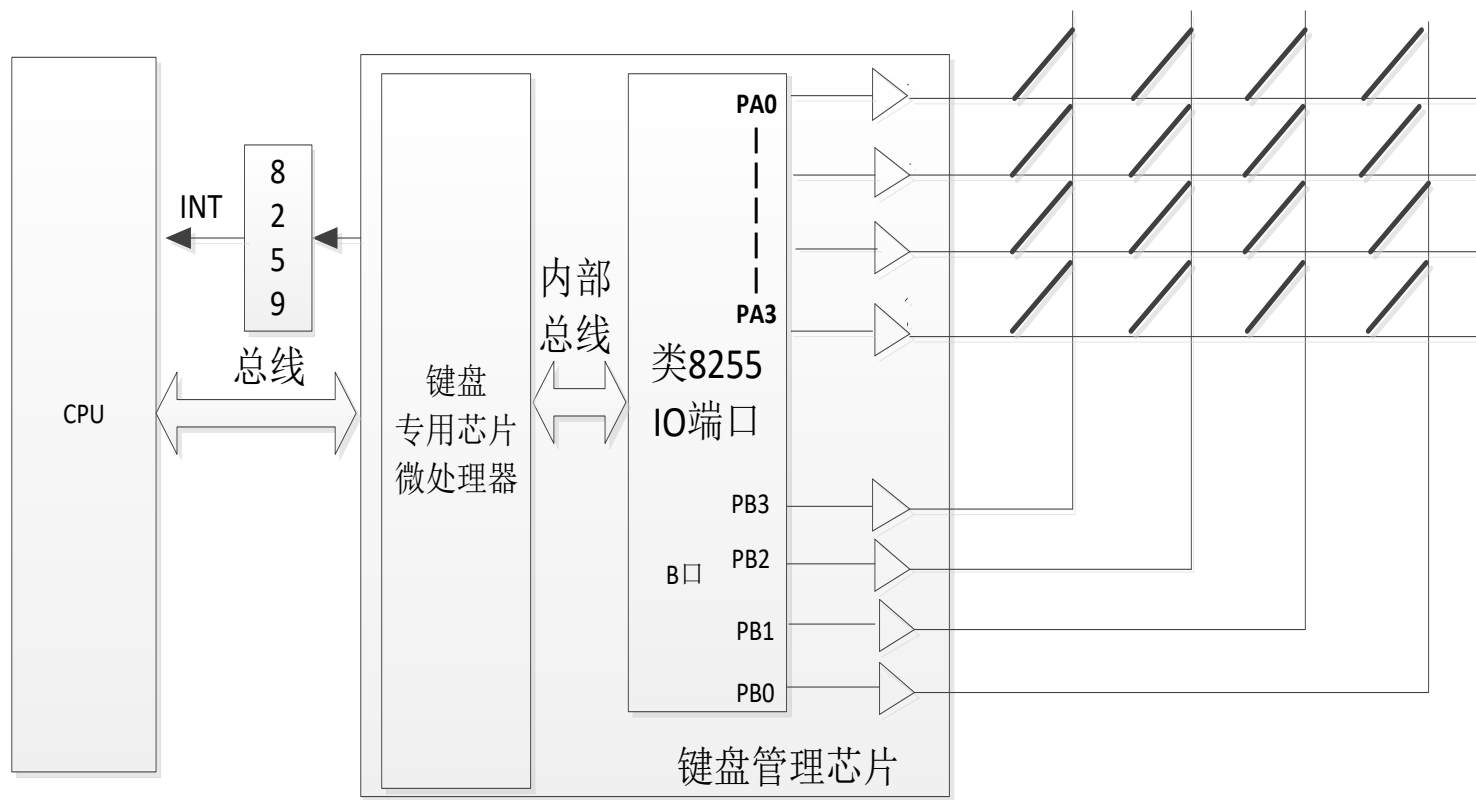
矩阵按盘

} 延时消耗**CPU**时间

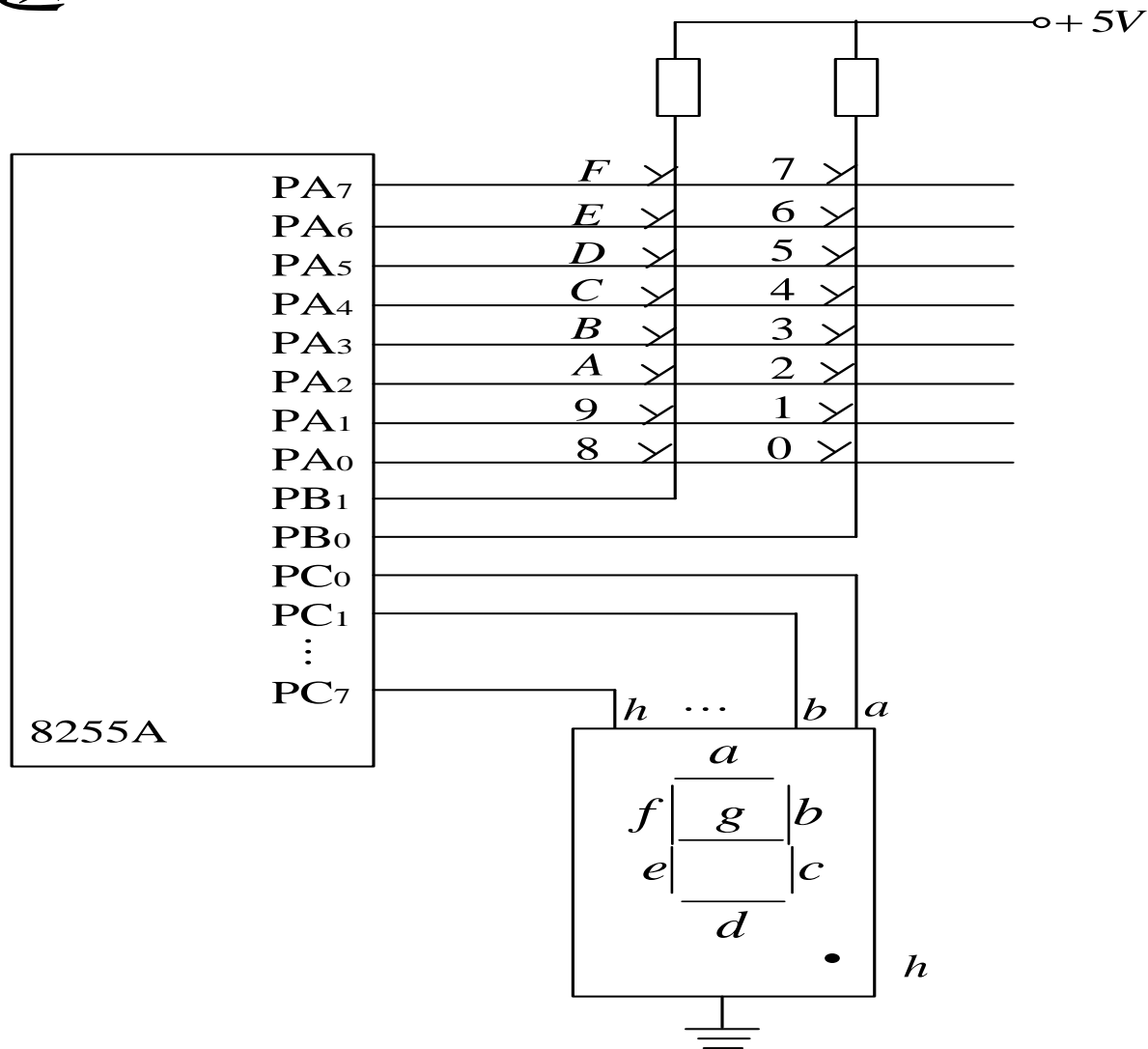
- 一次按键记为一次还是多次

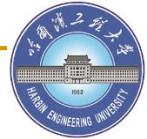
- 专用键盘管理芯片  
INTEL(NEC) 8279  
ZLG7290 （串行）

# 键盘专用芯片工作原理

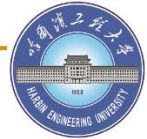


# 练习题





大致题意：采用查询方式检测按键，如果第2列（8-F）有键按下，在数码管（共阴极）上显示8，第1列（0-7）有键按下显示0。编写相关程序段（8255初始化，键盘扫描，七段码的建立，查表显示等功能）。



# DATA SEGMENT

```
TAB_SEG DB  
3FH,06H,5BH,4FH,66H,6DH,7DH,07H,7FH,  
6FH
```

```
DATA ENDS
```

# CODE SEGMENT

```
ASSUME CS:CODE,DS:DATA
```

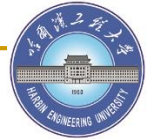
```
MOV AX,DATA
```

```
MOV DS,AX
```

```
LEA BX,TAB_SEG
```

```
MOV AL,10000010B
```

```
OUT 63H,AL
```



```
BEGIN: MOV AL,0
        OUT 60H,AL
        IN AL,61H
        AND AL,03H ;屏蔽PB2-PB7
        CMP AL,02H ;PB0=0
        JZ DISP_0 ;送0到数码管显示
        CMP AL,01H ;PB1=0
        JZ DISP_8 ;送8到数码管显示
        JMP BEGIN
DISP_0: MOV AL,0
```

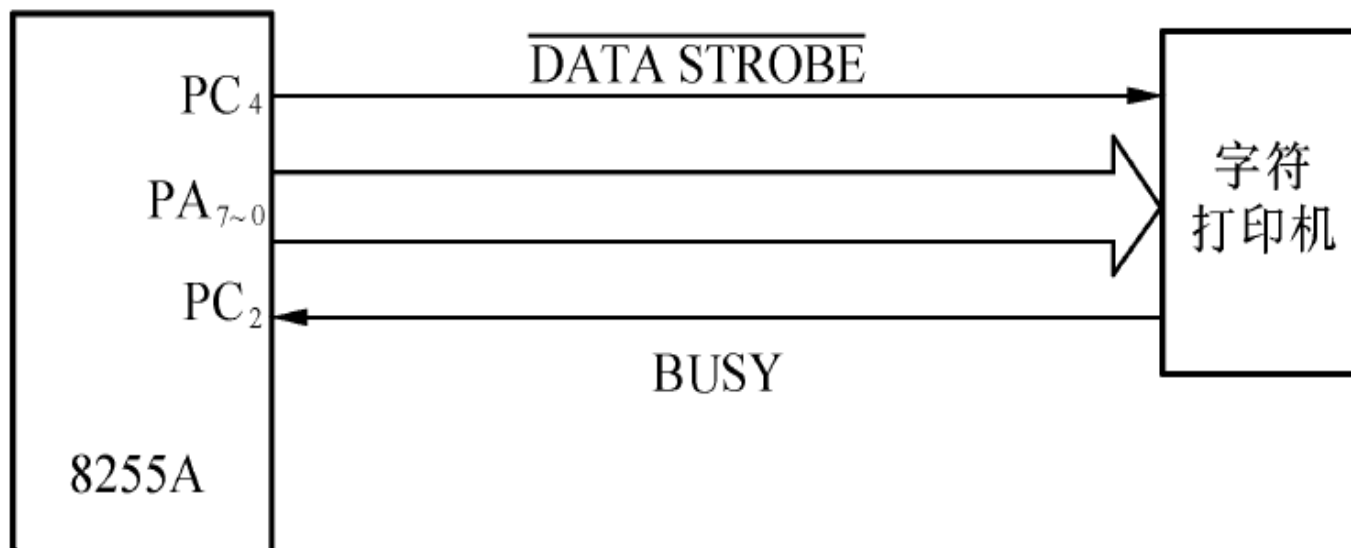


```
XLAT
OUT 62H,AL
JMP OUTT
DISP_8:MOV AL,8
XLAT
OUT 62H,AL
OUTT: JMP BEGIN
CODE ENDS
END
```

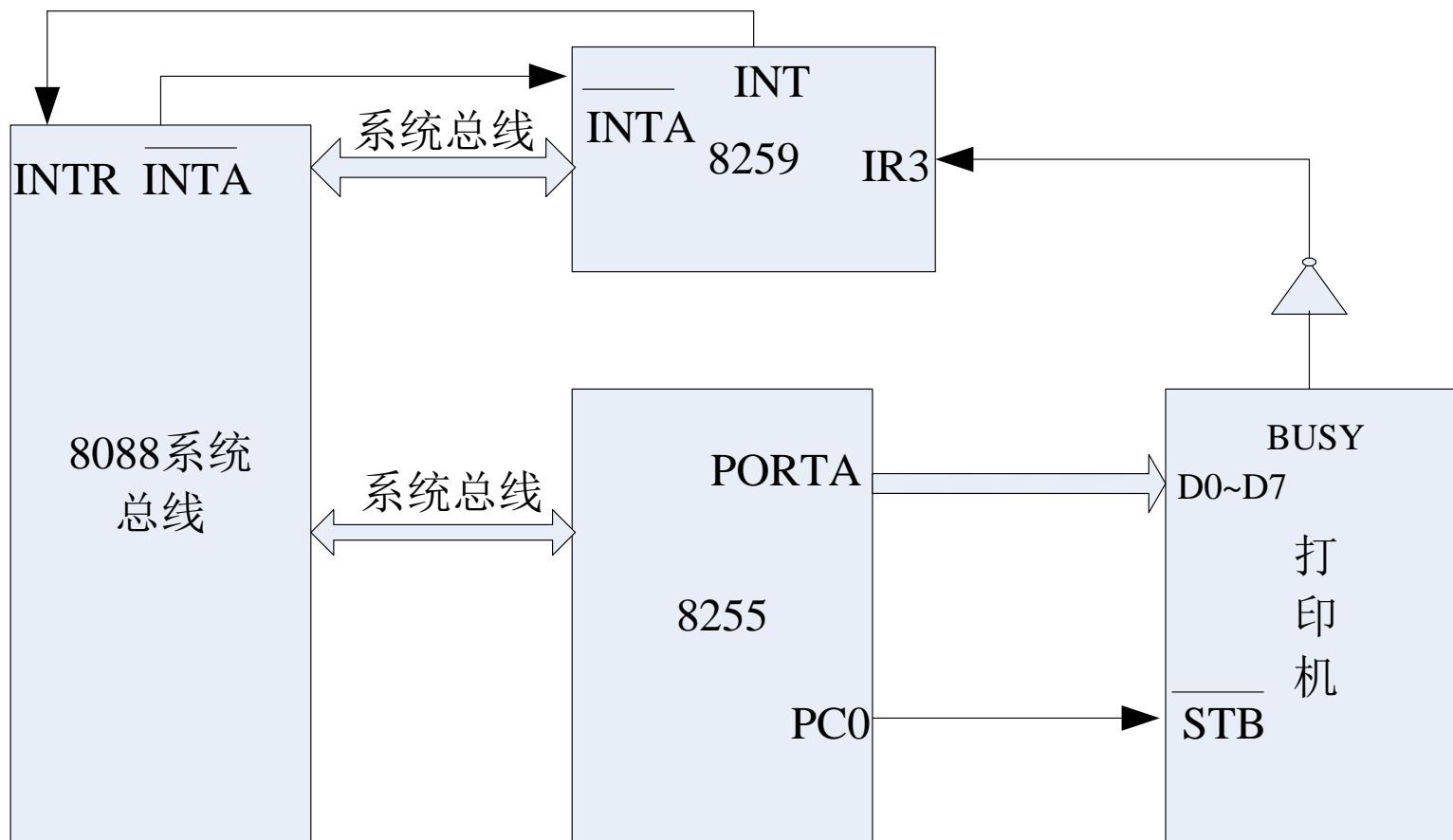
## 7.3 8255方式1应用

### 一、主机和外设交互数据形式(8255方式0)

#### 1. 查询方式输出数据

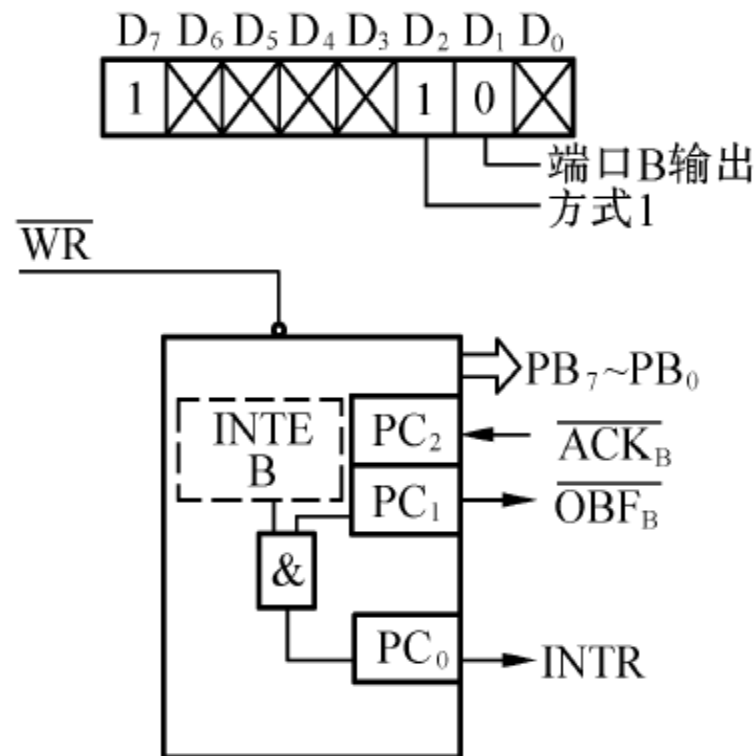
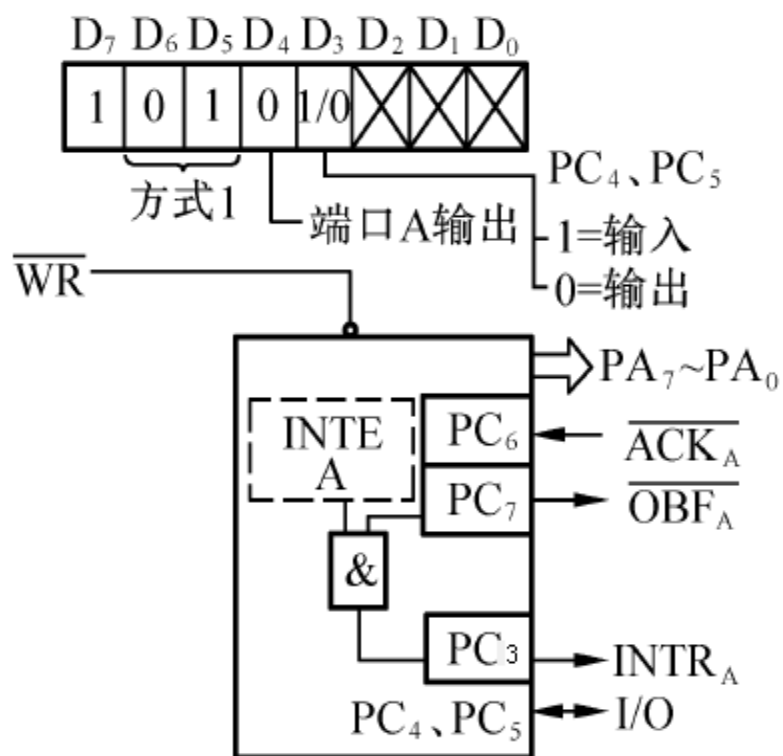


## 2、中断方式输出数据

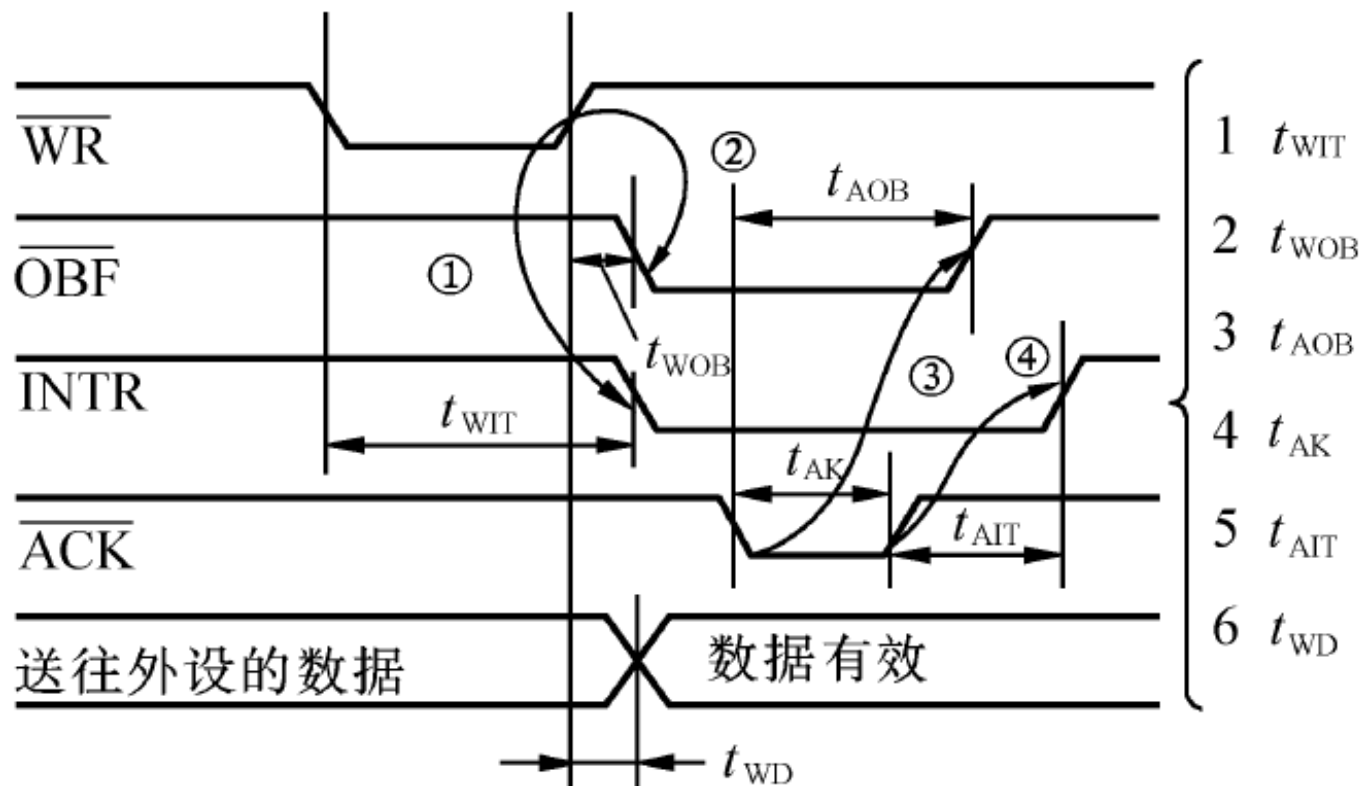


## 二、8255方式1

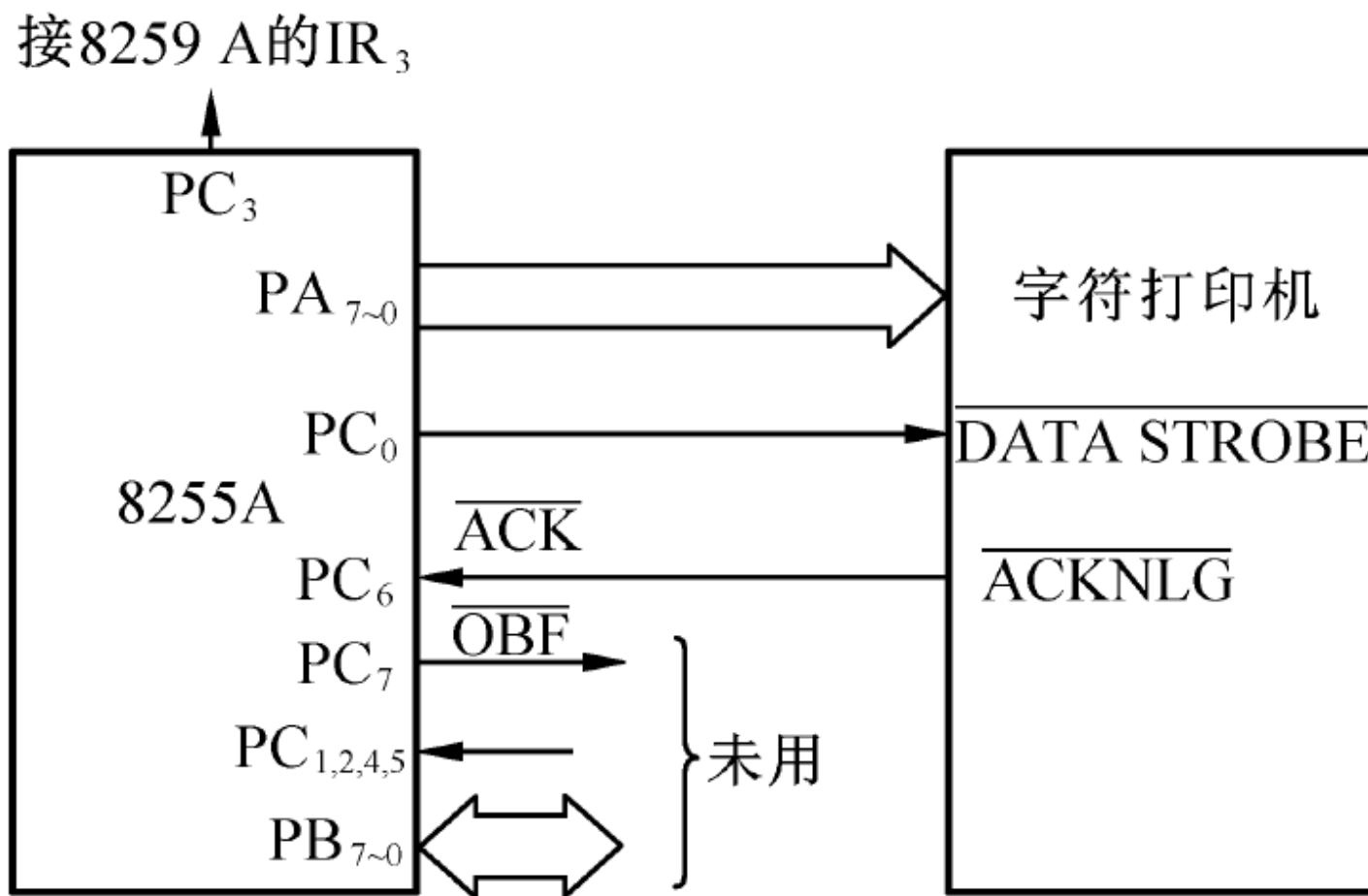
### 1.8255方式1输出

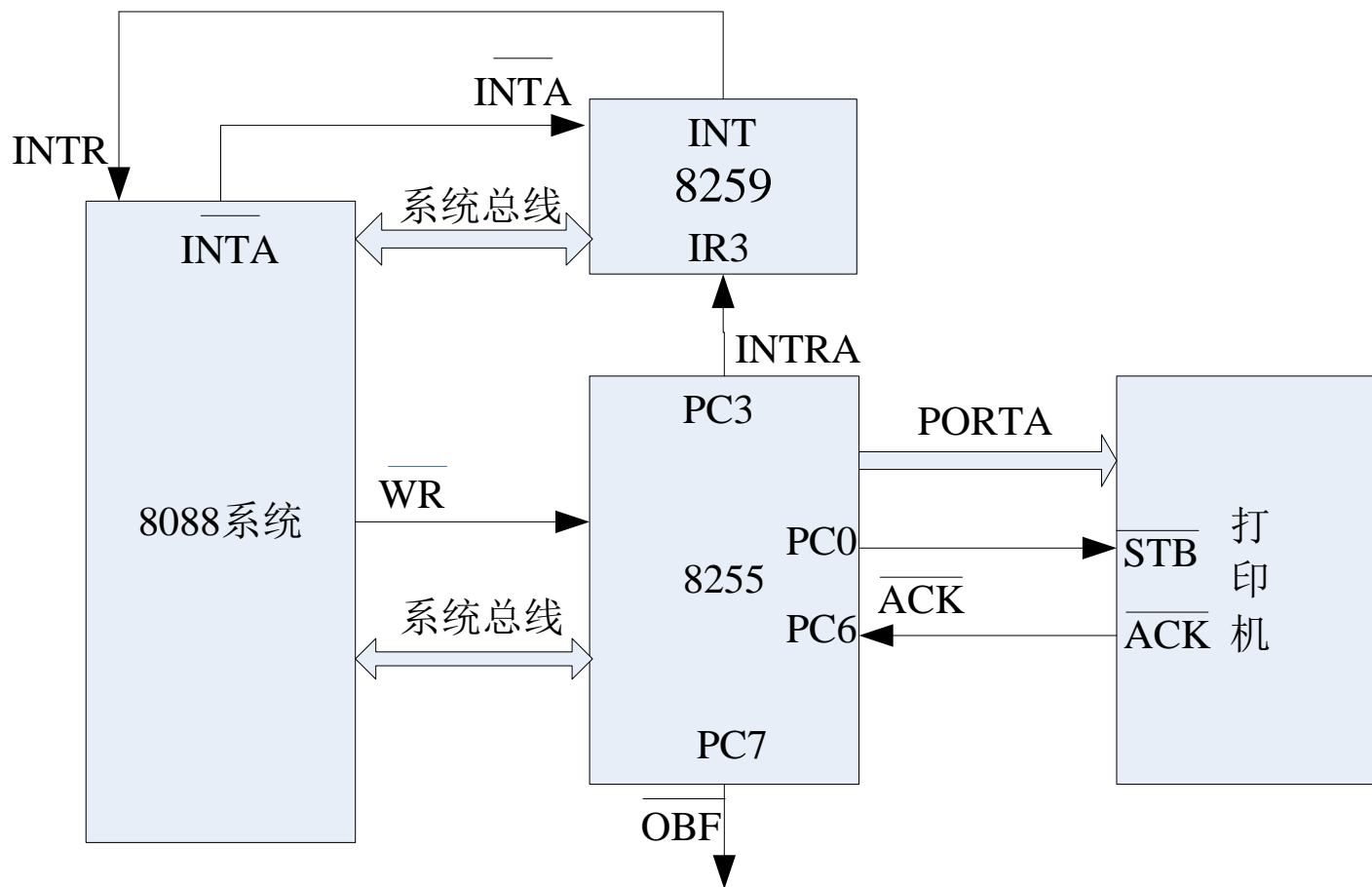


方式1输出控制信号

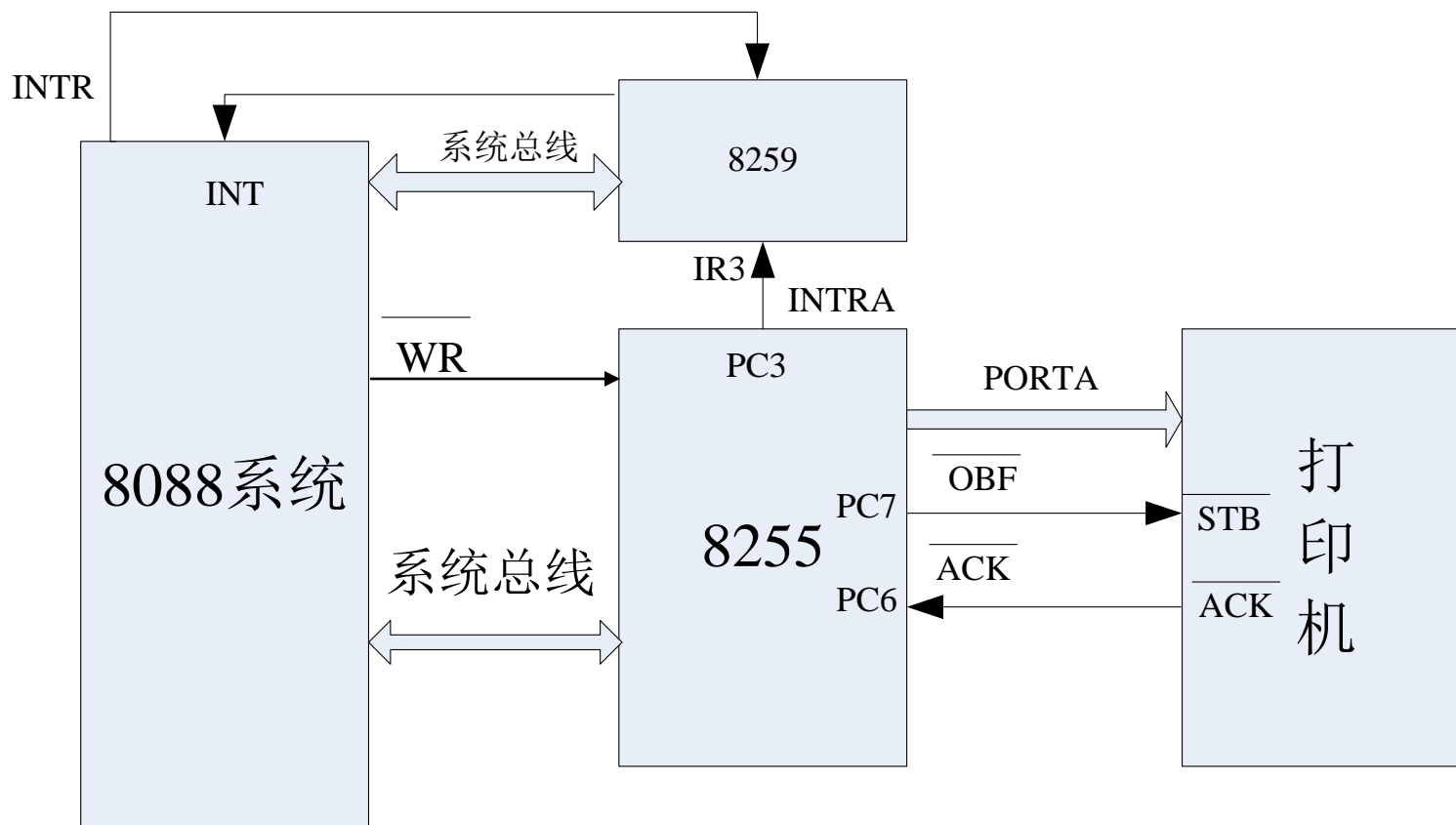


方式1 输出时序图

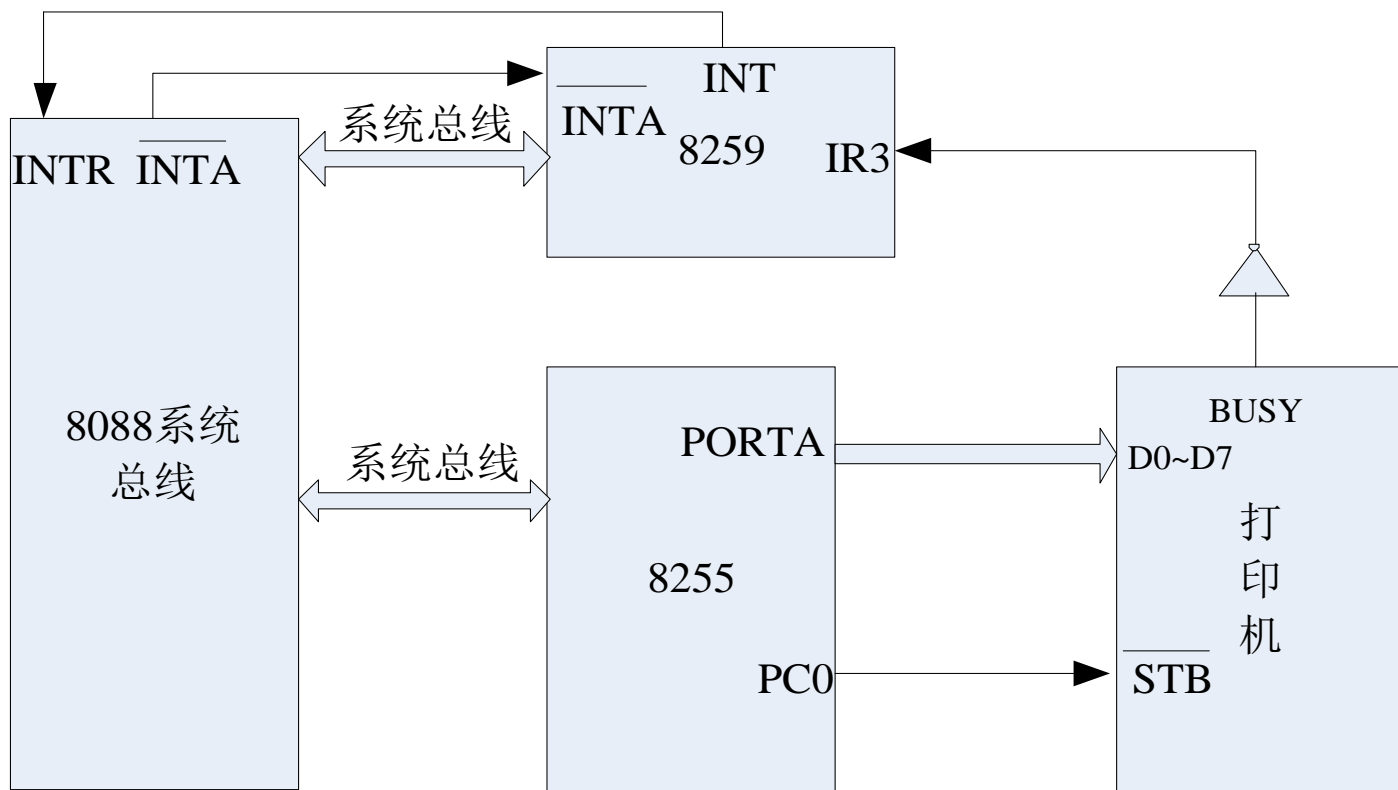




## 8255方式1输出中断方式（书上）



## 8255方式1输出中断方式



## 8255方式0 中断方式



**例** 内存中LP\_DATA有10个字符，它们被其他程序更新时就需要打印，否则不用打印。要求采用8255A口**方式1**输出打印机，8255端口地址：E0H，E2H，E4H，E6H，8259端口地址为20H，21H，中断矢量号为08H。

DATA SEGMENT

LP\_DATA DB X1,...X10 ; 待打印字符数据

CNT EQU \$-LP\_DATA

FLAG\_DATA\_CHG DB 0 ; 数据被更新的标志位

DATA ENDS



# CODE SEGMENT

```
ASSUME CS:CODE ,DS:DATA
```

```
GO: MOV AX , DATA
```

```
MOV DS , AX
```

```
MOV AL, 0A0H; 8255A输出方式1
```

```
OUT 0E6H, AL
```

```
MOV AL, 13H
```

```
OUT 20H, AL; ICW1的控制字
```

```
MOV AL, 08H
```

```
OUT 21H, AL; ICW2的控制字
```

```
MOV AL, 01H
```

```
OUT 21H, AL; ICW4的控制字
```



PUSH DS

MOV AX, 0

MOV DS, AX

MOV AX, OFFSET IRQ3\_PRT

MOV DS:[002CH], AX; 填IP

MOV AX, CS

MOV DS:[002EH], AX; 填CS

POP DS ; DS出栈

STI ; 开中断总开关

IN AL, 21H

AND AL, 11110111B ; OCW1, 打开中断3

OUT 21H, AL



```
MOV AL, 0DH ; INTEA=1, PC6=1  
OUT 0E6H, AL
```

```
MAIN: ; 循环监控主程序体
```

```
... ; 系统其他功能, 可更新打印数据
```

```
CMP FLAG_DATA_CHGL,1
```

```
JNZ MAIN
```

```
MOV FLAG_DATA_CHGL,0; 清打印标志位
```

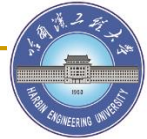
```
MOV BX, OFFSET LP_DATA
```

```
MOV AL, [BX] ;
```

```
OUT 0E0H, AL;打印第一个数
```

```
...
```

```
JMP MAIN
```



IRQ3\_PRT : ; 打印中断服务程序

PUSH AX

INC BX

MOV AL, [BX]

OUT 0E0H, AL ; 送A口

MOV AL, 20H

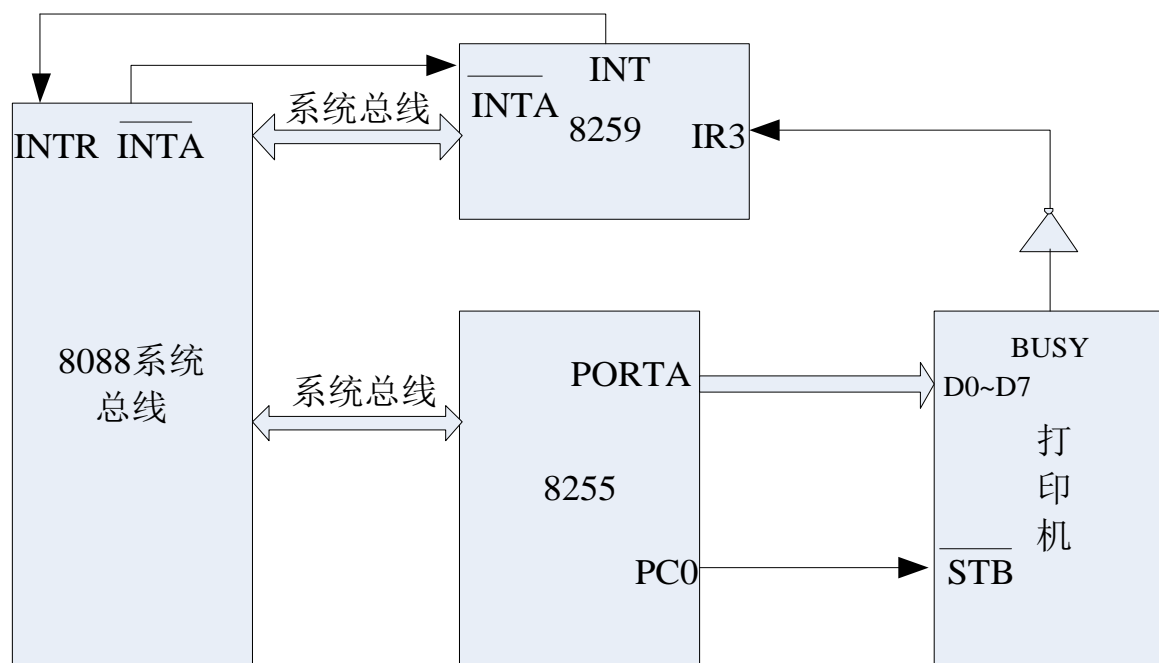
OUT 20H, AL

POP AX

IRET

CODE ENDS

END GO



## 8255方式0 中断方式



**例修改** 内存中LP\_DATA有10个字符，它们被其他程序更新时就需要打印，否则不用打印。要求采用8255A口**方式0**输出打印机，8255端口地址：E0H，E2H，E4H，E6H，8259端口地址为20H，21H，中断矢量号为08H。

DATA SEGMENT

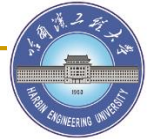
LP\_DATA DB X1,...X10 ; 待打印字符数据

CNT EQU \$-LP\_DATA

FLAG\_DATA\_CHG DB 0 ; 数据被更新的标志位

ADDRESS\_DATA DW ?

DATA ENDS



# CODE SEGMENT

```
ASSUME CS:CODE ,DS:DATA
```

```
GO: MOV AX , DATA
```

```
MOV DS , AX
```

```
MOV AL, 80H; 8255A,C输出,方式0
```

```
OUT 0E6H, AL
```

```
MOV AL, 13H
```

```
OUT 20H, AL; ICW1的控制字
```

```
MOV AL, 08H
```

```
OUT 21H, AL; ICW2的控制字
```

```
MOV AL, 01H
```

```
OUT 21H, AL; ICW4的控制字
```



PUSH DS

MOV AX, 0

MOV DS, AX

MOV AX, OFFSET IRQ3\_PRT

MOV DS:[002CH], AX; 填IP

MOV AX, CS

MOV DS:[002EH], AX; 填CS

POP DS ; DS出栈

STI ; 开中断总开关

IN AL, 21H

AND AL, 11110111B ; OCW1, 打开中断3

OUT 21H, AL



```
MOV AL, 01 ; PC0=1
OUT 0E6H, AL
```

```
MAIN: ; 循环监控主程序体
... ; 系统其他功能，可更新打印数据
```

```
CMP FLAG_DATA_CHGL,1
JNZ MAIN
```

```
MOV FLAG_DATA_CHGL,0; 清打印标志位
```

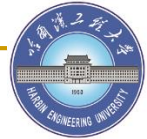
```
MOV BX,OFFSET LP_DATA
```

```
MOV AL, [BX] ;
```

```
OUT 0E0H, AL;打印第一个数
```

```
MOV AL, 00 ; PC0=0
OUT 0E6H, AL
```

```
MOV  AL, 01      ; PC0=1
OUT  0E6H, AL
INC  BX
MOV  ADDRESS_DATA, BX
LOP1:
.....
.....
JMP  LOP1
```



IRQ3\_PRT : ; 打印中断服务程序

PUSH AX

PUSH BX

MOV BX, ADDRESS\_DATA

MOV AL, [BX]

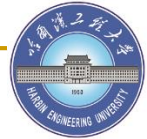
OUT 0E0H , AL ; 送A口

MOV AL, 00 ; PC0=0

OUT 0E6H, AL

MOV AL, 01 ; PC0=1

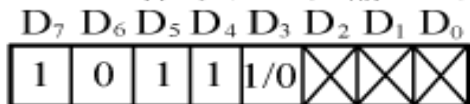
OUT 0E6H, AL



```
INC  BX
MOV  ADDRESS_DATA ,BX
MOV  AL ,20H
OUT  20H, AL
POP  BX
POP  AX
IRET
CODE ENDS
      END START
```

## 2.输入

A组工作于方式1输入的控制字



方式1

PC<sub>6</sub>、PC<sub>7</sub>  
1=输入  
0=输出

端口A为输入

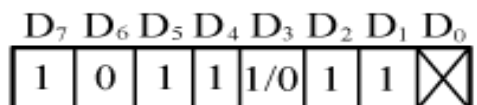
B组工作于方式1输入的控制字



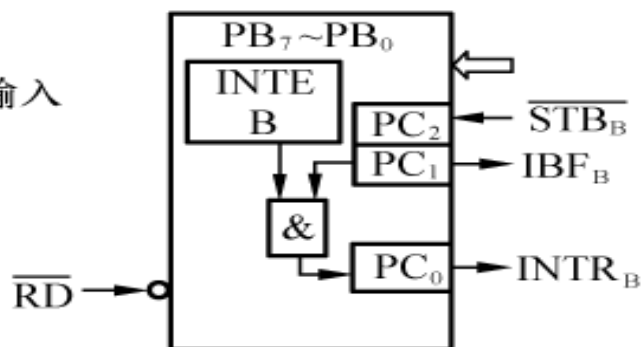
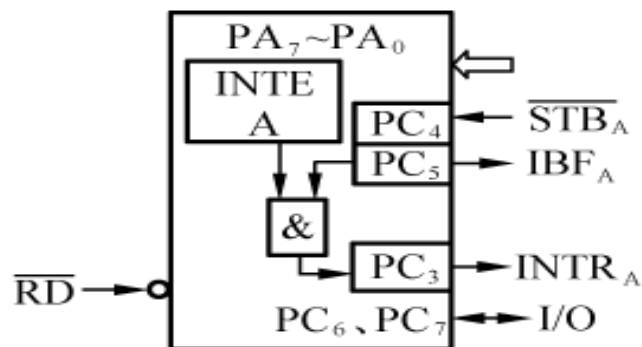
端口B为输入

方式1

A组和B组都工作于方式1输入的控制字



PC<sub>6</sub>、PC<sub>7</sub>  
1=输入  
0=输出



# 方式2

- 双向输入需要握手信号，两个（输入输出）方式1。

