8086微机接口技术

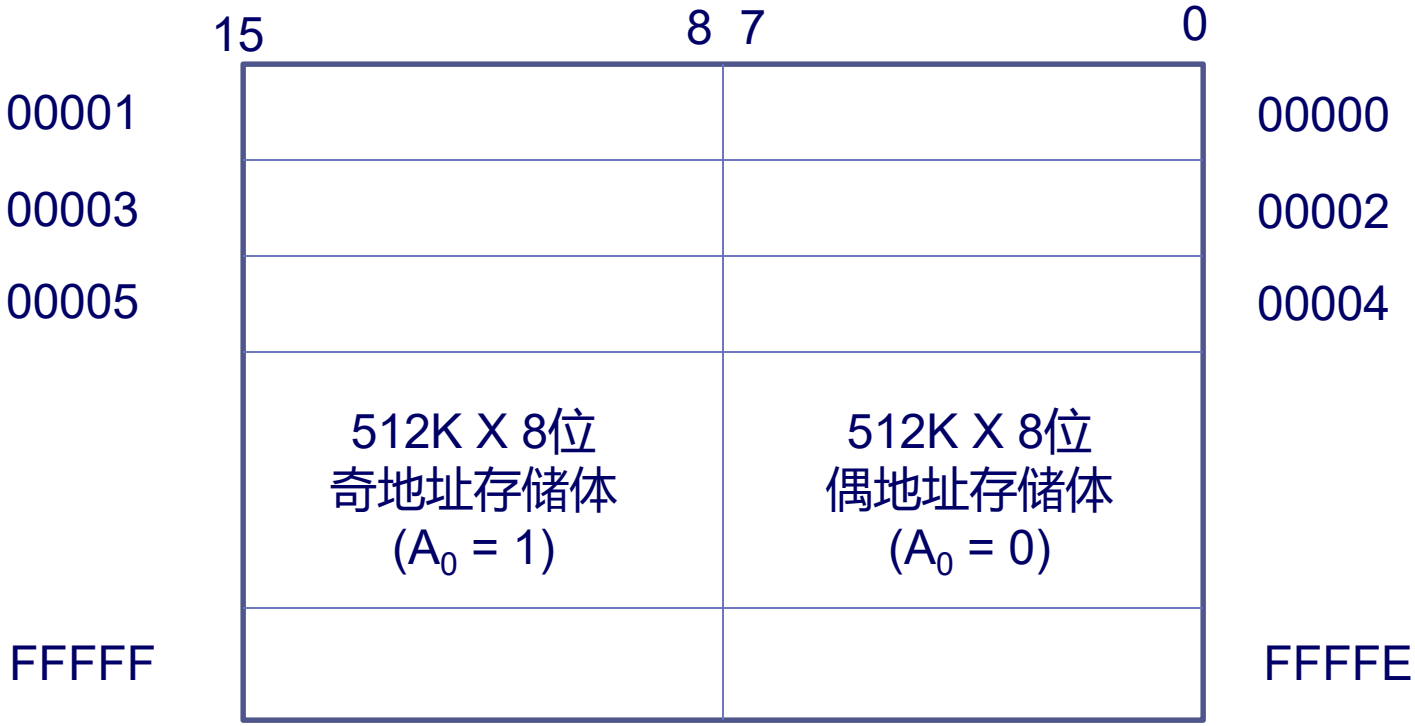
S.J.T.U.

陆海宁
hnlu@sjtu.edu.cn

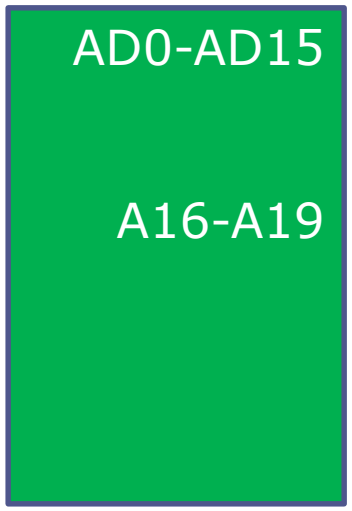
内容

8086存储器分体结构

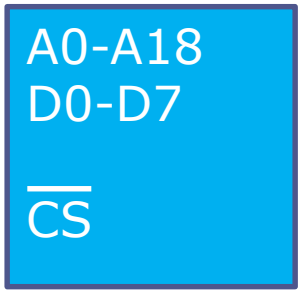
- **1MB存储空间分成两个存储体**
  - 偶地址存储体和奇地址存储器
  - 各512KB



8086存储器分体结构

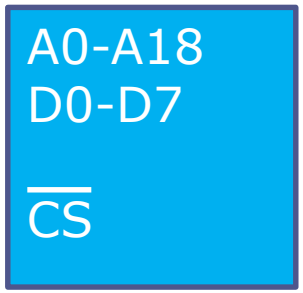


8086



512KB
8位存储器

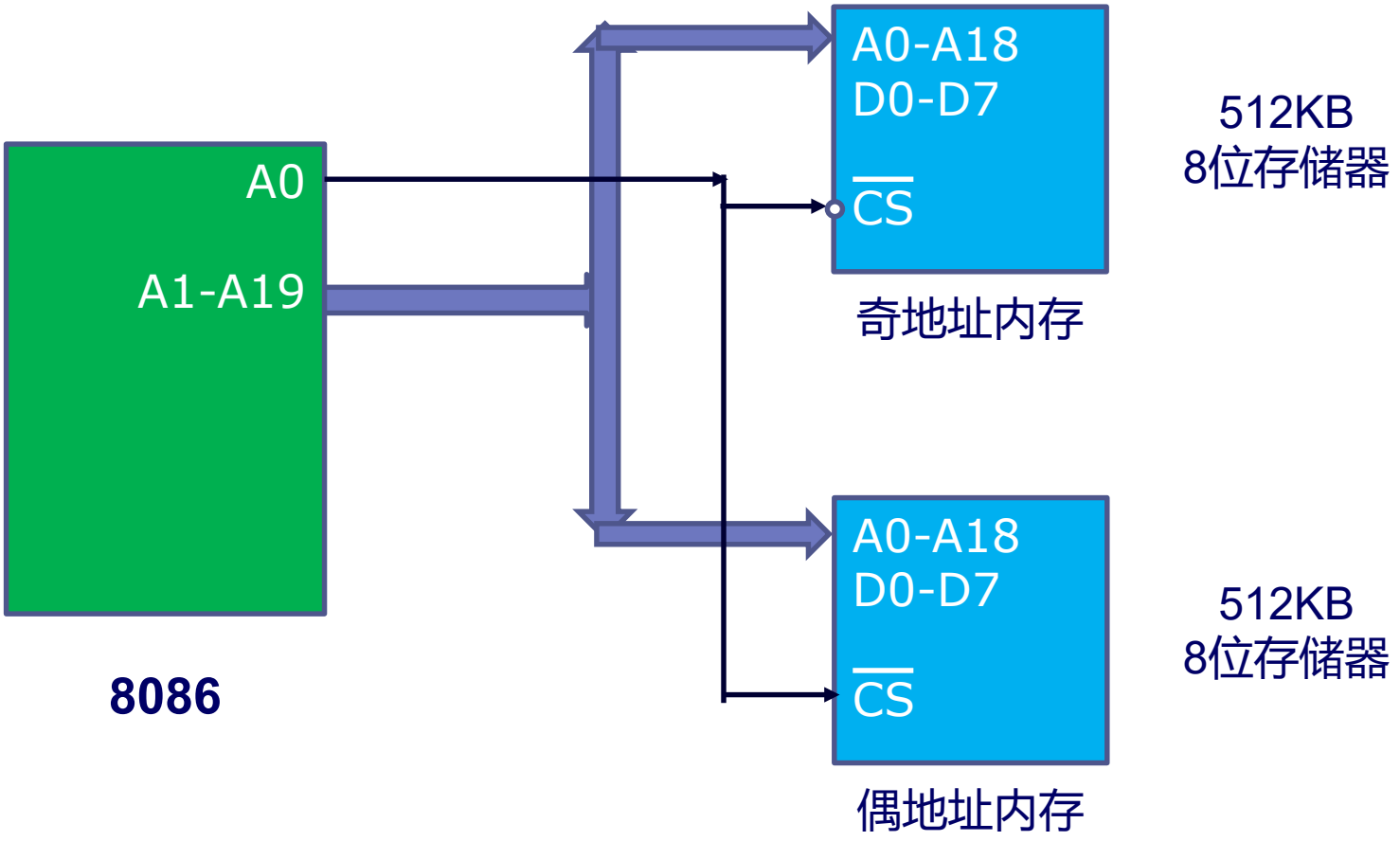
奇地址内存



512KB
8位存储器

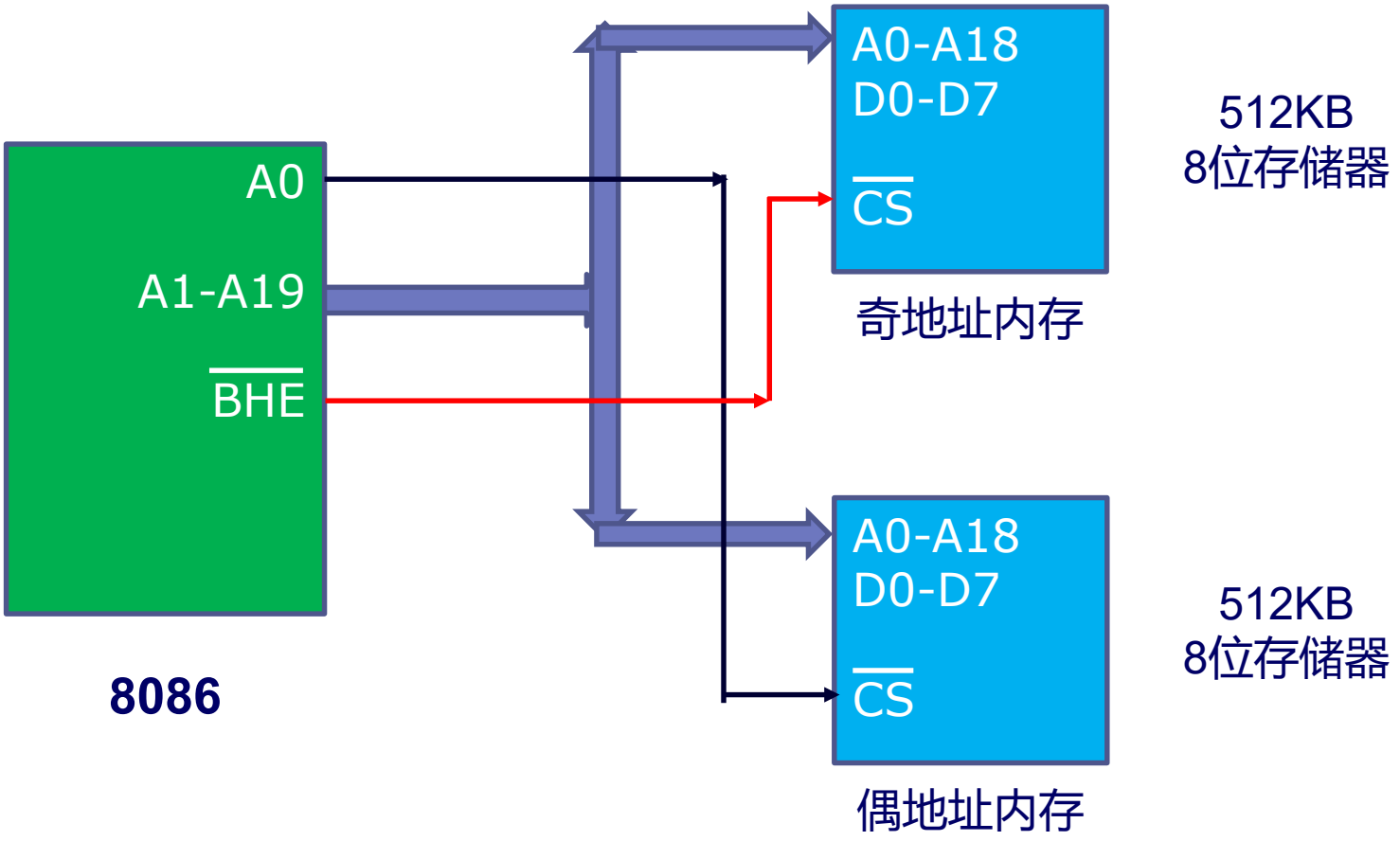
偶地址内存

8086存储器分体结构

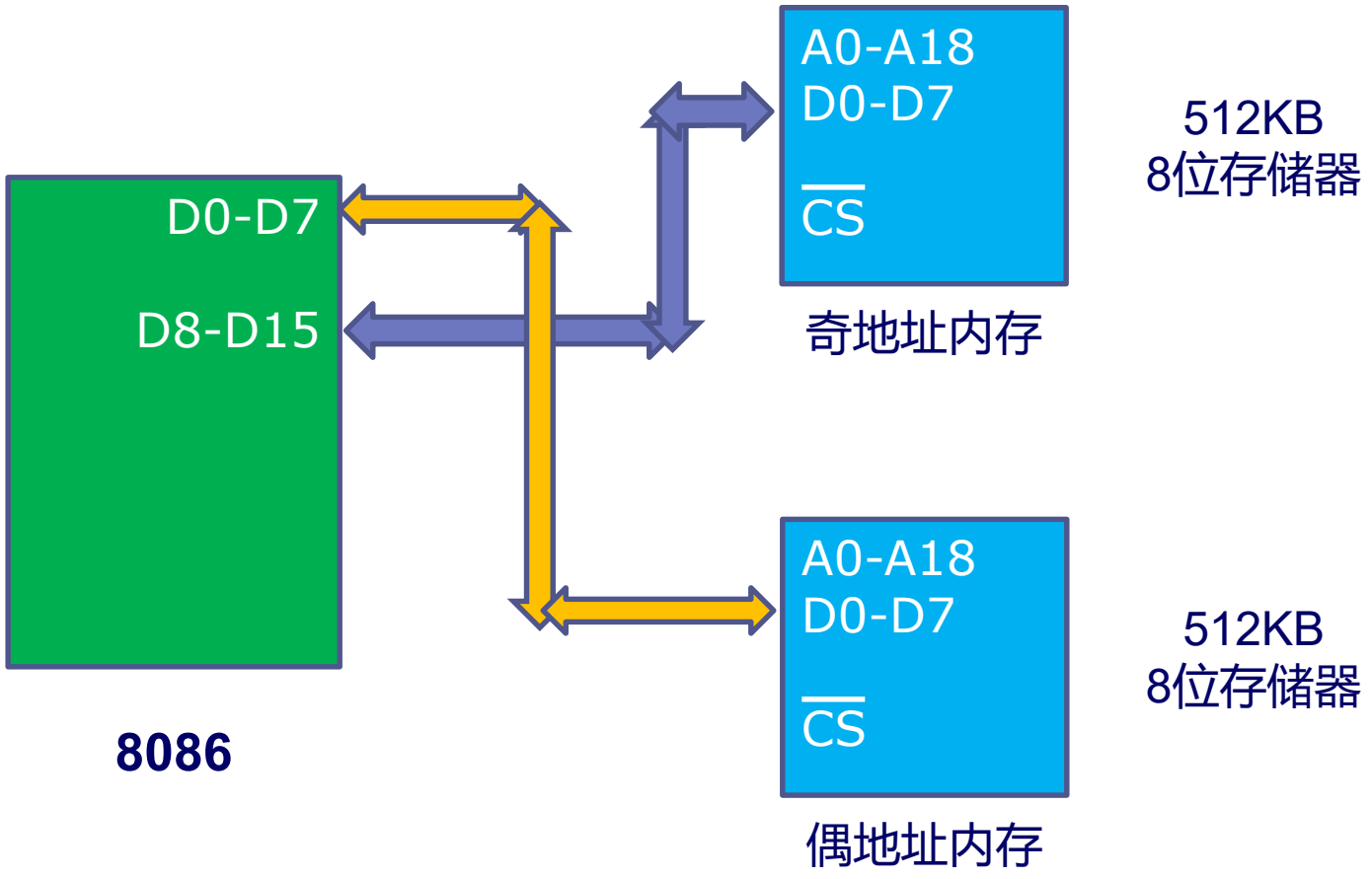


存在什么问题?

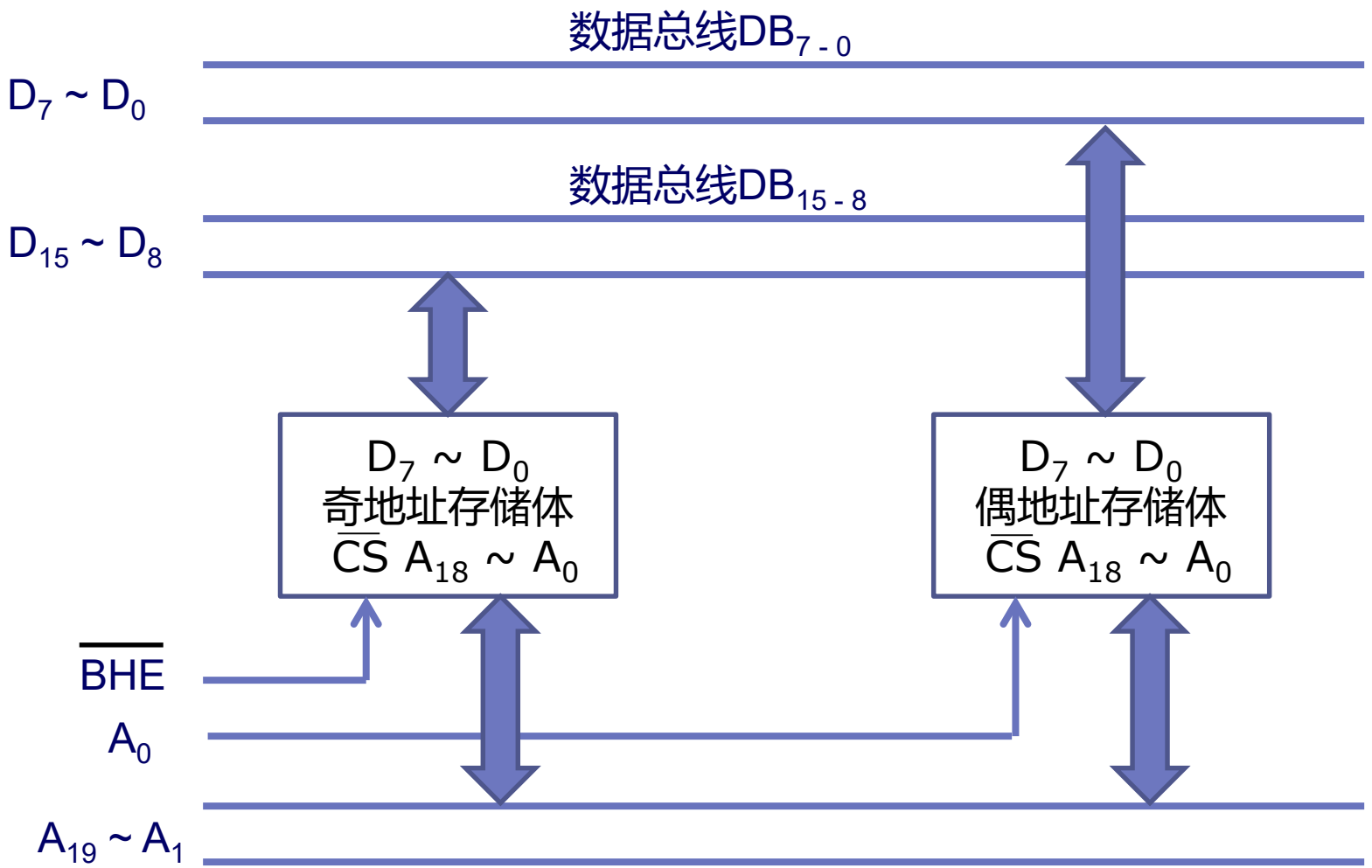
8086存储器分体结构



8086存储器分体结构



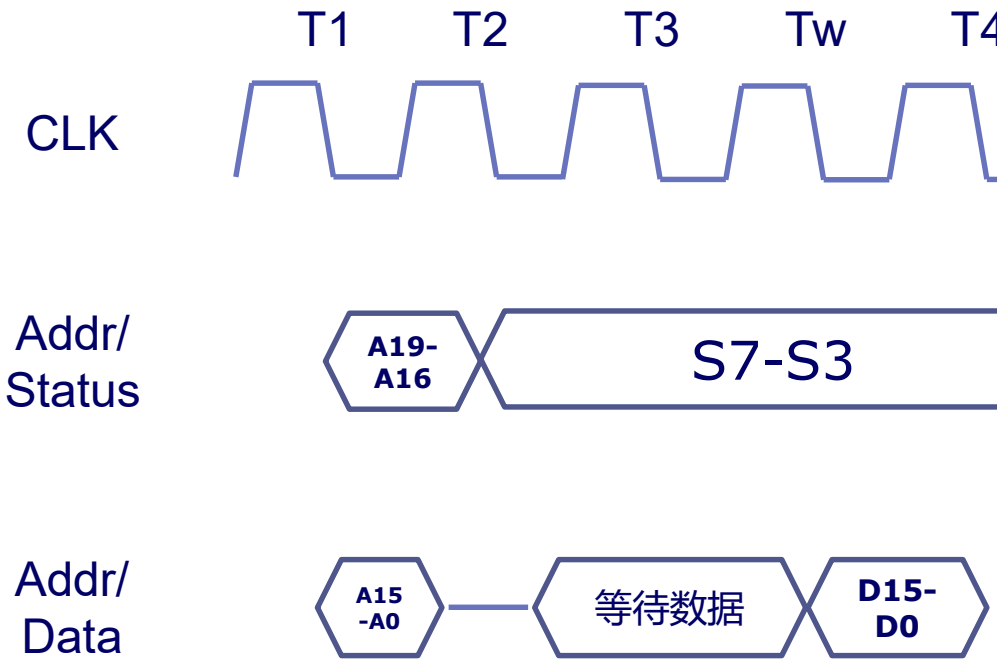
8086存储器与总线的连接



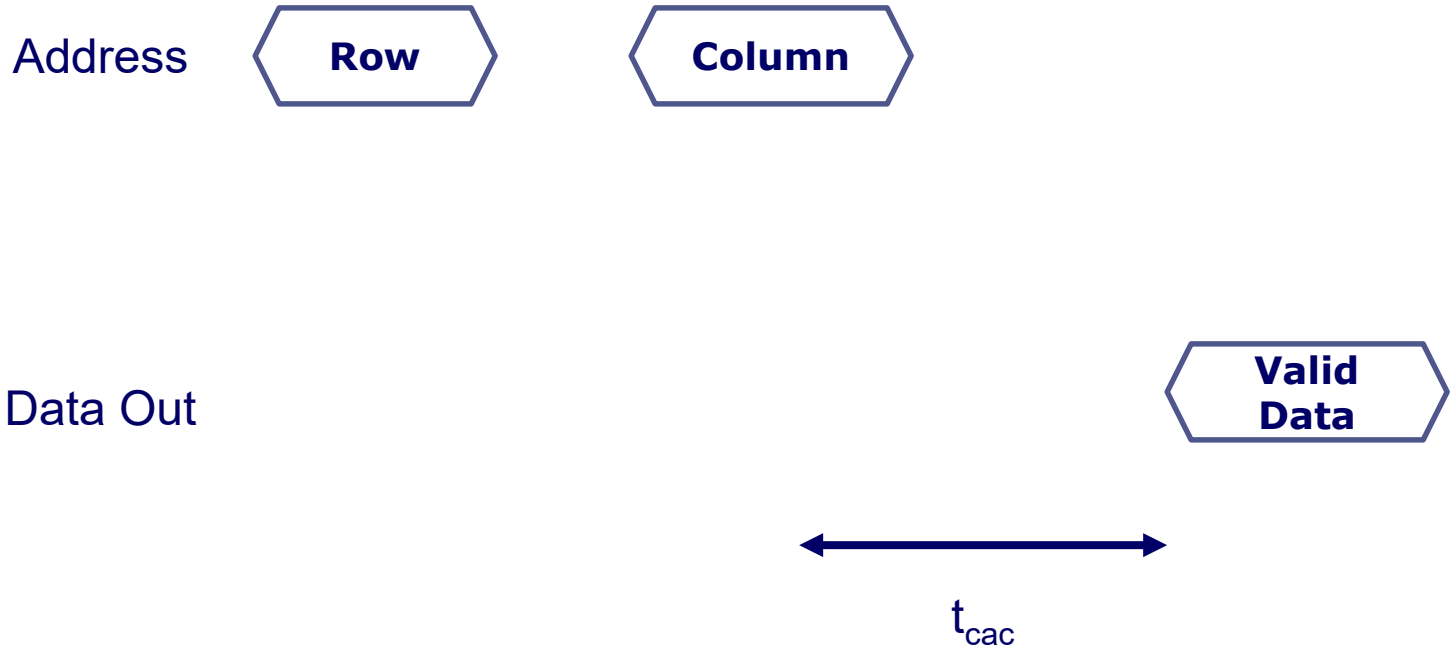
$\overline{BHE}$和A_0的编码含义

$\overline{BHE}$	A0	操作	总线使用情况
0	0	从偶地址开始读/写一个字	AD_{15} ~ AD_0
0	1	从奇地址单元开始读/写一个字节	AD_{15} ~ AD_8
1	0	从偶地址单元开始读/写一个字节	AD_7 ~ AD_0
1	1	无效	
0	1	从奇地址开始读/写一个字	AD_{15} ~ AD_8
1	0		AD_7 ~ AD_0

8086基本系统时序(读)(部分)



NEC μPD416内存时序(读)(部分)



T_{cac}最多135ns

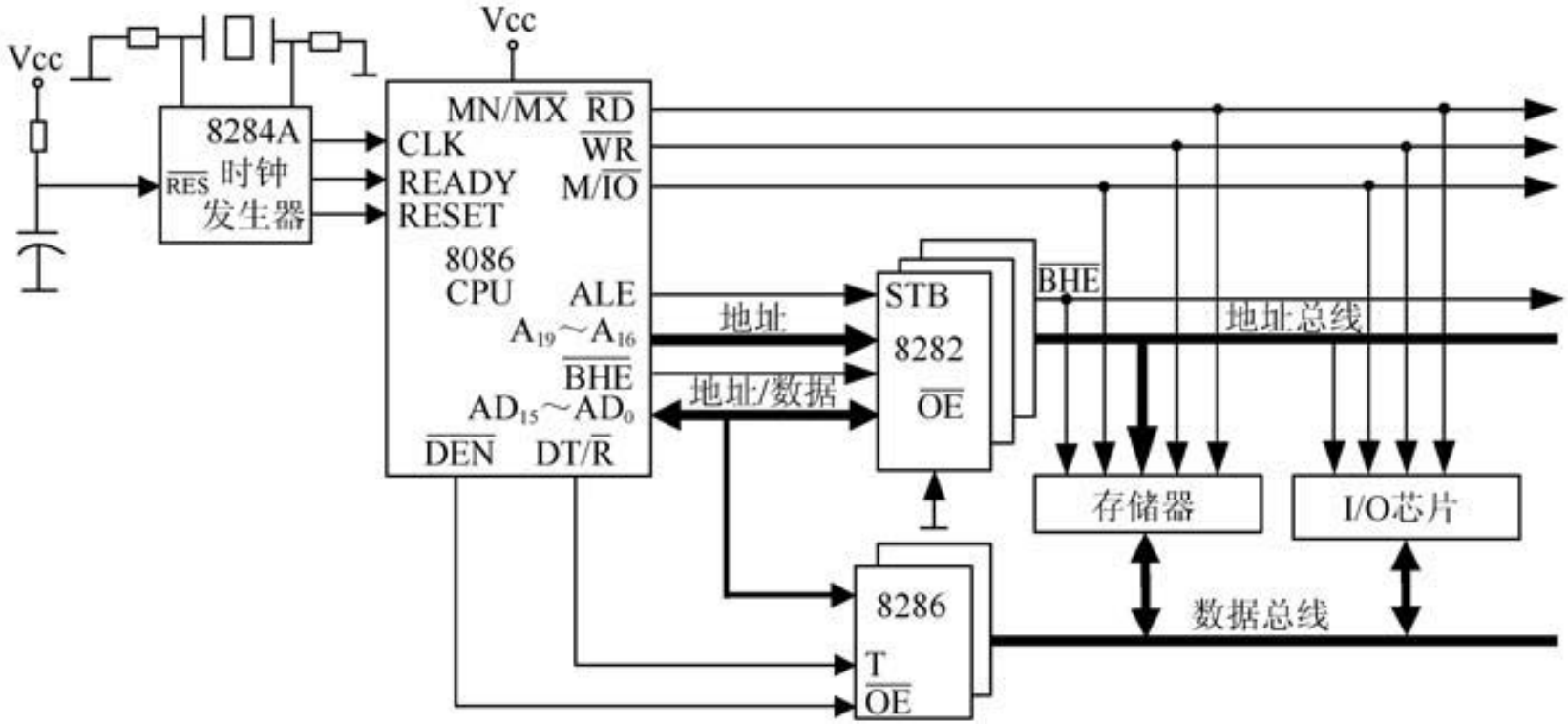
8086系统配置

- **最小模式**
  - MN/$\overline{MX}$接+5V
  - 构成单处理器系统
  - 系统控制信号由CPU提供
- **最大模式**
  - MN/$\overline{MX}$接地
  - 构成多处理器系统
  - 系统控制信号由总线控制器8288提供

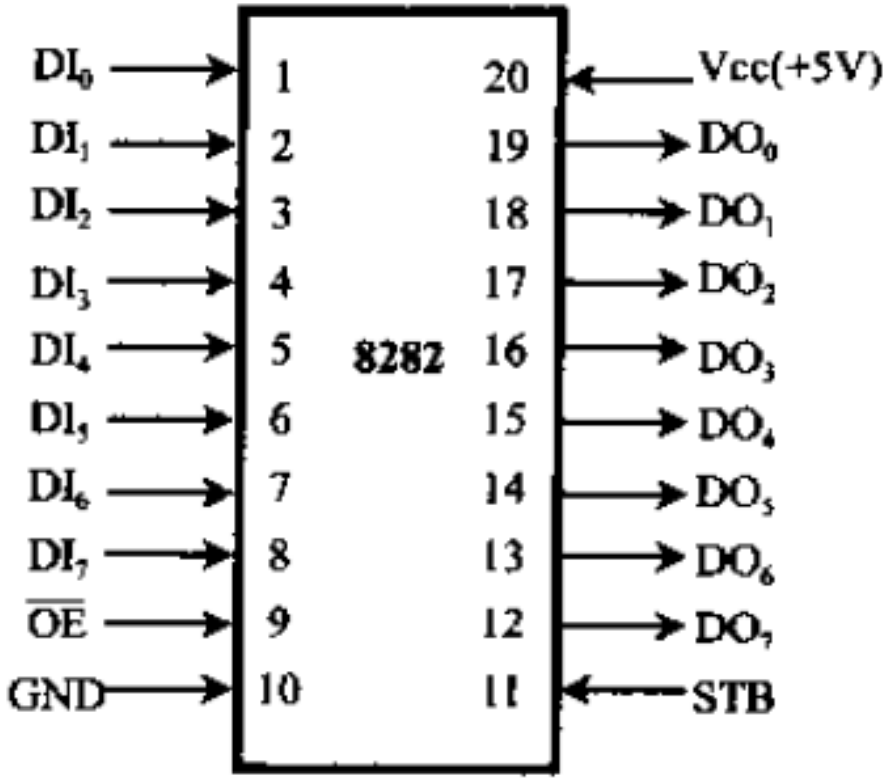
最小模式

- 1片8284A，作为时钟发生器
- 3片8282/8283或74LS373
  - 地址锁存器
- 2片8286/8287或74LS245
  - 作为双向数据总线收发器

最小模式系统配置


Vcc
Vcc
8284A
RES 时钟
发生器
MN/MX RD
CLK WR
READY M/IO
RESET
8086
CPU ALE
A19~A16
BHE
AD15~AD0
DEN DT/R
地址
地址/数据
STB
8282
OE
BHE
地址总线
存储器
I/O芯片
8286
T
OE
数据总线

8282引脚示意图

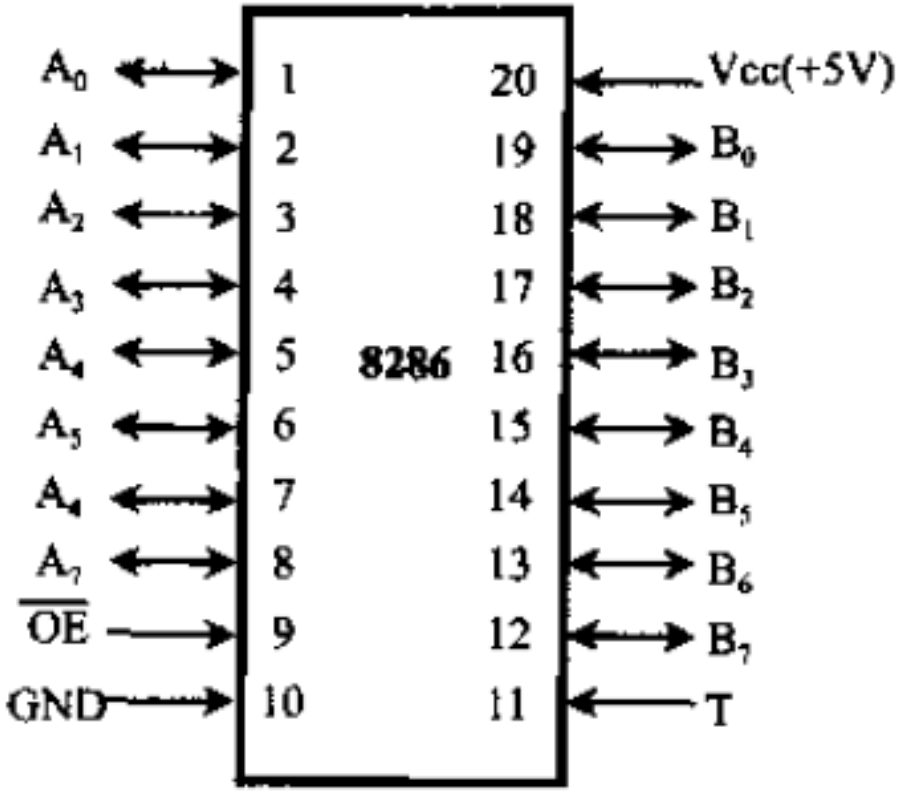

DI0
DI1
DI2
DI3
DI4
DI5
DI6
DI7
OE
GND
1
2
3
4
5
6
7
8
9
10
8282
20
19
18
17
16
15
14
13
12
11
Vcc(+5V)
DO0
DO1
DO2
DO3
DO4
DO5
DO6
DO7
STB

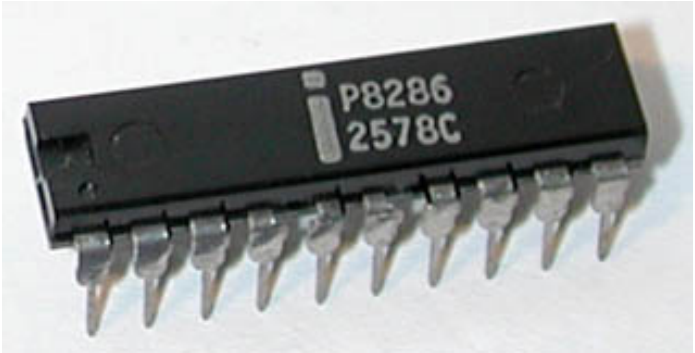

NEC
JAPAN
K0Y011-046
μPB8282C

8282引脚说明

- $DI_7 \sim DI_0$：8位数据输入
- $DO_7 \sim DO_0$：8位数据输出
- **STB**：
  - 选通信号
  - 为高时允许数据输入
- $\overline{OE}$：
  - 输出允许
  - 为低时允许数据输出

8286引脚示意图

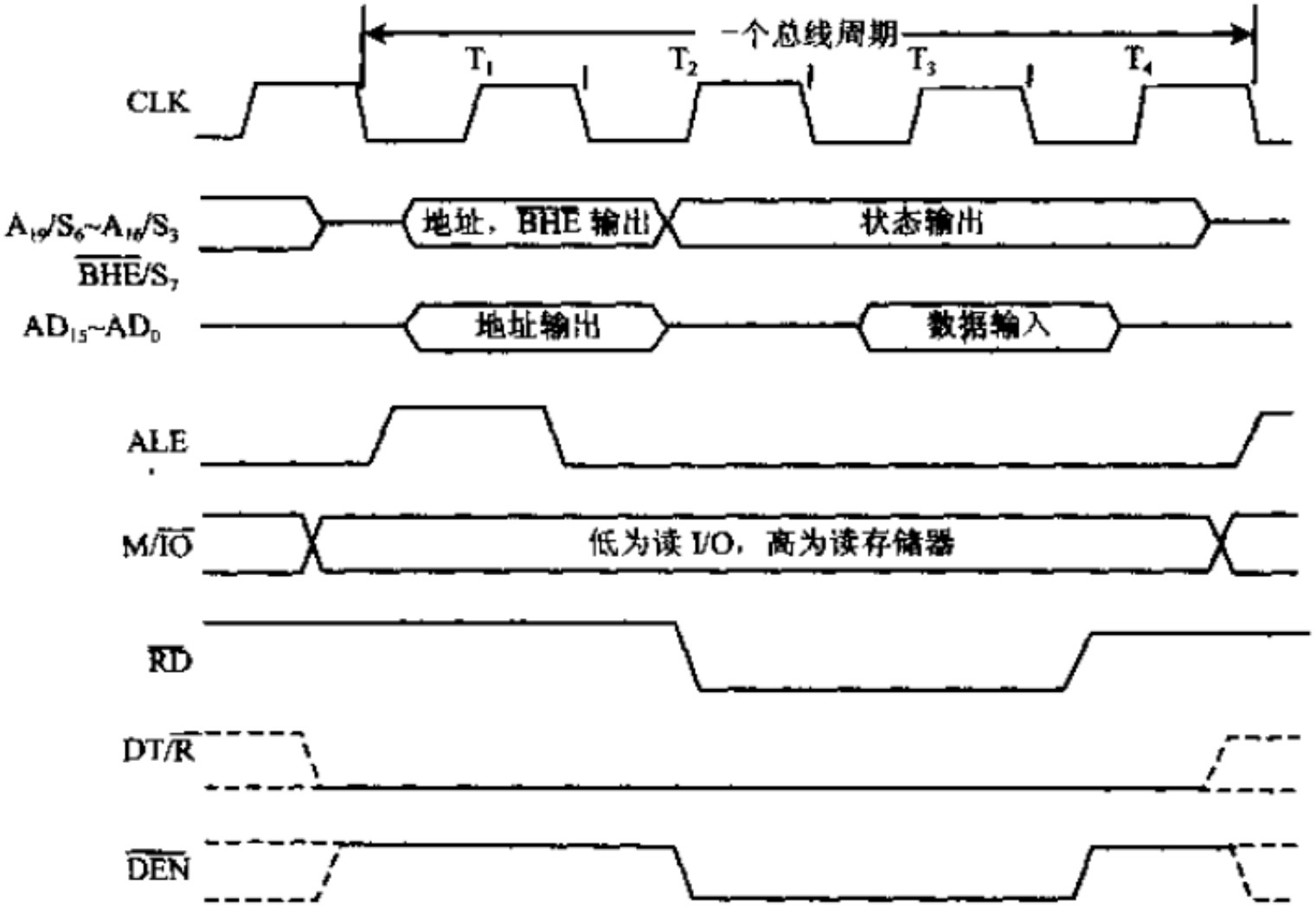

A0
A1
A2
A3
A4
A5
A6
A7
OE
GND
1
2
3
4
5
6
7
8
9
10
8286
20
19
18
17
16
15
14
13
12
11
Vcc(+5V)
B0
B1
B2
B3
B4
B5
B6
B7
T


P8286
2578C

8286引脚说明

- $A_7 \sim A_0$：输入/输出数据线
- $B_7 \sim B_0$：输入/输出数据线
- $\overline{OE}$：
  - 输出允许
  - 为低时允许数据通过8286
- T：
  - 数据传输方向
  - T = 1时：A → B
  - T = 0时：B → A

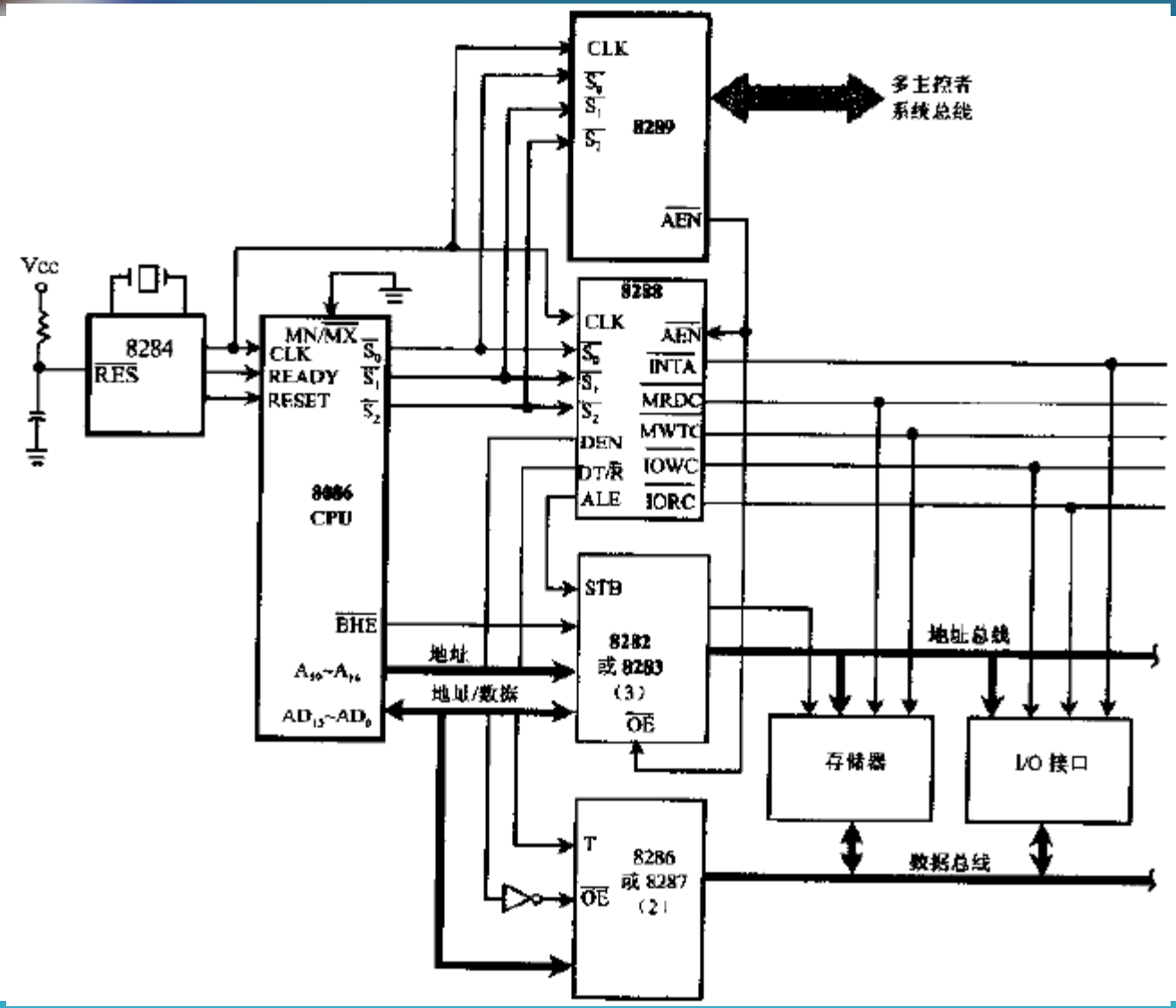
最小模式下的总线读操作



最大模式

- **8288总线控制器**
  - 输出原CPU所有的控制信号
- **8289总线总裁**
  - 裁决总线使用权赋予哪个处理器
- **形成以8086为中心的多处理器系统**

最大模式系统配置


CLK
$\overline{S_0}$
$\overline{S_1}$
$\overline{S_2}$
8289
$\overline{AEN}$
多主控者
系统总线
Vcc
8284
RES
MN/$\overline{MX}$
CLK
READY
RESET
$\overline{S_0}$
$\overline{S_1}$
$\overline{S_2}$
8086
CPU
$\overline{BHE}$
A_{19}~A_{16}
AD_{15}~AD_0
8288
CLK
$\overline{AEN}$
$\overline{S_0}$
$\overline{INTA}$
$\overline{S_1}$
$\overline{MRDC}$
$\overline{S_2}$
$\overline{MWTC}$
DEN
$\overline{IOWC}$
DT/$\overline{R}$
$\overline{IORC}$
ALE
STB
8282
或 8283
(3)
$\overline{OE}$
地址
地址/数据
地址总线
存储器
I/O 接口
T
8286
或 8287
(2)
$\overline{OE}$
数据总线

内容

I/O端口

- **CPU与外设通信时，传送的信息主要包括数据信息、状态信息和控制信息。在接口电路中，这些信息分别进入不同的寄存器，通常将这些寄存器和他们的控制逻辑统称为端口(Port)，CPU可对端口中的信息直接进行读写**
  - 数据端口
  - 状态端口
    - Ready, Busy, Error…
  - 命令端口
    - CPU发出的各种命令和控制字
    - 启动、停止、允许中断…

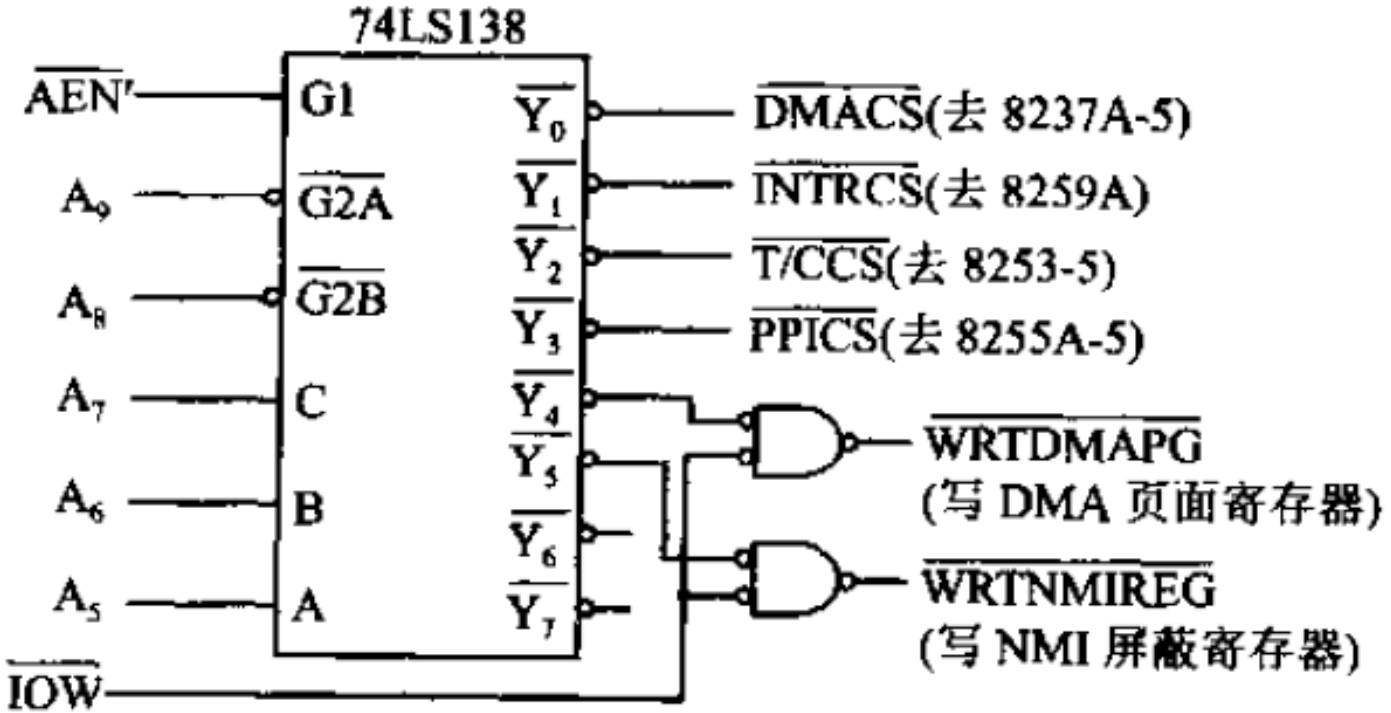
I/O端口的寻址方法

- **存储器映射式I/O**
- **独立编址I/O**
  - 8086采用这种方式
  - 地址总线低16位寻址I/O端口
  - 最多65536个输入或输出端口
  - 通过M/$\overline{IO}$控制信号区分

PC/XT的系统板I/O端口分布

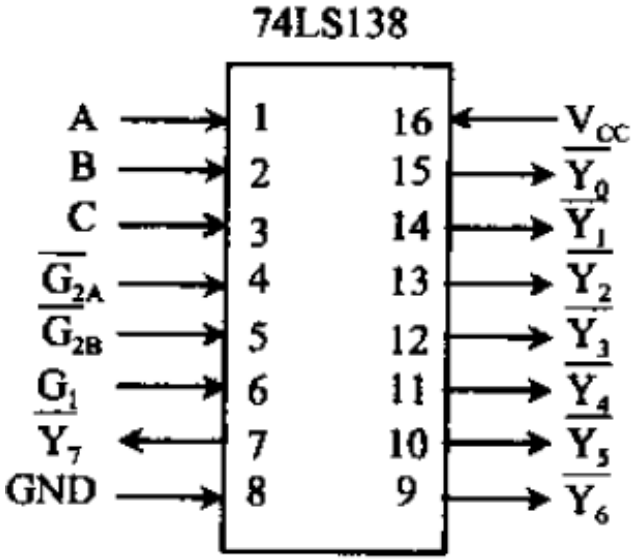
地址范围(H)	I/O设备(端口)
000 ~ 01F (00 ~ 0F)	8237A-5 DMA控制器
020 ~03F (20 ~ 21)	8259A 中断控制器
040 ~ 05F (40 ~ 43)	8253-5计数器/定时器
060 ~ 07F (60 ~ 63)	8255A-5 并行接口
080 ~ 09F (80 ~ 83)	DMA页寄存器
0A0 ~ 0BF (A0)	NMI屏蔽寄存器
0C0 ~ 0DF	保留
0E0 ~ 0FF	保留

PC/XT上I/O端口译码电路



MOV AL, 13H
OUT 20H, AL　**哪个设备?**
OUT 120H, AL　**哪个设备?**

74LS138



G_1	$\overline{G_{2A}}$	$\overline{G_{2B}}$	C	B	A	输 出
1	0	0	0	0	0	$\overline{Y_0}$ =0 其余为 1
1	0	0	0	0	1	$\overline{Y_1}$ =0 其余为 1
1	0	0	0	1	0	$\overline{Y_2}$ =0 其余为 1
1	0	0	0	1	1	$\overline{Y_3}$ =0 其余为 1
1	0	0	1	0	0	$\overline{Y_4}$ =0 其余为 1
1	0	0	1	0	1	$\overline{Y_5}$ =0 其余为 1
1	0	0	1	1	0	$\overline{Y_6}$ =0 其余为 1
1	0	0	1	1	1	$\overline{Y_7}$ =0 其余为 1

中断系统

- **中断源**
- **中断响应**
  - CPU检测INTR中断请求
  - CPU发送$\overline{\text{INTA}}$中断响应
  - 保护断点，转向中断服务程序
- **中断向量表**
- **中断优先级**
- **中断屏蔽**

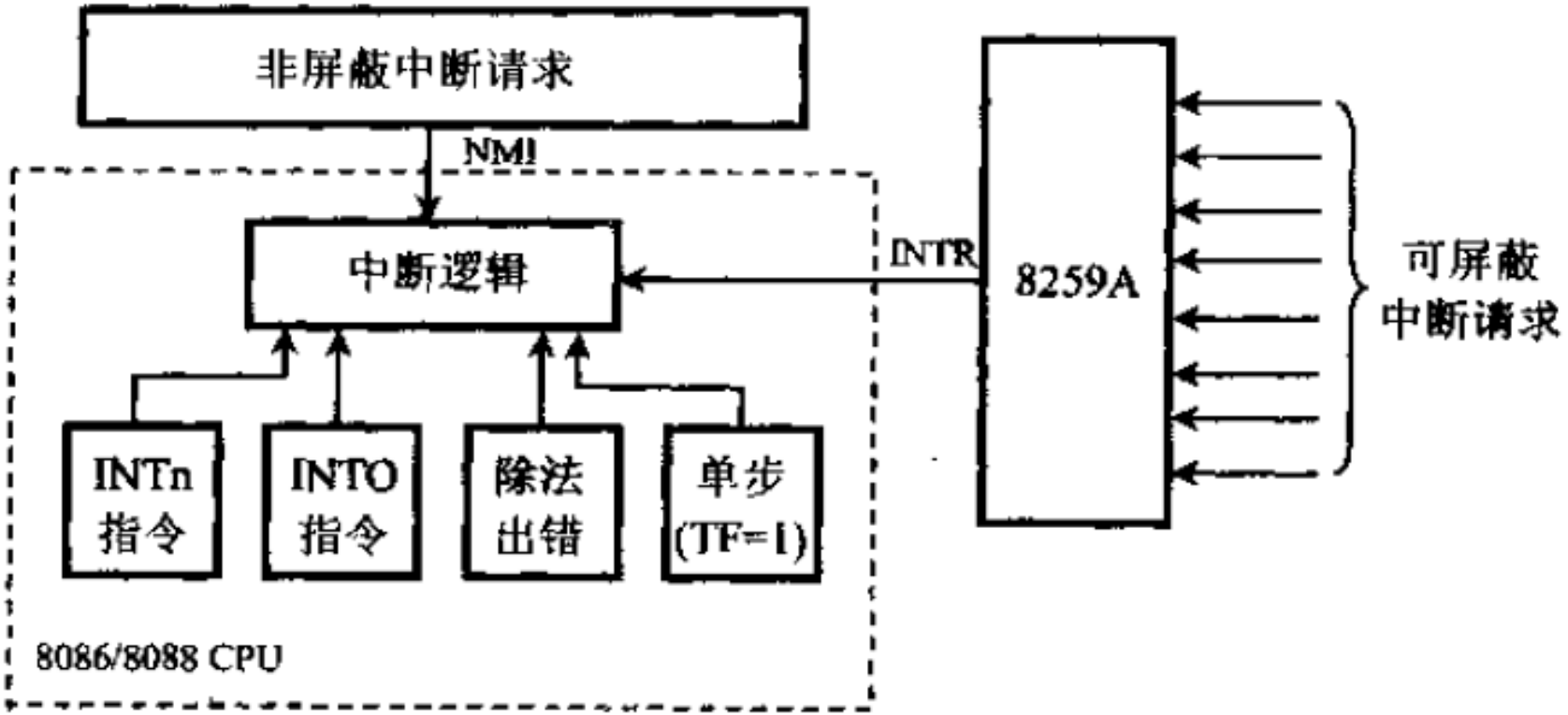
中断分类

- **8086支持256种中断**
- **外部中断**
  - 也称为硬件中断
  - 不可屏蔽中断
    - 从NMI引脚引入
    - 动态RAM奇偶校验错
    - I/O通道扩展版奇偶校验错
    - 8087协处理器中断请求
  - 可屏蔽中断
    - 从INTR引脚引入
    - 通过可编程中断控制器8259A管理

中断分类

- **内部中断**
  - 软件中断
  - 由中断指令INT引起
  - 由CPU某些运算错误引起
    - 除法错
    - 溢出
  - 由调试程序debug设置的中断

中断源

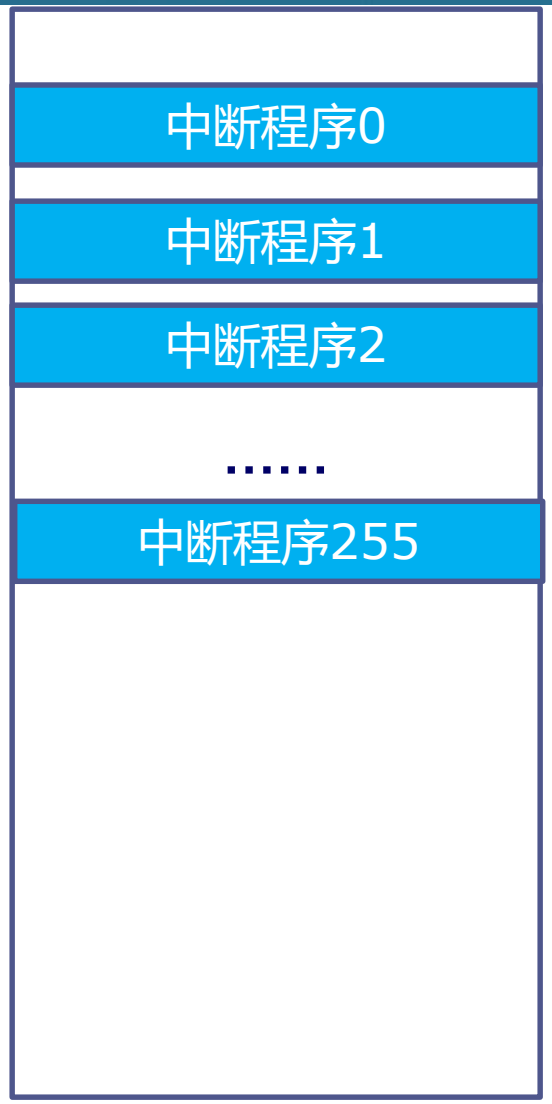

非屏蔽中断请求
NMI
中断逻辑
INTR
8259A
可屏蔽
中断请求
INTn
指令
INTO
指令
除法
出错
单步
(TF=1)
8086/8088 CPU

要解决的问题

- **如何知道是哪个中断源**
  - 究竟是256种中断中的哪一个
- **如何找到中断处理程序**

中断程序分布

0000：0000H



内存地址空间

9000：FFFFH

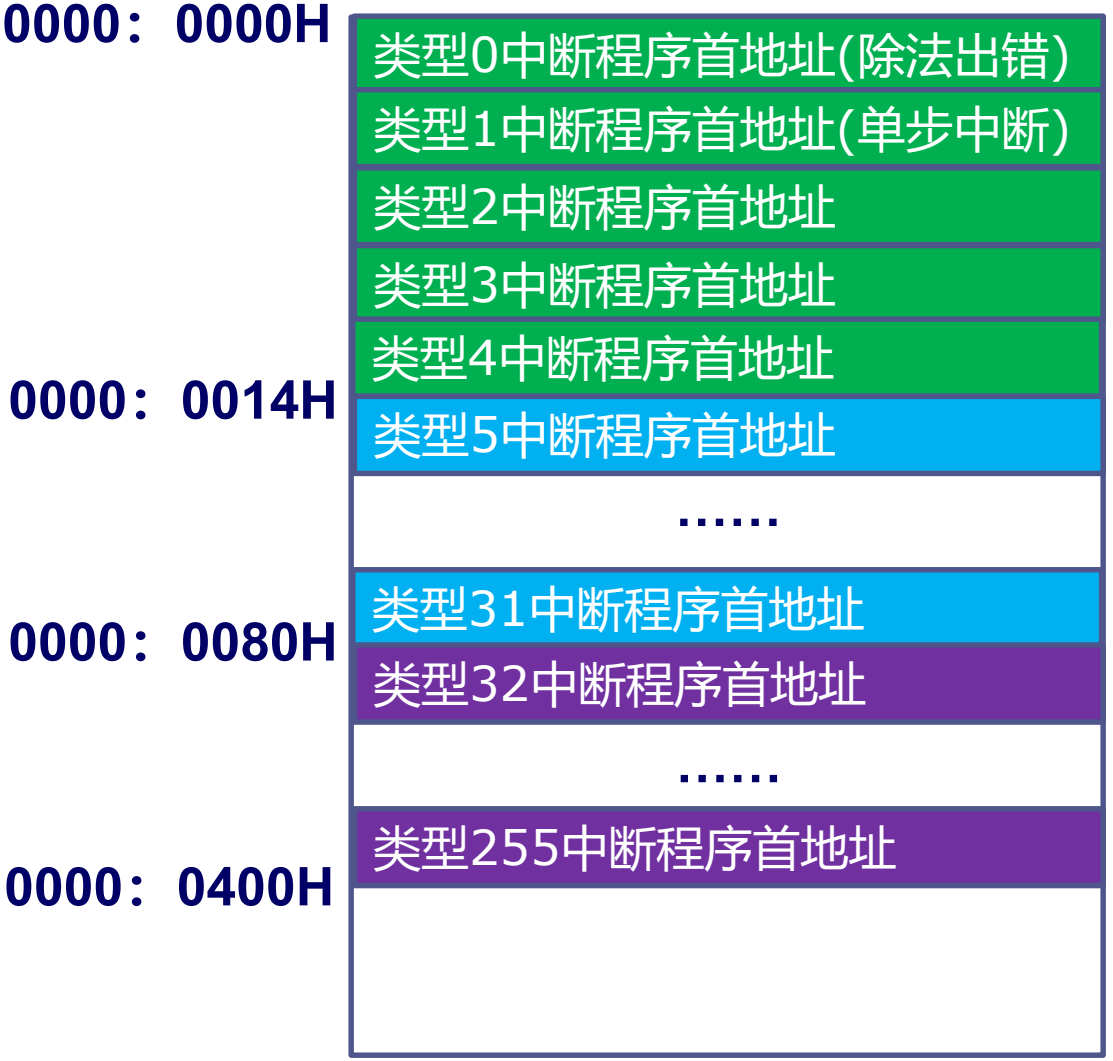
中断向量表

- 中断程序入口地址表
- 位于存储器 00000H – 003FFH

专用中断

系统中断

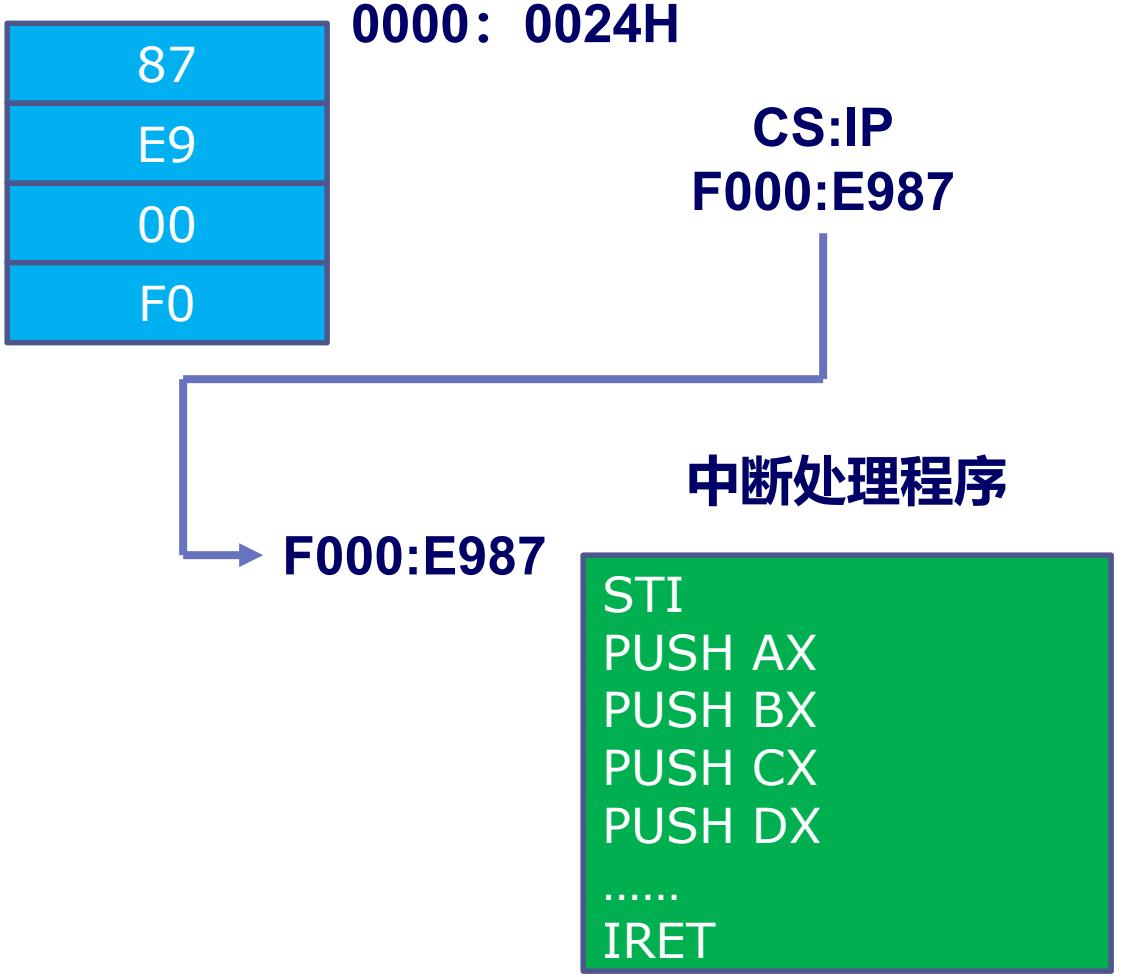
用户中断



中断操作过程

键盘中断向量9

9 × 4 = 24H



中断向量的设置

- **0-4为专用中断，入口地址由系统定义，用户不能修改**
- **5-31为系统使用中断，用户不应修改**
- **其余中断可由用户定义**
  - 部分中断已有定义
    - 如20H-3FH的DOS中断

标志操作指令

- **CLI**
  - 清除IF标志，禁止CPU响应中断
- **STI**
  - 设置IF标志，允许CPU响应中断

数据项定义

- **SEG**
  - SEG variable
  - 变量所在段的段基址
- **OFFSET**
  - OFFSET variable
  - 变量地址与所在段首地址之间的偏移字节数

修改9号中断入口地址

```
        CLI
        MOV AX, 0
        MOV ES, AX                  ; ES = 0
        MOV BX, 9 * 4               ; BX为中断向量表项地址
        MOV AX, OFFSET INTRAD
        MOV ES:[BX], AX             ; 中断程序入口段内偏移
                                    ; 0000:0024H
        MOV AX, SEG INTRAD
        MOV ES:[BX+2], AX           ; 中断程序入口段地址
        STI

INTRAD:                             ; 新的中断程序
        …
        IRET
```

中断程序编写

- **主程序中的初始化**
  - 设置中断向量
  - 设置8259A的中断屏蔽寄存器的中断屏蔽位
  - 设置CPU中断允许标志IF(开中断STI)
- **硬件(外设接口)和CPU自动完成**
  - 外部中断控制器接口向CPU INTR发中断请求
  - CPU发$\overline{\text{INTA}}$信号给外部中断控制器接口
  - CPU取中断向量号n
  - CPU自动将flags, CS, IP内容入栈保护
  - 清除IF, TF，禁止外部中断和单步中断
  - 从中断向量表中取CS:IP
  - 转向中断服务子程序

中断程序编写

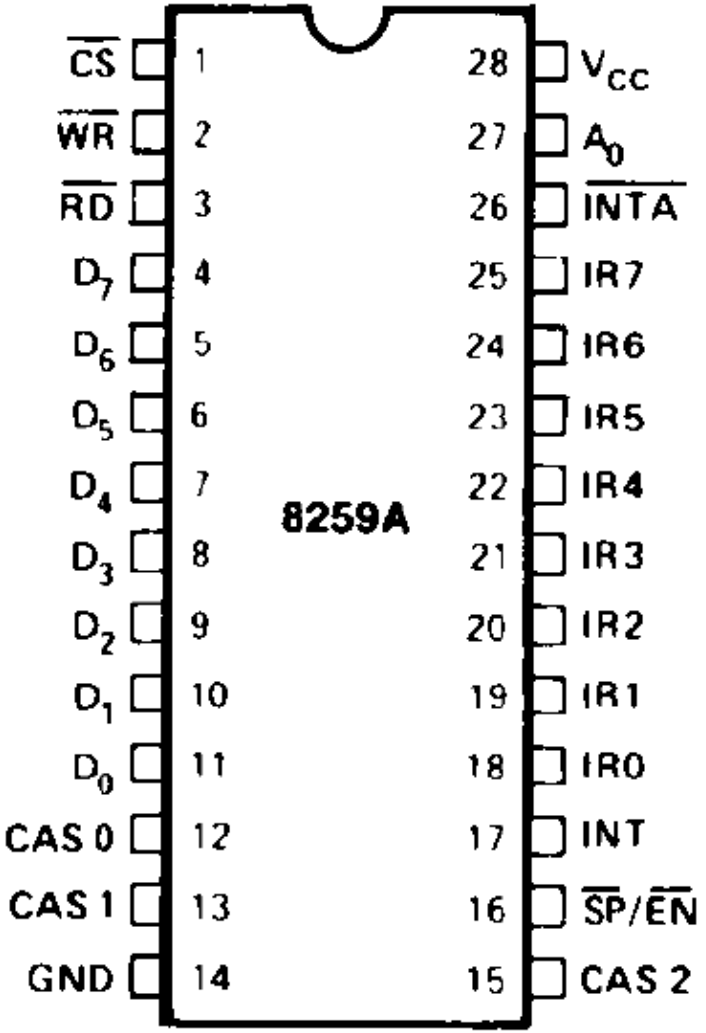
- **中断服务子程序**
  - 保护现场，通过一系列PUSH指令将各寄存器入栈
  - 若允许中断嵌套，使用STI指令开中断
  - 执行中断处理程序
  - 使用CLI关中断
  - 发送中断结束命令EOI，清除当前正在处理的中断请求标志
  - 恢复现场，通过一系列POP指令
  - 调用IRET返回

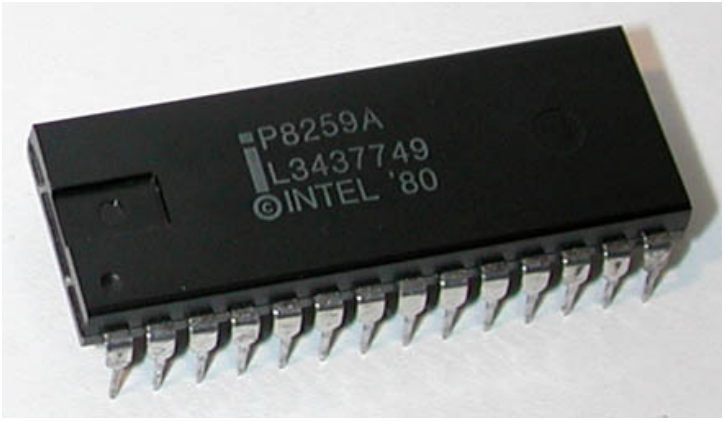
内容

8259A可编程中断控制器

- 具有8级优先级控制，通过级联可以扩展到64级优先级控制
- 每一级中断可由程序单独屏蔽或允许
- 可提供中断类型号给CPU
- 可以通过编程选择多种不同工作方式

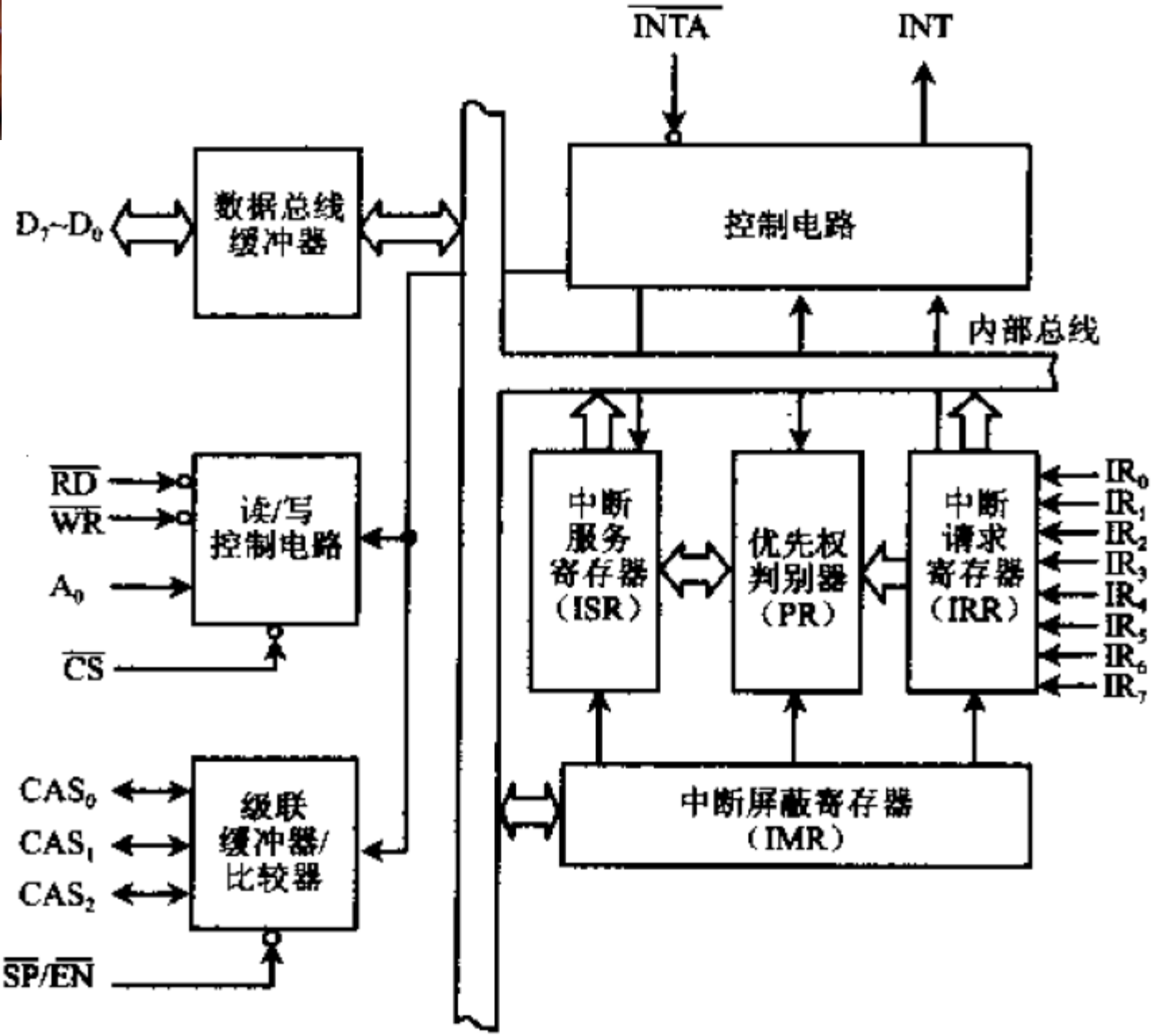
8259A引脚示意图


CS 1
WR 2
RD 3
D7 4
D6 5
D5 6
D4 7
D3 8
D2 9
D1 10
D0 11
CAS 0 12
CAS 1 13
GND 14
28 VCC
27 A0
26 INTA
25 IR7
24 IR6
23 IR5
22 IR4
21 IR3
20 IR2
19 IR1
18 IR0
17 INT
16 SP/EN
15 CAS 2
8259A


P8259A
L3437749
©INTEL '80

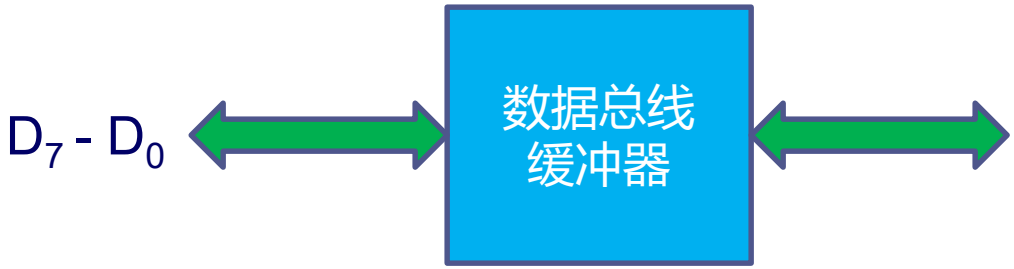
8259A引脚说明

- $D_7 \sim D_0$：双向数据线，与系统数据总线相连
- $IR_7 \sim IR_0$：外设的中断请求信号输入端
- $\overline{RD}$：读命令信号
- $\overline{WR}$：写命令信号
- $\overline{CS}$：片选信号
- A_0：选择8259A的两个端口，连低位地址线
- INT：向CPU发出中断请求，连CPU的INTR
- $\overline{INTA}$：CPU给8259A的中断响应信息
- $CAS_2 \sim CAS_0$：双向级联信号线
- $\overline{SP}/\overline{EN}$：从片编程/允许缓冲信号


INTA
INT
D_7~D_0
数据总线缓冲器
控制电路
内部总线
RD
WR
A_0
CS
读/写控制电路
中断服务寄存器（ISR）
优先权判别器（PR）
中断请求寄存器（IRR）
IR_0
IR_1
IR_2
IR_3
IR_4
IR_5
IR_6
IR_7
CAS_0
CAS_1
CAS_2
级联缓冲器/比较器
中断屏蔽寄存器（IMR）
SP/EN

8259A内部结构

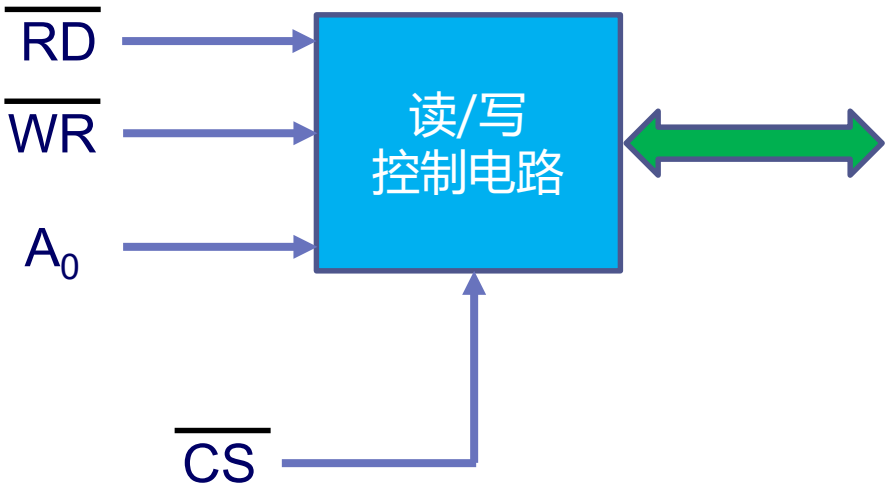
- **数据总线缓冲器**
  - 连低8位数据总线D_7 ~ D_0
  - 内部命令字的读写
  - 中断向量输出



中断向量值是多少?

8259A内部结构

- **读写控制电路**
  - $\overline{RD}$、$\overline{WR}$、$\overline{CS}$和A_0



MOV AL, 13H
OUT 21H, AL ?

在Intel 8086文档中

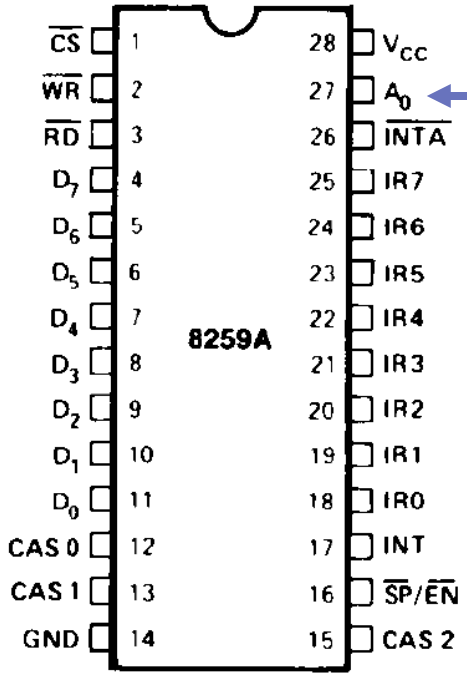
I/O ports are addressed in the same manner as memory locations.

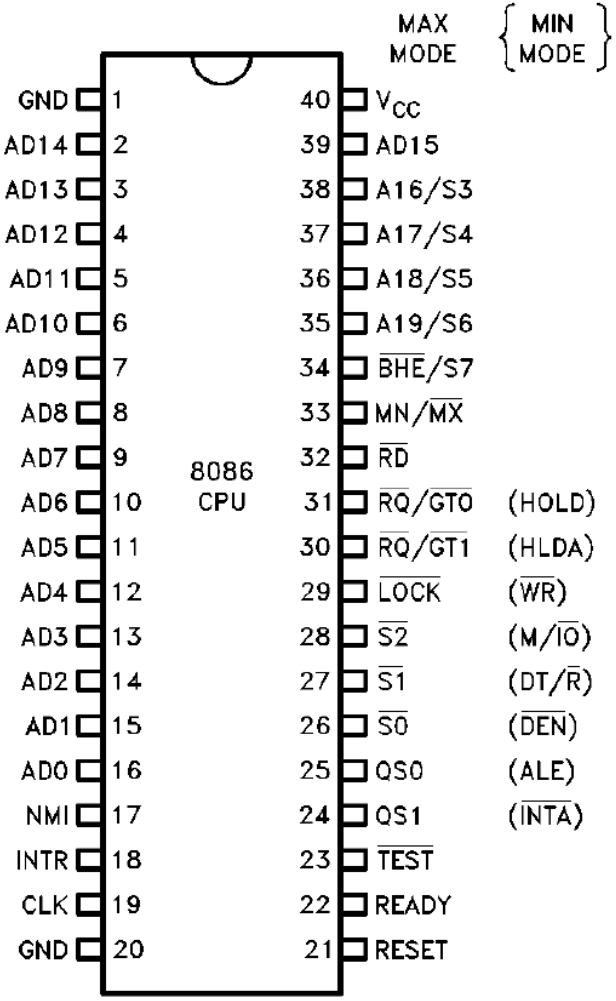
Even addressed bytes are transferred on the D7~D0 bus lines and odd addressed bytes on D15~D8.

在Intel 8086文档中 (续)

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**.

Intel 8086与8259地址线连接

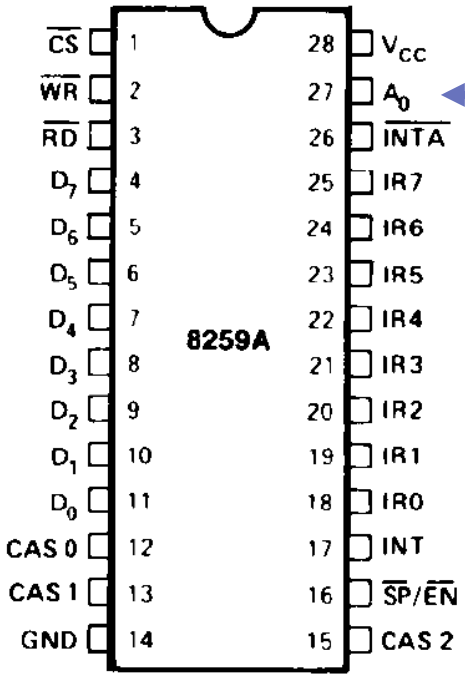


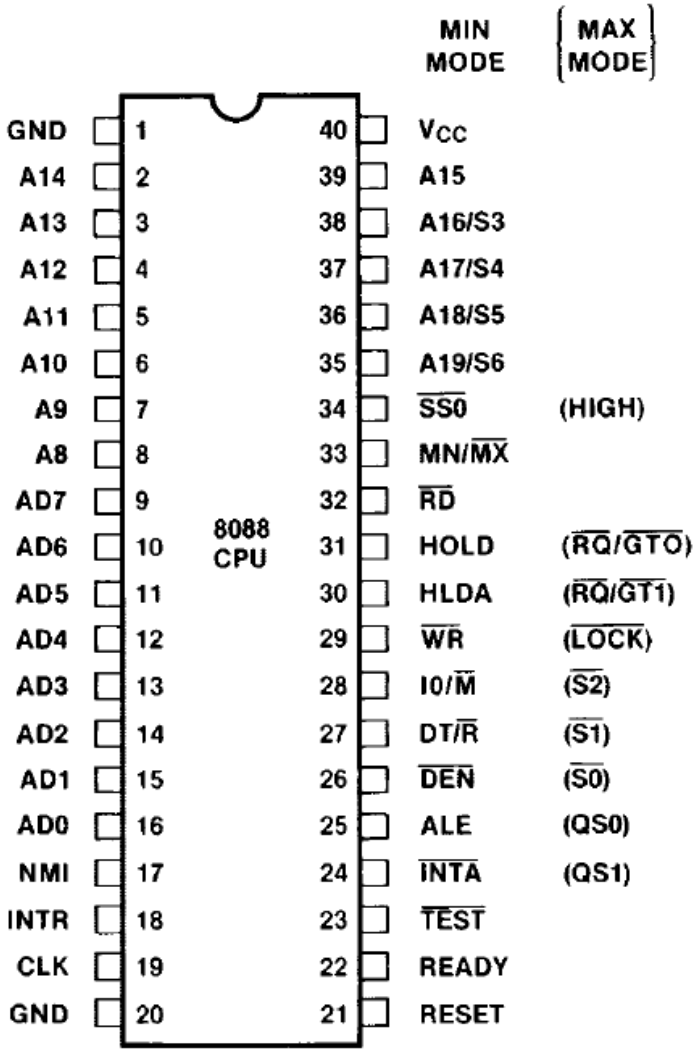


AD1

20, 22两个端口

Intel 8088与8259地址线连接





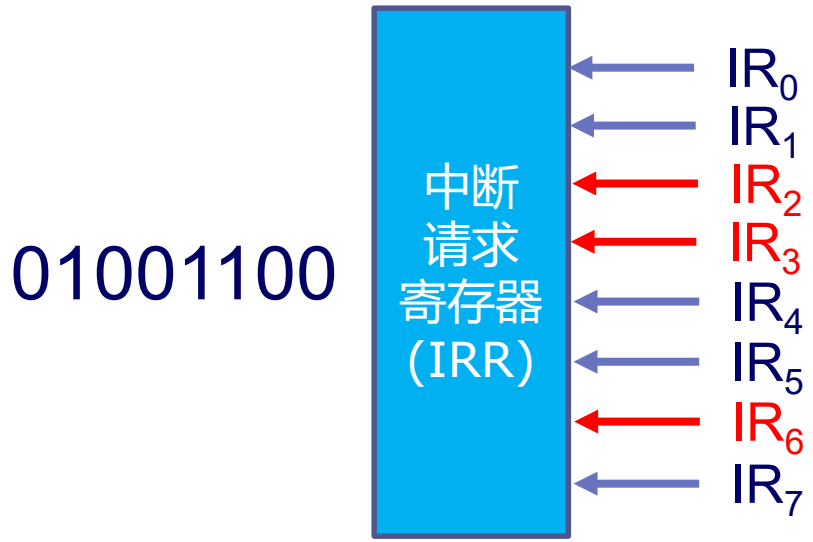
AD0

20, 21两个端口

8259A内部结构

中断请求寄存器IRR

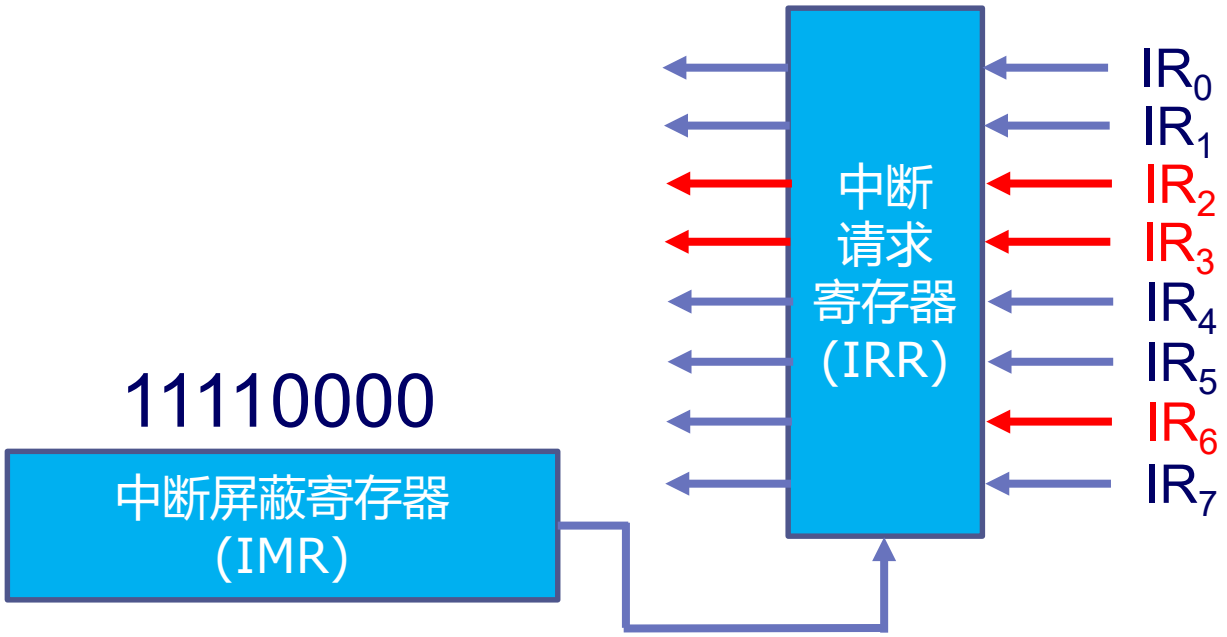
- 8位寄存器，存放中断请求信号IR_7 ~ IR_0
- IR有中断请求时，对应位置1
- 中断请求相应时，对应位置0



8259A内部结构

中断屏蔽寄存器IMR

- 8位寄存器，存放中断屏蔽信息
- 某一位置0，允许对应位的中断请求
- 某一位置1，不允许对应位的中断请求

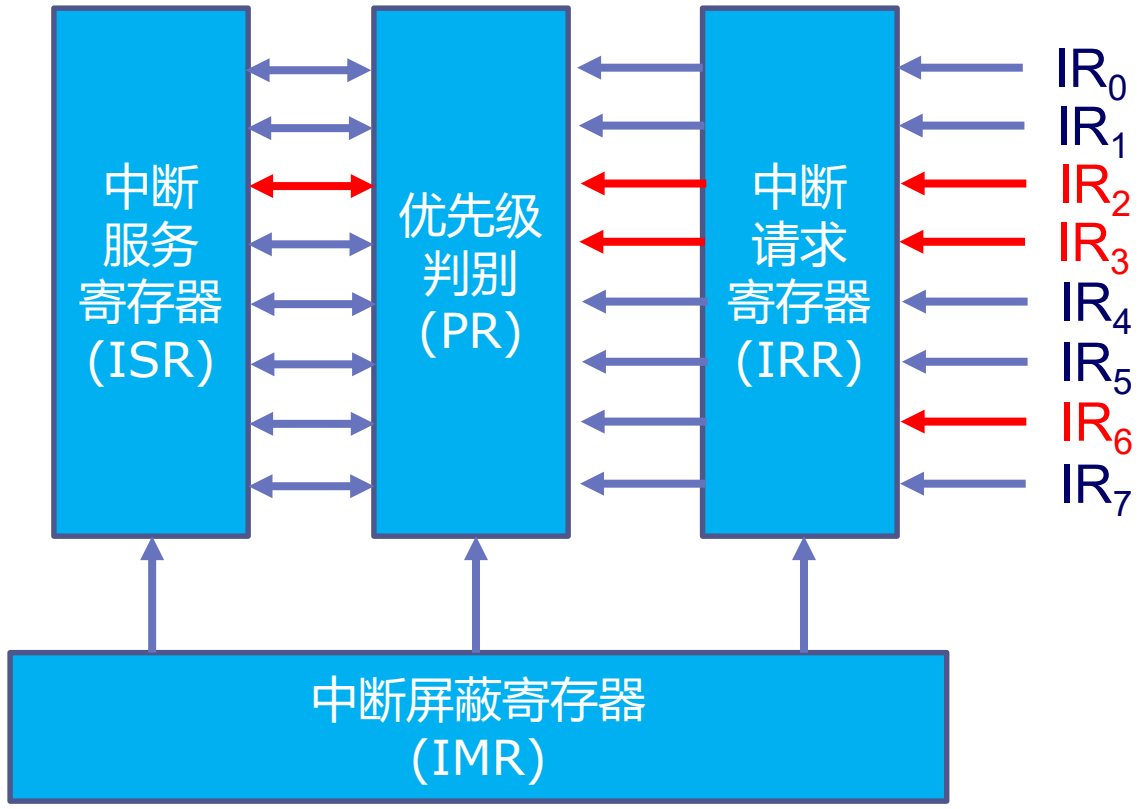


8259A内部结构

- **优先级判别逻辑PR**
  - 选出最高优先级的中断请求到中断服务寄存器ISR中
- **中断服务寄存器ISR**
  - 8位寄存器，保存正在处理中的中断请求信号
  - 1表示对应位的中断正在处理，0反之

8259A内部结构

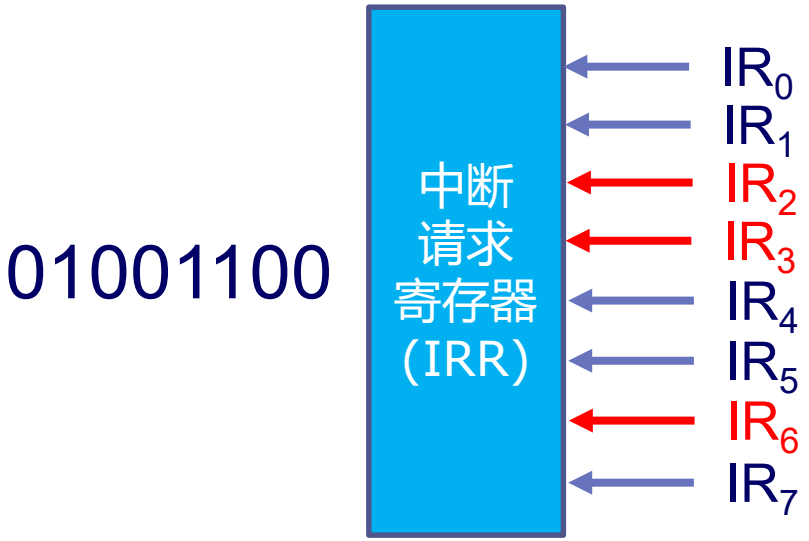
00000100



11110000

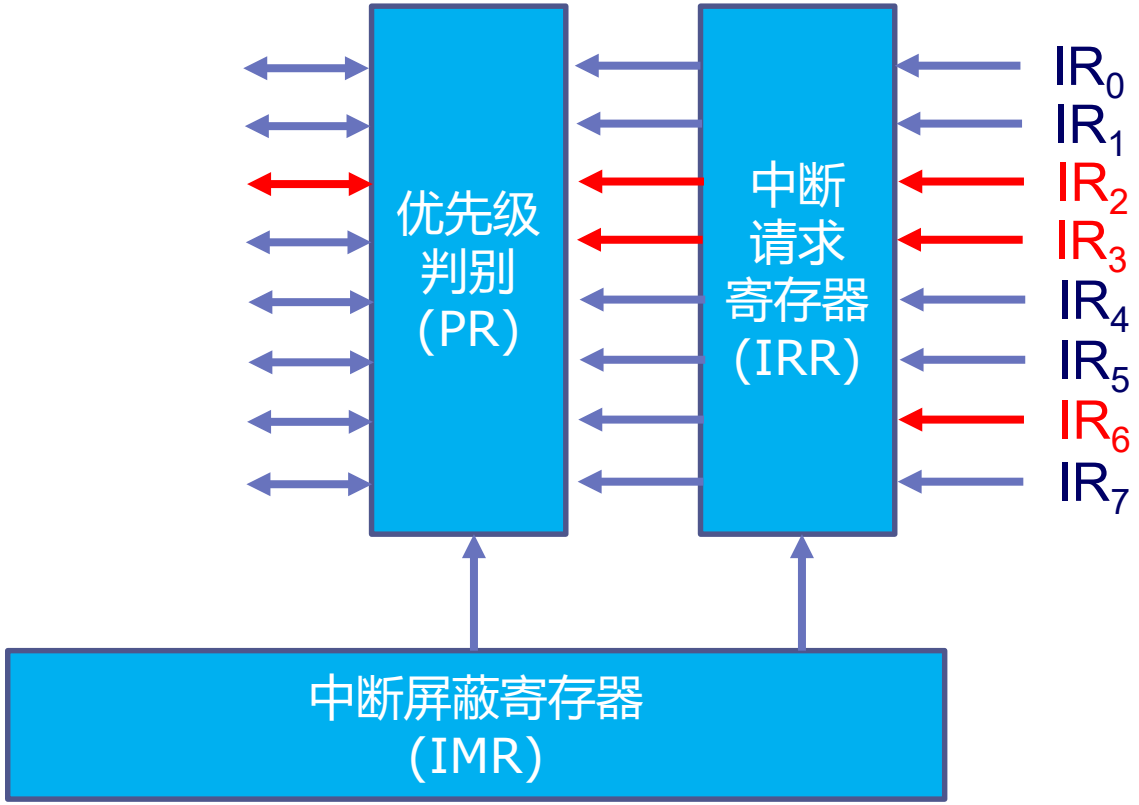
8259中断顺序

- 1个或多个中断输入线产生高电平，设置对应的IRR寄存器位



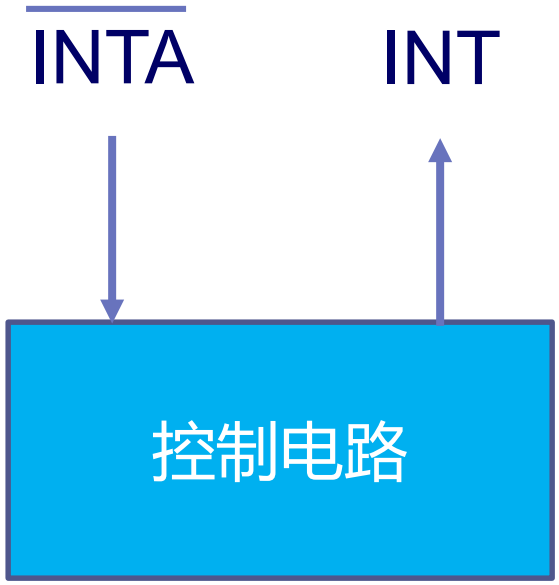
8259中断顺序

- 选出应该被最优先处理的中断



8259中断顺序

- 8259通过INT引脚通知CPU
- CPU通过$\overline{\text{INTA}}$引脚应答8259



8259中断顺序

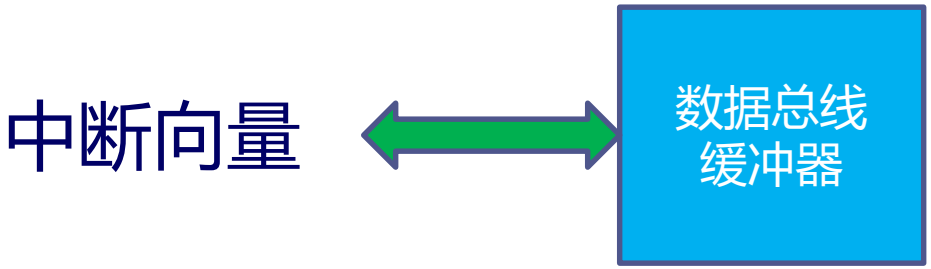
- 设置对应的ISR寄存器位
- 清除对应的IRR寄存器位

00000100

01001000

8259中断顺序

- CPU通过第二次$\overline{\text{INTA}}$信号通知8259
- 8259将中断向量值送到数据总线上



8259中断顺序

00000100

什么时候清除ISR寄存器位?

8259优先级

- **Fully Nested Mode**
  - 初始时IR0至IR7优先级从高到低(IR0最高)
  - 只允许更高优先级的中断(拒绝同级或低级中断)

- **Special Fully Nest Mode**
  - 允许同级中断
  - 用于级联的场景

8259优先级

❖ 优先级自动循环

- 处理完一级中断后，其优先级变为最低

原始状态	ISR内容	ISR_7	ISR_6	ISR_5	ISR_4	ISR_3	ISR_2	ISR_1	ISR_0
		0	0	**1**	0	0	**1**	0	0
	优先级	7	6	5	4	3	2	1	0
处理完IR_2	ISR内容	0	0	**1**	0	0	0	0	0
	优先级	**4**	**3**	**2**	**1**	**0**	**7**	**6**	**5**
处理完IR_5	ISR内容	0	0	0	0	0	0	0	0
	优先级	**1**	**0**	**7**	**6**	**5**	**4**	**3**	**2**

End of Interrupt (EOI)

- **普通EOI命令**
  - 中断例程返回前调用该命令
  - Specific EOI命令
    - 命令中指定当前正要结束的中断向量
    - 被指定的ISR寄存器位清零
  - Non-Specific EOI命令
    - ISR寄存器中，最高优先级的1清零
- **自动EOI命令**
  - 在CPU第二次$\overline{\text{INTA}}$信号后自动发送Non-Specific EOI命令

中断源屏蔽方式

- 通过对中断屏蔽寄存器IMR的操作可实现对某几位中断的屏蔽
- 普通中断屏蔽方式
  - 将IMR某一位或者几位置”1”，即可将对应中断屏蔽
- 特殊中断屏蔽方式

中断请求引入方式

- **边沿触发方式**
  - 输入端出现的上升沿作为中断请求信号
  - 输入端可一直维持高电平
- **电平触发方式**
  - 输入端出现的高电平作为中断请求信号
  - 中断响应后，输入端必须及时撤出高电平
- **中断查询方式**
  - 通过软件查询确定中断源

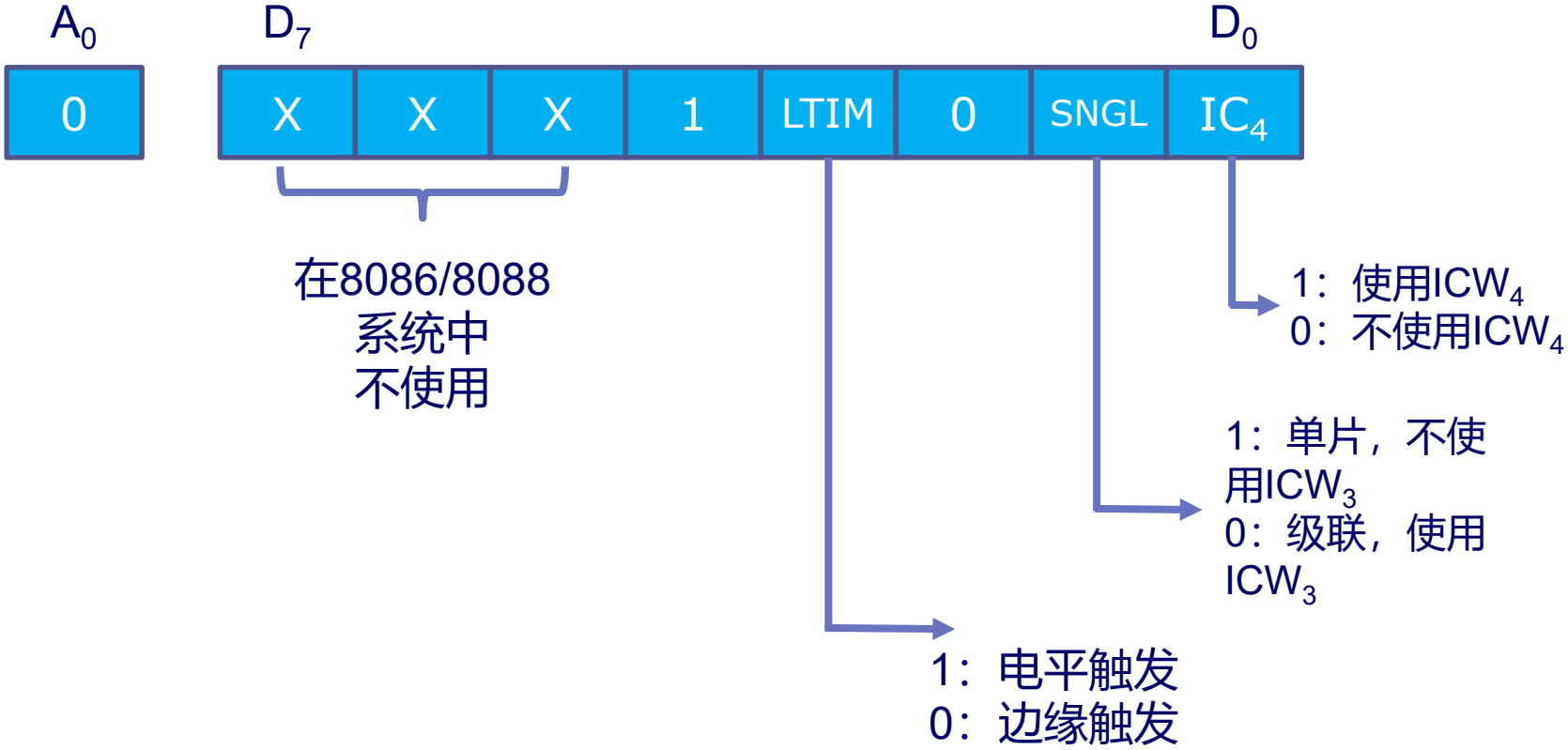
8259A编程结构

- **初始化命令字**
  - 寄存器ICW_1 ~ ICW_4
  - 初始化程序设置
  - 一经设定，工作过程中就不再改变
- **操作命令字**
  - 寄存器OCW_1 ~ OCW_3
  - 对中断处理过程进行控制
  - 可以在运行过程中重新设置

8259A读写功能

$\overline{CS}$	$\overline{RD}$	$\overline{WR}$	A0	D4	D3	读写操作	指令
0	1	0	0	1	X	CPU → ICW1	OUT
0	1	0	1	X	X	CPU → ICW2, ICW3, ICW4, OCW1	
0	1	0	0	0	0	CPU → OCW2	
0	1	0	0	0	1	CPU → OCW3	
0	0	1	0			IRR/ISR → CPU	IN
0	0	1	1			IMR → CPU	
1	X	X	X			高阻	
X	1	1	X			高阻	

ICW_1

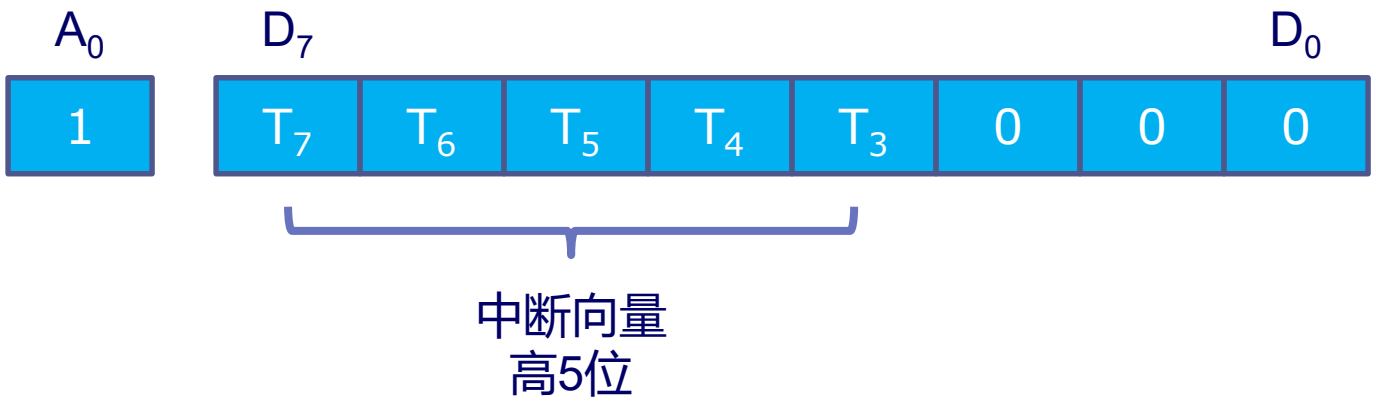


IBM PC/XT：

```
MOV AL, 13H
OUT 20H, AL
```

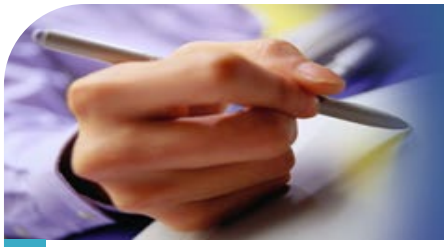
ICW_2

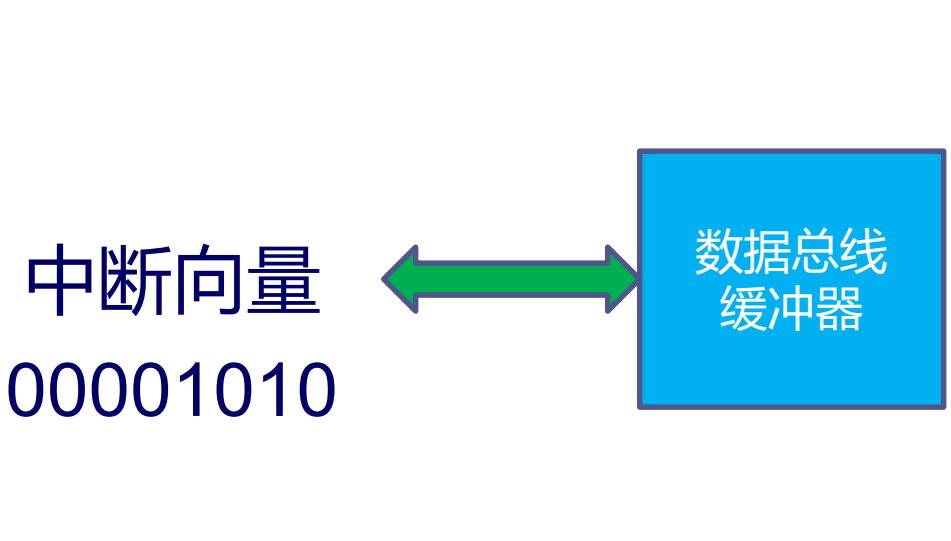
- 设置中断向量初始值



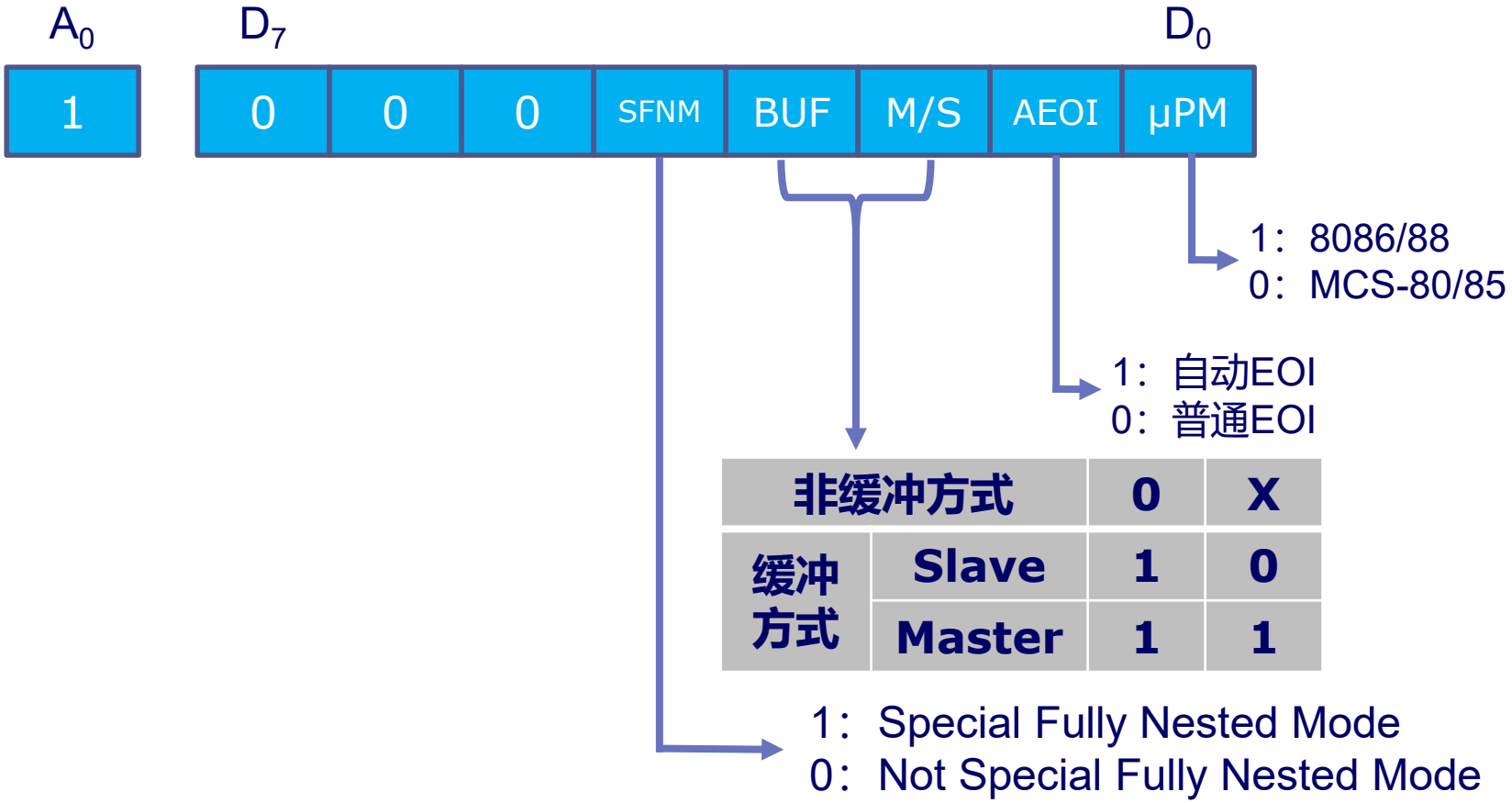
IBM PC/XT：

```
MOV AL, 8
OUT 21H, AL
```




中断向量
00001010
数据总线
缓冲器
中断
服务
寄存器
(ISR)
00000100

ICW_4



IBM PC/XT：

```
MOV AL, 9
OUT 21H, AL
```

IBM PC BIOS对8259的初始化

- MOV AL, 13H
- OUT 20H, AL
- MOV AL, 8
- OUT 21H, AL
- MOV AL, 8
- OUT 21H, AL

OCW_1

- 中断屏蔽操作命令字

M_X= { 1：屏蔽由IR_X引入的中断
0：允许由IR_X引入的中断

```
MOV AL, 0FFH
OUT 21H, AL
```

OCW_2

- 优先权循环方式和中断结束方式操作字

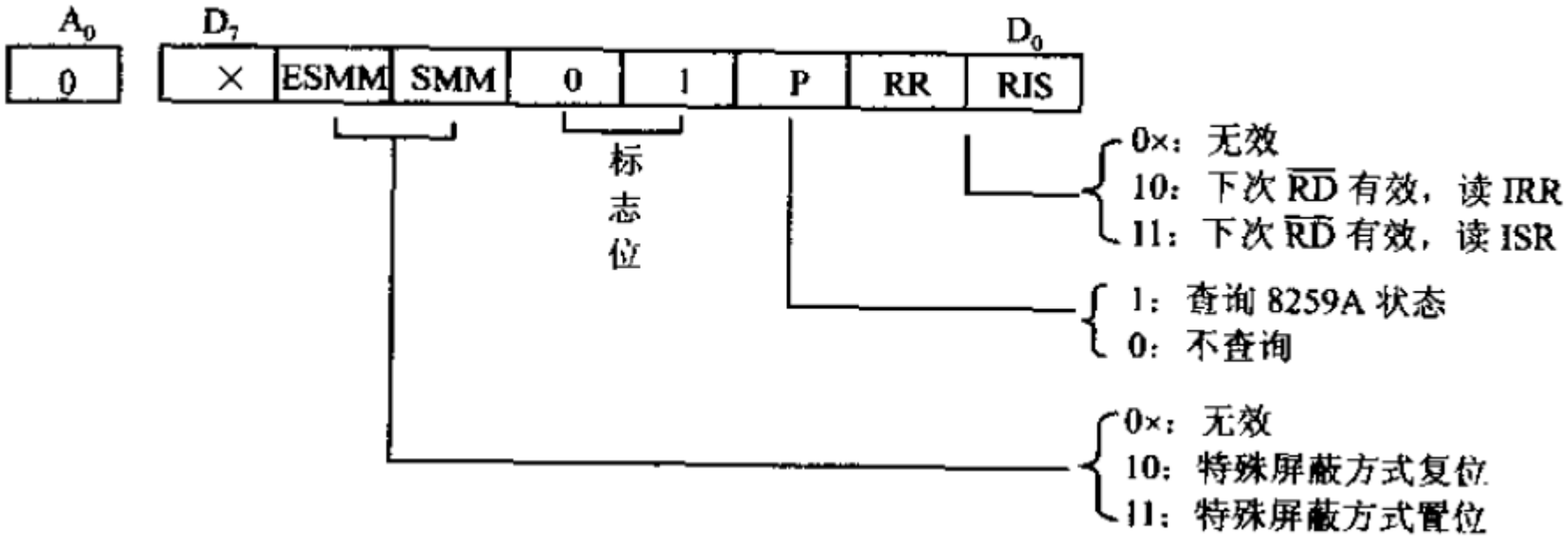
001: Non-Specific EOI Command
011: Specific EOI Command
101: Rotate on Non-Specific EOI Command
100: Rotate in Automatic EOI Mode (Set)
000: Rotate in Automatic EOI Mode (Clear)
111: Rotate on Specific EOI Command
110: Set Priority Command
010: No Operation

000: IR_0
001: IR_1
010: IR_2
011: IR_3
100: IR_4
101: IR_5
110: IR_6
111: IR_7

普通EOI命令

- **Non-Specific EOI Command**
  - MOV AL, 20H
  - OUT 20H, AL

OCW_3


A0
D7
D0
0
×
ESMM
SMM
0
1
P
RR
RIS
标志位
0×：无效
10：下次 RD 有效，读 IRR
11：下次 RD 有效，读 ISR
1：查询 8259A 状态
0：不查询
0×：无效
10：特殊屏蔽方式复位
11：特殊屏蔽方式置位

IBM PC/XT的键盘中断例程

```
STI
PUSH AX
PUSH BX
……
CLI
MOV AL, 20H
OUT  20H, AL              ; 普通EOI命令
….
POP BX
POP AX
IRET
```

IBM PC BIOS中断

- **从0FE000H开始的8K ROM存放BIOS(Basic Input/Output System)**
- **BIOS提供**
  - 系统加电自检
  - 引导装入
  - 主要I/O设备的处理程序以及接口控制等功能模块，以处理系统中断
- **程序员可以直接用指令设置参数，然后中断调用BIOS中的程序**

IBM PC BIOS中断

CPU中断			
0	除法错	4	溢出
1	单步	5	打印屏幕
2	非屏蔽中断	6	保留
3	断点	7	保留

8259中断			
8	8253系统定时器	C	保留(通讯)
9	键盘	D	Alternate Printer
A	保留	E	软盘
B	保留(通讯)	F	打印机

IBM PC BIOS中断

BIOS中断			
10	显示器	16	键盘
11	设备检验	17	打印机
12	内存大小	18	驻留BASIC
13	磁盘	19	引导
14	通讯	1A	时钟
15	I/O系统扩充	40	软盘

中断指令

- **INT**
  - INT imm8
  - 通过指令调用相应的中断，其操作数就是中断向量值
    - INT 9可以在8259没有报告中断的情况下调用键盘中断例程

范例：屏幕显示

- 使用BIOS显示中断10H

AH	功能	调用参数	返回参数
2	置光标位置	BH = 页号 DH = 行 DL = 列	
3	读光标位置	BH = 页号	CH = 光标开始行 CL = 光标结束行 DH = 行 DL = 列
A	在光标位置显示字符	BH = 显示页 AL = 字符 CX = 字符重复数	

范例：屏幕显示

- **置光标到第5行第6列**
  - MOV DH, 4　　;第5行
  - MOV DL, 5　　;第6列
  - MOV BH, 0　　;单色显示器总是0
  - MOV AH, 2　　;置光标
  - INT 10H
- **读光标位置**
  - MOV AH, 3
  - MOV BH, 0
  - INT 10H

DOS中断

- DOS(Disk Operating System)是IBM PC的磁盘操作系统
- 部分BIOS中断的封装，使用更简便，提供更多必要的测试

DOS中断类型

DOS中断类型			
20	程序结束	26	绝对盘写入
21	功能调用	27	结束并留在内存
22	结束地址	28–2E	保留给DOS
23	Ctrl-Break出口地址	2F	打印机
24	严重错处理	30-3F	保留给DOS
25	绝对盘读取		

范例：屏幕显示

- 使用DOS中断(21H)

AH	功能	调用参数	
2	显示一个字符 (检验Ctrl-Break)	DL = 字符	光标随字符移动
6	显示一个字符 (不检验Ctrl-Break)	DL = 字符	光标随字符移动
9	显示字符串	DS:DX = 串地址	串必须以$结束，光标随字符移动

范例：屏幕显示

- MESSAGE DB ‘Hello World.$’

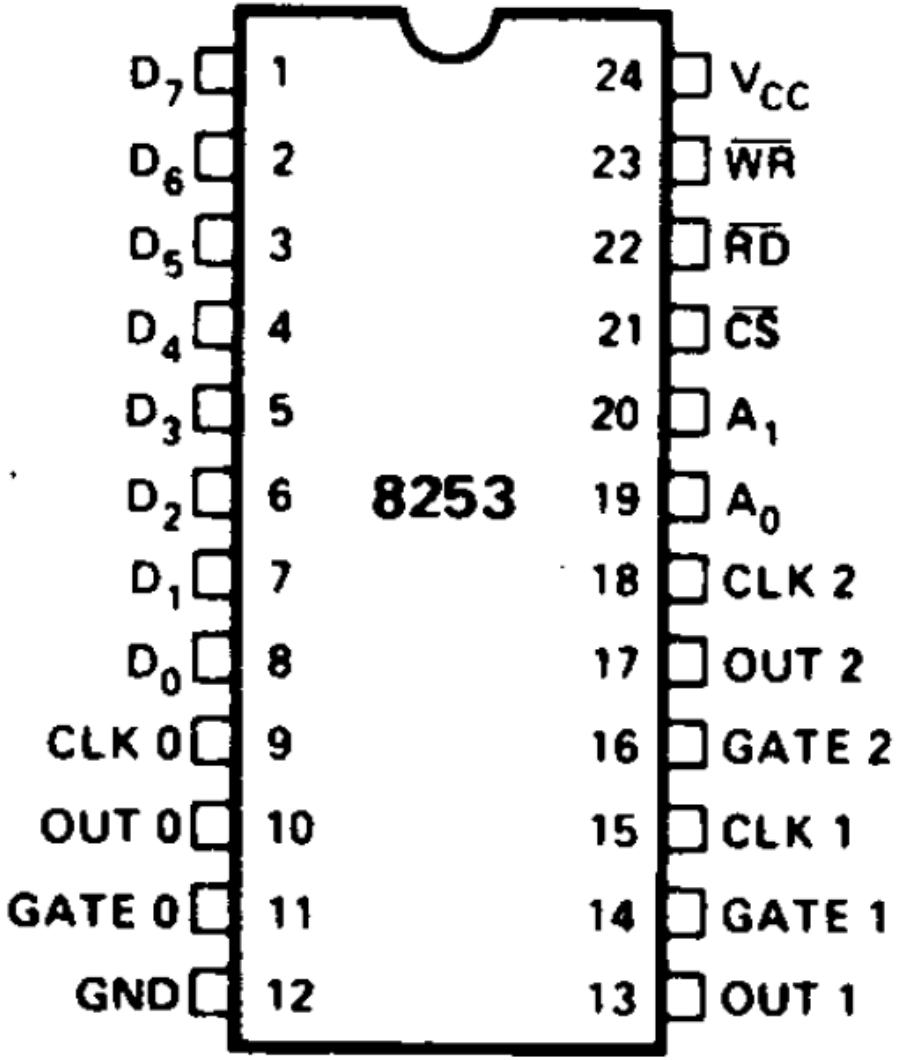
- MOV AH, 9
- MOV DX, SEG MESSAGE
- MOV DS, DX
- MOV DX, OFFSET MESSAGE
- INT 21H

内容

8253可编程计数器芯片

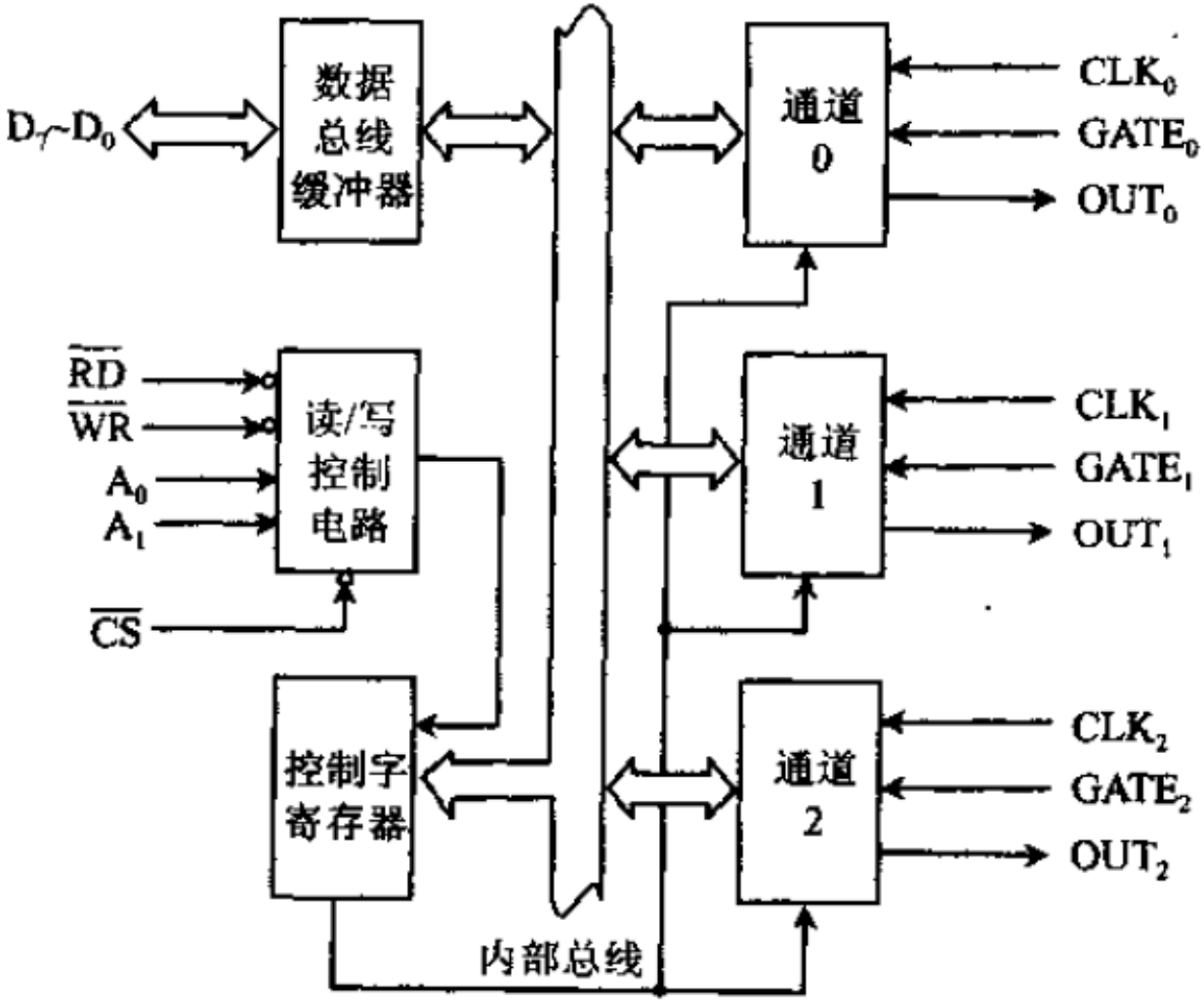
- 可编程硬件定时
- 用指令对其设定计数初值，然后启动芯片
- 预订时间到来后，形成输出信号，可用来向CPU提起中断请求
- Intel 8253
  - 3个独立的16位计数器通道
  - 6种不同的工作方式
  - 二进制/十进制计数
  - 最高计数频率2MHz

8253的引脚


D7 1
D6 2
D5 3
D4 4
D3 5
D2 6
D1 7
D0 8
CLK 0 9
OUT 0 10
GATE 0 11
GND 12
8253
24 VCC
23 WR
22 RD
21 CS
20 A1
19 A0
18 CLK 2
17 OUT 2
16 GATE 2
15 CLK 1
14 GATE 1
13 OUT 1

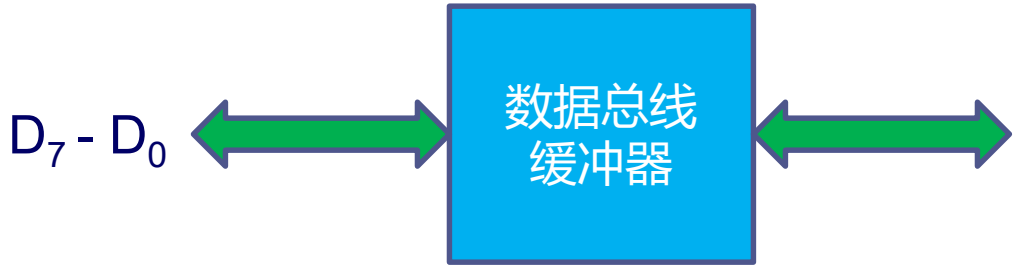

D8253
L7210435
© INTEL '80

8253内部结构


数据
总线
缓冲器
D_7~D_0
通道
0
CLK_0
$GATE_0$
OUT_0
$\overline{RD}$
$\overline{WR}$
A_0
A_1
读/写
控制
电路
$\overline{CS}$
通道
1
CLK_1
$GATE_1$
OUT_1
控制字
寄存器
通道
2
CLK_2
$GATE_2$
OUT_2
内部总线

8253内部结构

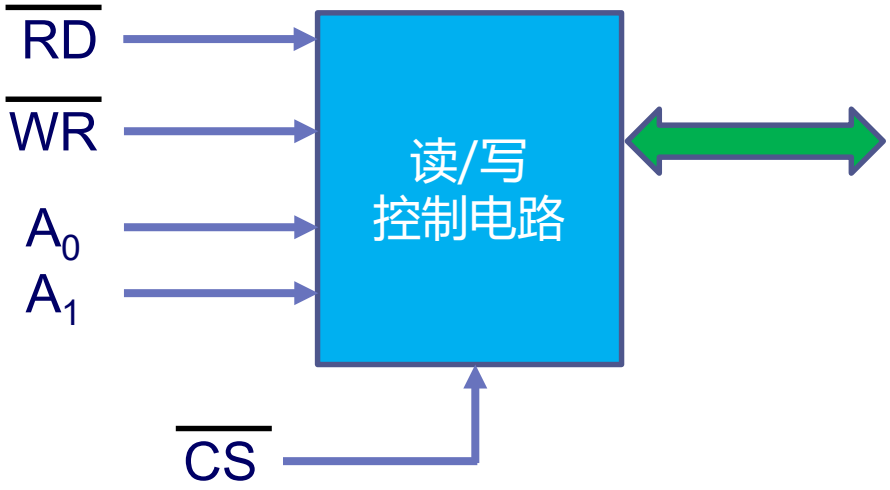
- **数据总线缓冲器**
  - $D_7 \sim D_0$：与系统总线低8位相连



8253内部结构

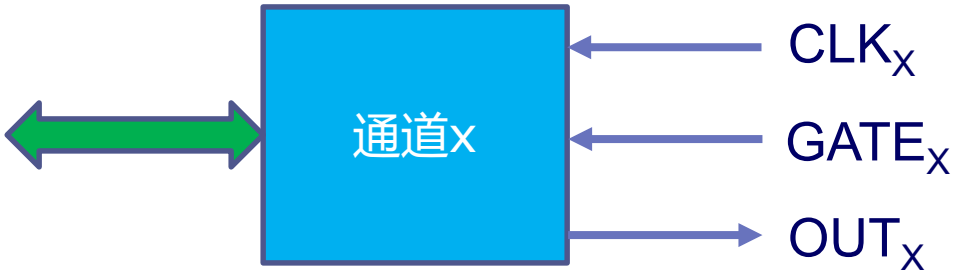
读/写控制逻辑

- $\overline{CS}$：片选信号，低电平有效
- $\overline{RD}$：读信号，低电平有效
- $\overline{WR}$：写信号，低电平有效
- A_1A_0：端口选择信号



8253内部结构

- **计数器0 ~ 2**
  - 3个独立通道
  - 每个通道包括
    - 一个8位控制字寄存器
    - 一个16位计数器初值寄存器
    - 一个计数器执行部件
    - 一个输出锁存器
  - 对输入到CLK引脚的脉冲进行计数
    - 每输入一个脉冲，计数器减一
    - 计数器归零时，从OUT引脚输出一个脉冲信号
    - GATE引脚控制是否允许计数



只有8位数据线，
如何写入16位计数器初值

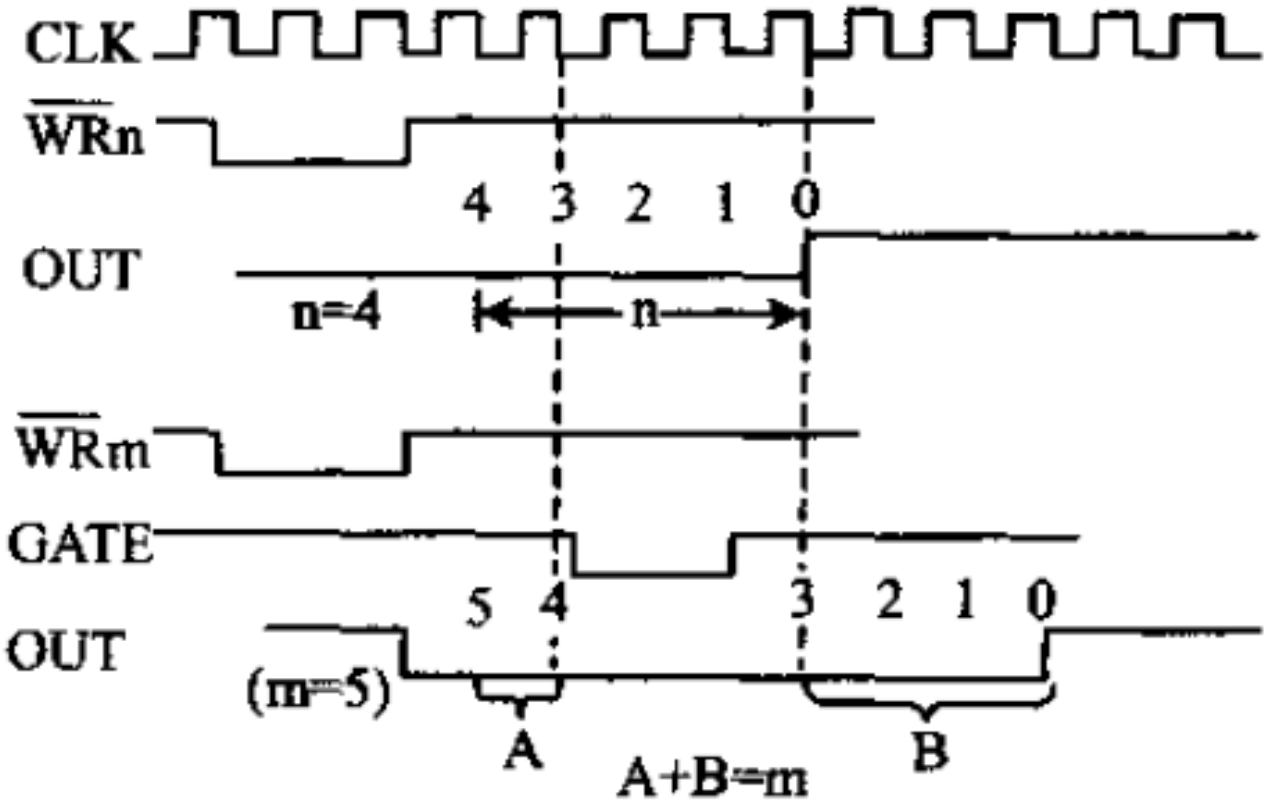
8253工作方式

- 方式0：计数结束中断方式
- 方式1：可编程单稳态输出方式
- 方式2：比率发生器
- 方式3：方波发生器
- 方式4：软件触发选通
- 方式5：硬件触发选通

计数结束中断方式

- 选定方式0时，OUT立即变为低电平
- GATE为高时允许计数，否则计数暂停
- 向计数通道写入初值n时，$\overline{WR}$变为低电平
- 在$\overline{WR}$上升沿后的下一个时钟下降沿，才进行计数
- 计数计到终点时，OUT从低电平变为高电平

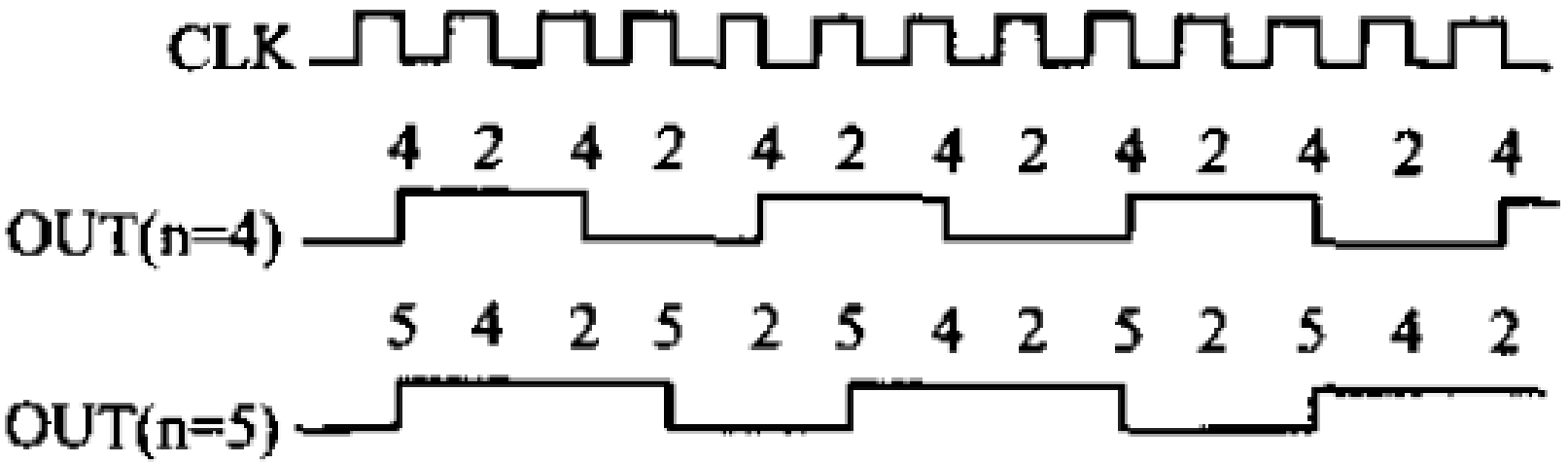
计数结束中断方式



方波生成器

- 输入控制字后，OUT端变为高电平
- 写入计数值后下一个时钟脉冲开始计数
- GATE为高电平时允许计数
- 如果计数器初值为偶数，则每个时钟脉冲，计数器值减2，计数值归零时，OUT引脚电平翻转
- 如果计数器初值为计数，则对于奇数轮计数，第一个脉冲计数器值减1；对于偶数轮计数，第一个脉冲计数器值减3

方波生成器



8253编程方式

- 通过写入控制字设置通道模式及其他设置
- 向所需通道写入计数器初始值

8253输入信号功能表

$\overline{CS}$	$\overline{RD}$	$\overline{WR}$	A_1 A_0	功能
0	1	0	0 0	写入计数器0
0	1	0	0 1	写入计数器1
0	1	0	1 0	写入计数器2
0	1	0	1 1	写入控制字寄存器
0	0	1	0 0	读计数器0
0	0	1	0 1	读计数器1
0	0	1	1 0	读计数器2
0	0	1	1 1	无操作
1	X	X	X X	禁止使用
0	1	1	X X	无操作

控制字寄存器

1: BCD码计数
0: 2进制计数

000: 方式0
001: 方式1
X10: 方式2
X11: 方式3
100: 方式4
101: 方式5

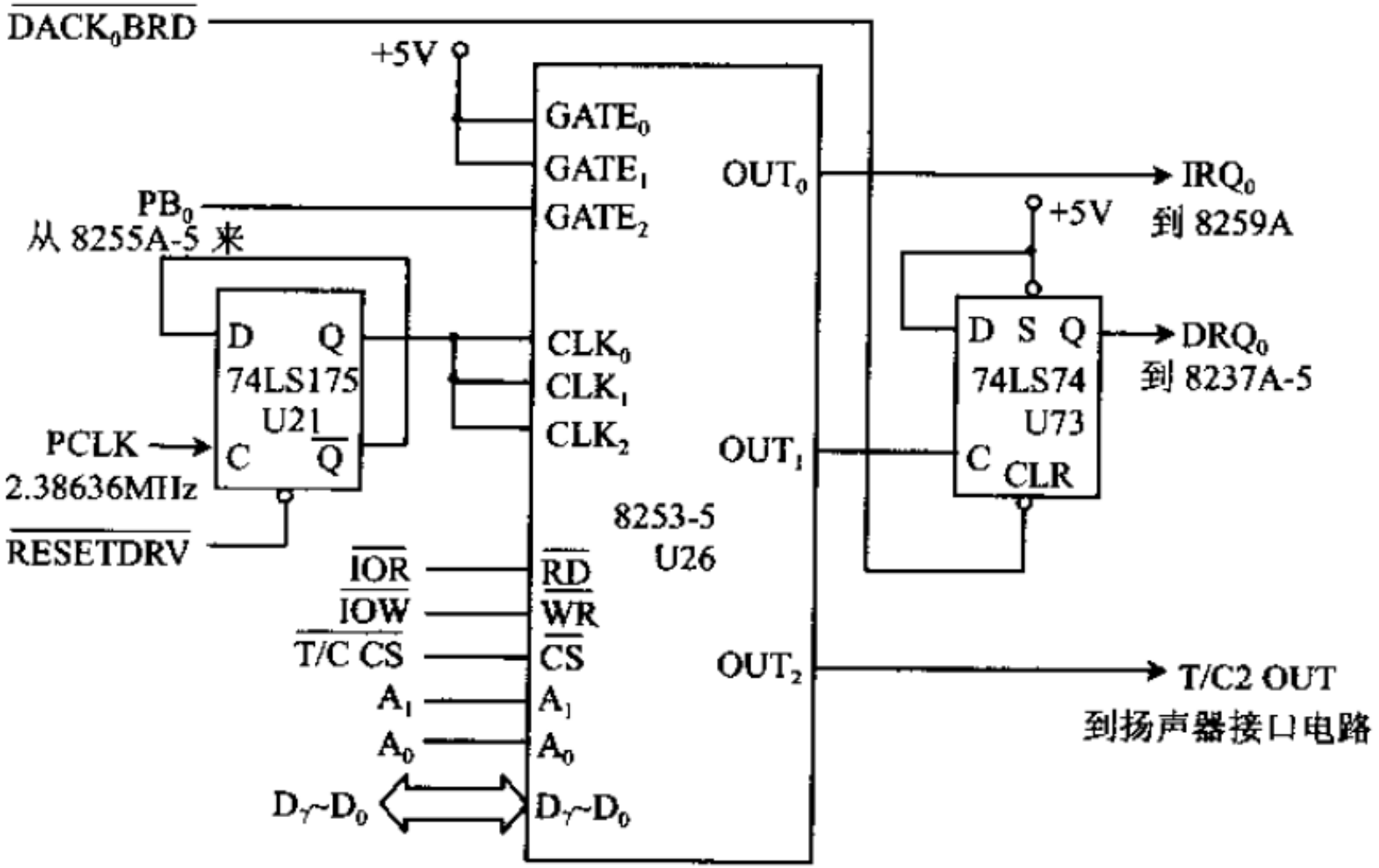
00: 选择通道0
01: 只读/写计数器低字节
10: 只读/写计数器高字节
11: 先读/写计数器低字节，
后读/写计数器高字节

00: 选择通道0
01: 选择通道1
10: 选择通道2
11: 无效

IBM PC/XT：

```
MOV AL, 36H
OUT 43H, AL
```

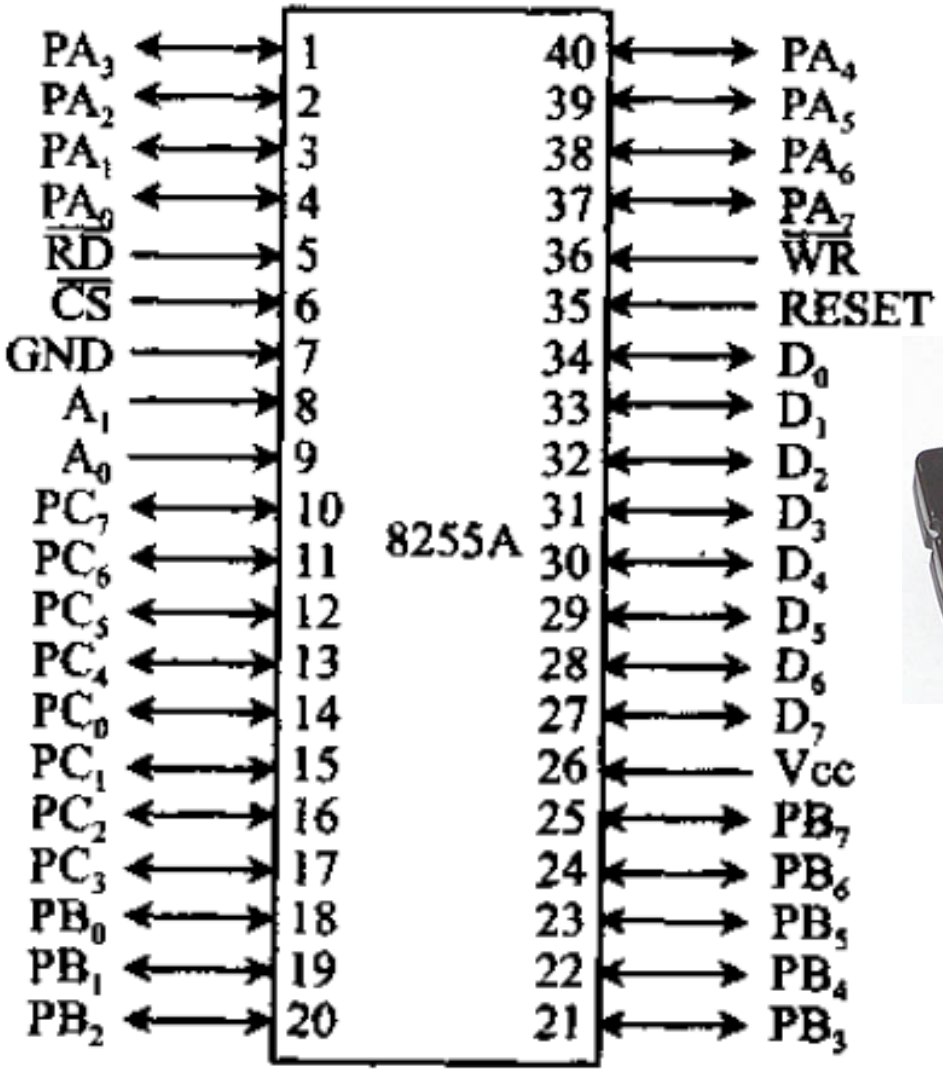
8253在PC/XT机中的连线

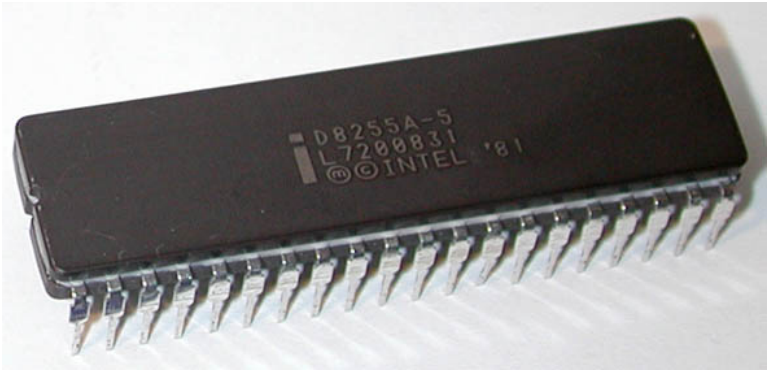


8255A可编程外围接口芯片

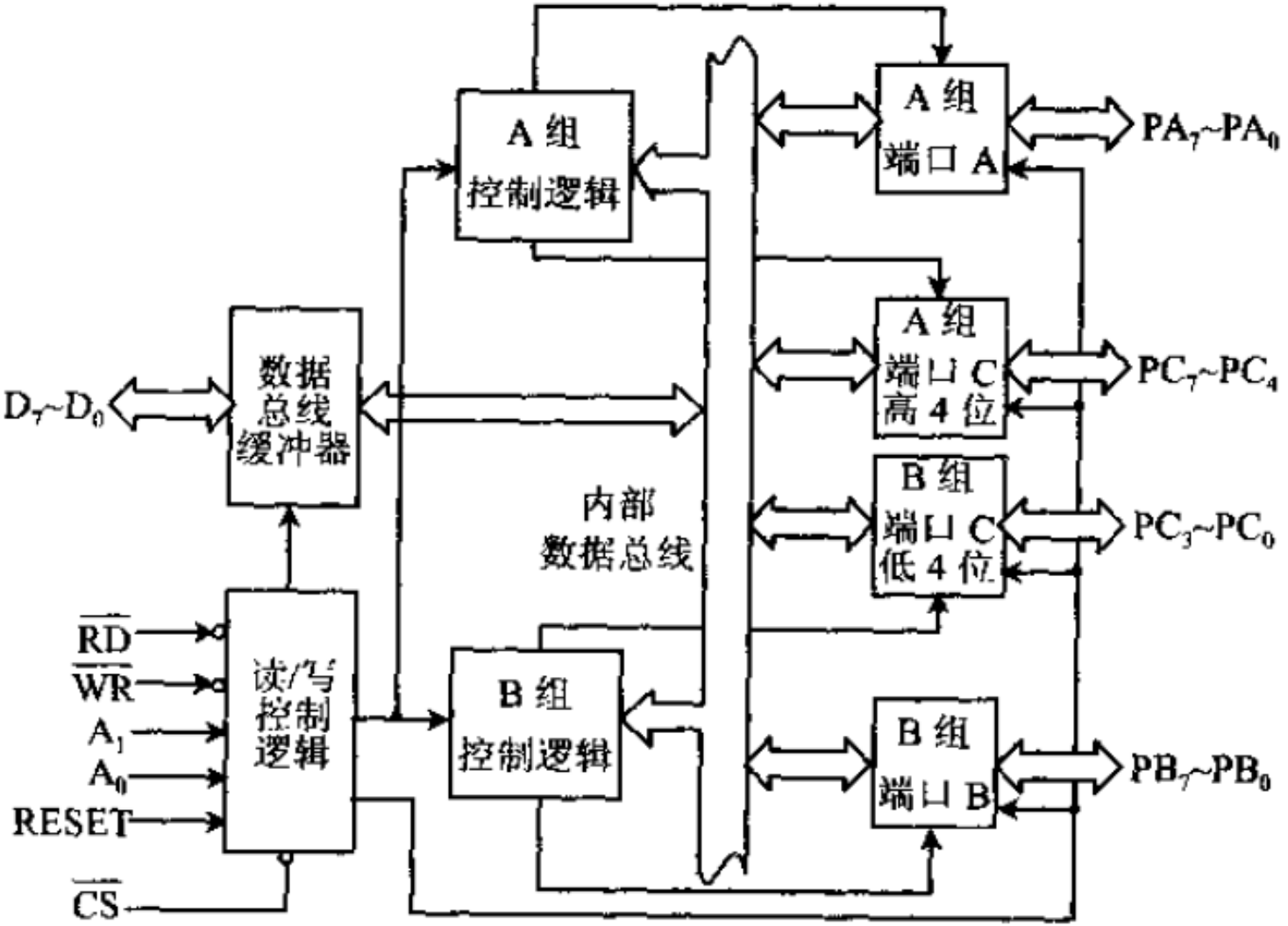
- **为CPU和外设之间提供数据通道**
  - 无需附加外部逻辑电路
  - 键盘
  - 扬声器
  - 打印机
  - ……

8255A的引脚


PA3 1
PA2 2
PA1 3
PA0 4
RD 5
CS 6
GND 7
A1 8
A0 9
PC7 10
PC6 11
PC5 12
PC4 13
PC0 14
PC1 15
PC2 16
PC3 17
PB0 18
PB1 19
PB2 20
8255A
40 PA4
39 PA5
38 PA6
37 PA7
36 WR
35 RESET
34 D0
33 D1
32 D2
31 D3
30 D4
29 D5
28 D6
27 D7
26 Vcc
25 PB7
24 PB6
23 PB5
22 PB4
21 PB3

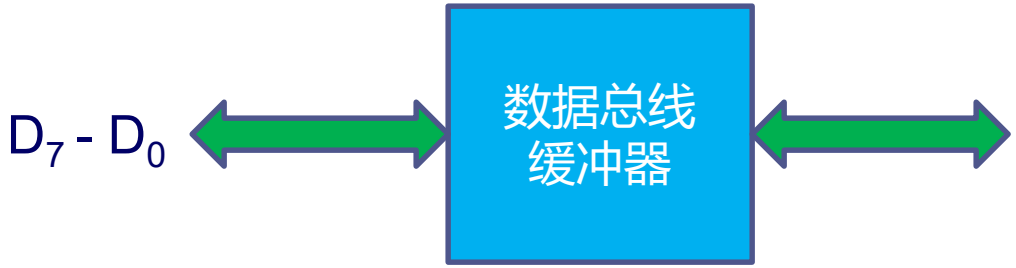

D8255A-5
INTEL '81

8255A的内部结构


A 组
控制逻辑
A 组
端口 A
PA_7~PA_0
数据
总线
缓冲器
D_7~D_0
A 组
端口 C
高 4 位
PC_7~PC_4
B 组
端口 C
低 4 位
PC_3~PC_0
内部
数据总线
$\overline{RD}$
$\overline{WR}$
A_1
A_0
RESET
$\overline{CS}$
读/写
控制
逻辑
B 组
控制逻辑
B 组
端口 B
PB_7~PB_0

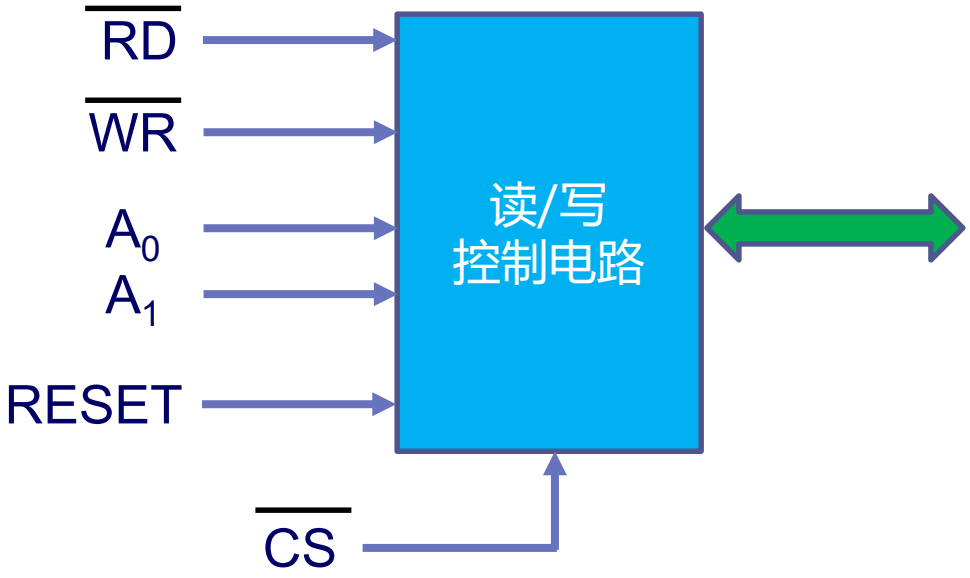
8255A的内部结构

- **数据总线缓冲器**
  - 双向三态8位缓冲器
  - 与系统数据总线D_7 ~ D_0相连



8255A的内部结构

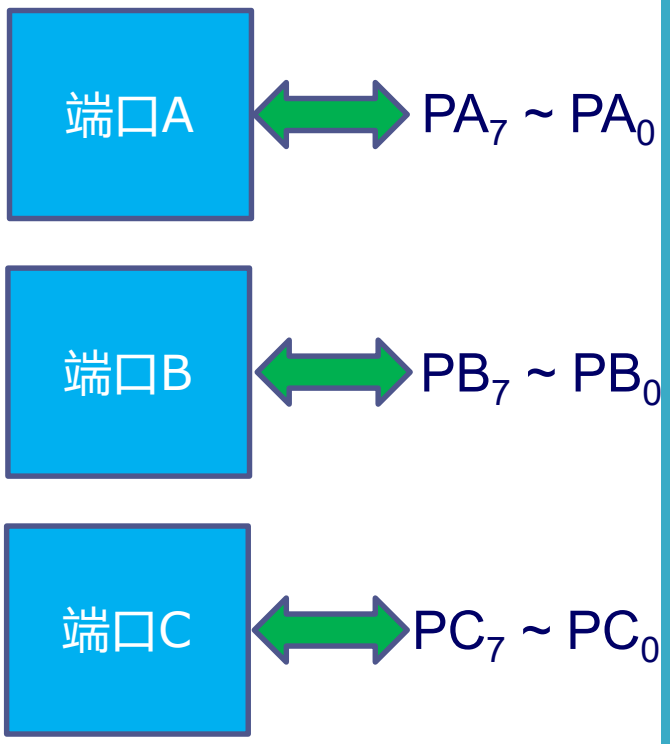
- **读/写控制逻辑**
  - RESET：复位，高电平有效。控制寄存器清0，所有端口均置为输入方式。
  - CS：片选信号，低电平有效
  - RD：读信号，低电平有效
  - WR：写信号，低电平有效
  - A_1A_0：端口选择信号



8255A的内部结构

数据端口A，B和C

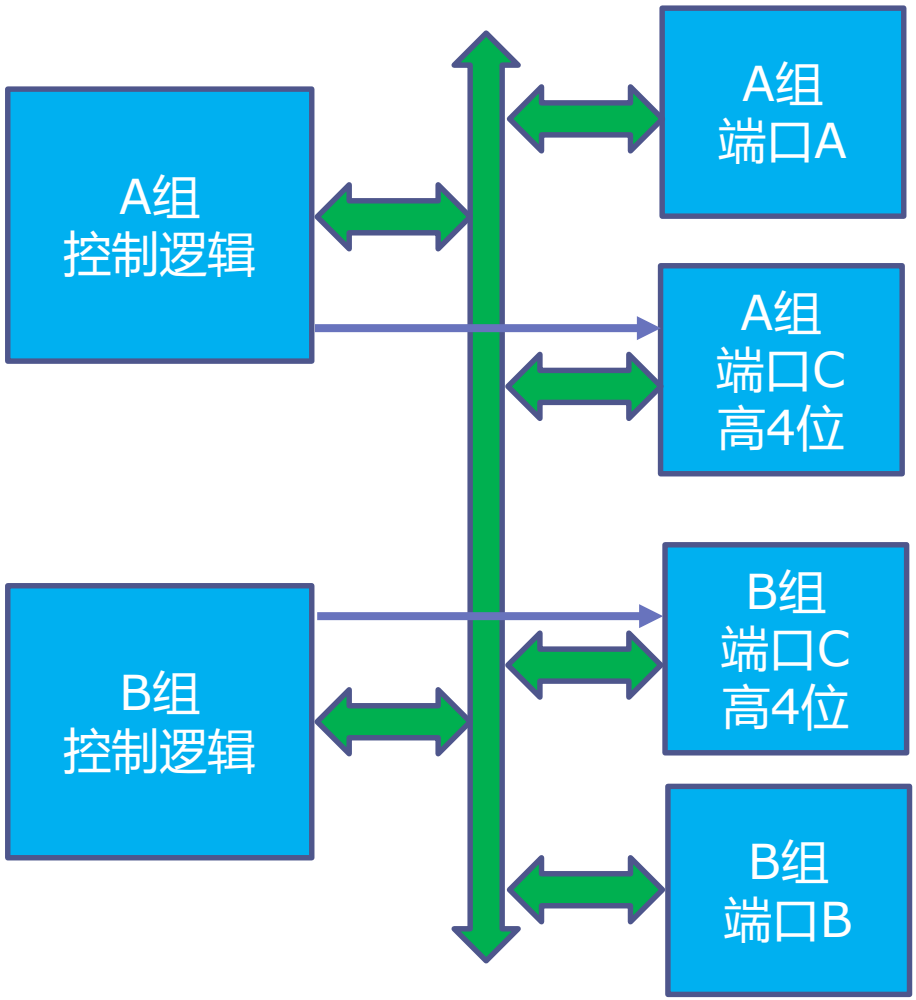
- 3个8位输入输出端口与外设交换数据
  - $PA_7 \sim PA_0$, $PB_7 \sim PB_0$, $PC_7 \sim PC_0$
- 端口A
  - 1个8位数据输出锁存器/缓冲器
  - 1个8位数据输入锁存器
- 端口B
  - 1个8位数据输入/输出锁存器/缓冲器
  - 1个8位数据输入缓冲器
- 端口C
  - 1个8位数据输出锁存器/缓冲器
  - 1个8位数据输入缓冲器
  - 分成两个4位端口时，每个端口有1个4位输出锁存器



8255A的内部结构

A组和B组控制逻辑

- 控制寄存器，决定A组和B组的工作方式
- 端口A和端口C的上半部分($PC_7 \sim PC_4$)由A组控制逻辑管理
- 端口B和端口C的下半部分($PC_3 \sim PC_0$)由B组控制逻辑管理



工作方式

- **8255A的端口工作方式**
  - 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：双向总线I/O方式
  - 端口A可工作于3种方式的任一种
  - 端口B只能工作于方式0和方式1
  - 端口C常被分成两个4位端口，用以配合A口和B口工作

工作方式0

- 基本输入输出方式
- 每个端口都可以配置成输入或者输出
- 输出为锁存，输入不锁存
- CPU可以直接用输入指令读取数据，或用输出指令写入数据
- 不需要其他用于应答的联络信号

8255A控制字

- **方式选择控制字**
  - 定义各端口的工作方式
- **置位复位控制字**
  - 对C端口的任一位进行置位或者复位操作

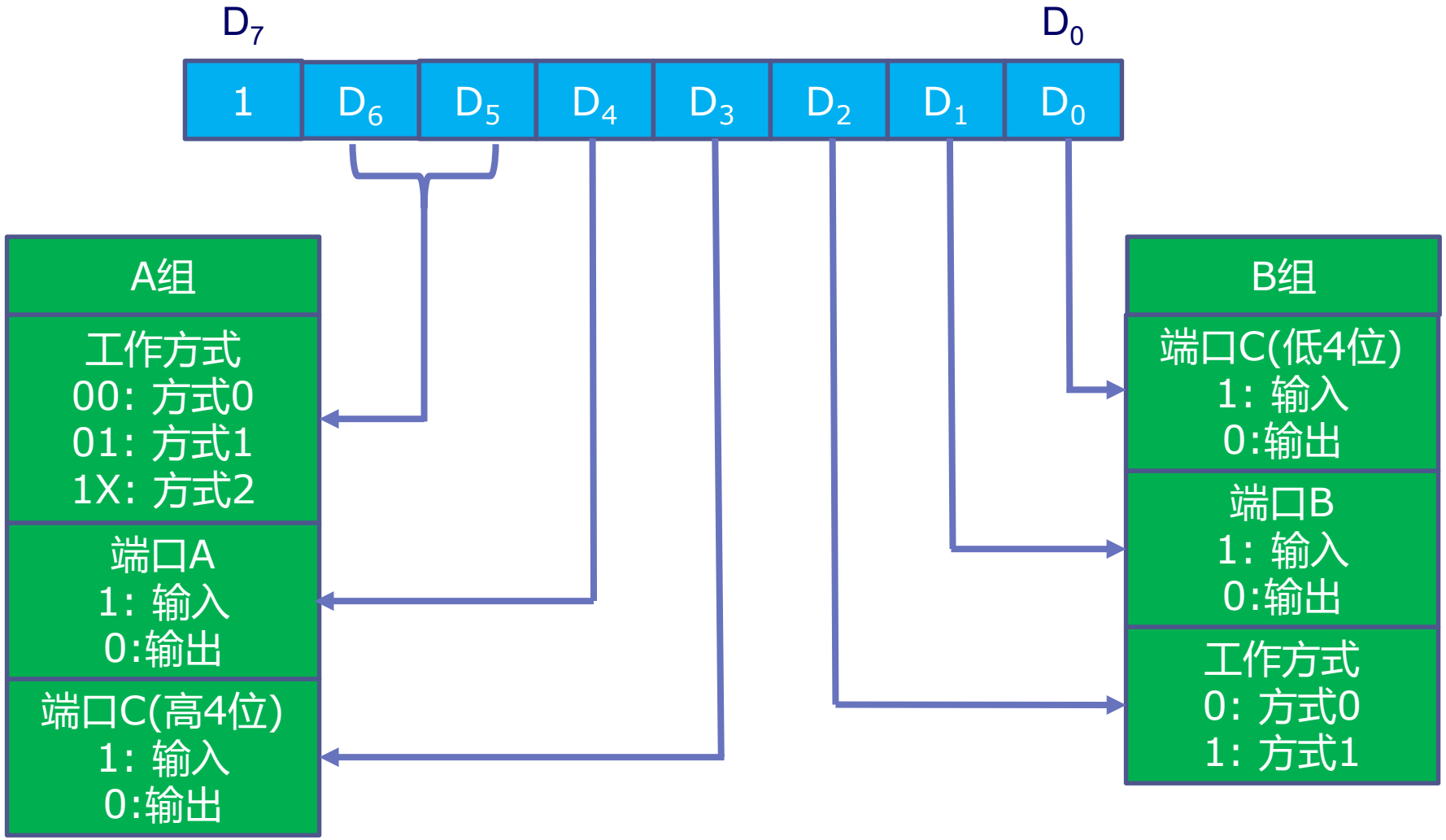
8255编程方式

- 通过写入控制字设置各端口的工作方式和输入输出状态
- 从所需端口读取/写入数据

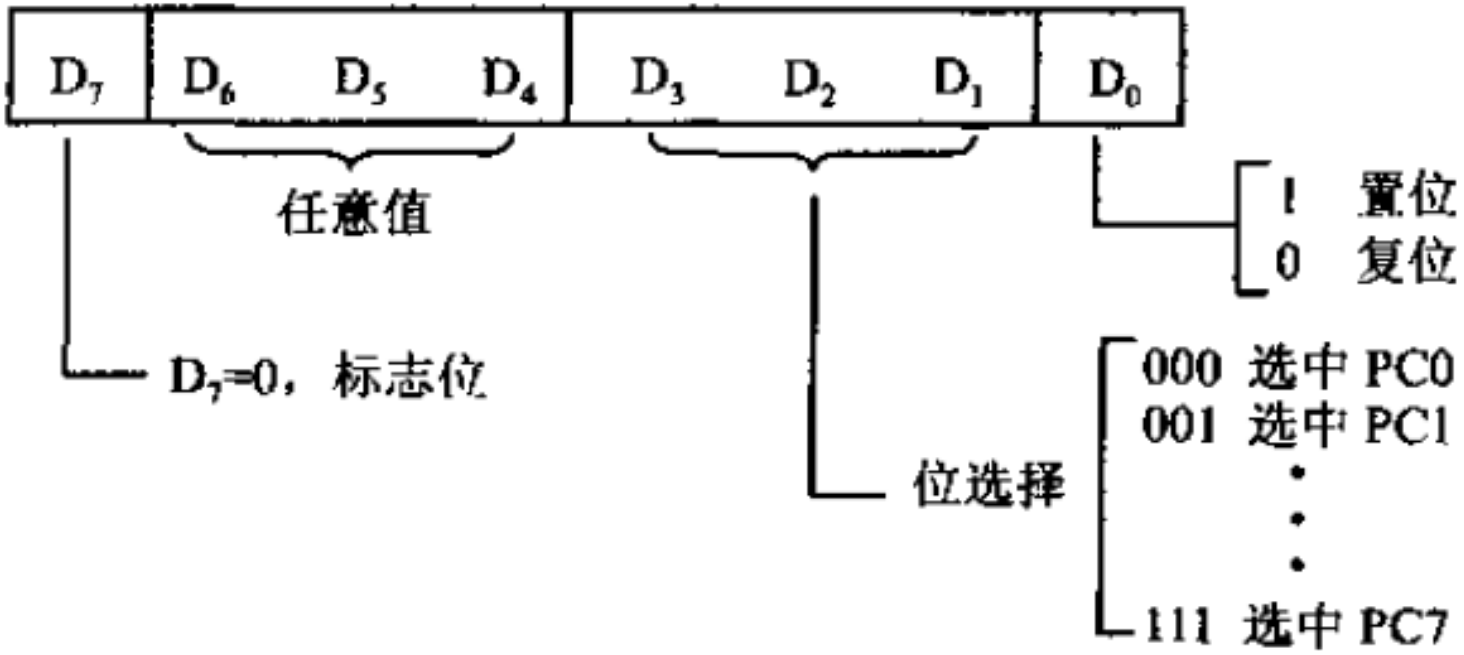
8255A的基本操作

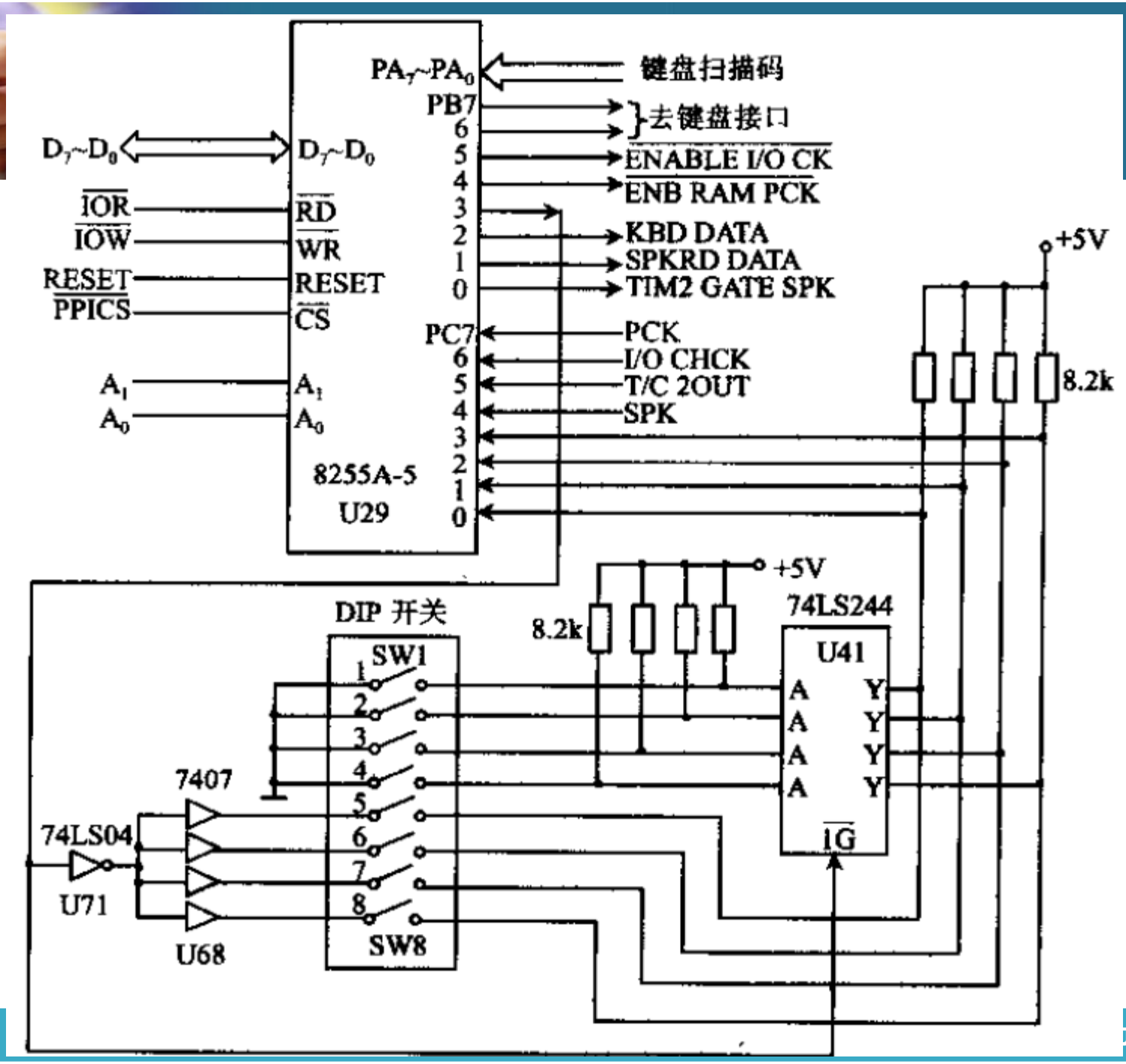
A1	A0	$\overline{RD}$	$\overline{WR}$	$\overline{CS}$	操作
0	0	0	1	0	端口A → 数据总线
0	1	0	1	0	端口B → 数据总线
1	0	0	1	0	端口C → 数据总线
0	0	1	0	0	数据总线 → 端口A
0	1	1	0	0	数据总线 → 端口B
1	0	1	0	0	数据总线 → 端口C
1	1	1	0	0	数据总线 → 控制字寄存器
X	X	X	X	1	数据总线三态
1	1	0	1	0	非法状态
X	X	1	1	0	数据总线三态

方式选择控制字


D7
D0
1
D6
D5
D4
D3
D2
D1
D0
A组
工作方式
00: 方式0
01: 方式1
1X: 方式2
端口A
1: 输入
0:输出
端口C(高4位)
1: 输入
0:输出
B组
端口C(低4位)
1: 输入
0:输出
端口B
1: 输入
0:输出
工作方式
0: 方式0
1: 方式1

置位复位控制字


D7 D6 D5 D4 D3 D2 D1 D0
任意值
D7=0，标志位
位选择
1 置位
0 复位
000 选中 PC0
001 选中 PC1
111 选中 PC7


PA7~PA0
键盘扫描码
PB7
6
5
4
3
2
1
0
去键盘接口
ENABLE I/O CK
ENB RAM PCK
KBD DATA
SPKRD DATA
TIM2 GATE SPK
D7~D0
IOR
IOW
RESET
PPICS
RD
WR
RESET
CS
A1
A0
8255A-5
U29
PC7
6
5
4
3
2
1
0
PCK
I/O CHCK
T/C 2OUT
SPK
+5V
8.2k
DIP 开关
SW1
SW8
1
2
3
4
5
6
7
8
7407
74LS04
U71
U68
74LS244
U41
A
Y
1G

8255A在PC/XT中的应用

- 8255A的D_7 ~ D_0与数据总线D_7 ~ D_0相连
- 8255A的$\overline{RD}$，$\overline{WR}$与$\overline{IOR}$和$\overline{IOW}$相连，接受系统I/O总线的读写控制命令
- $\overline{CS}$与I/O译码电路的$\overline{PPICS}$相连
- 端口范围是60H ~ 63H
- A, B和C口均处于工作方式0

各端口功能

- **端口A**
  - 上电自检时处于输出方式，输出当前检测部件的标志
  - 自检完成后，工作于输入方式，读取键盘的8位扫描码
- **端口B**
  - 工作于输出方式，用以输出控制信号
  - PB_0：输出TIM2 GATE SPK信号，送到8253芯片的$GATE_2$端。
  - PB_1：输出SPK DATA信号，控制扬声器发声。
  - PB_2：保留的输出信号，控制键盘接口。

各端口功能

端口B

- PB_3：控制系统配置DIP开关的状态的读入。
- PB_4：输出$\overline{\text{ENB RAM PCK}}$信号，是否允许RAM奇偶校验电路工作
- PB_5：输出$\overline{\text{ENABLE I/O CK}}$信号，用于控制I/O通道上扩展RAM奇偶校验电路的工作
- PB_6：Hold Keyboard Clock Low
- PB_7：0：允许键盘读；1：清除键盘信息

各端口功能

端口C

- C口工作于输入方式，用来读取系统内部的状态
- $PC_3 \sim PC_0$：系统配置开关DIP的设置状态
- PC_4：扬声器电路的发声数据SPK
- PC_5：扬声器的音调信号状态，8253的OUT_2信号
- PC_6：I/O通道奇偶校验结果I/O CHECK
- PC_7：系统板上奇偶检验电路的结果PCK

键盘中断处理

```
pushall
in al,060h      ;get key code
push ax         ;save it
in al,061h      ;get current control
mov ah,al       ;save PB control
or al,80h       ;set keyboard bit
out 061h,al     ;keyboard acknowledge
xchg ah,al      ;get back PB
out 061h,al     ;reset PB control
pop ax          ;get back code
```

- ... ;save code in buffer
- cli
- mov al,20h
- out 20h,al ;send eoi to interrupt controller
- popall
- iret

作业2

- **写一个能实现以下功能的程序**
  - 运行之后进入以下循环
    - 10秒禁止键盘响应(即按任何键都没效果)
    - 10秒密码键盘：简单古典密码，对于按键$x \in a-z$，显示$(x+1)\ mod\ 26$ (如键入a，显示b……)

作业2相关提示

- 程序应以驻留内存形式退出，故应使用INT 27H(驻留方式)代替原来的INT 20H/21H，请自行查阅INT 27H的使用方法
- 可以通过设置8259芯片IMR寄存器实现屏蔽键盘输入
- 修改键盘中断例程中得到的key code，可实现密码键盘——不过并不容易，这是本题的最难点，需要有较强的文献查阅能力

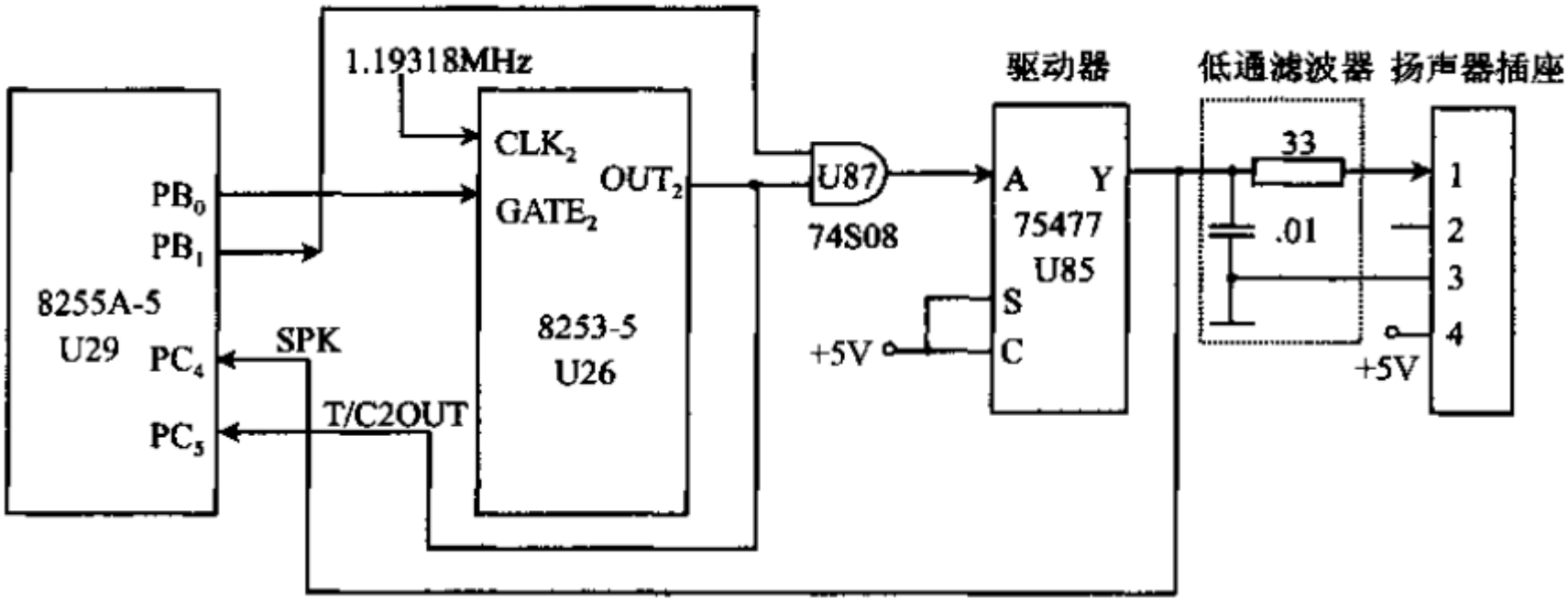
DIP开关

- **8位双列直插式(DIP)开关**
- **8个小开关，开关接通，电位为0；开关断开，电位为1**
- **读取开关状态可以了解系统配置**
  - 显示器类型
  - 存储器容量

DIP开关

- **$PB_3 = 0$**
  - U41选通
  - DIP开关低4位送到PC_0 ~ PC_3
  - U68输入高电平
- **$PB_3 = 1$**
  - U41不选通，高阻状态
  - U68输出低电平
  - DIP开关高4位送到PC_4 ~ PC_7

PC/XT中的扬声器接口电路



PC/XT中的扬声器接口电路

- 8253的CLK_2接1.19318MHz时钟脉冲
- 8255A的PB_0接8253的$GATE_2$
- 8255A的PB1和8253的OUT2经与门电路U87输出到驱动器U85
- U85通过低通滤波，送到扬声器插座
- 8253的计数器2工作于方式3
- U85的输出为一定频率的方波
- 通过编程控制8253的定时时间，可使扬声器发出不同频率的声音，控制其发声时间，可形成乐曲

DMA控制器

- 向CPU提出DMA请求，请求信号加到CPU的HOLD引脚上
- CPU相应DMA请求后，DMA控制器获得总线控制权
- 提供读/写存储器或I/O设备的各种控制命令
- 确定数据传输的起始地址和数据长度，每传送一个数据，能自动修改地址：地址增1，长度减1
- 数据传送完毕，能发出结束DMA传送的信号

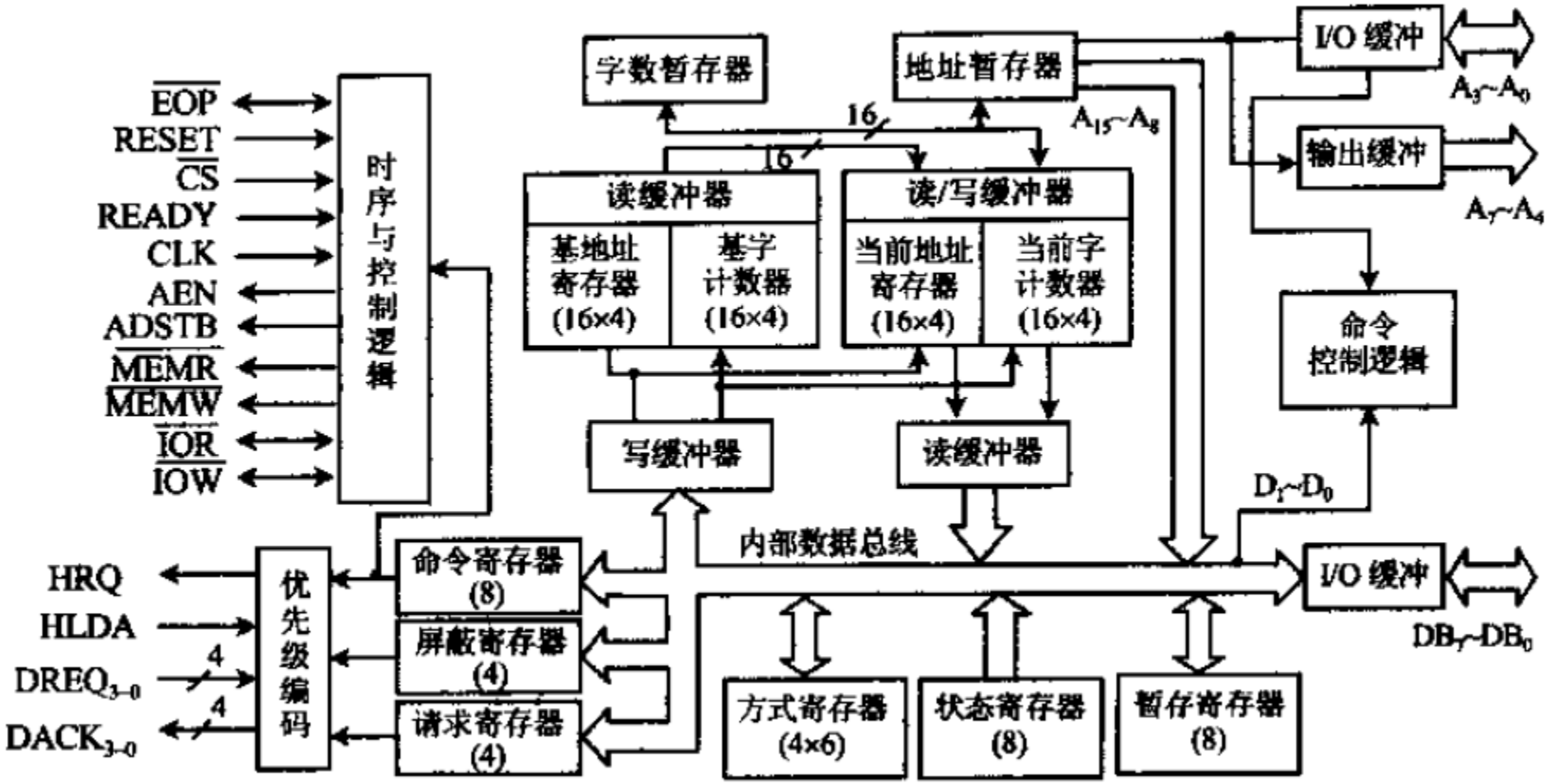
8237A DMA控制器

- **可编程DMA控制器**
- **4个独立通道**
  - 64K地址和字节的计数能力
  - 4种不同的传送方式
  - 可以允许或者禁止
  - 具有不同的优先级

8237A的工作状态

- **从态方式**
  - 未取得总线控制权时
  - CPU对8237A进行编程
  - CPU可读取8237A的状态
- **主态方式**
  - 取得总线控制权时
  - 进行DMA数据传送

8237A的内部结构


EOP
RESET
CS
READY
CLK
AEN
ADSTB
MEMR
MEMW
IOR
IOW
时序与控制逻辑
字数暂存器
地址暂存器
16
16
A15~A8
I/O 缓冲
A3~A0
输出缓冲
A7~A4
读缓冲器
基地址寄存器 (16×4)
基字计数器 (16×4)
读/写缓冲器
当前地址寄存器 (16×4)
当前字计数器 (16×4)
命令控制逻辑
写缓冲器
读缓冲器
D7~D0
内部数据总线
HRQ
HLDA
DREQ3-0
DACK3-0
4
4
优先级编码
命令寄存器 (8)
屏蔽寄存器 (4)
请求寄存器 (4)
I/O 缓冲
DB7~DB0
方式寄存器 (4×6)
状态寄存器 (8)
暂存寄存器 (8)

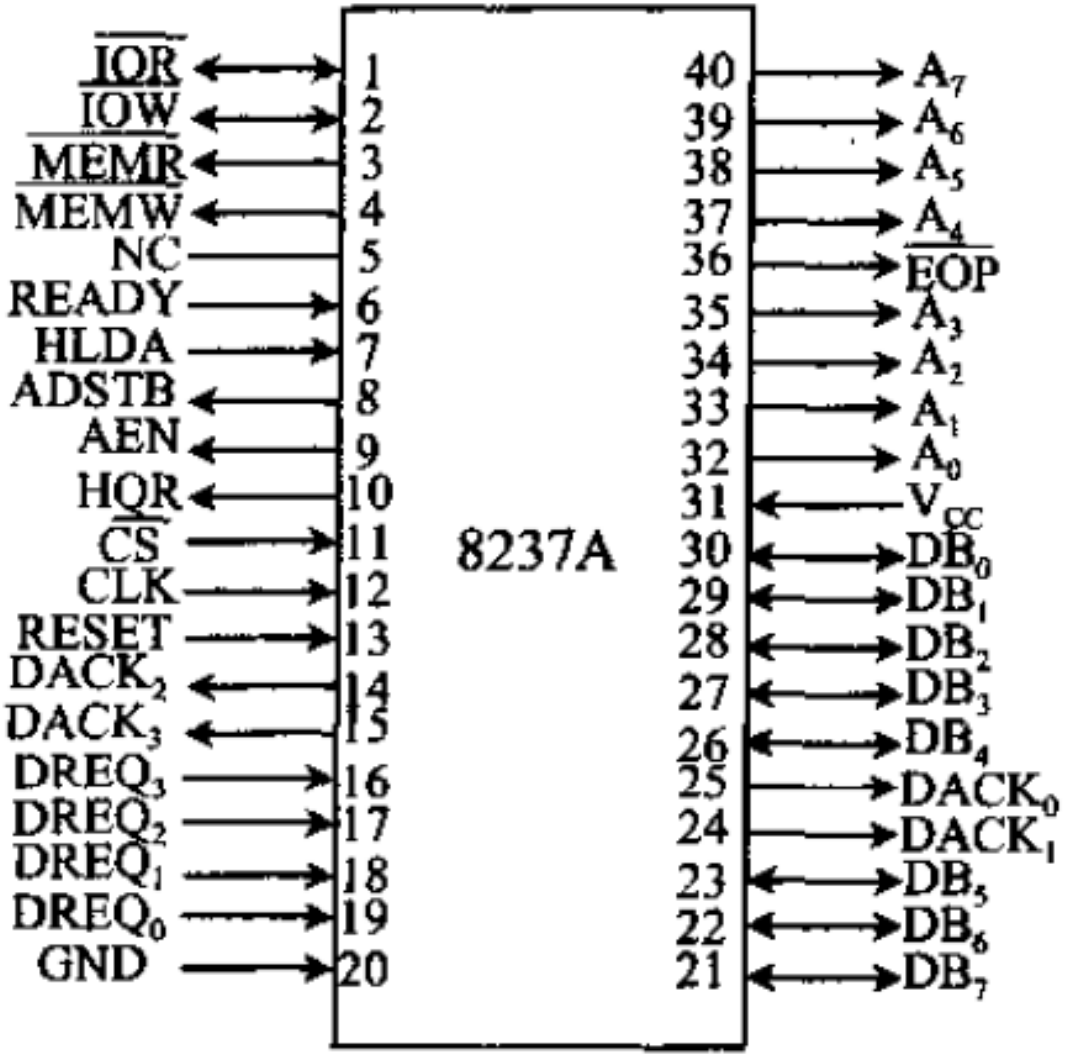
8237A的内部结构

- **时序与控制逻辑**
  - 在从态时接收从系统送来的时钟、复位、片选和读/写控制等信号
  - 在主态时向系统发出相应的控制信号
- **优先级编码电路**
  - 确定哪个通道优先级最高
  - 高优先级设备服务时，其他通道请求被禁止
- **数据和地址缓冲器组**

8237A的内部结构

- **命令控制逻辑**
  - 从态时接收CPU的寄存器选择信号
  - 主态时对方式字译码，确定DMA的操作类型
  - 组成各种操作命令
- **内部寄存器组**
  - 每个通道：
    - 16位基地址寄存器、基字计数器、当前地址寄存器和当前字计数器
    - 6位工作方式寄存器
  - 通道共享：
    - 命令寄存器、屏蔽寄存器、请求寄存器、状态寄存器和暂存寄存器
    - ……

8237A的引脚


IOR 1
IOW 2
MEMR 3
MEMW 4
NC 5
READY 6
HLDA 7
ADSTB 8
AEN 9
HQR 10
CS 11
CLK 12
RESET 13
DACK2 14
DACK3 15
DREQ3 16
DREQ2 17
DREQ1 18
DREQ0 19
GND 20
8237A
40 A7
39 A6
38 A5
37 A4
36 EOP
35 A3
34 A2
33 A1
32 A0
31 VCC
30 DB0
29 DB1
28 DB2
27 DB3
26 DB4
25 DACK0
24 DACK1
23 DB5
22 DB6
21 DB7

8237A的引脚

- **CLK：时钟信号，输入**
- **$\overline{CS}$：片选信号，输入，低电平有效**
- **READY：准备好，输入，高电平有效**
- **$A_3 \sim A_0$：最低4位地址线，三态，双向**
  - 从态时，输入信号，寻址内部寄存器
  - 主态时，输出所访问内存的最低4位
- **$A_7 \sim A_4$：4位地址线，三态，输出**
  - 主态时输出4位地址$A_7 \sim A_4$

8237A的引脚

- **DB_7 ~ DB_0：8位数据线，三态，双向**
  - 从态时用于读写8237A的内部寄存器
  - 主态时输出地址信号A_{15} ~ A_8，输出到外部锁存器
- **AEN：地址允许信号，输出，高电平有效**
  - 使A_{15} ~ A_8地址锁存器输出地址到地址总线
  - 使与CPU相连的锁存器无效
- **ADSTB：地址选通信号，输出，高电平有效**
  - 选通A_{15} ~ A_8地址锁存器

8237A的引脚

- **$\overline{\text{IOR}}$：I/O读信号，双向，三态，低电平有效**
  - 从态时，输入信号，有效时CPU读取内部寄存器
  - 主态时，输出信号，与$\overline{\text{MEMW}}$配合控制数据从外设传送到存储器
- **$\overline{\text{IOW}}$：I/O写信号，双向，三态，低电平有效**
  - 从态时，输入信号，有效时CPU写内部寄存器
  - 主态时，输出信号，与$\overline{\text{MEMR}}$配合控制数据从存储器传送到外设

8237A的引脚

- **$\overline{\text{MEMR}}$：存储器读，三态，输出，低电平有效**
- **$\overline{\text{MEMW}}$：存储器写，三态，输出，低电平有效**
- **$DREQ_3 \sim DREQ_0$：通道3 – 0的DMA请求信号，输入**
  - 外设请求DMA服务器时，向DREQ引脚送出有效信号
- **HRQ：保持请求信号，输出，高电平有效**
  - 送至CPU的HOLD引脚，申请总线控制
- **HLDA：保持相应信号，输入，高电平有效**
  - 与CPU的HLDA引脚相连，CPU在收到HRQ信号至少一个周期后，使HLDA置高，表示交出总线控制权

8237A的引脚

- **$DACK_3$ ~ $DACK_0$：通道3 – 0的DMA相应信号，输出**
  - 进入DMA传送后，相应通道的DACK有效，通知外部电路已进入DMA周期
- **$\overline{EOP}$：传输过程结束信号，双向，低电平有效**
  - DMA传送时，计数器减至0，再减为FFFFH而终止计数时，从该引脚输出低电平信号，表示传送结束
  - 外部输入低电平信号至该引脚，表示终止DMA传送

8237A的内部寄存器

名称	位数	数量
当前地址寄存器	16	4 (每通道一个)
当前字计数寄存器	16	4 (每通道一个)
基地址寄存器	16	4 (每通道一个)
基字计数寄存器	16	4 (每通道一个)
工作方式寄存器	6	4 (每通道一个)
命令寄存器	8	1 (4个通道公用一个)
状态寄存器	8	1 (4个通道公用一个)
请求寄存器	4	1 (每通道1位)
屏蔽寄存器	4	1 (每通道1位)
暂存寄存器	8	1 (每通道1位)

8237A的内部寄存器

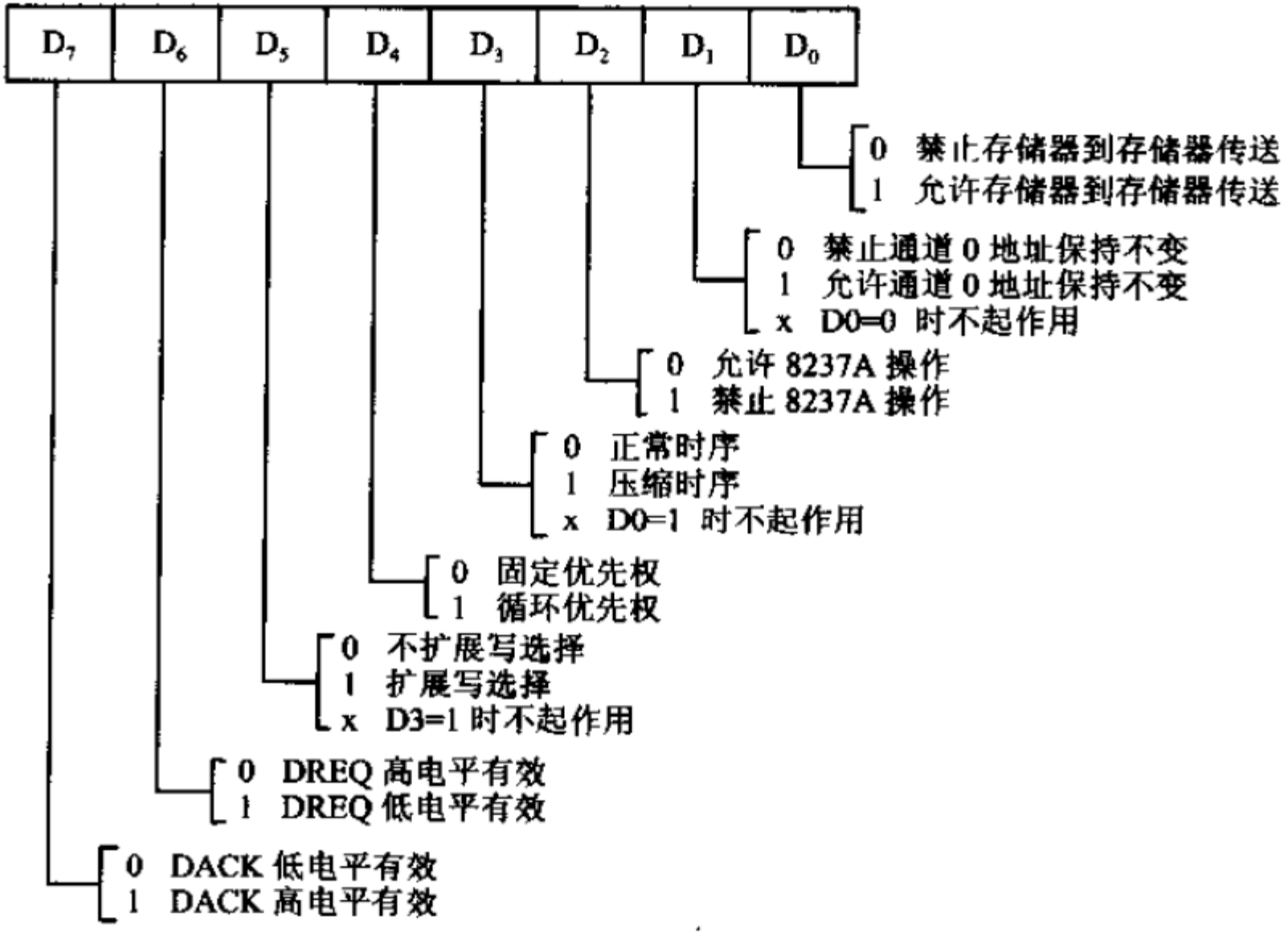
- **当前地址寄存器**
  - 存放DMA传送的存储器地址值
  - 每传送一个数据，地址值自动增1或减1
  - 可读写，每次8位，需分两次进行

- **当前字计数寄存器**
  - 为实际传送的字节数减1
  - 每传送一个字节，自动减1
  - 减为0，再由0减为FFFFH时，产生终止计数信号TC

8237A的内部寄存器

- **基地址寄存器**
  - 存放当前地址寄存器的初值
  - 不能被读出
  - 用于自动预置操作时的初值恢复
- **基字计数寄存器**
  - 存放当前字计数器的初值
  - 不能被读出
  - 用于自动预置操作时的初值恢复

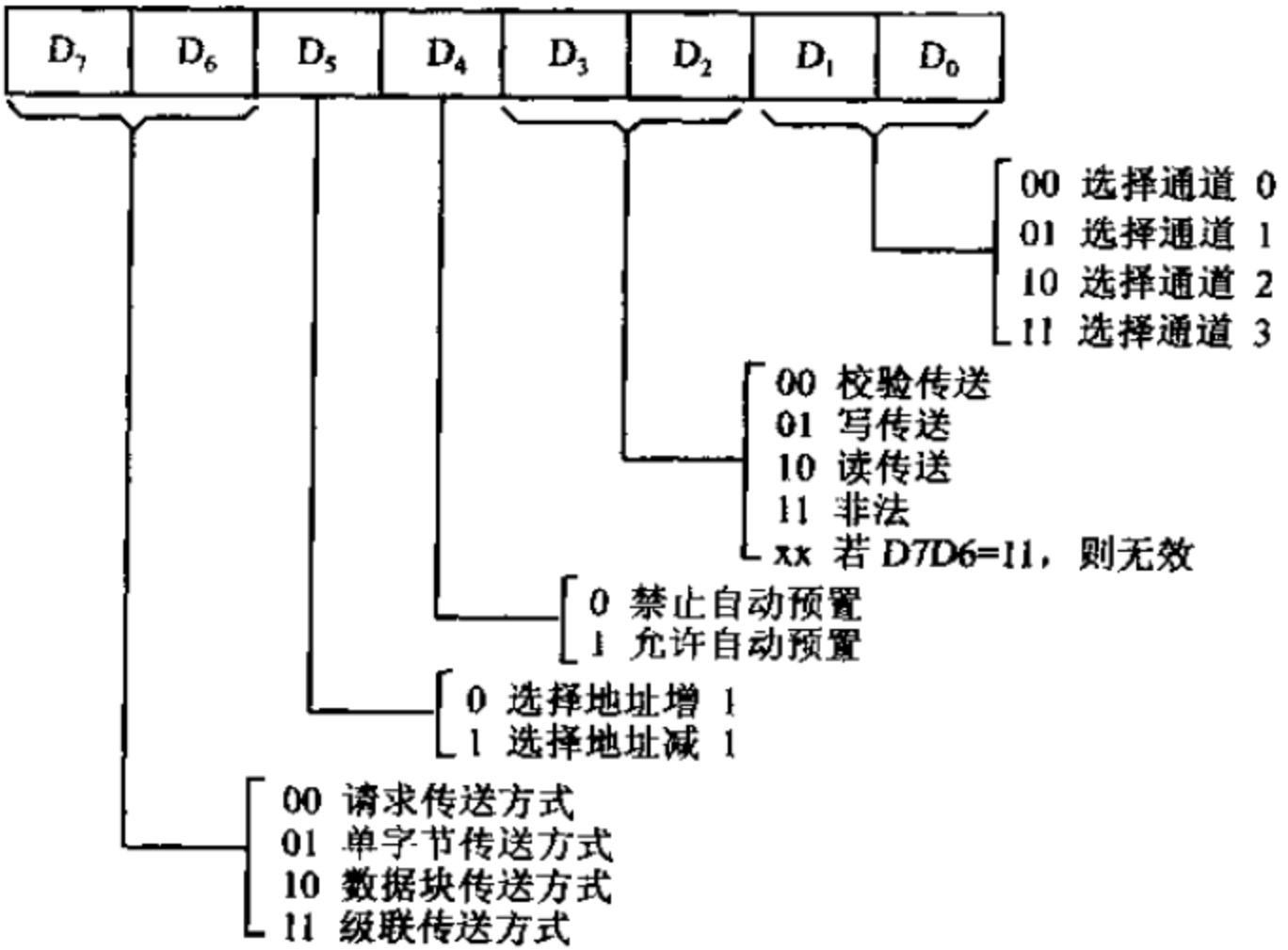
8237A的内部寄存器(命令寄存器)



8237A的优先级

- **固定优先权**
  - 通道0最高，通道3最低
  - $D_4 = 0$
- **循环优先权**
  - 使服务过的通道优先级变为最低，而让通道i + 1的优先级变为最高
  - D1 = 1
- **任一通道进入DMA服务时，其他通道均不能打断其操作**

8237A的内部寄存器(工作方式寄存器)



8237A的工作方式

单字节传送方式

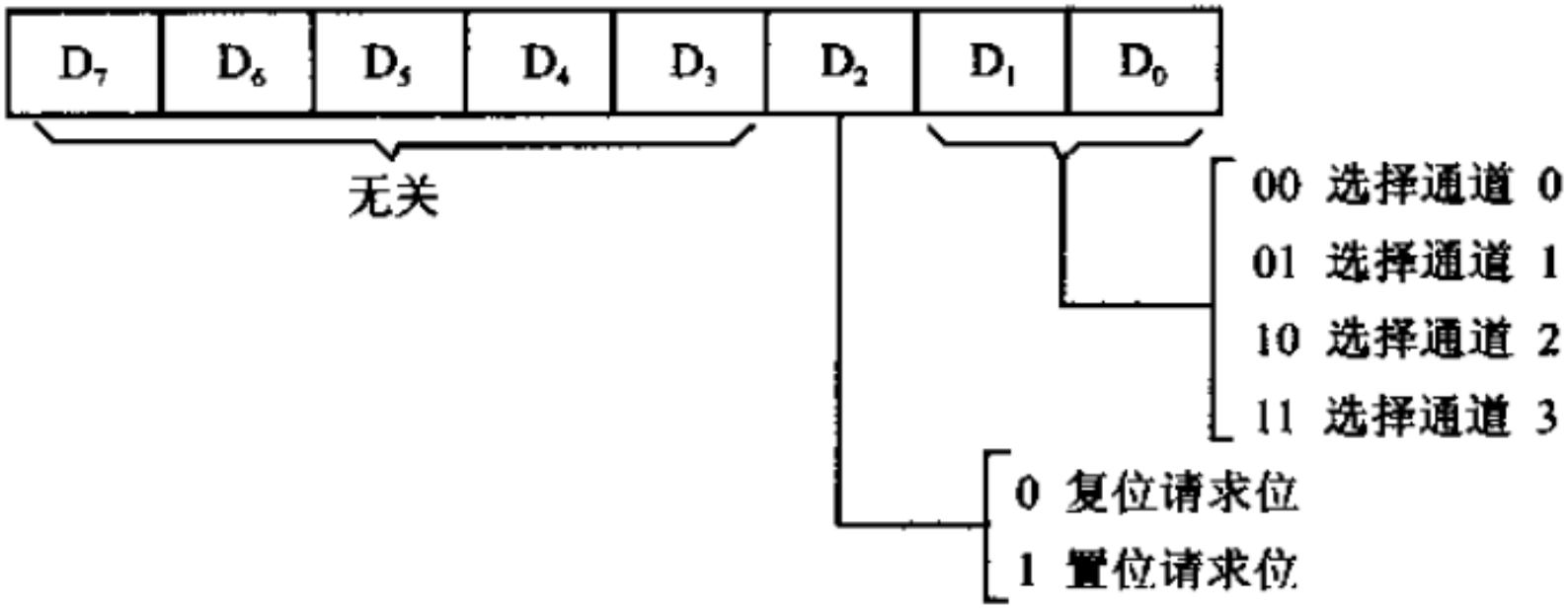
- 每进行一次DMA操作，只传送一个字节，字计数器减1，地址寄存器加1或减1
- 保持请求信号HRQ无效，释放系统总线
- HRQ会很快再次变为有效，从而开始下一个字节的传送
- 保证两次DMA传送之间，CPU可以执行一次完整的总线操作

8237A的工作方式

- **数据块传送方式**
  - DREQ有效，芯片进入DMA服务后，可以连续传送数据，一直到结束为止才释放总线
- **请求传送方式**
  - 与数据块传送方式类似
  - 不同点在于：每传送一个字节，检测一下DREQ，如果无效则马上暂停传送
  - 外设再次使DREQ有效后，又可以继续传送数据
- **级联传送方式**

8237A的内部寄存器(请求寄存器)

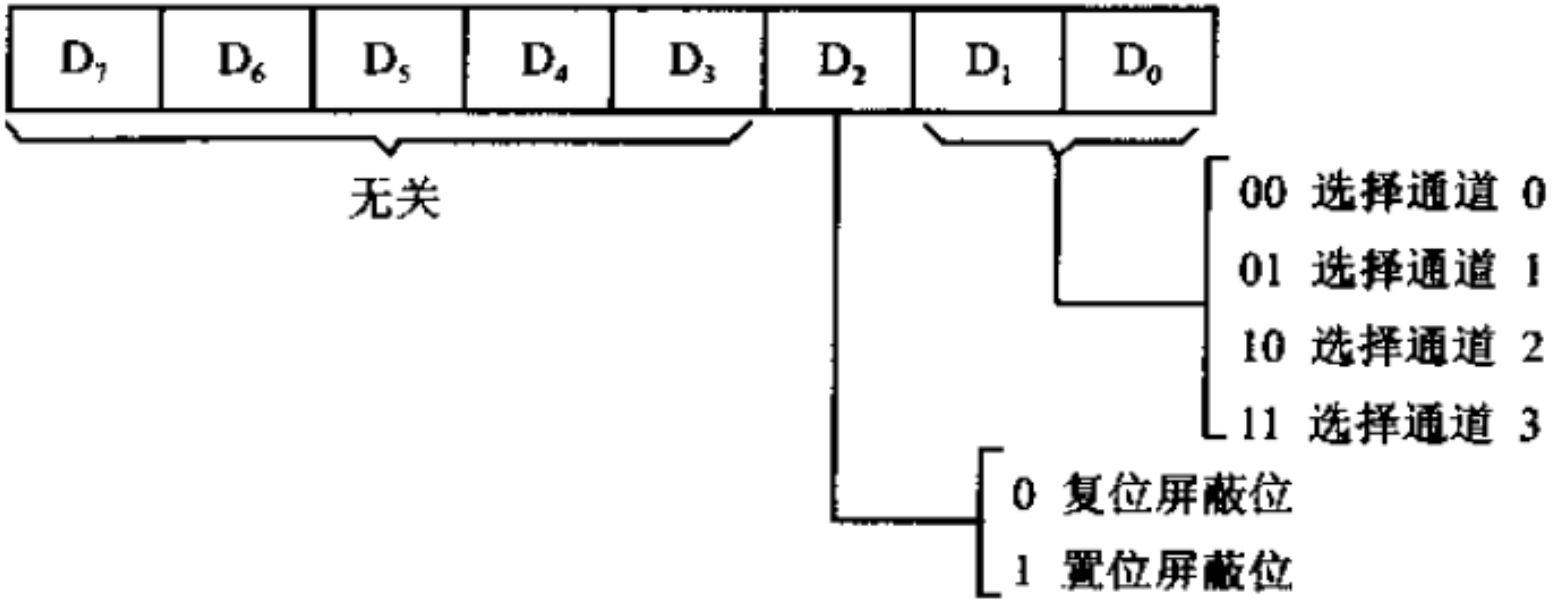
- 置1时产生请求，清0时不产生请求
- 可由软件置位，也可由硬件置位



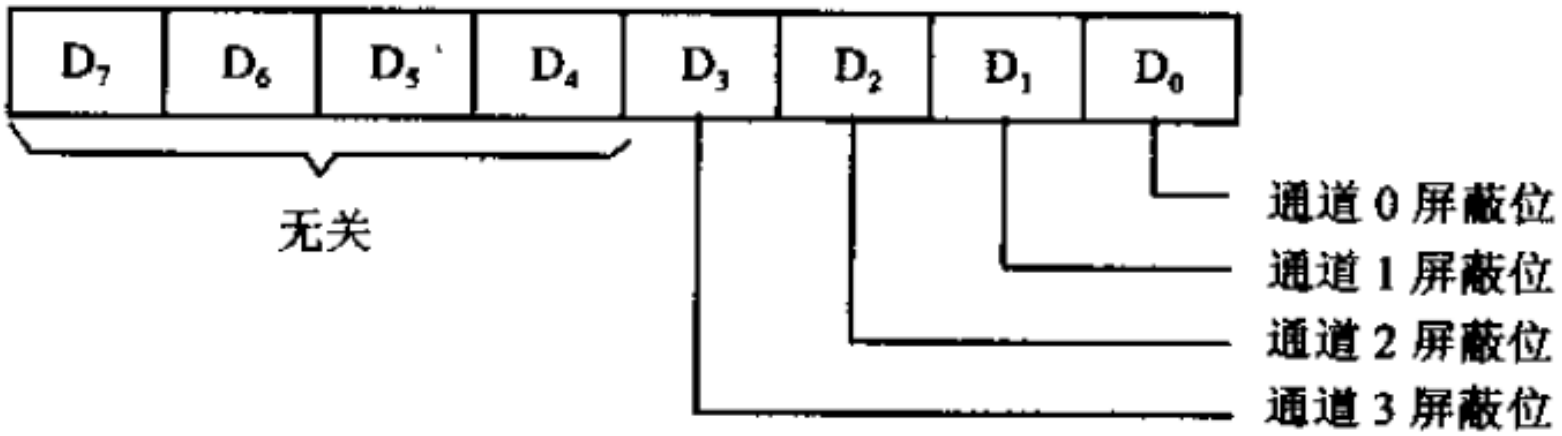
8237A的内部寄存器(屏蔽寄存器)

- **置1时禁止对应通道的DREQ请求**
- **清0时允许DREQ请求**
- **通道屏蔽字**
  - 对单个通道进行屏蔽置位或复位
- **主屏蔽字**
  - 一次完成对所有通道的置位或复位

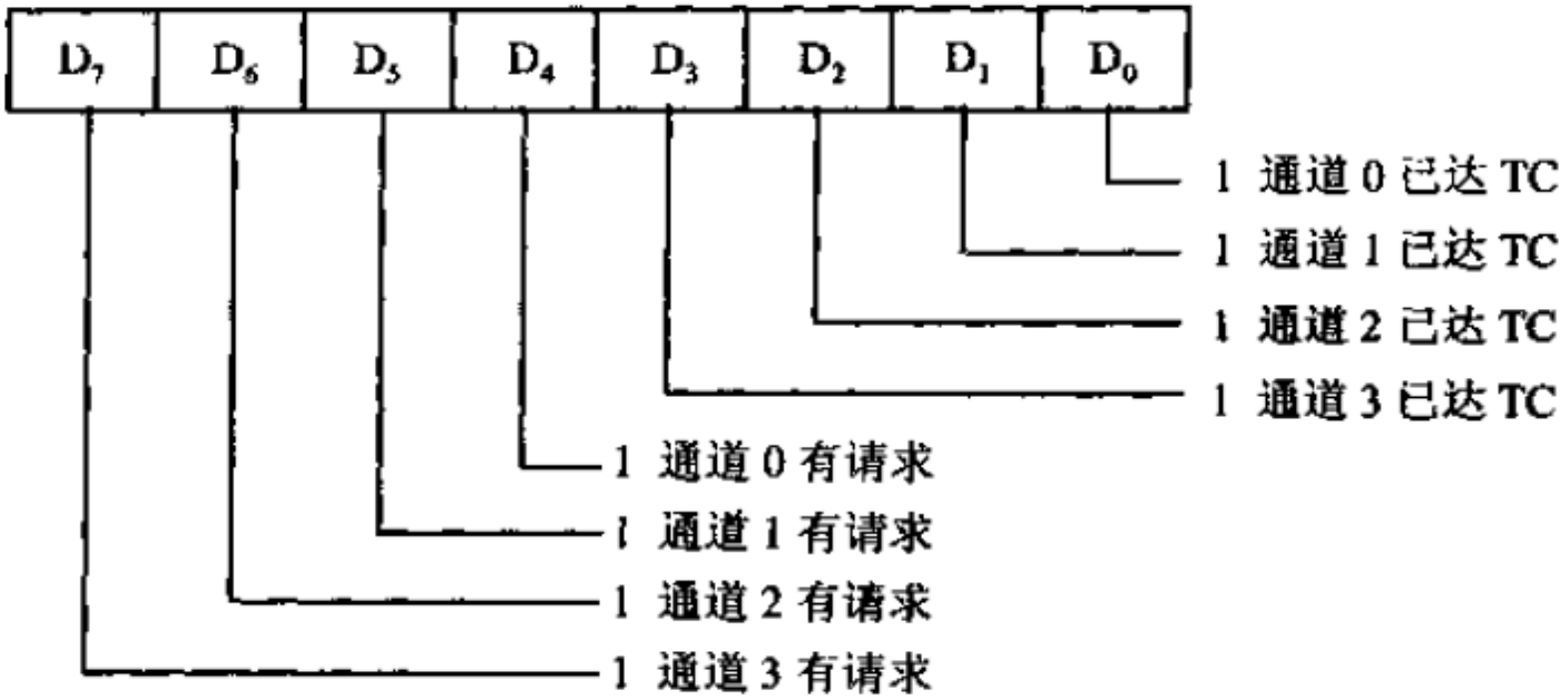
通道屏蔽字


D7 D6 D5 D4 D3 D2 D1 D0
无关
00 选择通道 0
01 选择通道 1
10 选择通道 2
11 选择通道 3
0 复位屏蔽位
1 置位屏蔽位

主屏蔽字



8237A的内部寄存器(状态寄存器)



软件命令

- **清除先/后触发器**
  - 清0时读/写低8位数据
  - 置1时读/写高8位数据
  - 每读写一次数据，该触发器自动翻转
  - 在读写16位寄存器时，为保证正确的顺序，应先对该寄存器清0
- **主清命令**
  - 使命令寄存器、状态寄存器、请求寄存器、暂存寄存器和内部先/后触发器均清0
  - 屏蔽寄存器置1
- **清除屏蔽寄存器**
  - 清除4个通道的屏蔽位

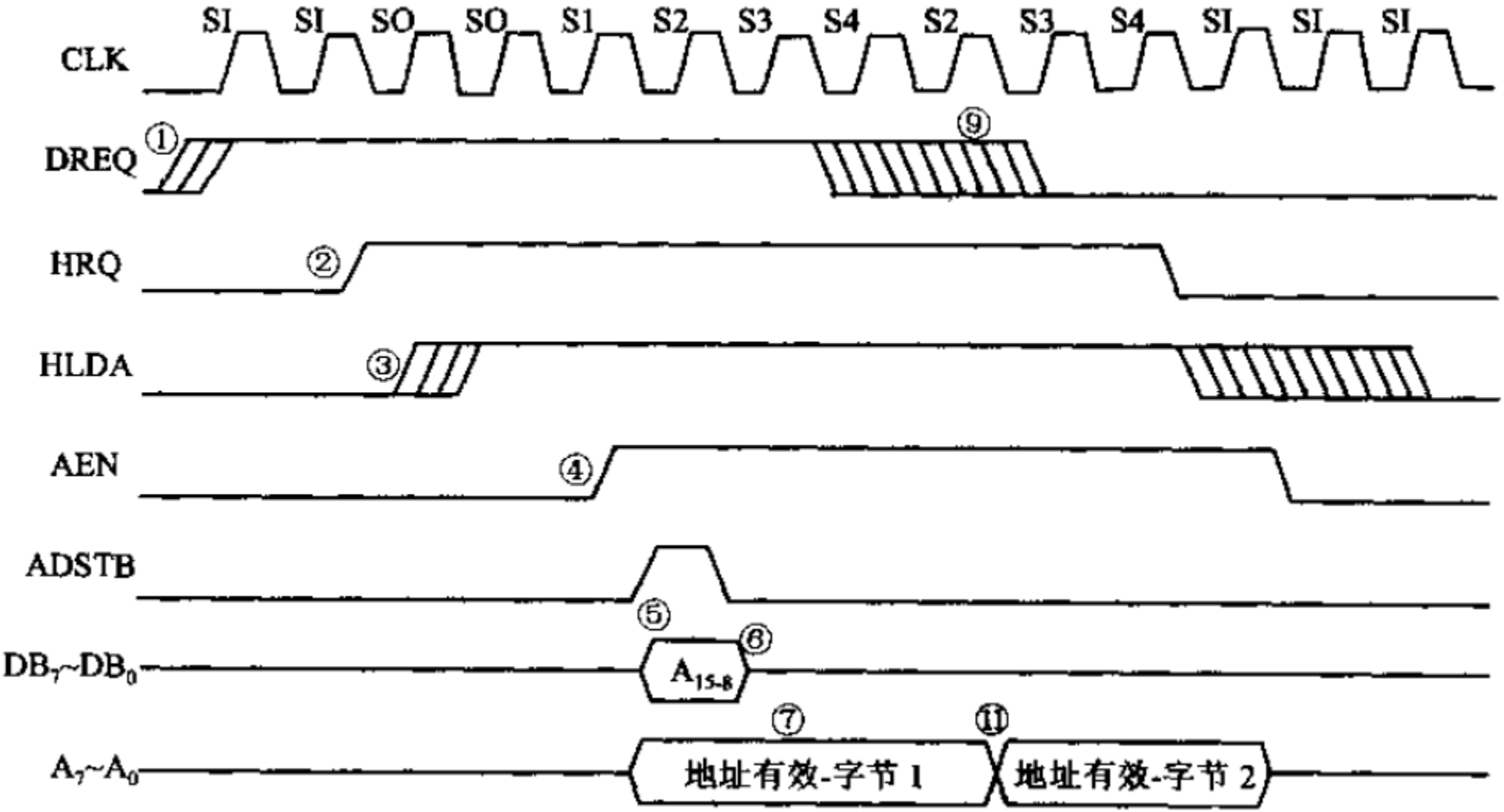
8237A内部寄存器口地址分配

I/O口地址16进制	寄存器	
	读($\overline{IOR}$)有效	写($\overline{IOW}$)有效
00	通道0当前地址寄存器	通道0基地址与当前地址寄存器
01	通道0当前字计数寄存器	通道0基字计数与当前字计数寄存器
02	通道1当前地址寄存器	通道1基地址与当前地址寄存器
03	通道1当前字计数寄存器	通道1基字计数与当前字计数寄存器
04	通道2当前地址寄存器	通道2基地址与当前地址寄存器
05	通道2当前字计数寄存器	通道2基字计数与当前字计数寄存器
06	通道3当前地址寄存器	通道3基地址与当前地址寄存器
07	通道3当前字计数寄存器	通道3基字计数与当前字计数寄存器

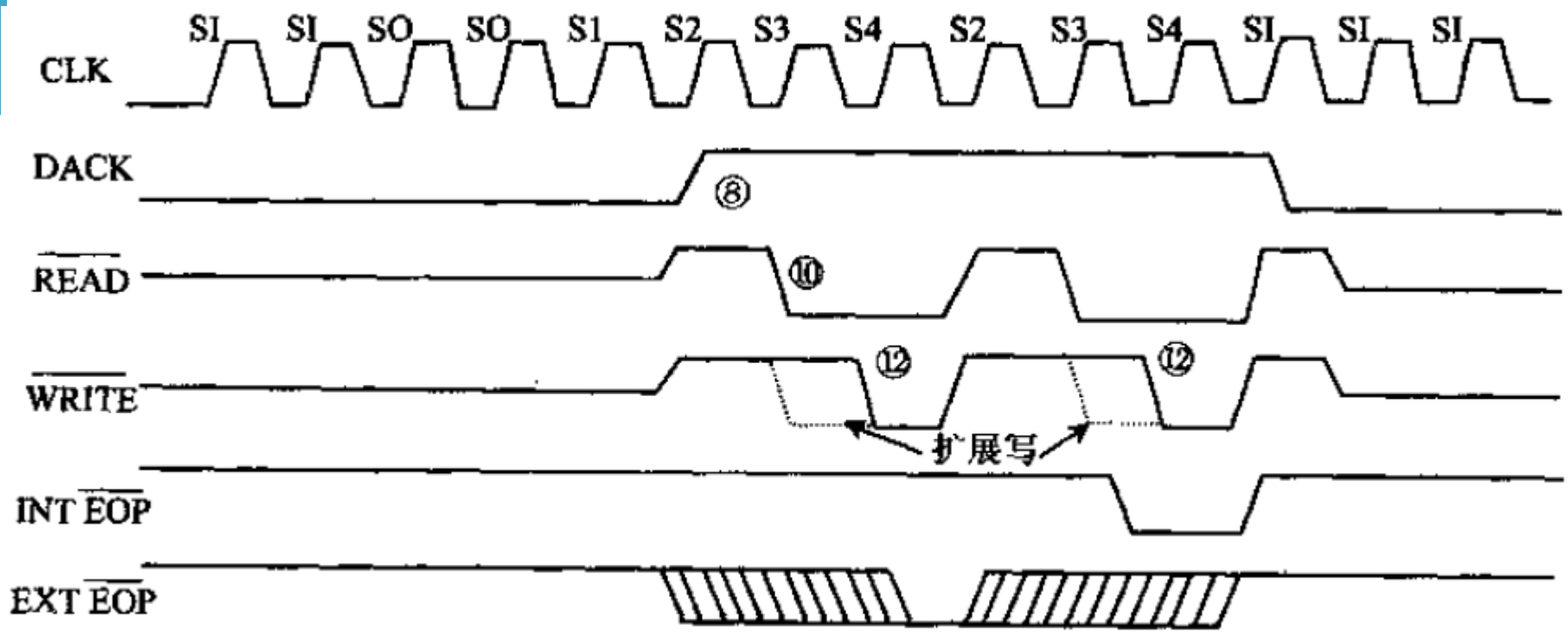
8237A内部寄存器口地址分配

I/O口地址16进制	寄存器	
	读($\overline{IOR}$)有效	写($\overline{IOW}$)有效
08	状态寄存器	命令寄存器
09	-	请求寄存器
0A	-	屏蔽寄存器(通道屏蔽字)
0B	-	工作方式寄存器
0C	-	清除先/后触发器
0D	暂存寄存器	主清命令寄存器
0E	-	屏蔽寄存器(清除屏蔽)
0F	-	屏蔽寄存器(主屏蔽字)

8237A的时序

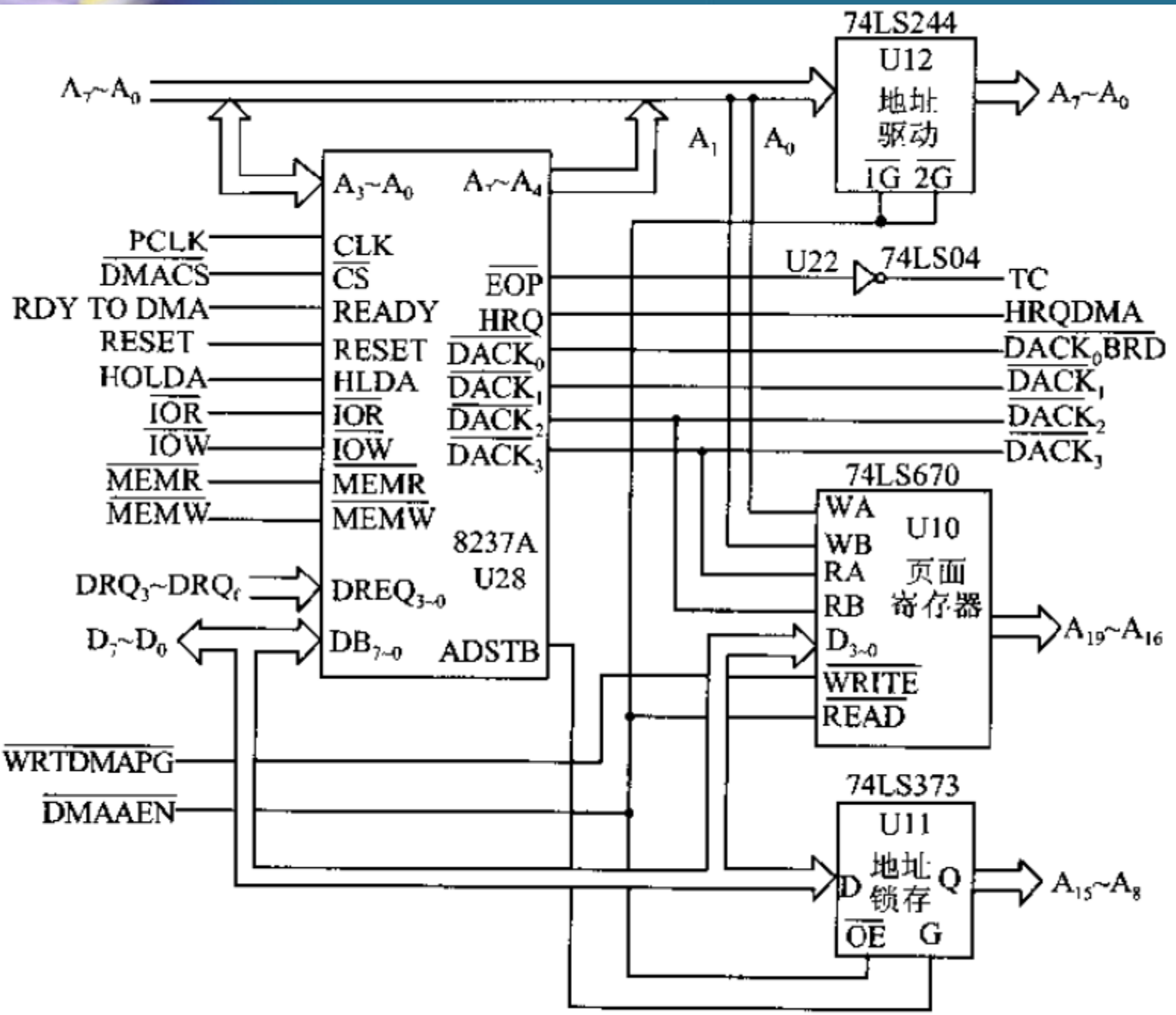

SI SI SO SO S1 S2 S3 S4 S2 S3 S4 SI SI SI
CLK
DREQ
①
⑨
HRQ
②
HLDA
③
AEN
④
ADSTB
⑤
⑥
DB_7~DB_0
A_{15-8}
⑦
⑪
A_7~A_0
地址有效-字节 1
地址有效-字节 2

8237A的时序


SI SI SO SO S1 S2 S3 S4 S2 S3 S4 SI SI SI
CLK
DACK
⑧
READ
⑩
WRITE
⑫
⑫
扩展写
INT EOP
EXT EOP

PC/XT机中的8237A

- **各通道功能**
  - 通道0：用于动态RAM的刷新
  - 通道1：为用户保留
  - 通道2：用于软盘DMA传送
  - 通道3：用于硬盘DMA传送
- **BIOS禁用了存储器-存储器的DMA传输**


74LS244
U12
地址
驱动
1G 2G
A_7~A_0
A_1
A_0
A_3~A_0
A_7~A_4
8237A
U28
PCLK
CLK
DMACS
CS
RDY TO DMA
READY
RESET
RESET
HOLDA
HLDA
IOR
IOR
IOW
IOW
MEMR
MEMR
MEMW
MEMW
EOP
HRQ
$DACK_0$
$DACK_1$
$DACK_2$
$DACK_3$
U22
74LS04
TC
HRQDMA
$DACK_0$BRD
$DACK_1$
$DACK_2$
$DACK_3$
DRQ_3~DRQ_0
$DREQ_{3-0}$
D_7~D_0
DB_{7-0}
ADSTB
74LS670
WA
WB
RA
RB
U10
页面
寄存器
D_{3-0}
WRITE
READ
A_{19}~A_{16}
WRTDMAPG
DMAAEN
74LS373
U11
D
地址
锁存
Q
OE
G
A_{15}~A_8

PC/XT的DMA控制逻辑

- DMA服务期间，A_7 ~ A_0输出低8位地址，送至地址驱动器U12的输入端
- S1, S2状态，DB_7 ~ DB_0输出高8位地址，通过ADSTB选通信号锁存至U11
- 为实现对1M空间的寻址，使用页面寄存器74LS670(U10)，以产生高4位地址A_{19} ~ A_{16}
- 在DMA传输过程中，页面寄存器值固定

页面寄存器

- 包含4个4位寄存器，对应4个DMA通道
- 4位输入接系统数据总线的$D_3 \sim D_0$
- 4位输出接系统地址总线的$A_{19} \sim A_{16}$
- WA，WB分别接地址总线的A_0和A_1
- $\overline{\text{WRITE}}$与片选信号$\overline{\text{WRTDMAPG}}$相连
- $\overline{\text{READ}}$与$\overline{\text{DMAAEN}}$信号相连，该信号在DMA传输过程中保持低电平
- RA与$DACK_3$相连，RB与$DACK_2$相连

74LS670内部寄存器写入

$\overline{\text{WRITE}}$	WB	WA	功能	对应通道
0	0	0	写入0号寄存器	未用
0	0	1	写入1号寄存器	通道2
0	1	0	写入2号寄存器	通道3
0	1	1	写入3号寄存器	通道1

74LS670内部寄存器读出

$\overline{\text{READ}}$	RB	RA	功能	对应通道
0	0	0	读出0号寄存器	未用
0	0	1	读出1号寄存器	通道2
0	1	0	读出2号寄存器	通道3
0	1	1	读出3号寄存器	通道1

8237A一般编程方法

- 输出主清命令，复位
- 写入基地址和现行地址寄存器，确定起始地址
- 写入基字和现行字寄存器，确定数据长度
- 写入方式寄存器，指定工作方式
- 写入屏蔽寄存器
- 写入命令寄存器
- 写入请求寄存器

Thank You !

S.J.T.U.