

《微机原理与接口》

第2章 微机原理(8088)

教师：苏曙光

华中科技大学软件学院

I 第二章 微机原理

n第1节 8088CPU内部组成

n第2节 数字电路和常用IC

n第3节 8088CPU外部结构

n第4节 8088CPU总线时序

I 本章重点

n8088内部结构

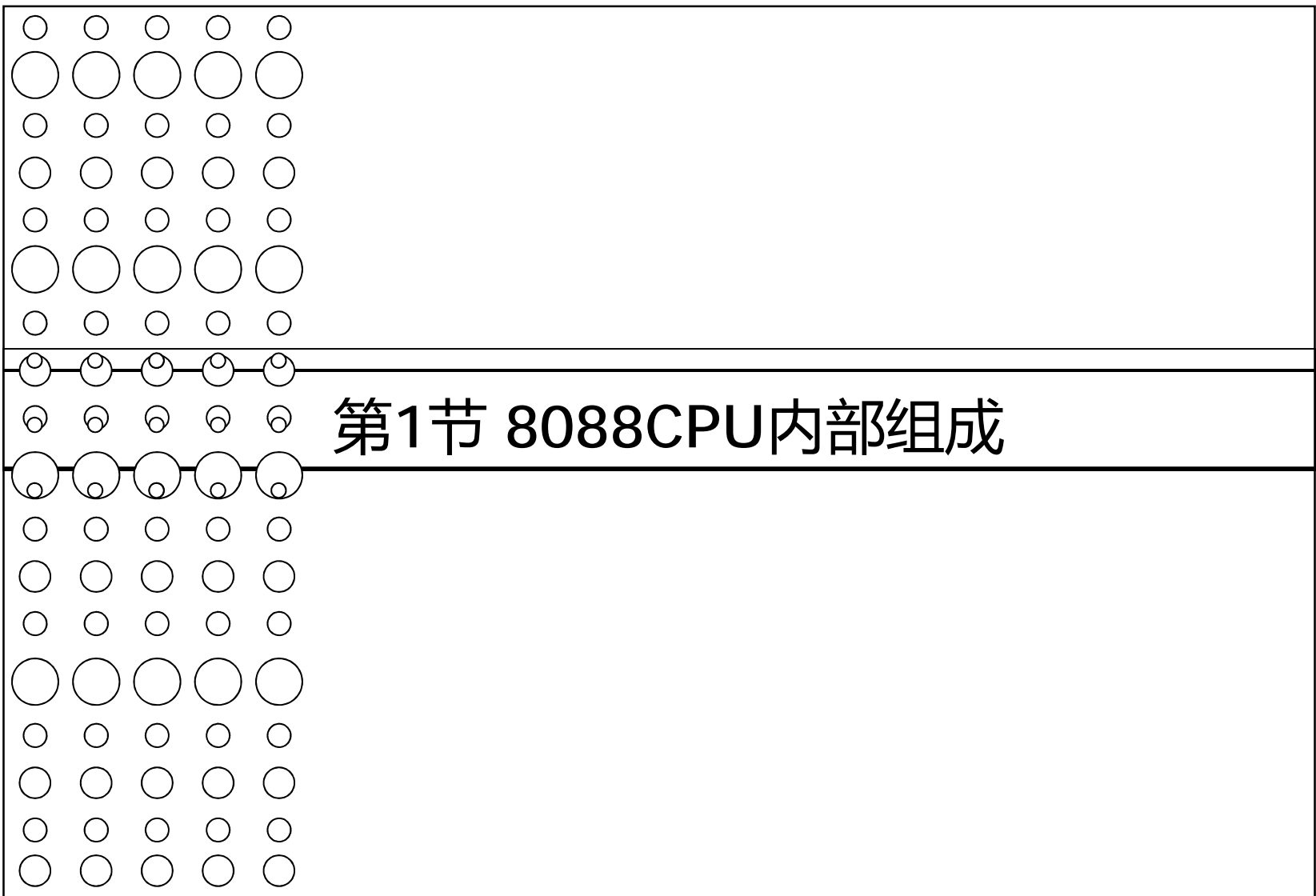
n8088外部引脚；

n8088内部寄存器；

n8088的存储器组织；

n8088的工作时序

n数字电路回顾：常用门和IC芯片



I 8088/8086 MPU

n相同点

- u16位CPU：内部寄存器16位

- u20根地址线：1MB内存

n差异

- u数据总线

 - p8086：内外16根

 - p8088：内部16根，外部8根：准16位机；

- u指令队列

 - p8086：6字节

 - p8088：4字节

I 8086/8088 MPU管脚

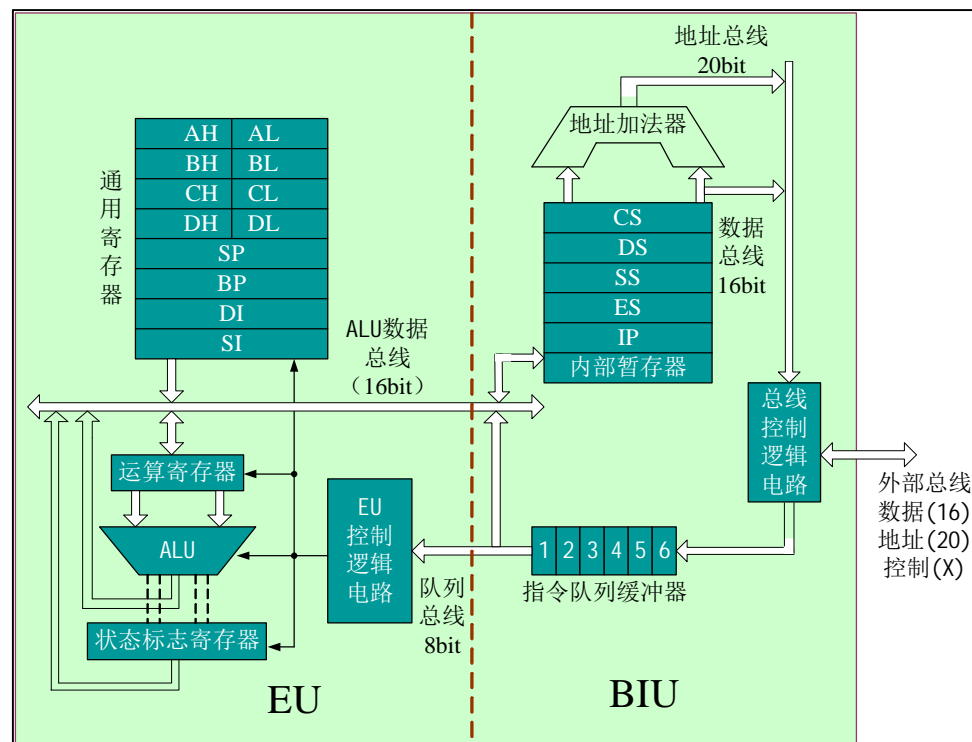
地	1	40	Vcc(+5V)
AD14	2	39	AD15
AD13	3	38	A16 / S3
AD12	4	37	A17 / S4
AD11	5	36	A18 / S5
AD10	6	35	A19 / S6
AD9	7	34	BHE / S7
AD8	8	33	MN / MX
AD7	9	32	RD
AD6	10	31	HOLD ($\overline{RQ}/\overline{GT0}$)
AD5	11	30	HLDA ($\overline{RQ}/\overline{GT1}$)
AD4	12	29	WR (\overline{LOCK})
AD3	13	28	M / IO ($\overline{S2}$)
AD2	14	27	DT / R ($\overline{S1}$)
AD1	15	26	\overline{DEN} ($\overline{S0}$)
AD0	16	25	ALE (QS0)
NMI	17	24	\overline{INTA} (QS1)
INTR	18	23	TEST
CLK	19	22	READY
地	20	21	RESET

地	1	40	Vcc(+5V)
A14	2	39	A15
A13	3	38	A16 / S3
A12	4	37	A17 / S4
A11	5	36	A18 / S5
A10	6	35	A19 / S6
A9	7	34	$\overline{SS0}$ (HIGH)
A8	8	33	MN / \overline{MX}
AD7	9	32	RD
AD6	10	31	HOLD ($\overline{RQ}/\overline{GT0}$)
AD5	11	30	HLDA ($\overline{RQ}/\overline{GT1}$)
AD4	12	29	WR (\overline{LOCK})
AD3	13	28	M / IO ($\overline{S2}$)
AD2	14	27	DT / R ($\overline{S1}$)
AD1	15	26	\overline{DEN} ($\overline{S0}$)
AD0	16	25	ALE (QS0)
NMI	17	24	\overline{INTA} (QS1)
INTR	18	23	TEST
CLK	19	22	READY
地	20	21	RESET



8088 内部结构：EU和BIU

- I EU (Execute Unit, 执行单元)：负责执行指令或运算
- I BIU (Bus Interface Unit, 总线接口单元)：负责读指令或数据



BIU功能和内部构成

I 功能

n 预取指令

u 执行指令的同时从内存取下一条或几条指令放在队列中。

u 指令队列：6字节或4字节。

n 指令执行顺序

u 顺序指令执行。

u 转移指令执行：清除队列。

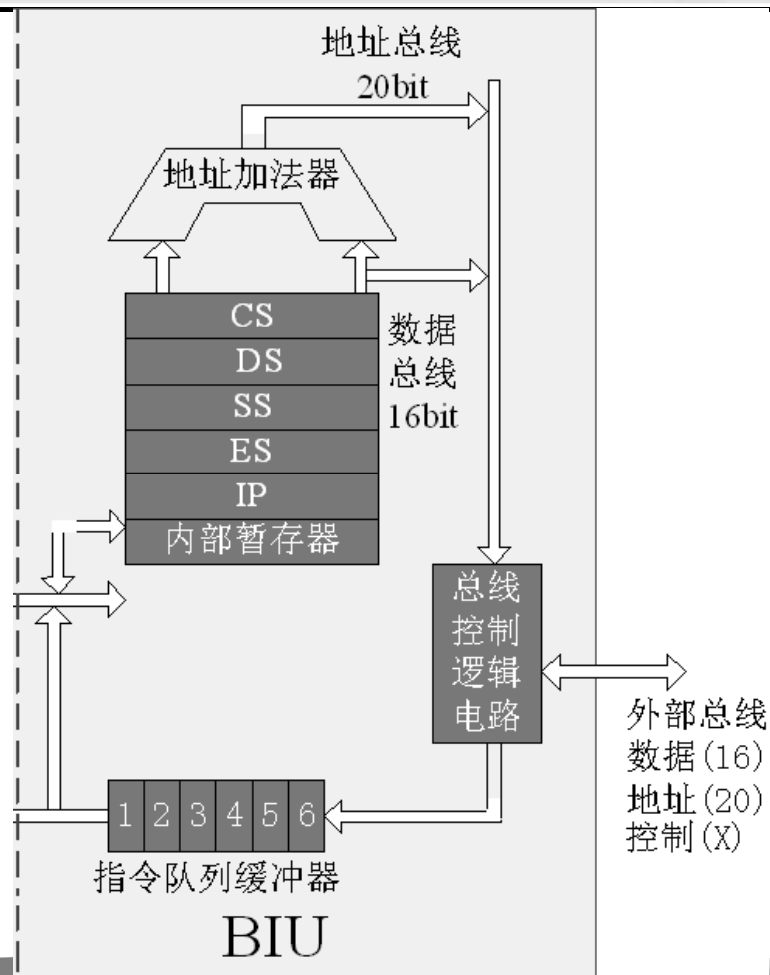
I 构成

n 1) 一组段寄存器+ 指令指针IP

n 2) 地址加法器：将段地址和偏移地址相加，形成20位物理地址

n 3) 指令队列缓冲器：寄存指令。

n 4) 总线控制逻辑：内外总线接口。



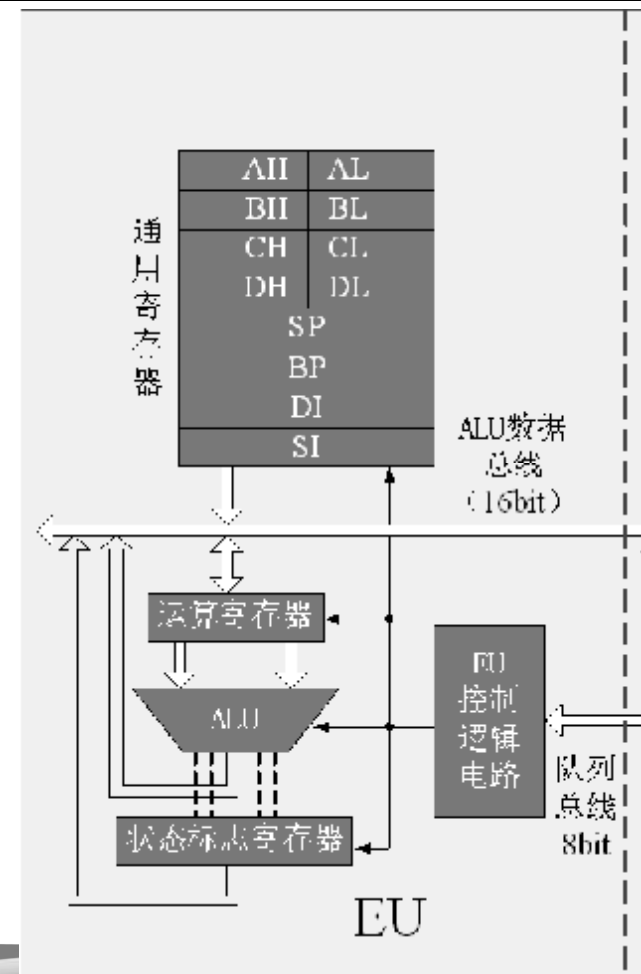
EU功能和内部构成

I 功能:负责执行指令或运算

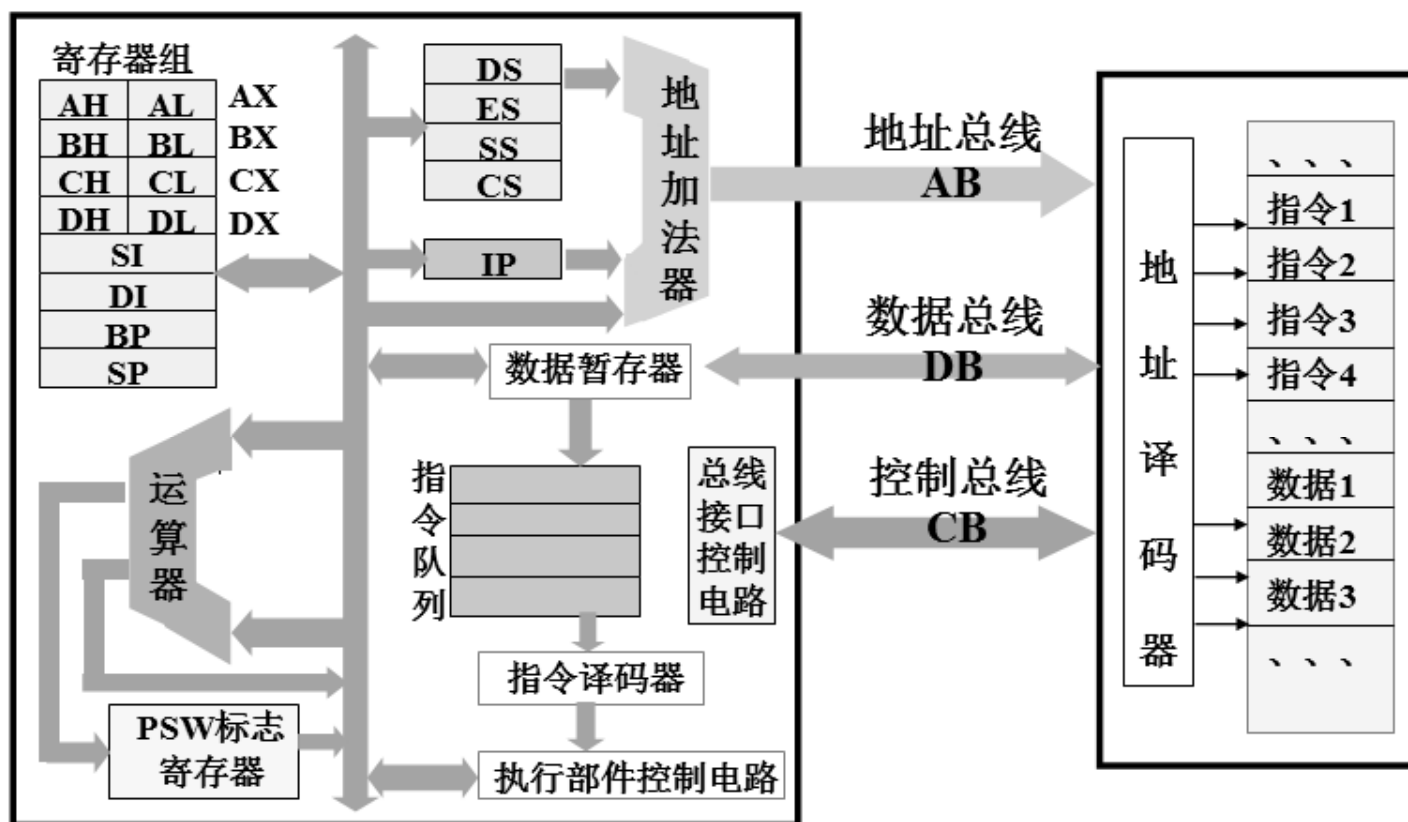
- n 从指令队列中取指令代码,译码,在 **ALU**中完成数据的运算,结果的特征保存在标志寄存器中。

I 内部构成

- n 1) **ALU**: 执行基本运算和处理.
- n 2) 一组通用寄存器 + 标志寄存器
- n 3) **EU**控制系统: 队列控制和时序控制



8088工作原理：取指令, 执行指令



CPU

总线

内存

8088并行工作方式：流水线

- 指令预取队列的存在使**EU**和**BIU**可同时工作
- 2级流水线



8086/8088 CPU的特点

- 丨 采用并行流水线工作方式
- 丨 支持多处理器系统
- 丨 片内无浮点运算部件，浮点运算由数学协处理器**8087**支持（也可用软件模拟）
 - n注：**80486DX**以后的**CPU**均将数学协处理器作为标准部件集成到**CPU**内部
- 丨 对内存空间实行分段管理



I 8088的运行（执行）环境

n寄存器

n内存空间

n堆栈（Stack）

nI/O端口

14个基本寄存器

- | 8个通用寄存器(**General Registers**)
- | 1个标志寄存器(**F: Flags Register**);
- | 1个指令指针寄存器 (**IP: Instruction Point Register**)
- | 4个段寄存器(**Segment Registers**)。

nCS,DS,SS,ES

数据
寄存
器

AH	AL
BH	BL
CH	CL
DH	DL

AX 累加器

BX 基址寄存器

CX 计数据器

DX 数据寄存器

FLAGS
IP

标志寄存器
指令指针

地址
寄存
器

SI
DI
BP
SP

源地址寄存器

目的寄存器

基址寄存器

堆栈指针

CS
SS
DS
ES

代码段寄存器

堆栈段寄存器

数据段寄存器

附加段寄存器

4个数据寄存器：AX,BX,CX和DX

- | 常用来存放参与运算的操作数或运算结果
- | 16位数据寄存器，分为8个8位寄存器
 - n AX: AH, AL
 - n BX: BH, BL
 - n CX: CH, CL
 - n DX: DH, DL
- | 8位可以单独操作，不影响另外8位

I 习惯使用

nAX: 累加器。多用于存放中间运算结果。使用频率最高，用于算术、逻辑运算以及与外设传送信息等；

nBX: 基址寄存器。常用于存放内存地址；

nCX: 计数寄存器。用于在循环或串操作指令中存放循环次数或重复次数；

nDX: 数据寄存器。在32位乘法运算存放高16位数；

I 寄存器的特殊使用

nAX—操作数和结果数据的累加器;

nBX—在**DS**段中数据的指针;

nCX—串和循环操作的计数器;

nDX—I/O指针(端口地址);

4个段寄存器：CS,DS,ES和SS

I 用于存放逻辑段的段基地址

nCS：代码段寄存器 Code Segment

└ 代码段用于存放指令代码

nDS：数据段寄存器 Data Segment

nES：附加段寄存器 Extended Segment

└ 数据段和附加段用来存放操作数

nSS：堆栈段寄存器 Stack Segment

└ 堆栈段用于存放返回地址，

└ 保存寄存器内容，传递参数

2个指针寄存器：SP和BP

I SP, BP

n指针寄存器，用于寻址堆栈内的数据

n与段寄存器**SS**联合使用，确定堆栈中的单元地址

I SP

n堆栈指针寄存器，其内容为栈顶的偏移地址；

I BP

n基址指针寄存器，表示数据在堆栈中的基地址。

I **BX**与**BP**在应用上的区别

n作为通用寄存器，二者均可用于存放数据；

nBX通常用于寻址数据段**DS**或扩展**ES**段

nBP则通常用于寻址堆栈段**SS**。

2个变址寄存器：SI和DI

I SI,DI

n常用于指令的间接寻址或变址寻址。

I SI: 源变址寄存器

n指向**DS**段中的数据指针、串操作的源指针；

I DI: 目的变址寄存器

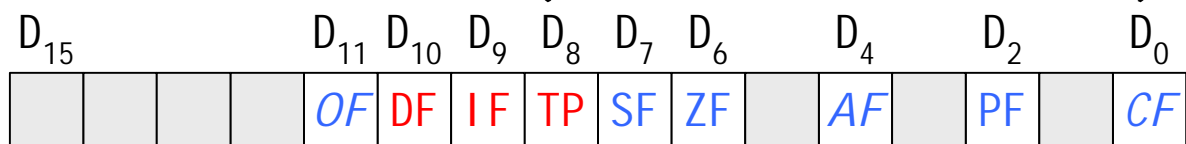
n指向**ES**段中的数据指针、串操作的目标指针；

1个指令指针寄存器：IP

- I 存储**CPU**将要执行的下一条指令的偏移地址；
- I **CPU**在执行完一条指令之后，会自动将下一条指令的偏移地址存入到**IP**中。

1个状态标志寄存器：F

1 16位，包含9个标志位(6个状态位，3个控制位)



n状态位：标示算术、逻辑运算的结果状态

n控制位：控制CPU的下一步操作

n状态位的例子

uOF (Overflow Flag)，溢出标志位

p功能：标示符号数的运算结果是否溢出

n控制位的例子

uDF (Direction Flag)，方向位

p功能：用于控制字符串操作的地址步进方向

8088的存储器结构

I 存储空间

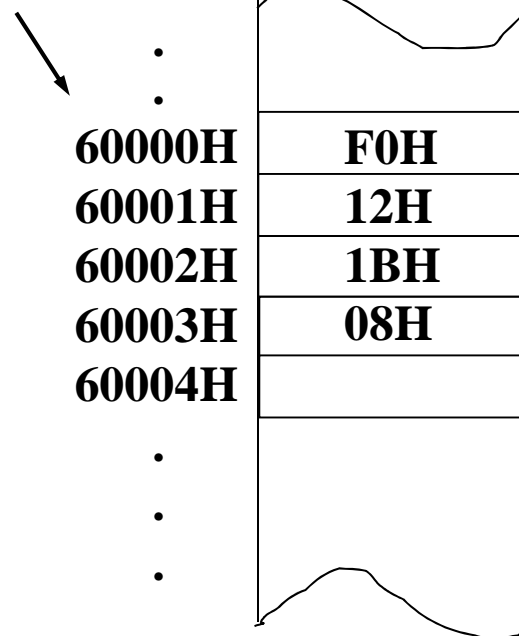
n20根地址线: 1M

I 问题:

n寄存器16位, 如何生成20位地址?

n解决: 存储器分段

物理地址

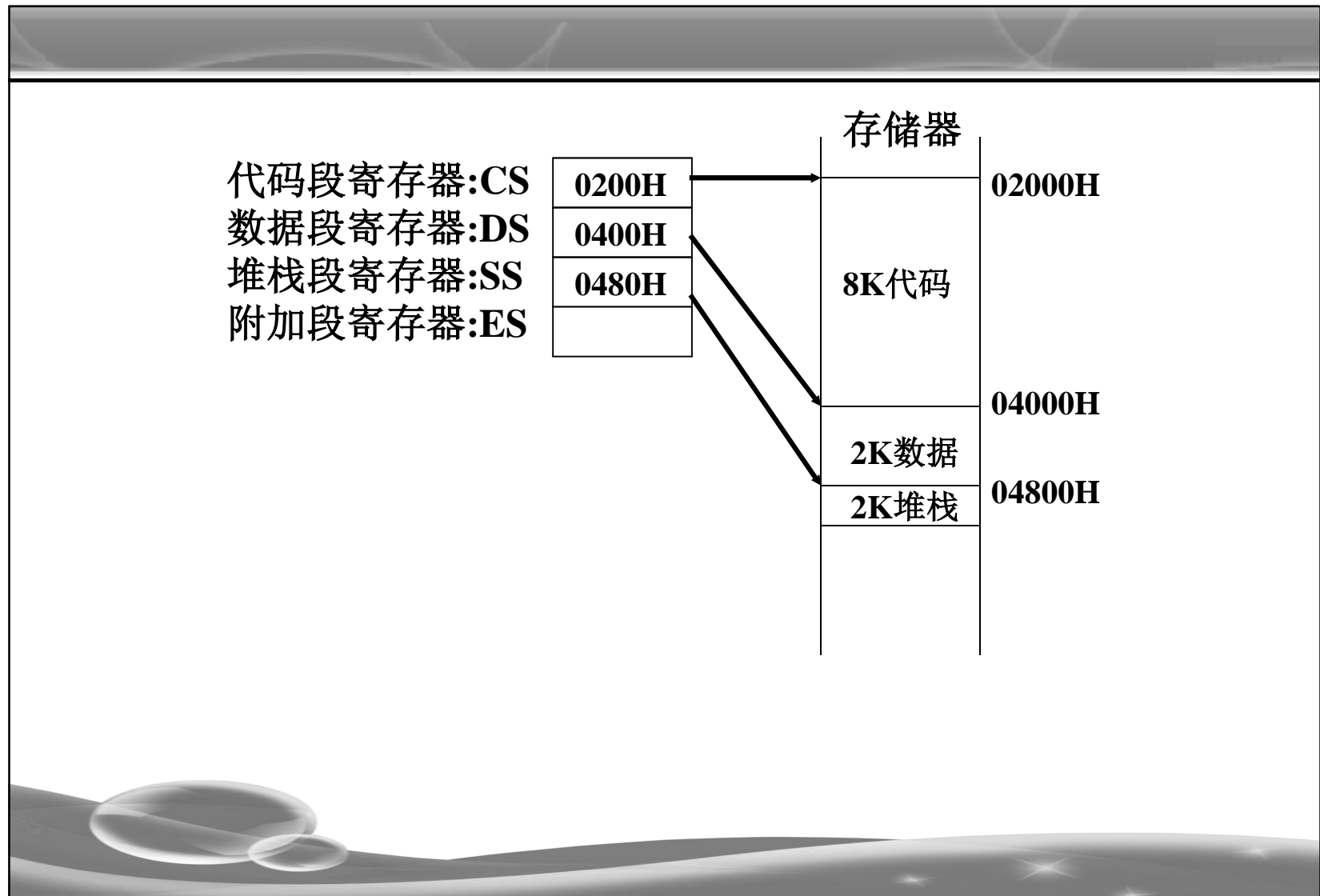


:	
:	
60000H	F0H
60001H	12H
60002H	1BH
60003H	08H
60004H	
.	
.	
.	

I 分段管理

n 段起始地址(20位)的高十六位称为该段的段地址

u 段地址放在**DS**、**SS**、**CS**、**ES**中



I 分段管理

n 段起始地址(20位)的高十六位称为该段的段地址

u 段地址放在**DS、SS、CS、ES**中

n 段内某个单元到段首的偏移称为偏移地址（有效地址**EA**）；

u 偏移地址放在**BX、SI、DI、BP、SP、IP**中。

u 最大段： $64K = 2^{16}B$

n 物理地址 = 段地址（16位）* 10H + 偏移（16位）

n 例如

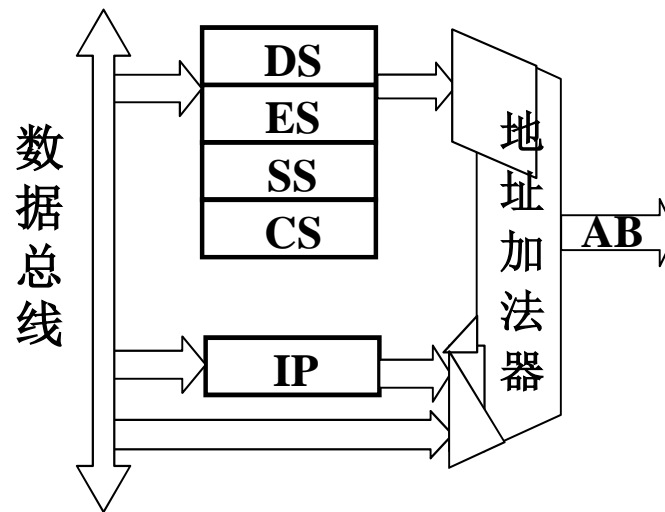
DS:BX (数据区)

DS:SI (数据区)

SS:SP (堆栈区)

CS:IP (代码区)

I 物理地址 = 段地址 (16位) * 10H + 偏移 (16位)



数据存放规律

I 字节数据

n 一单元存放一个数

例子: **E4H** 放在 **00001H**单元;

I 字数据

n 2单元: “低对低, 高对高”

n 字的地址: 2 个单元中的低地址

例子: **76E4H**放在**00001H**地址中

I 字符串

n 按字符顺序按地址递增存放。

I 机器指令(机器码):

n 按字节顺序按地址递增存放, 同字符串的放置方式

如: **MOV BX, AX ; 89C3H**, 放在**00004H**单元

0 0000H	23H
0 0001H	E4H
0 0002H	76H
0 0003H	1 0 1 0 0 1 0 1
0 0004H	89H
0 0005H	C3H
0 0006H	21H
...	...
F FFEH	41H
F FFFH	42H

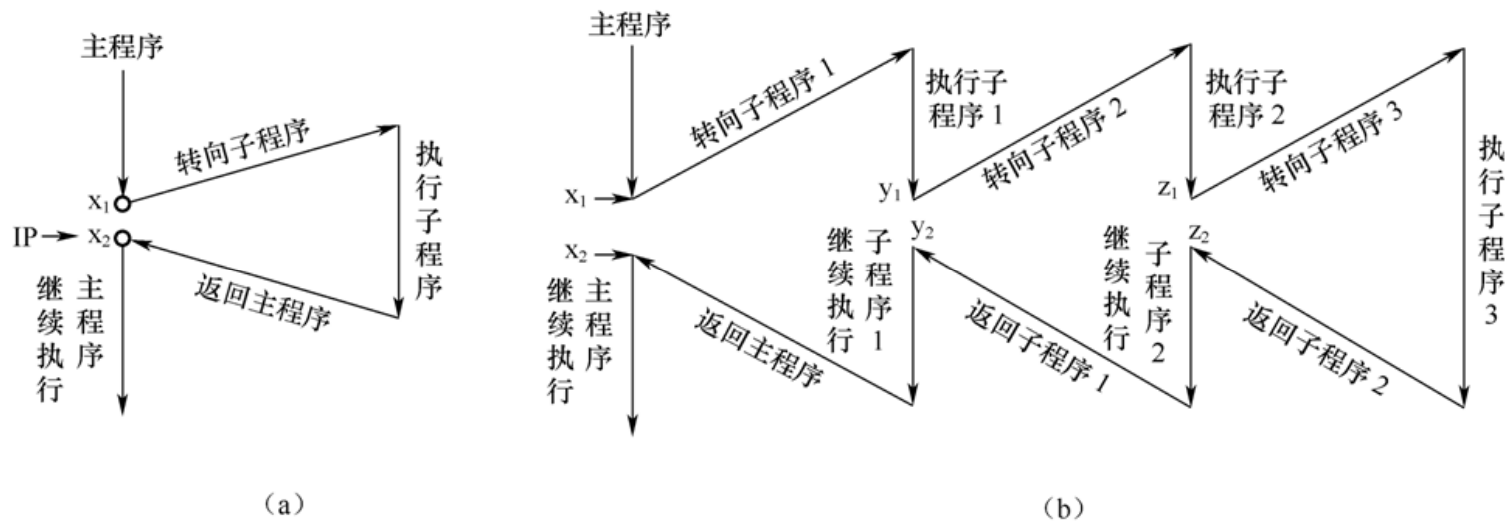
I 练习题

n00002H单元存放的字节/字/双字节指令为多少?

0 0000H	23H
0 0001H	E4H
0 0002H	76H
0 0003H	1 0 1 0 0 1 0 1
0 0004H	89H
0 0005H	C3H
0 0006H	21H
...	...
F FFEH	41H
F FFFH	42H

堆栈

I 例子：子程序调用的过程



n调用发生后，主程序在**CPU**中的运行环境被破坏。

n调用返回时，必须恢复主程序之前的运行环境

I 堆栈 (STACK)

n 功能一：在子程序调用和中断服务时存储现场数据；

n 特殊内存

- u “后进先出” (LIFO)存储

- u 堆栈一端固定(栈底)，另一端活动(栈顶)，数据只允许从栈顶存取（进或出）

- u 栈指针：指示栈顶位置 (Stack Poniter, SP)

n 堆栈的伸展方向

- u 栈底的地址大，栈顶的地址小

n 栈的操作 (PC)

- u 入栈：将一个数存入栈顶，并改变SP（变小）

- u 出栈：从栈顶读出一个数据,并改变SP（变大）

I 入栈操作

n **PUSH SRC**; **SRC** 代表寄存器或存储单元地址

n 功能：将寄存器或存储单元中的一个字压入堆栈

n 操作：

u “先减后入”：**SP-1 → SP**，字高位 → **[SP]**

SP-1 → SP，字低位 → **[SP]**

n 结果：**SP-2**，数据高对高，低对低存放。

n 例：**AX=1122H**，**BX=3344H**

SS=095BH，**SP=0040H**

执行：**PUSH AX ;SP=003EH**

PUSH BX ;SP=003CH

<i>SP</i> →	003CH	44H	0 95ECH
	003DH	33H	0 95EDH
↑	→ 003EH	22H	0 95EEH
	003FH	11H	0 95EFH
<i>SP</i> →	0040H		0 95F0H

I 出栈操作

n **POP DST** ; **DST** 代表寄存器或存储单元地址

n 功能：将栈顶一个字传送到寄存器或存储单元中

n 操作

u “先出后加”：[**SP**] → 字低位， **SP+1** → **SP**

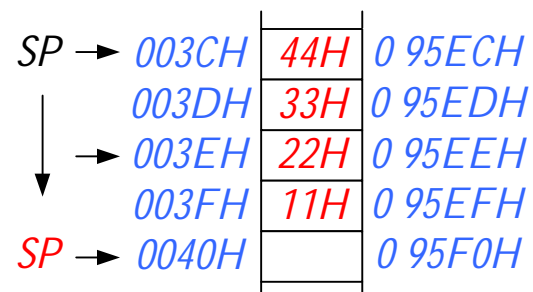
[**SP**] → 字高位， **SP+1** → **SP**

n 结果：**SP+2**, 数据低对低,高对高存放

n 例：上述前一例子中再执行：

POP CX ; **CX=3344H, SP=003EH**

POP DX ; **DX=1122H, SP=0040H**



PUSHA

I (Push All) 将所有 (8个) 16位通用寄存器存入堆栈。

nTemp ← (SP) ;

nPush (AX) ;

nPush (CX) ;

nPush (DX) ;

nPush (BX) ;

nPush (Temp) ; //SP

nPush (BP) ;

nPush (SI) ;

nPush (DI) ;

POPA

I Pop All, 自堆栈弹出至相应的**16**位通用寄存器。

nDI←Pop();

nSI←Pop();

nBP←Pop();

nSP增量**2**（跳过堆栈的下**2**个字节）

nBX←Pop();

nDX←Pop();

nCX←Pop();

nAX←Pop();

I **Flag**寄存器出/入栈

n命令格式

PUSHF; F入栈, $SP-2 \rightarrow SP$

POPF ; F出栈, $SP+2 \rightarrow SP$

n功能: 保护和恢复状态标志寄存器**Flag**

I 注意:

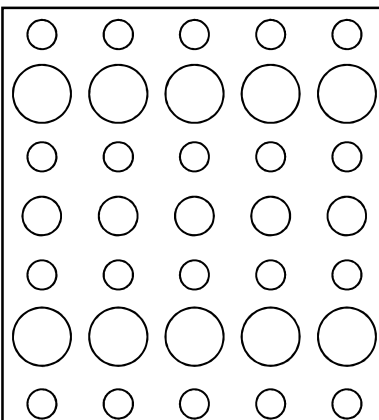
n栈操均以字为单位, 下列指令均错:

PUSH AL

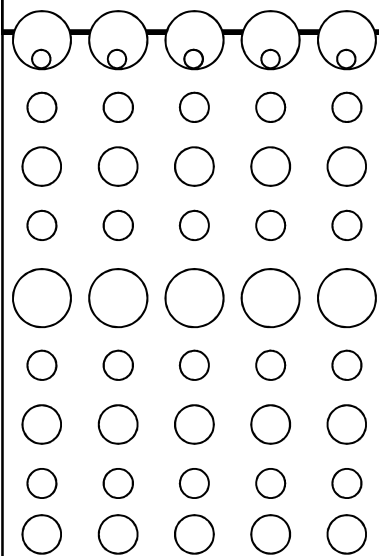
POP DH

n**PUSH**与**POP**成对, 避免堆栈溢出或程序出错;

n堆栈实为内存区, 还可按数据区的方法对其操作。



第3节 数字电路和常用IC芯片



I 本节基本内容

- n 数字电路基本概念

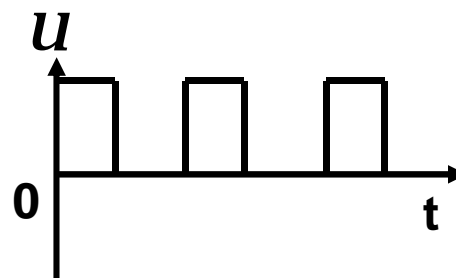
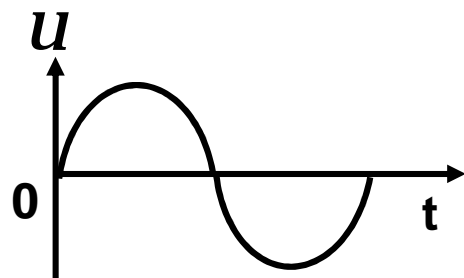
- n 常用门电路

- n 常用IC芯片

I 电路中两类信号

n模拟信号：在时间上和幅值上均连续的信号

n数字信号：在时间上和幅值上均离散的信号

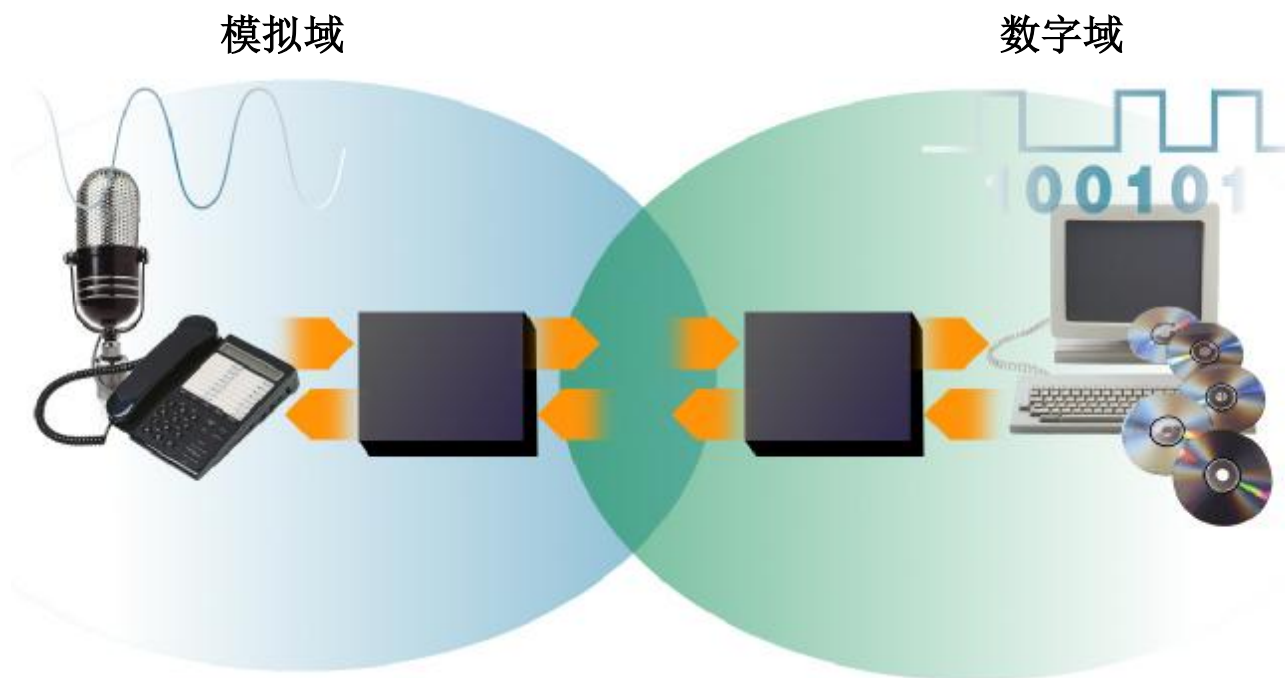


I 两类电路

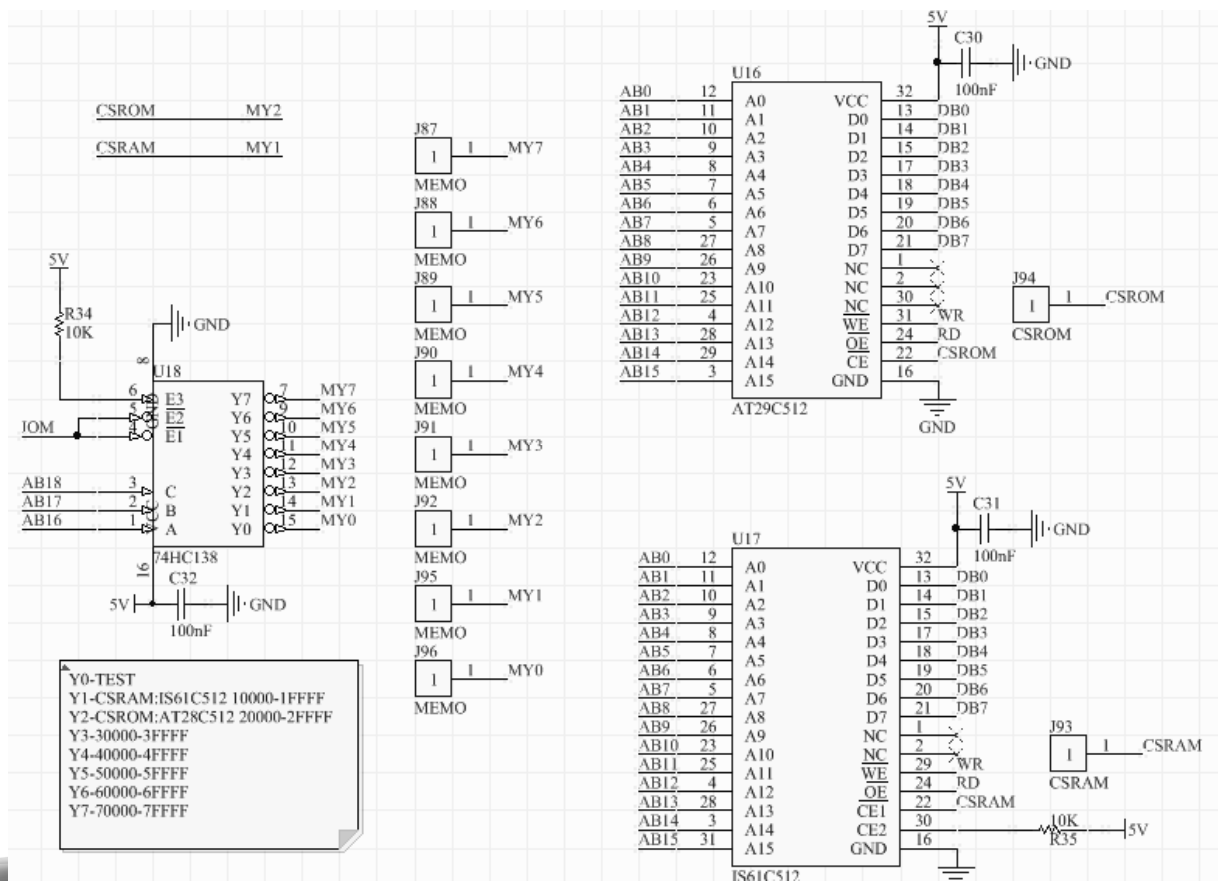
n模拟电路：处理模拟信号的电路

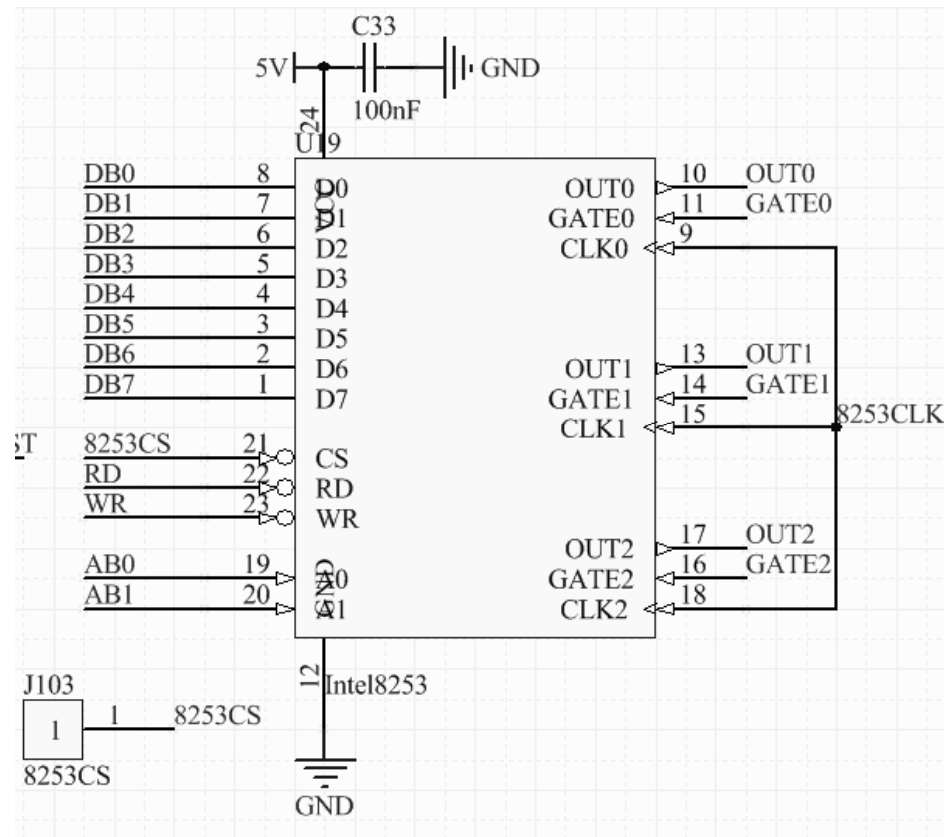
n数字电路：处理数字信号的电路

I 数字电路与模拟电路的联系

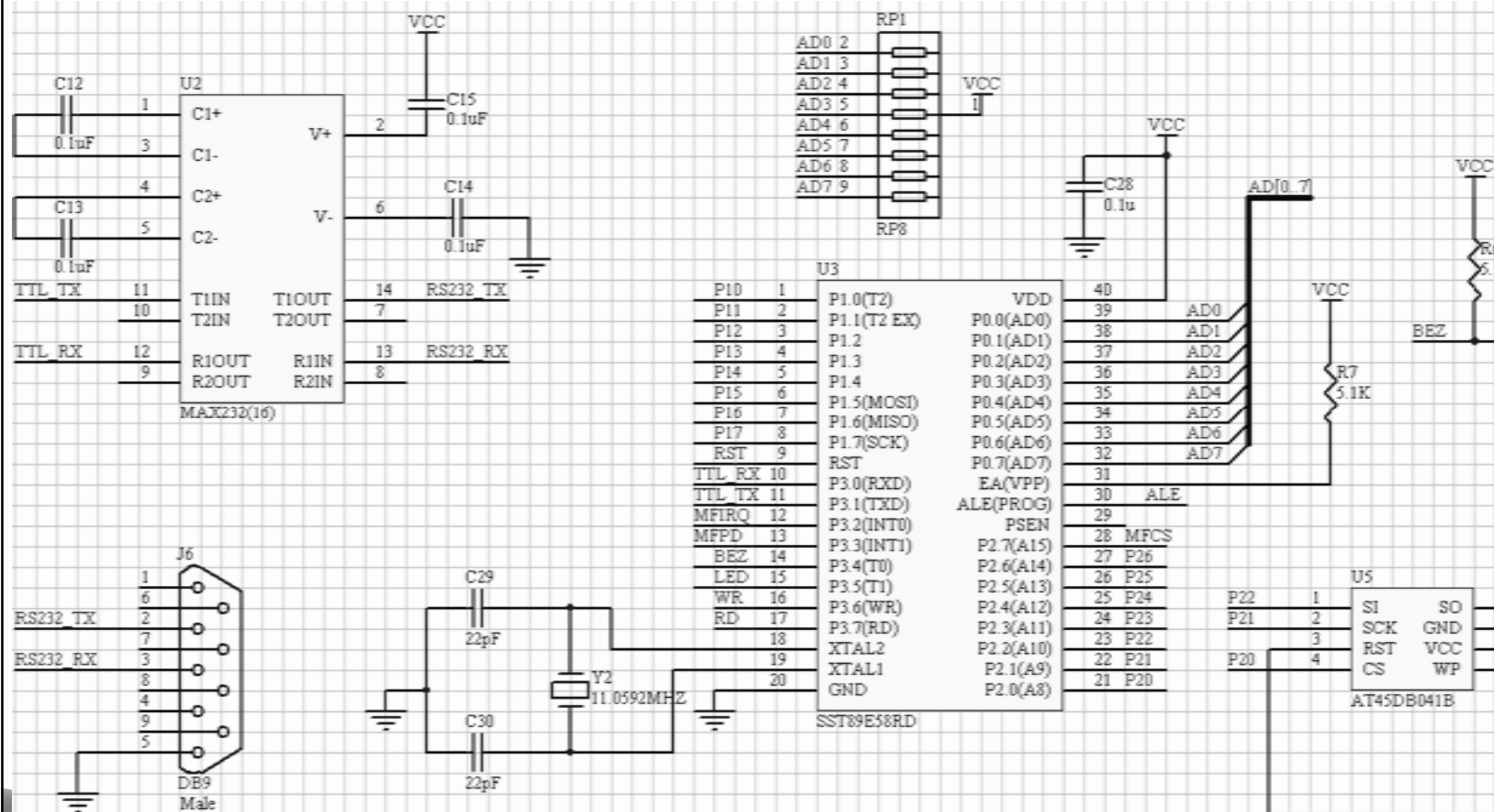


- | **0和1**: 两个数字, 高低两种电平, 两种逻辑, ...
- | 电路中半导体管工作在开关状态
 - n 二极管工作在导通态和截止态
 - n 三极管工作在饱和态和截止态
- | 基本逻辑运算
 - n 与、或、非。
 - n 任何复杂逻辑运算通过三种基本运算来实现。
- | 使用标准化, 积木式的元件/芯片构建电路系统。
 - n 硬件设计“软件化”
 - n **CPLD/FPGA**





数字电路的实例



I 数字电路的应用

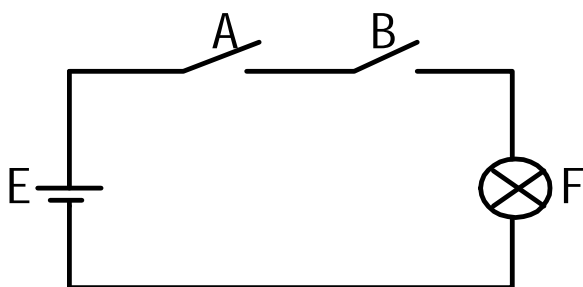
- n 通信、计算机、网络、家电、数码产品、雷达、自动控制（数控等）、仪器仪表、电子测量等。
- n 《数字电子技术》是一门极其重要的技术基础课。



逻辑代数（续）

I “与”运算

n只有决定一事件的全部条件都具备时，这件事才成立；如果有一个或一个以上条件不具备，则这件事就不成立。这样的因果关系称为“与”逻辑关系



与逻辑电路

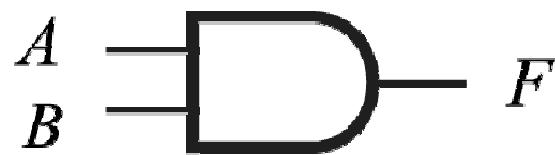
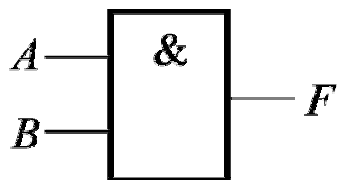
与逻辑真值表

A	B	$F=A \cdot B$
0	0	0
0	1	0
1	0	0
1	1	1

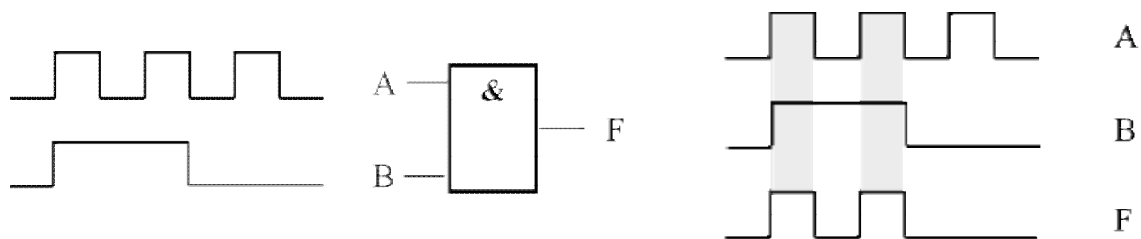
与的逻辑功能概括：

- 1) 有“0”出“0”；
- 2) 全“1”出“1”。

I “与”运算的符号



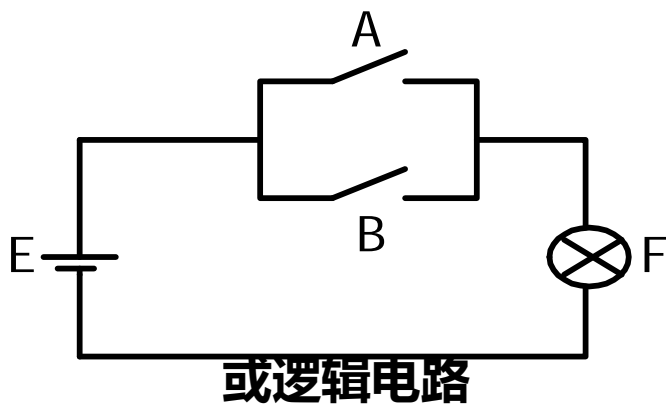
I 例：已知与门的输入波形，求其输出波形**F**。



逻辑代数（续）

I “或” 运算

n在决定一事件的各种条件中，只要有一个或一个以上条件具备时，这件事就成立；只有所有条件都不具备时，这件事就不成立。这样的因果关系称为“或”逻辑关系。



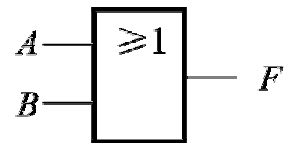
或逻辑真值表

A	B	$F=A+B$
0	0	0
0	1	1
1	0	1
1	1	1

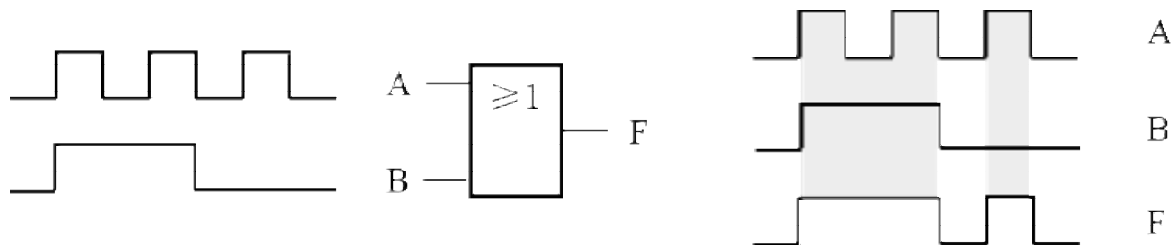
或门的逻辑功能概括：

- 1) 有“1”出“1”；
- 2) 全“0”出“0”。

I “或”运算的符号



I 例：已知或门的输入波形，求其输出波形**F**。



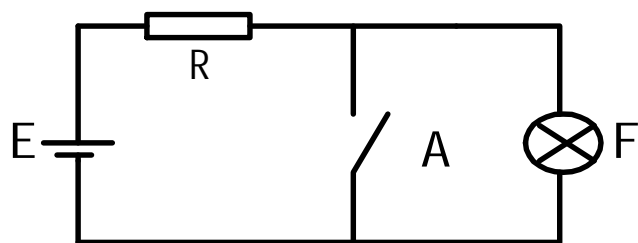
逻辑代数（续）

I “非”运算

n 假定事件F成立与否同条件A的具备与否有关；若A具备，则F不成立；若A不具备，则F成立。F和A之间这种因果关系称为“非”逻辑关系

n 逻辑表达式

$$F = \bar{A}$$

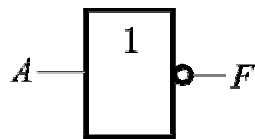


非逻辑电路

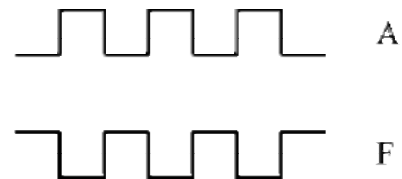
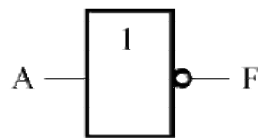
非逻辑真值表

A	$F = \bar{A}$
0	1
1	0

I “非”运算的符号



I 例：已知非门的输入波形，求其输出波形**F**。



复合逻辑门

I 基本逻辑运算的复合叫做复合逻辑运算。而实现复合逻辑运算的电路叫复合逻辑门。

n与非门

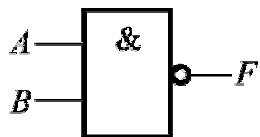
n或非门

n异或门

n同或门

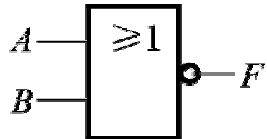
与非门

- 丨 “与”运算后再进行“非”运算的复合运算称为“与非”运算。
- 丨 实现“与非”运算的逻辑电路称为与非门。
- 丨 与非门的逻辑关系表达式为： $F = \overline{A \cdot B}$
- 丨 与非门的逻辑符号

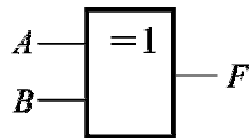


- 丨 “或”运算后再进行“非”运算的复合运算称为“或非”运算，实现“或非”运算的逻辑电路称为或非门。
- 丨 或非门的逻辑关系表达式为： $F = \overline{A + B}$

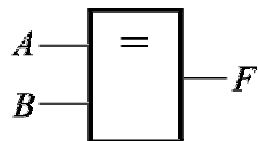
- 丨 或非门的逻辑符号：



I 异或门的逻辑符号 $F = A \oplus B = \overline{A}B + A\overline{B}$

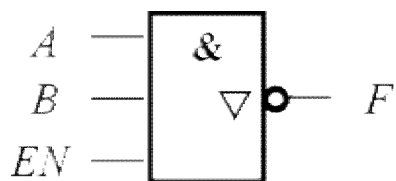


I 同或门的逻辑符号 : $F = A \quad B = \overline{A \oplus B} = \overline{A}B + AB$

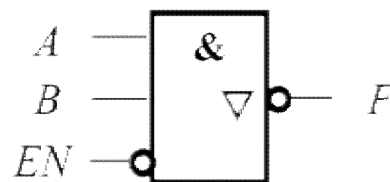


其它逻辑门：三态门

- 三态门（简称**TS**门，有倒三角符号）有三种逻辑状态，即**0**、**1**、**Z**。第三种状态为高阻状态(**Z**)，或禁止状态。
- 在普通门的基础上增加一个使能端（**EN**），使门原来的输出增加了一个**Z**态。
 - 当**EN**有效时，门按原逻辑工作，输出**0**或**1**；
 - 当**EN**无效时，门输出高阻态（**Z**）。
- 例：三态门的例子



EN = 1 : 使能
EN = 0 : 失能，输出Z态



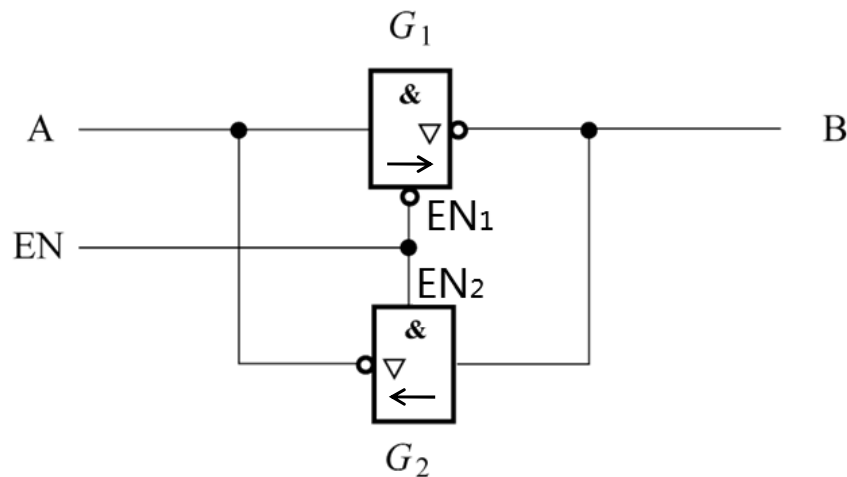
EN = 0 : 使能
EN = 1 : 失能，输出Z态

三态门的典型应用

- 丨 数据传输方向控制
- 丨 模拟开关
- 丨 总线存取控制

数据传输方向控制

I **A和B之间进行数据传输....., EN控制传输方向**



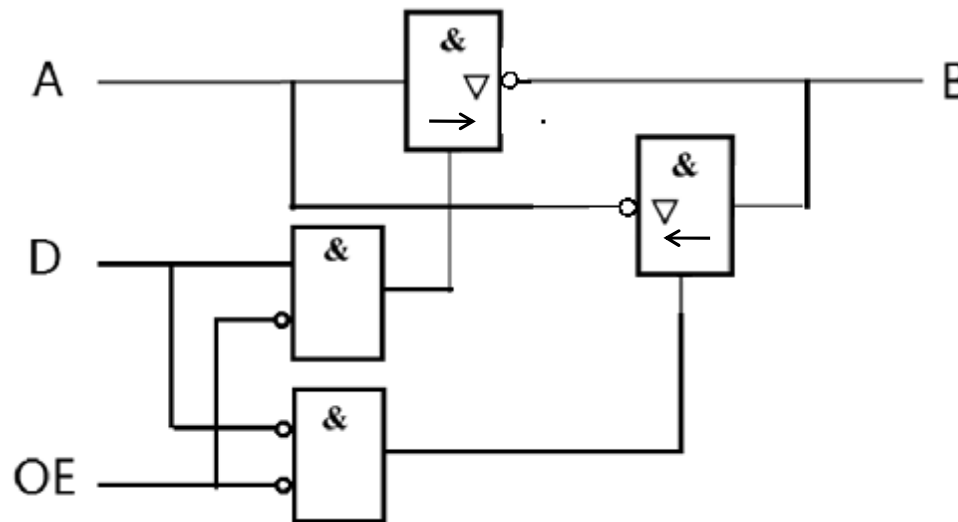
I **思考:**

$nEN=1$ 时, 数据能从 B 端传到 A 端;

$nEN=0$ 时, 数据能从 A 端传到 B 端;

数据传输方向控制

I **A和B之间进行数据传输...**, **D控制方向**, **OE是使能端**。



I **思考:**

n OE = 1 时, **A和B之间高阻**

n OE = 0 时, **A和B之间连通**

u D = 0 时, 数据能从 **B** 端传到 **A** 端;

u D = 1 时, 数据能从 **A** 端传到 **B** 端;

模拟开关

┆ 实现单刀双掷开关

┆ 思考

$nC=0$, $F = A$ 还是 $=B$?

$nC=1$, $F = A$ 还是 $=B$?

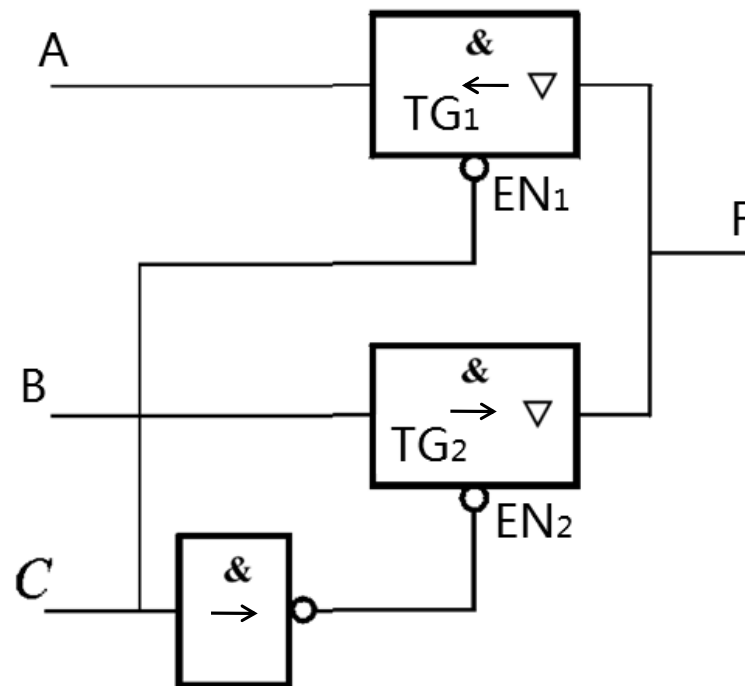
┆ 答案

$nC = 0$

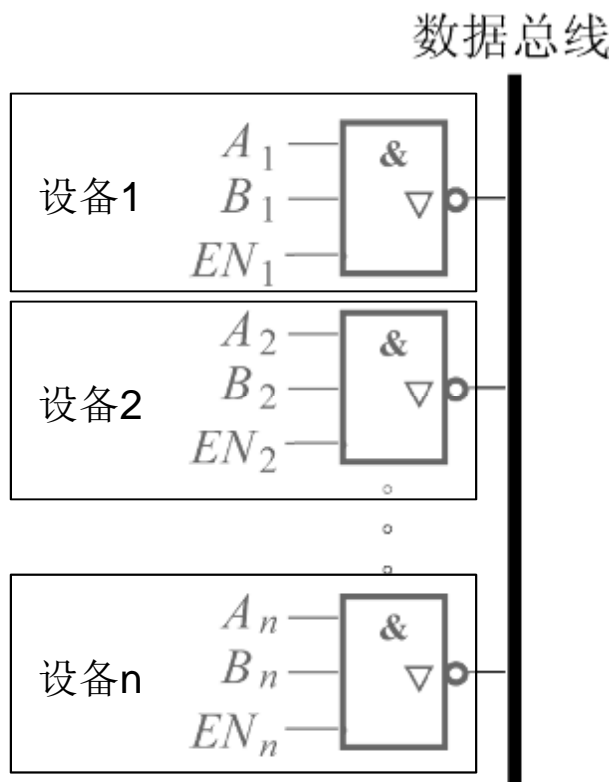
┆ TG_1 通, $F = A$;

$nC = 1$

┆ TG_2 通, $F = B$ 。



总线存取控制



多个设备物理上连在数据总线上。
必须控制存取权限，只能让最多1
个设备逻辑连接（占用）总线

$EN_i = 1$: 设备*i*逻辑连上总线

$EN_i = 0$: 设备*i*脱离总线（Z态）

OC门

I 特点

nOC门即集电极开路门。

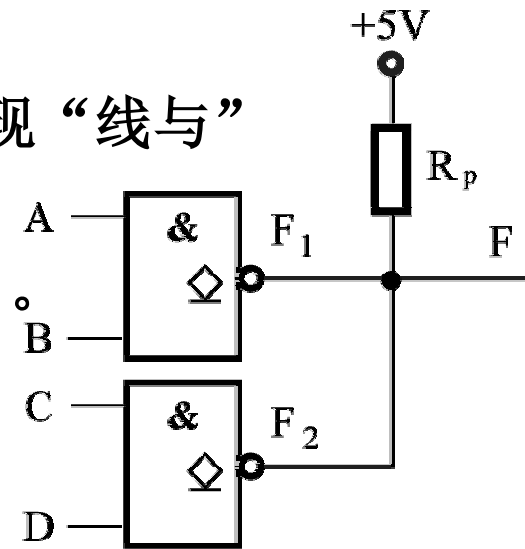
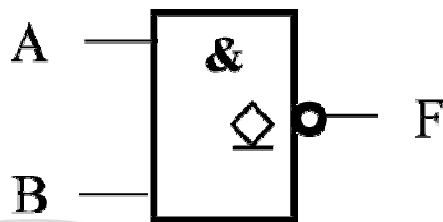
n门电路内部输出三极管的集电极C开路

n使用时必须外接“上拉电阻”

nOC门输出端可以直接相连，实现“线与”

I 例子：OC与非门

n注意：“线与”和“上拉电阻”



$$F = F_1 \cdot F_2 = \overline{AB} \cdot \overline{CD}$$

触发器——D触发器

I 触发器的功能和特点

- n 能储存一位二进制信息的单元电路。
- n 用于信号保持
- n 用做导通开关
- n 特点：0-1双稳态电路

I 触发器与门的联系

- n 联系：触发器是在门电路的基础上引入反馈构成的。
- n 区别：门是组合电路，触发器是时序电路。

I 触发器的种类

- n 基本RS触发器、同步RS触发器、主从型JK触发器、维持阻塞型D触发器、T和T'触发器等。

结构和原理

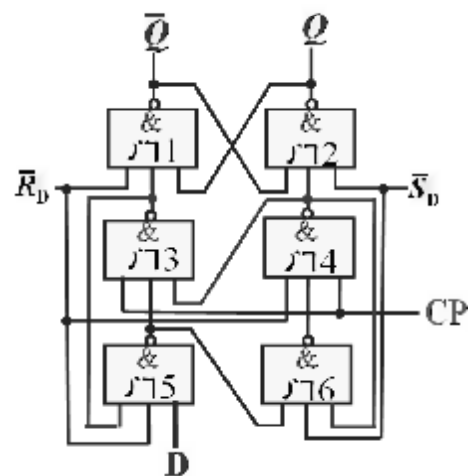
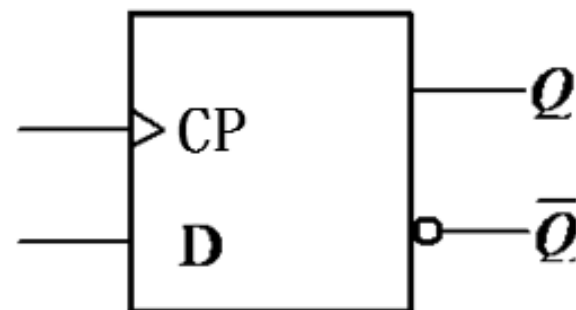
I 外部引脚

nD: 信号输入端

nCP: 时钟输入端

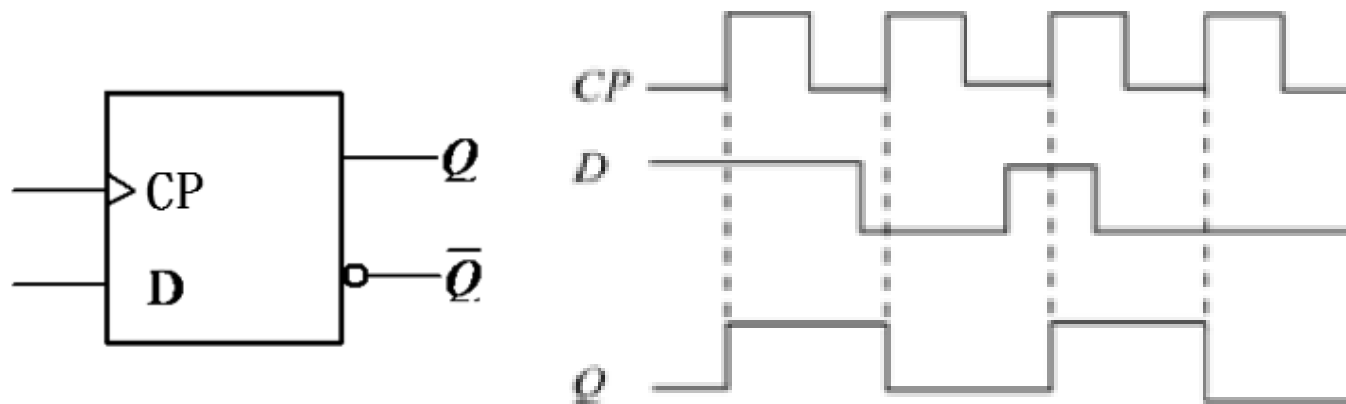
nQ, \bar{Q} : 输出端（反相）

I 内部结构

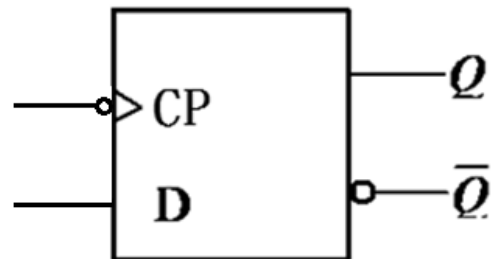


I 工作方式

- 触发器在CP脉冲的上升沿产生状态变化: $Q=D$ 。而在上升沿后, D端信号变化对触发器输出状态没有影响。触发器的次态取决于CP脉冲上升沿时的D信号,
- CP上沿锁存D (阻塞D), 上升后Q端维持不变。



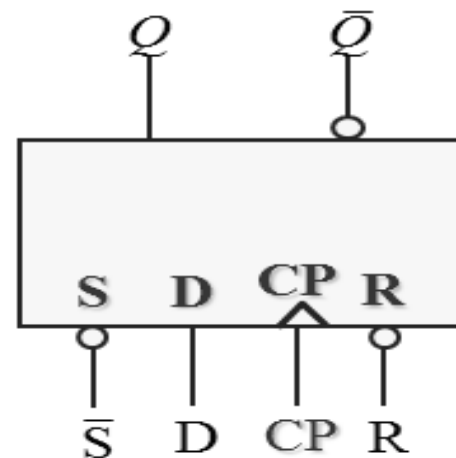
I CP下降沿触发的触发器



I 带清零和置1端的D触发器

I R: 置0端

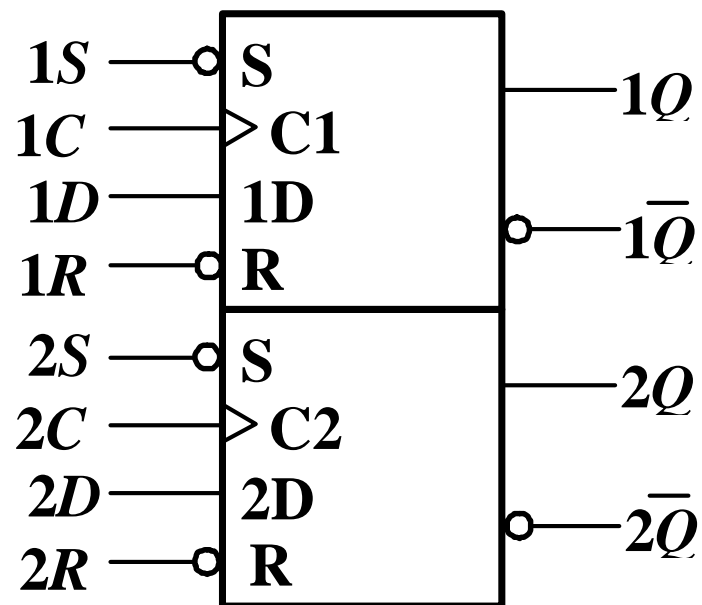
I S: 置1端



I D触发器集成电路

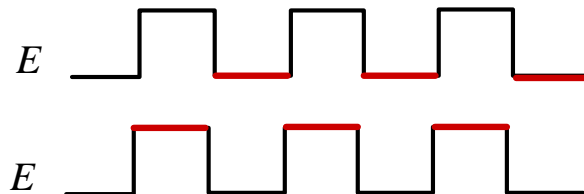
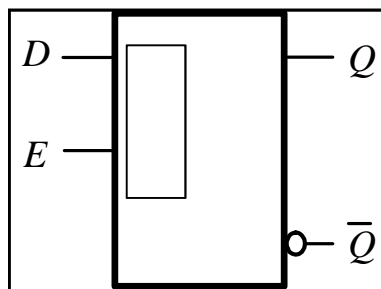
n74HC74

n2D触发器

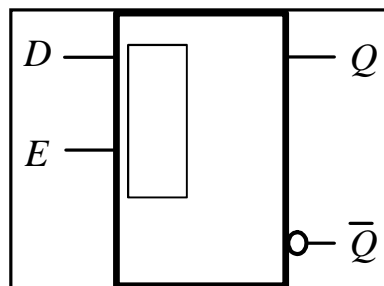


锁存器——D锁存器

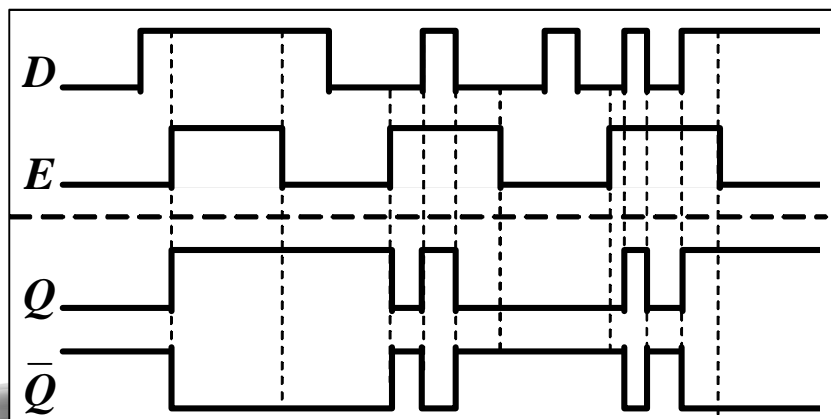
- I 和触发器有类似的功能
 - n 具有0和1两个稳态，能自行保持。
 - n 能存储一位二进制码。
- I 区别
 - n 锁存器对电平敏感，触发器对边沿敏感



D锁存器的功能表



E	D	Q	\bar{Q}	功能
0	×	不变	不变	保持
1	0	0	1	置0
1	1	1	0	置1



$E=1$ 时 $Q = D$

$E=0$ 时 Q 不变

IC的基本概念

I IC

n把若干个有源器件或无源器件及其连线，按照一定的功能要求，制作在一块半导体基片上，这样的产品叫集成电路，即**IC**

n最简单的数字集成电路是集成逻辑门。

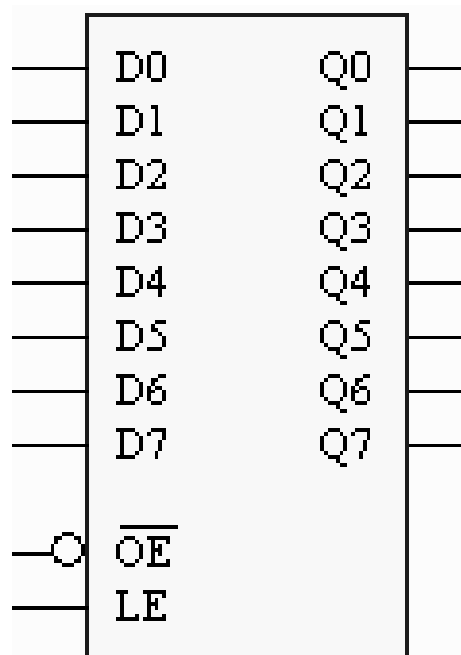
u常用集成门电路(TTL系列)

常用集成门电路(TTL系列)

序号	型号	名称	主要功能
1	74LS00	四2输入与非门	
2	74LS02	四2输入或非门	
3	74LS04	六反相器	
4	74LS05	六反相器	OC门
5	74LS08	四2输入与门	
6	74LS10	三3输入与非门	
7	74LS14	六反相器	施密特触发
8	74LS20	双4输入与非门	
9	74LS21	双4输入与门	
10	74LS30	8输入与非门	
11	74LS32	四2输入或门	
12	74LS64	4-2-3-2输入与或非门	
13	74LS86	四2输入异或门	
14	74LS125	四总线缓冲器	三态输出

锁存器74HC373 : 8D三态锁存器

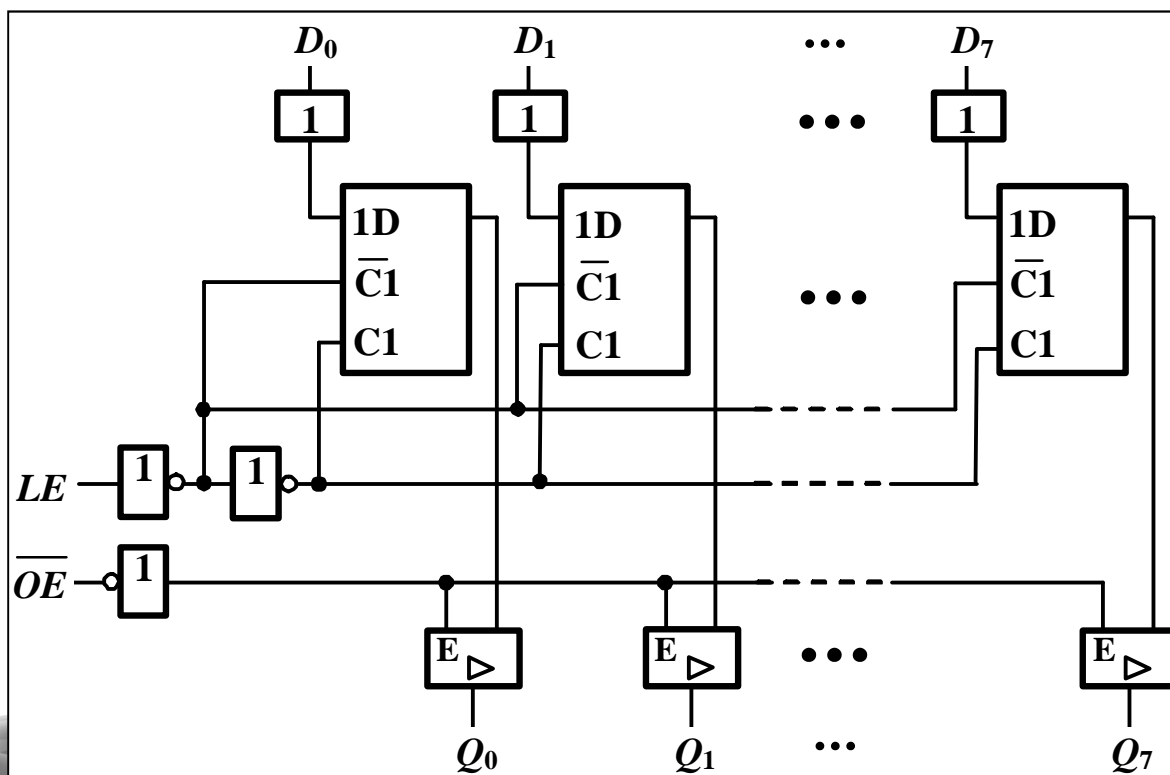
- I **LE (Lock Enable)** $LE=1$: $Q = D$; $LE=0$: Q 不变
- I **\overline{OE} (三态输出使能)**: **0**输出使能, **1**输出失能

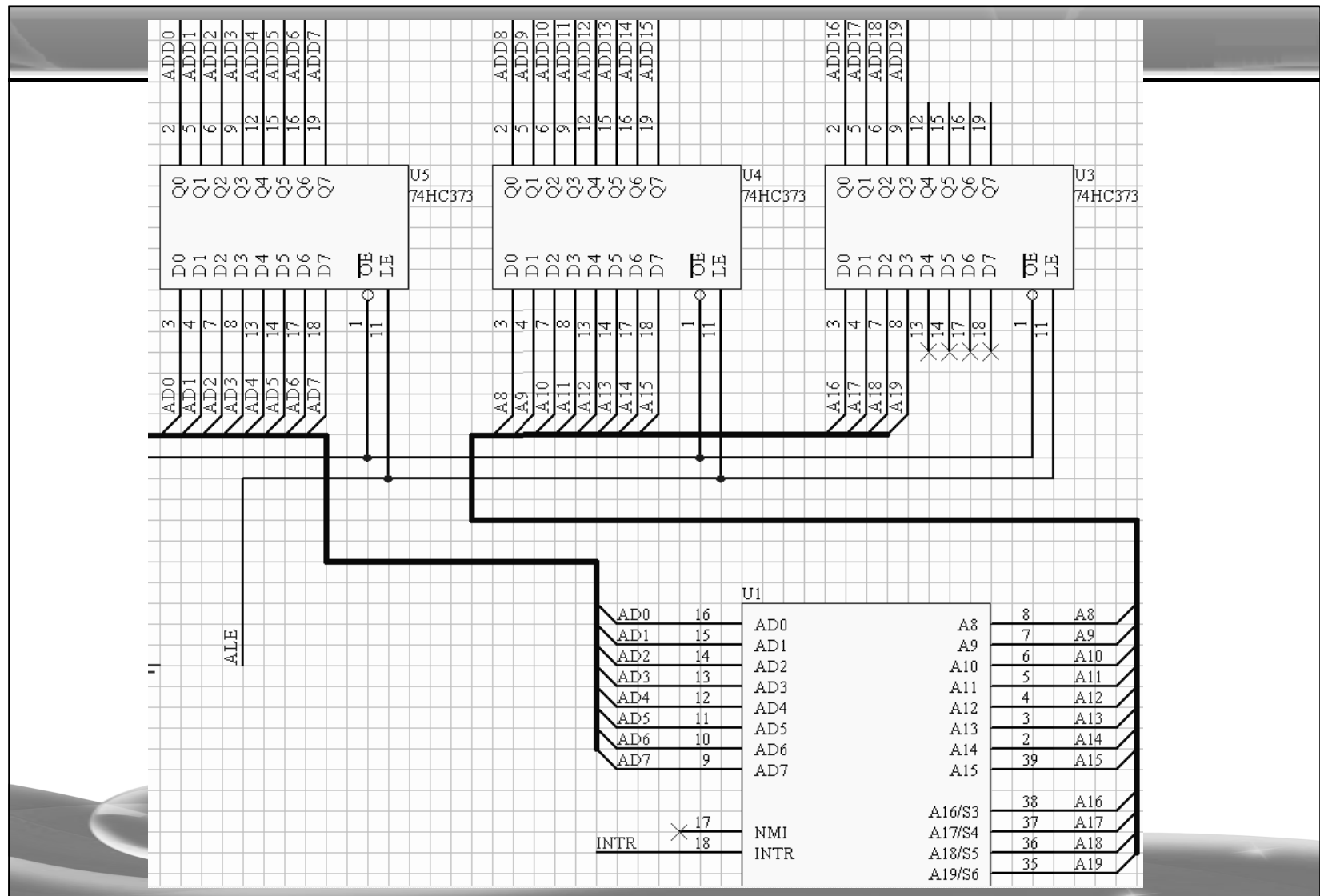


锁存器74HC373 : 8D三态锁存器

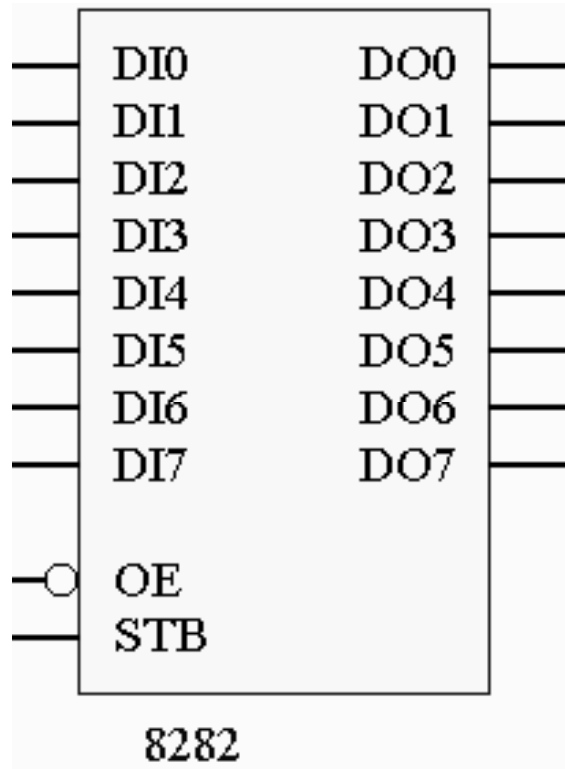
I **LE: Lock Enable**

I **OE: 三态输出使能**





Intel 8282 : 三态锁存器



DI: 输入;

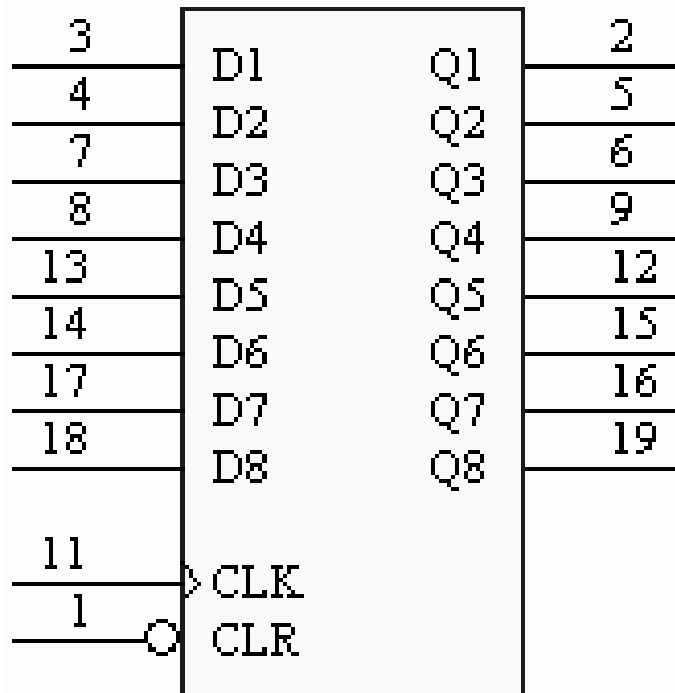
DO: 输出

STB: 选通控制(等同373的LE)

OE : 输出使能

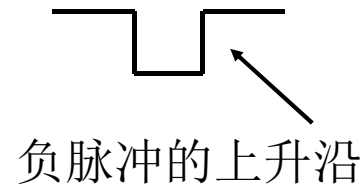
异步清零的锁存器：74LS273

D: 输入 ; **Q:** 输出 **CLK:** 控制 **CLR:** Clear

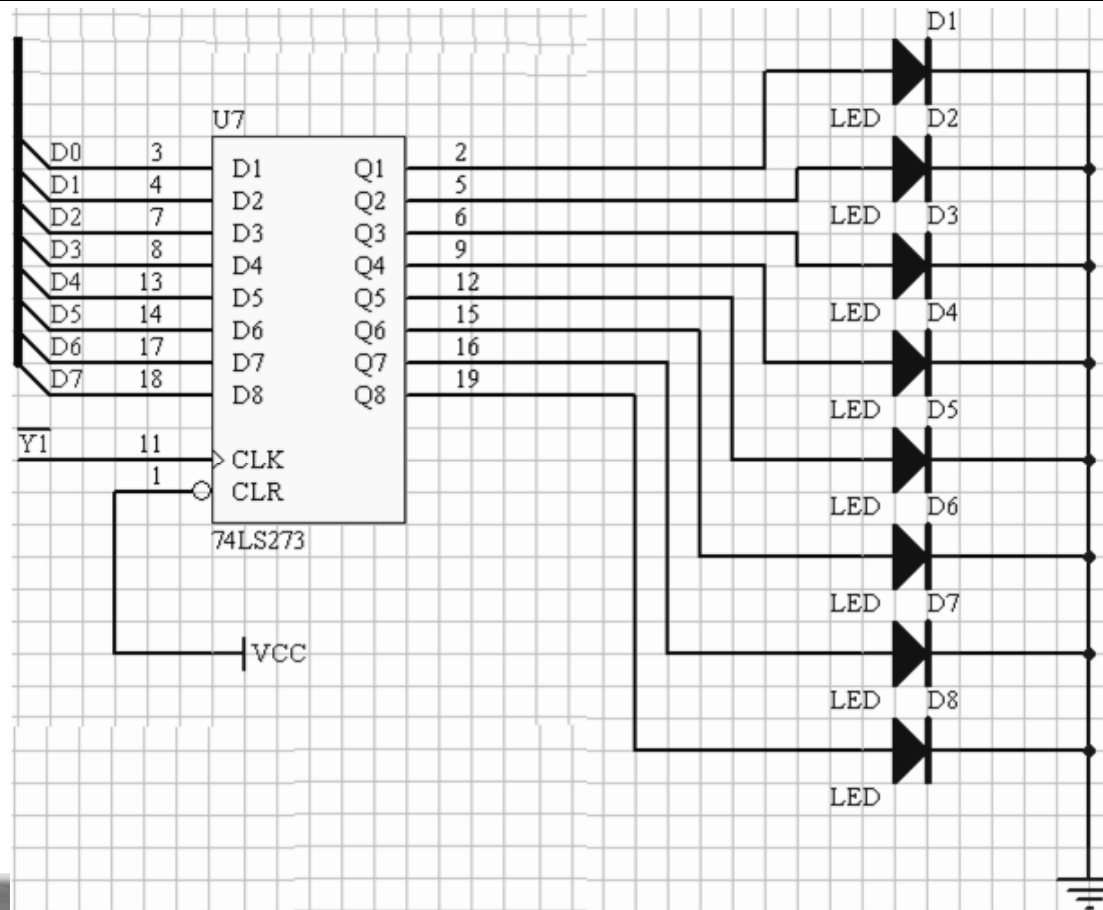


74LS273

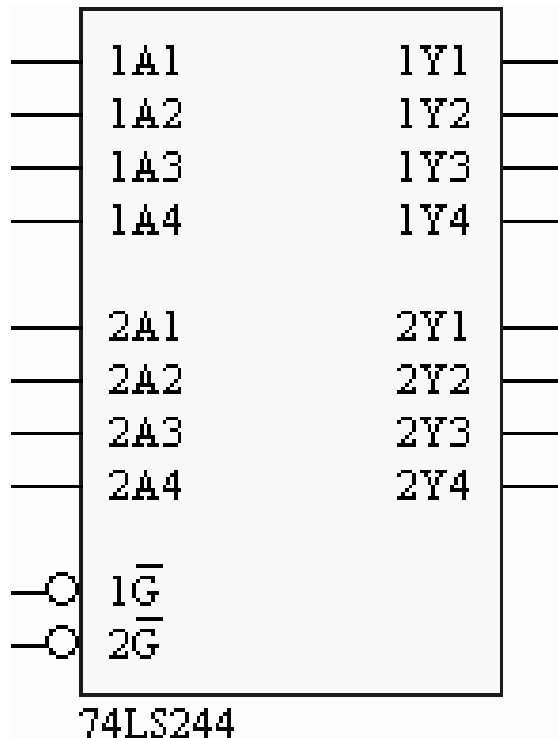
具有异步清零的
上升沿锁存器



74LS273



74LS244 —— 缓冲器(实质是三态开关)



$\overline{1G} = 0: 1Y = 1A$

$\overline{1G} = 1: 1Y = Z$

$\overline{2G} = 0: 2Y = 2A$

$\overline{2G} = 1: 2Y = Z$

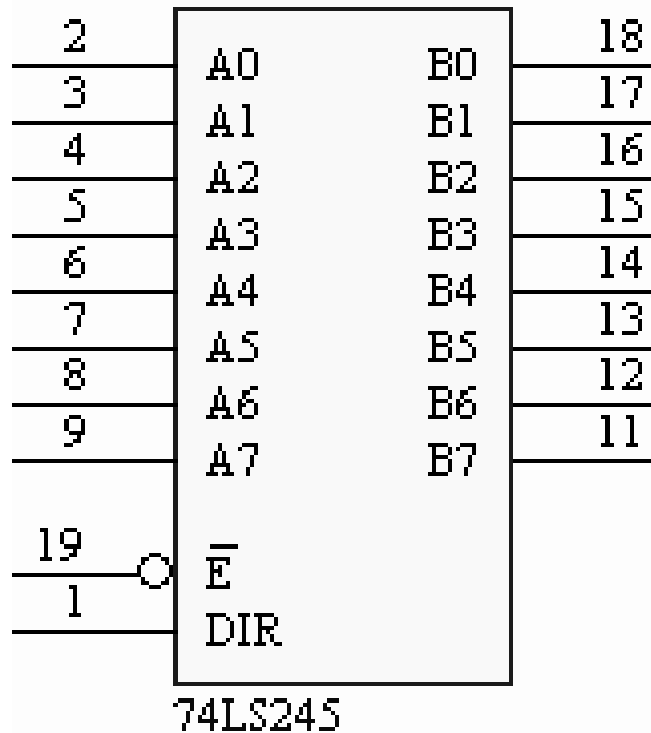
特点:

8位单向缓冲器

双4位分两组

输出与输入同相

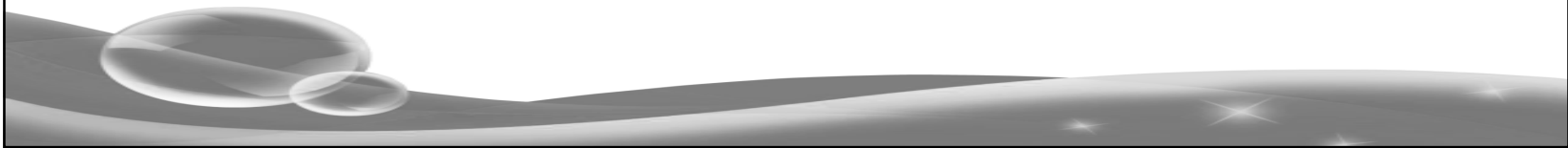
74LS245 —— 缓冲器(实质是三态开关)



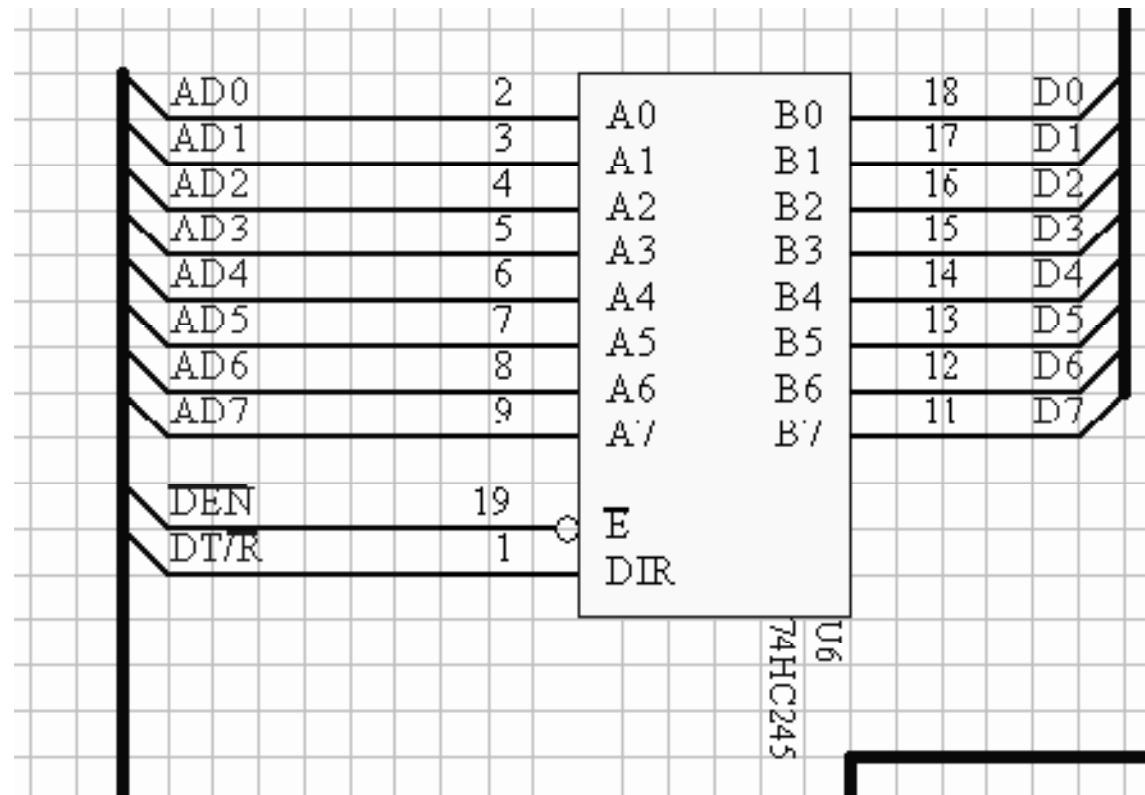
8位双向缓冲器
控制端E低电平有效
输出与输入同相

E: 低电平导通
DIR: 决定传输方向
DIR=1: A → B
DIR=0: B → A

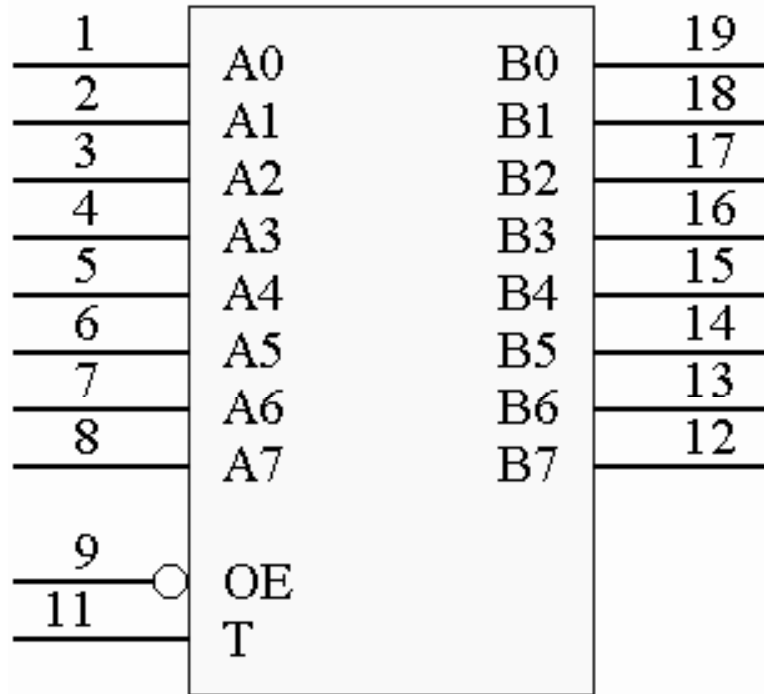
74LS244



74LS245



缓冲器—— Intel 8286 (实质是三态开关)



8286

8位双向缓冲器
控制端**OE**低电平有效
输出与输入同相

E: 低电平导通
T: 决定传输方向

DIR = 1: A → B

DIR = 0: B → A

数据收发器

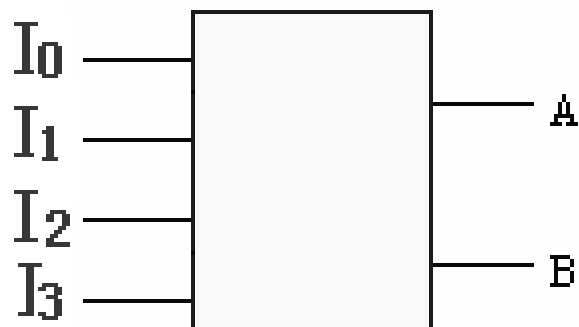
基本芯片--编码器

I 编码器

n 功能：将某个事件用若干（**n**）位二进制代码表示。

n 定义：执行编码功能的电路通称为编码器。

n 结构： 2^n 个输入脚（每个脚对应1个事件），**n**个输出引脚。

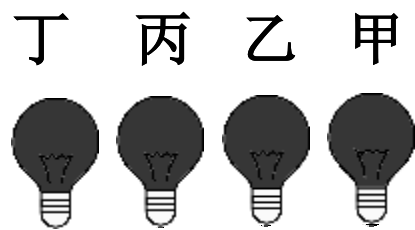


┌ 输入： I0, I1, I2 , I3 (确保每次仅一个脚输入有效)

┌ 输出： A,B=[00, 01, 10, 11]

基本芯片--编码器

例子：甲，乙，丙，丁 四人抢答



I 甲，乙，丙，丁 四人抢答 (用2位表示抢答结果)

丁 丙 乙 甲

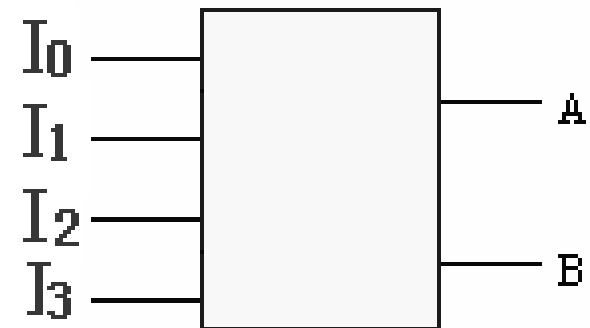


















0 0

0 1

1 0

1 1



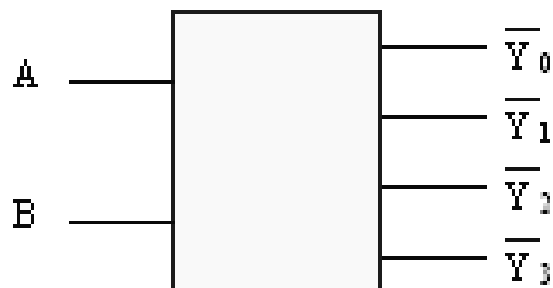
	丁	丙	乙	甲	输 入				输出	
					I_3	I_2	I_1	I_0	A	B
(1)					0	0	0	1	0	0
(2)					0	0	1	0	0	1
(3)					0	1	0	0	1	0
(4)					1	0	0	0	1	1

基本芯片--译码器

I 译码器

n 功能：将某个二进制数据的含义“翻译”出来，指示唯一的某1个事件有效。

n 结构： n 位输入脚， 2^n 个输出脚（每脚对应1个事件）

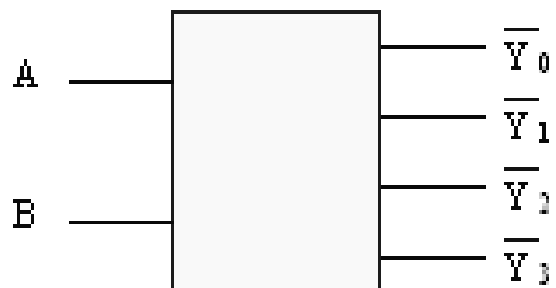


n 特点：每次译码，仅唯一的1个输出引脚为有效电平

u 输入： $AB = [00, 01, 10, 11]$

u 输出： $Y_0Y_1Y_2Y_3 = 0111, 1011, 1101, 1110$

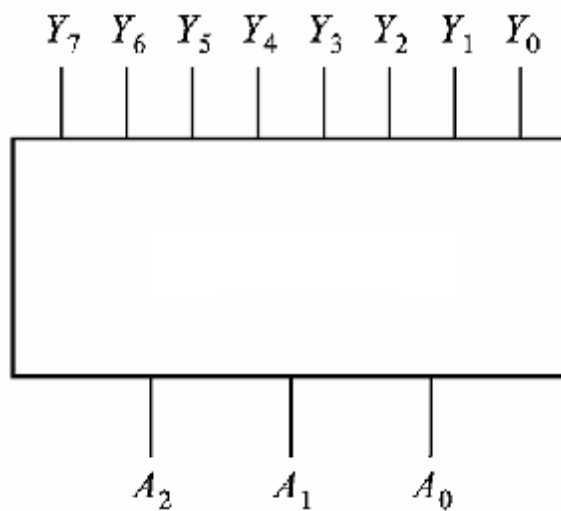
2-4译码器的真值表



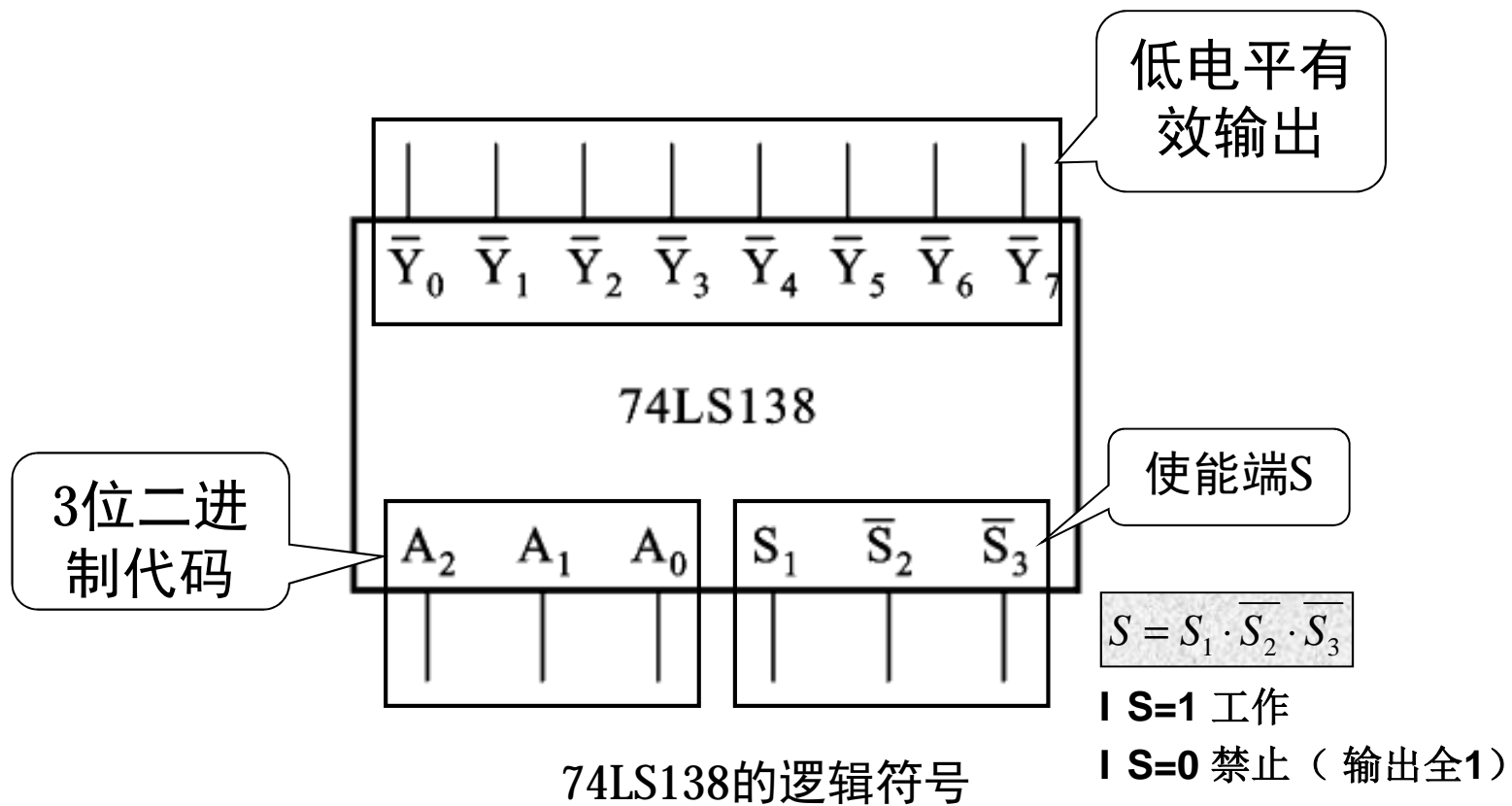
输 入		输 出			
A	B	Y_3	Y_2	Y_1	Y_0
0	0	1	1	1	0
0	1	1	1	0	1
1	0	1	0	1	1
1	1	0	1	1	1

3-8译码器

- 丨 结构：3个输入引脚，8个（ $=2^3$ ）输出引脚
- 丨 功能：输入3位二进制代码 $A_2A_1A_0$ （000~111），在唯一的某1个输出引脚上出现有效电平。



74LS138 —— 3-8译码器



74LS138的功能表

输 入			输 出							
S_1	$\bar{S}_2 + \bar{S}_3$	$A_2 A_1 A_0$	\bar{Y}_0	\bar{Y}_1	\bar{Y}_2	\bar{Y}_3	\bar{Y}_4	\bar{Y}_5	\bar{Y}_6	\bar{Y}_7
\times	1	$\times \times \times$	1	1	1	1	1	1	1	1
0	\times	$\times \times \times$	1	1	1	1	1	1	1	1
1	0	0 0 0	0	1	1	1	1	1	1	1
1	0	0 0 1	1	0	1	1	1	1	1	1
1	0	0 1 0	1	1	0	1	1	1	1	1
1	0	0 1 1	1	1	1	0	1	1	1	1
1	0	1 0 0	1	1	1	1	0	1	1	1
1	0	1 0 1	1	1	1	1	1	0	1	1
1	0	1 1 0	1	1	1	1	1	1	0	1
1	0	1 1 1	1	1	1	1	1	1	1	0

禁止译码

译码工作

译中为0

高电平有效

低电平有效

74LS138

I 译码工作时

I **A2 A1 A0**和输出

n000 → **Y0**低电平

n001 → **Y1**低电平

n010 → **Y2**低电平

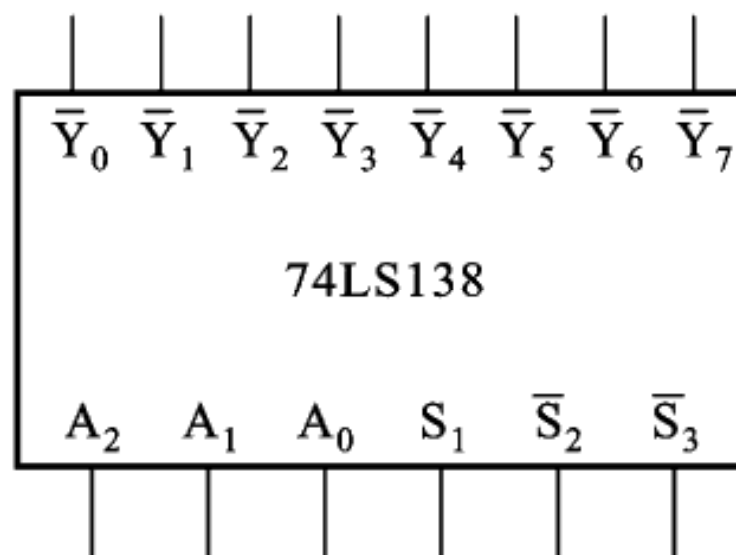
n011 → **Y3**低电平

n100 → **Y4**低电平

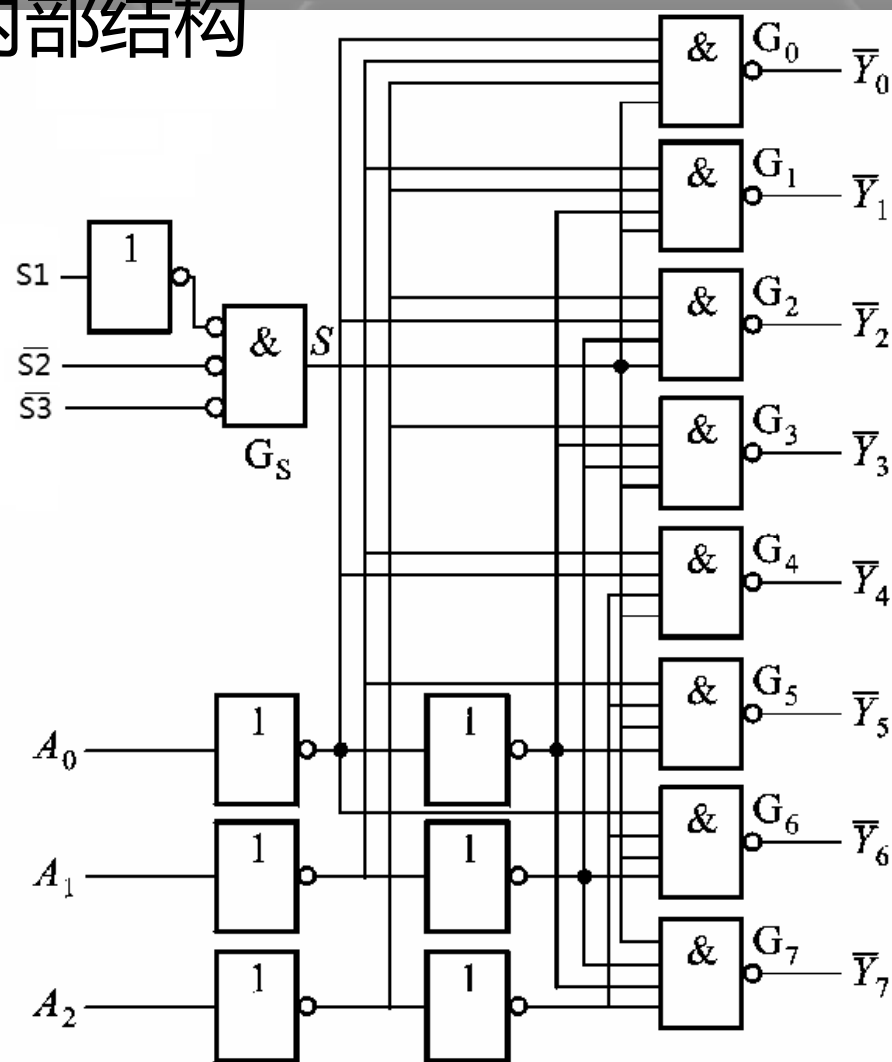
n101 → **Y5**低电平

n110 → **Y6**低电平

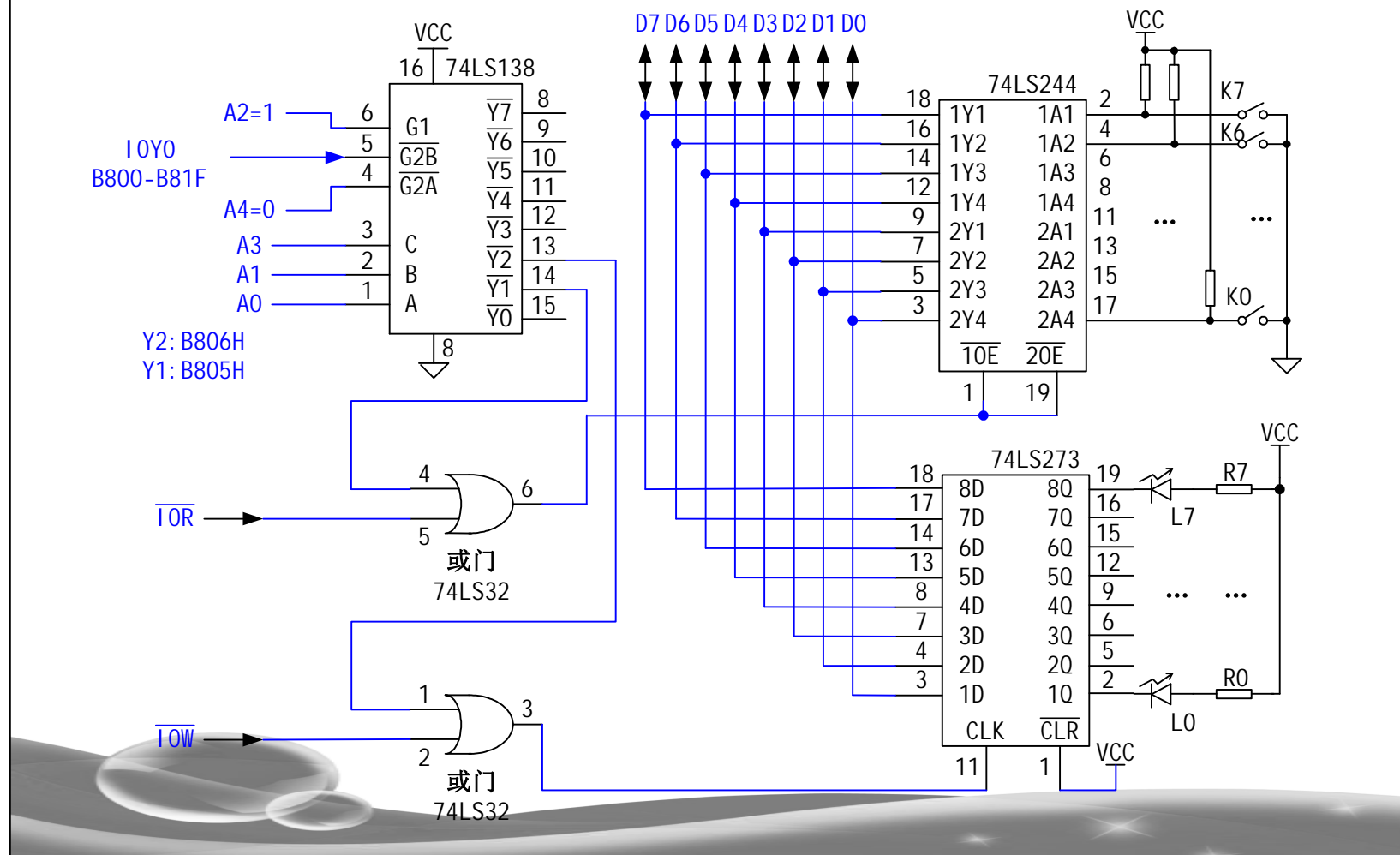
n111 → **Y7**低电平

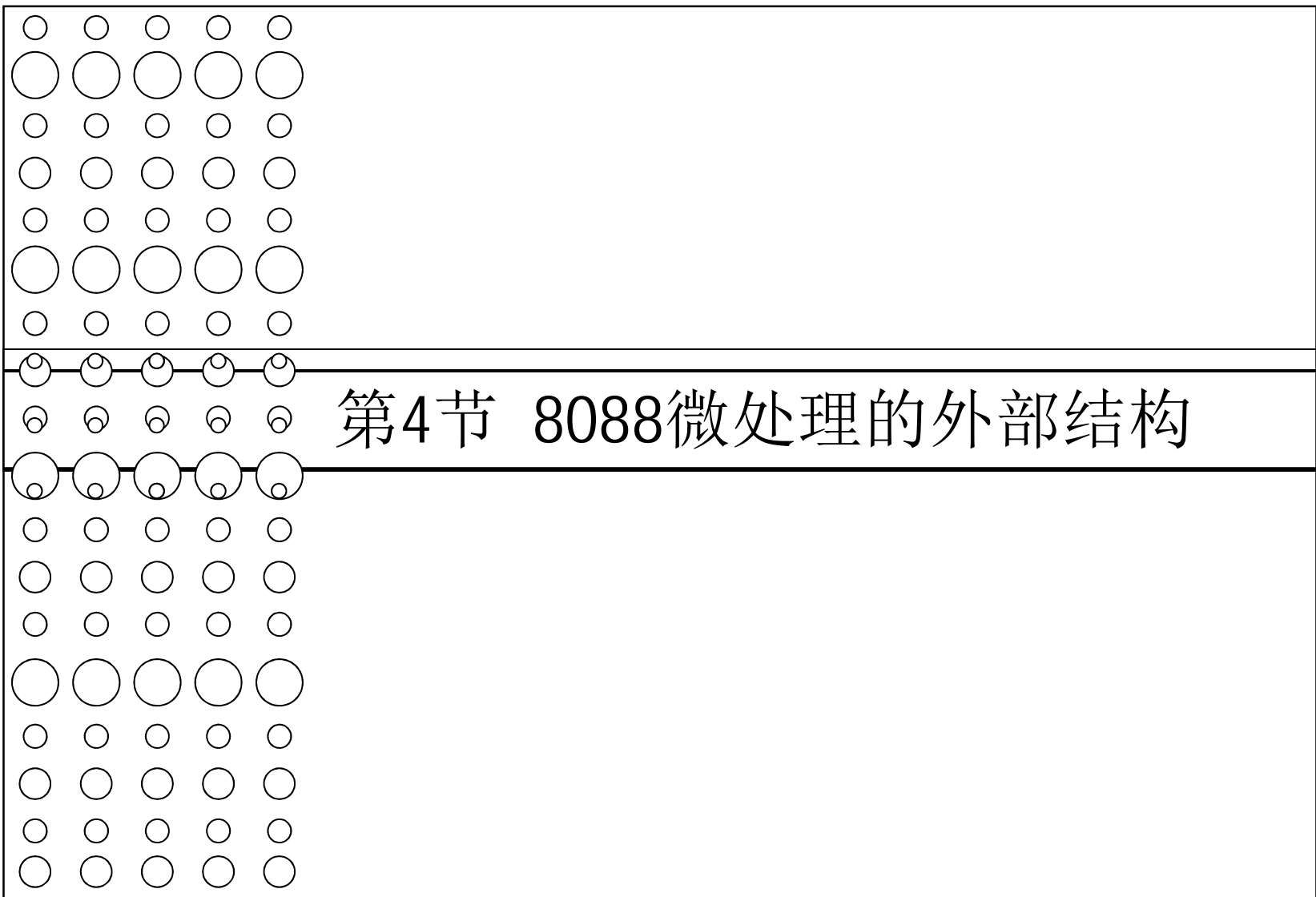


74LS138内部结构



门和IC在CPU典型接口电路中的应用





8088的两种工作模式

I 最小模式

- n 构成小规模的应用系统——单处理器系统

- n 8088本身提供所有的系统总线信号

I 最大模式

- n 构成较大规模的应用系统——多处理器系统，例如可以接入数值协处理器8087

- n 控制信号较多，8088和总线控制器8288共同形成系统总线信号

8088的两种工作模式

- I 两种模式利用**MN/MX***引脚区别
 - n**MN/MX***接高电平为最小模式
 - n**MN/MX***接低电平为最大模式
 - n硬件决定工作方式
- I 两种模式内部操作没有区别
 - n本书以最小模式展开基本原理
 - nIBM PC/XT采用最大模式

8088的外部引脚——电气特性

I 电气特性

n 电源VCC

u 5V \pm 10% 的条件下能够工作;

n 输入特性:

低电平 0.8 V (0)

高电平 2.0 V (1)

n 输出特性:

低电平 0.4 V (0)

高电平 2.4 V (1)

8088的外部引脚

I 外部特性

n 引脚的功能

n 是否复用

n 信号的流向

n 有效电平

n 三态能力

地	1	40	Vcc(+5V)
A14	2	39	A15
A13	3	38	A16 / S3
A12	4	37	A17 / S4
A11	5	36	A18 / S5
A10	6	35	A19 / S6
A9	7	34	$\overline{SS0}$ (HIGH)
A8	8	33	$\overline{MN} / \overline{MX}$
AD7	9	32	RD
AD6	10	31	HOLD ($\overline{RQ}/GT0$)
AD5	11	30	HLDA ($\overline{RQ}/GT1$)
AD4	12	29	\overline{WR} (LOCK)
AD3	13	28	$\overline{M} / \overline{IO} (\overline{S2})$
AD2	14	27	DT / R (S1)
AD1	15	26	$\overline{DEN} (\overline{S0})$
AD0	16	25	ALE (QS0)
NMI	17	24	\overline{INTA} (QS1)
INTR	18	23	TEST
CLK	19	22	READY
地	20	21	RESET

8088的外部引脚——电源，时钟，复位

I 电源管脚

n VCC(40)、GND(1/20)

n 5V±10%

I 时钟管脚

n CLK(19)

n 外接4.77MHz外部时钟

I 复位管脚

n RESET(21)

n 正脉冲复位(4T)

u 标志清0: Flag=0000H;

u DS,SS,ES复位为0H;

u CS8088FH, IP复位为0H;

p FFFF0H放第一条指令。

地	1	40	Vcc(+5V)
A14	2	39	A15
A13	3	38	A16 / S3
A12	4	37	A17 / S4
A11	5	36	A18 / S5
A10	6	35	A19 / S6
A9	7	34	$\overline{SS0}$ (HIGH)
A8	8	33	$\overline{MN} / \overline{MX}$
AD7	9	32	RD
AD6	10	31	HOLD ($\overline{RQ}/GT0$)
AD5	11	30	HLDA ($\overline{RQ}/GT1$)
AD4	12	29	\overline{WR} (LOCK)
AD3	13	28	$\overline{M} / \overline{IO} (\overline{S2})$
AD2	14	27	DT / R (S1)
AD1	15	26	$\overline{DEN} (\overline{S0})$
AD0	16	25	ALE (QS0)
NMI	17	24	\overline{INTA} (QS1)
INTR	18	23	TEST
CLK	19	22	READY
地	20	21	RESET

8088的外部引脚

I 主要功能引脚

- n 读、写信号 \overline{RD} 、 \overline{WR}
- n 端口/存储器选择信号 \overline{M}/IO
- n 数据传送/接收信号 DT/\overline{R}
- n 数据线 $AD7 \sim AD0$
- n 数据允许 \overline{DEN}
- n 地址线 $A19 \sim A0$
- n 地址锁存 ALE
- n $TEST$
- n $READY$
- n $INTR$
- n \overline{INTA}
- n NMI

地	1	40	Vcc(+5V)
A14	2	39	A15
A13	3	38	A16 / S3
A12	4	37	A17 / S4
A11	5	36	A18 / S5
A10	6	35	A19 / S6
A9	7	34	$\overline{SS0}$ (HIGH)
A8	8	33	$\overline{MN} / \overline{MX}$
AD7	9	32	\overline{RD}
AD6	10	31	\overline{HOLD} ($\overline{RQ}/GT0$)
AD5	11	30	\overline{HLDA} ($\overline{RQ}/GT1$)
AD4	12	29	\overline{WR} (LOCK)
AD3	13	28	$\overline{M} / \overline{IO}$ ($\overline{S2}$)
AD2	14	27	DT / R ($S1$)
AD1	15	26	\overline{DEN} ($S0$)
AD0	16	25	ALE ($QS0$)
NMI	17	24	\overline{INTA} ($QS1$)
INTR	18	23	$TEST$
CLK	19	22	$READY$
地	20	21	$RESET$

8088的外部引脚

I \overline{RD} (32) ——读(输出, 三态)

n 读I/O端口或存储器时, \overline{RD} 输出低电平; 否则为高阻态。

IN AL, 25H ; 读I/O
MOV AX, [BX] ; 读MEM

I \overline{WR} (29) ——写(输出, 三态)

n 写I/O端口或存储器时, \overline{WR} 输出低电平; 否则为高阻态。

OUT 25H, AX ; 写I/O
MOV [BX], AX ; 写MEM

地	1	40	Vcc(+5V)
A14	2	39	A15
A13	3	38	A16 / S3
A12	4	37	A17 / S4
A11	5	36	A18 / S5
A10	6	35	A19 / S6
A9	7	34	$\overline{SS0}$ (HIGH)
A8	8	33	$\overline{MN} / \overline{MX}$
AD7	9	32	\overline{RD}
AD6	10	31	HOLD ($\overline{RQ}/GT0$)
AD5	11	30	HLDA ($\overline{RQ}/GT1$)
AD4	12	29	\overline{WR} (LOCK)
AD3	13	28	$\overline{M} / \overline{IO}$ ($\overline{S2}$)
AD2	14	27	DT / R ($\overline{S1}$)
AD1	15	26	\overline{DEN} ($\overline{S0}$)
AD0	16	25	ALE (QS0)
NMI	17	24	\overline{INTA} (QS1)
INTR	18	23	TEST
CLK	19	22	READY
地	20	21	RESET

8088的外部引脚

I $\text{IO}/\overline{\text{M}}(28)$ ——端口/存储器选择(输出, 三态)

n 对端口读或写, $\text{IO}/\overline{\text{M}}$ 输出高电平

IN AL, 25H ; 读I/O

OUT 52H, AL ; 写I/O

n 对存储器读或写, $\text{IO}/\overline{\text{M}}$ 输出低电平

MOV AX, [BX] ; 读MEM

MOV [BX], AX ; 写MEM

n 其它情况处于高阻状态

地	1	40	Vcc(+5V)
A14	2	39	A15
A13	3	38	A16 / S3
A12	4	37	A17 / S4
A11	5	36	A18 / S5
A10	6	35	A19 / S6
A9	7	34	$\overline{\text{SS0}}$ (HIGH)
A8	8	33	$\overline{\text{MN}} / \overline{\text{MX}}$
AD7	9	32	$\overline{\text{RD}}$
AD6	10	31	$\overline{\text{HOLD}}$ ($\overline{\text{RQ}}/\text{GT0}$)
AD5	11	30	$\overline{\text{HLDA}}$ ($\overline{\text{RQ}}/\text{GT1}$)
AD4	12	29	$\overline{\text{WR}}$ (LOCK)
AD3	13	28	$\overline{\text{M}} / \overline{\text{IO}} (\overline{\text{S2}})$
AD2	14	27	$\overline{\text{DT}} / \overline{\text{R}} (\text{S1})$
AD1	15	26	$\overline{\text{DEN}} (\text{S0})$
AD0	16	25	ALE (QS0)
NMI	17	24	$\overline{\text{INTA}}$ (QS1)
INTR	18	23	TEST
CLK	19	22	READY
地	20	21	RESET

读写信号和IO/ \overline{M} 信号的组合

- I 由IO/ \overline{M} 、 \overline{WR} 和 \overline{RD} 三个信号生成存储器读、存储器读写、I/O读、I/O写等4种信号（低电平有效）

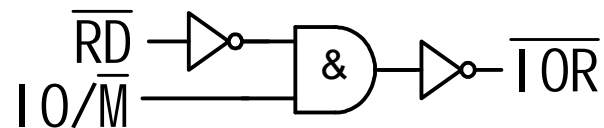
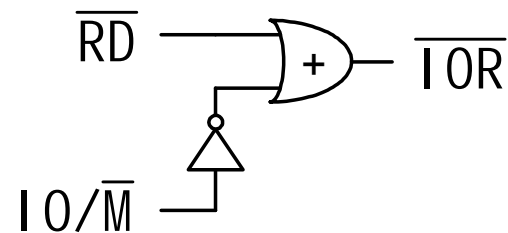
信号名称	IO/ \overline{M}	\overline{WR}	\overline{RD}
存储器读(MEMR)	低	高	低
存储器写(MEMW)	低	低	高
I/O读(IOR)	高	高	低
I/O写(IOW)	高	低	高

读写信号和IO/M信号的组合——实现IO读信号

I 通过RD、WR、IO/M得到IO读信号(IOR)

IO/M	RD	IOR
1	0	0
0	X	1
X	1	1

$$\overline{IOR} = \overline{IO/M + RD} = \overline{IO/M} \cdot \overline{RD}$$

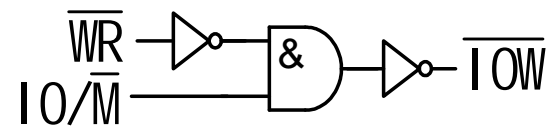
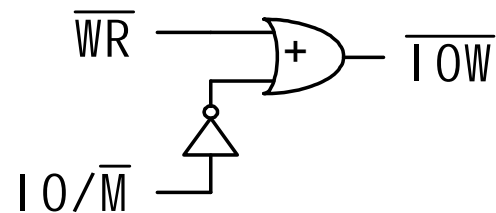


读写信号和IO/M信号的组合——实现IO写信号

I 通过RD、WR、IO/M得到IO写信号(IOW)

IO/M	WR	IOW
1	0	0
0	X	1
X	1	1

$$\overline{\text{IOW}} = \overline{\text{IO/M} + \text{WR}} = \overline{\text{IO/M}} \cdot \overline{\text{WR}}$$



课堂练习——实现存储器读/写信号

I 通过RD、WR、IO/M得到存储器读/写信号(MEMR/MEMW)

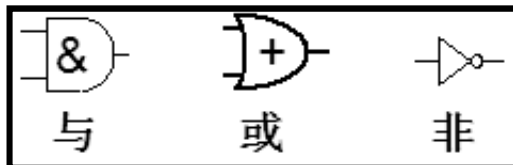
IO/M	RD	MEMR
0	0	0
1	X	1
X	1	1

IO/M	WR	MEMW
0	0	0
1	X	1
X	1	1

$$\overline{\text{MEMR}} = \text{IO}/\overline{\text{M}} + \overline{\text{RD}} = \overline{\overline{\text{IO}/\overline{\text{M}}} \cdot \overline{\overline{\text{RD}}}}$$

$$\overline{\text{MEMW}} = \text{IO}/\overline{\text{M}} + \overline{\text{WR}} = \overline{\overline{\text{IO}/\overline{\text{M}}} \cdot \overline{\overline{\text{WR}}}}$$

I 画出它们各自或或与两种形式的电路图

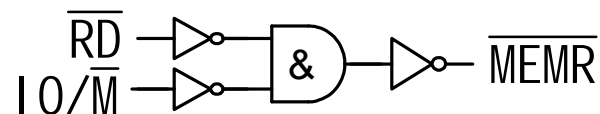
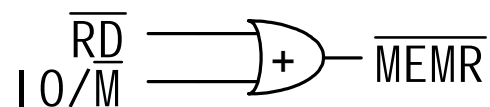


读写信号和IO/M信号的组合——实现存储器读信号

I 通过RD、WR、IO/M得到存储器读信号

IO/M	RD	MEMR
0	0	0
1	X	1
X	1	1

$$\overline{\text{MEMR}} = \text{IO}/\overline{\text{M}} + \overline{\text{RD}} = \overline{\overline{\text{IO}/\overline{\text{M}} \cdot \overline{\text{RD}}}}$$

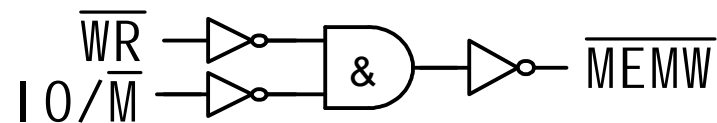
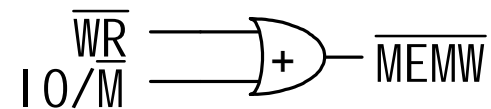


读写信号和IO/M信号的组合——实现存储器写信号

I 通过RD、WR、IO/M得到信号存储器写信号

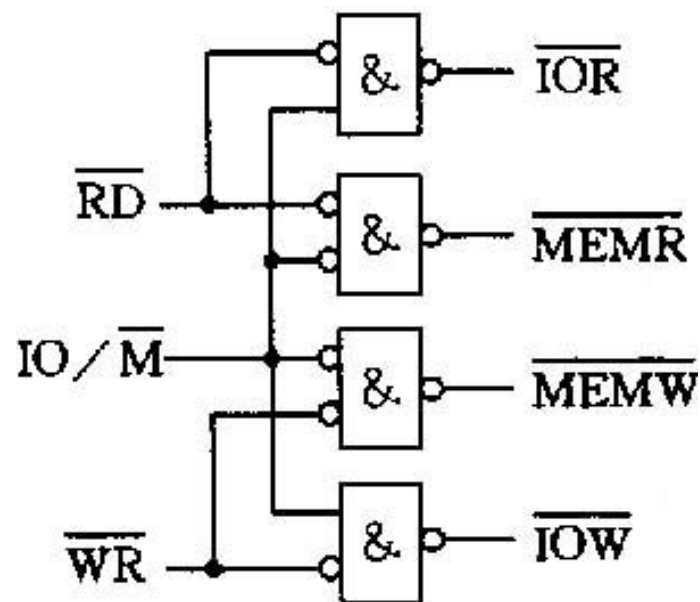
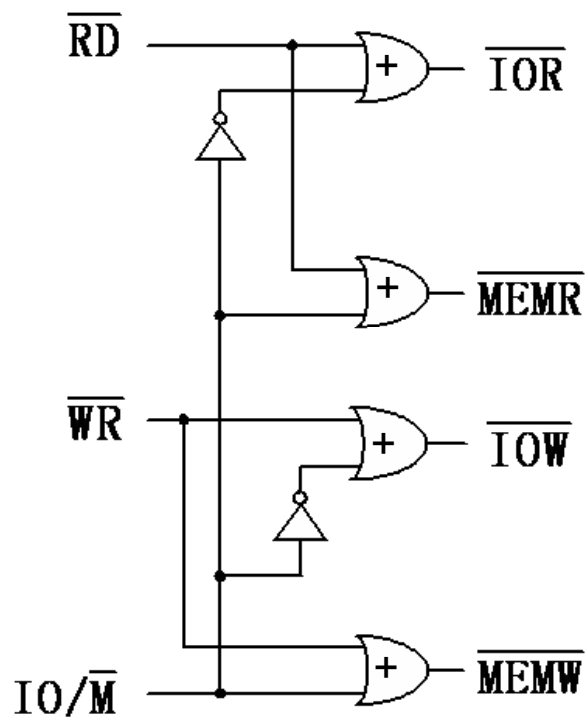
IO/M	WR	MEMW
0	0	0
1	X	1
X	1	1

$$\overline{\text{MEMW}} = \text{IO}/\overline{\text{M}} + \overline{\text{WR}} = \overline{\overline{\text{IO}/\overline{\text{M}} \cdot \overline{\text{WR}}}}$$



读写信号和IO/ \overline{M} 信号的组合——综合实现存储器和I/O读写信号

I 通过单一电路同时实现存储器和I/O的读写4个信号



8088的外部引脚

I DT/ \overline{R} (27)

n 数据传送/接收信号 (出, 三态)

n 向I/O或存储器传送数据, 输出高

OUT 25H, AL ; 写I/O

MOV [BX], AX ; 写MEM

n 从I/O或存储器接收数据, 输出低

IN AL, 25H; 读I/O

MOV AX, [BX]; 读MEM

n 其它情况下处于高阻状态

地	1	40	Vcc(+5V)
A14	2	39	A15
A13	3	38	A16 / S3
A12	4	37	A17 / S4
A11	5	36	A18 / S5
A10	6	35	A19 / S6
A9	7	34	$\overline{SS0}$ (HIGH)
A8	8	33	$\overline{MN} / \overline{MX}$
AD7	9	32	RD
AD6	10	31	HOLD ($\overline{RQ}/GT0$)
AD5	11	30	HLDA ($\overline{RQ}/GT1$)
AD4	12	29	\overline{WR} (LOCK)
AD3	13	28	$\overline{M} / \overline{IO} (\overline{S2})$
AD2	14	27	DT / R (S1)
AD1	15	26	$\overline{DEN} (\overline{S0})$
AD0	16	25	ALE (QS0)
NMI	17	24	\overline{INTA} (QS1)
INTR	18	23	TEST
CLK	19	22	READY
地	20	21	RESET

8088的外部引脚

I D7~D0(9~16)数据线(双, 三态)

n 读写I/O或存储器时出现相应数据

IN AL, 25H ; 读I/O

OUT 25H, AL ; 写I/O

MOV BX, 0000H

MOV AL, [BX] ; 读MEM

MOV [BX], 32H ; 写MEM

n 注意: D7~D0和A7~A0复用, 先
传送地址, 后传送数据。

I \overline{DEN} (26)数据允许 (出, 三态)

n 低电平: $AD_{7\sim0}$ 传送的是数据。

地	1	40	Vcc(+5V)
A14	2	39	A15
A13	3	38	A16 / S3
A12	4	37	A17 / S4
A11	5	36	A18 / S5
A10	6	35	A19 / S6
A9	7	34	$\overline{SS0}$ (HIGH)
A8	8	33	$\overline{MN} / \overline{MX}$
AD7	9	32	RD
AD6	10	31	HOLD ($\overline{RQ}/GT0$)
AD5	11	30	HLDA ($\overline{RQ}/GT1$)
AD4	12	29	\overline{WR} (LOCK)
AD3	13	28	$\overline{M} / \overline{IO} (\overline{S2})$
AD2	14	27	DT / R (S1)
AD1	15	26	\overline{DEN} (S0)
AD0	16	25	ALE (QS0)
NMI	17	24	\overline{INTA} (QS1)
INTR	18	23	TEST
CLK	19	22	READY
地	20	21	RESET

8088的外部引脚

I A19~A0地址线(出, 三态)

n 读写I/O或存储器, 出现地址信息

IN AL, 25H ; 读I/O

OUT 25H, AL ; 写I/O

MOV BX, 0000H

MOV AL, [BX] ; 读MEM

MOV [BX], 32H ; 写MEM

n 注意: D7~D0和A7~A0复用, 先传送地址, 后传送数据。

u 逻辑地址由地址加法器转为20位物理地址。

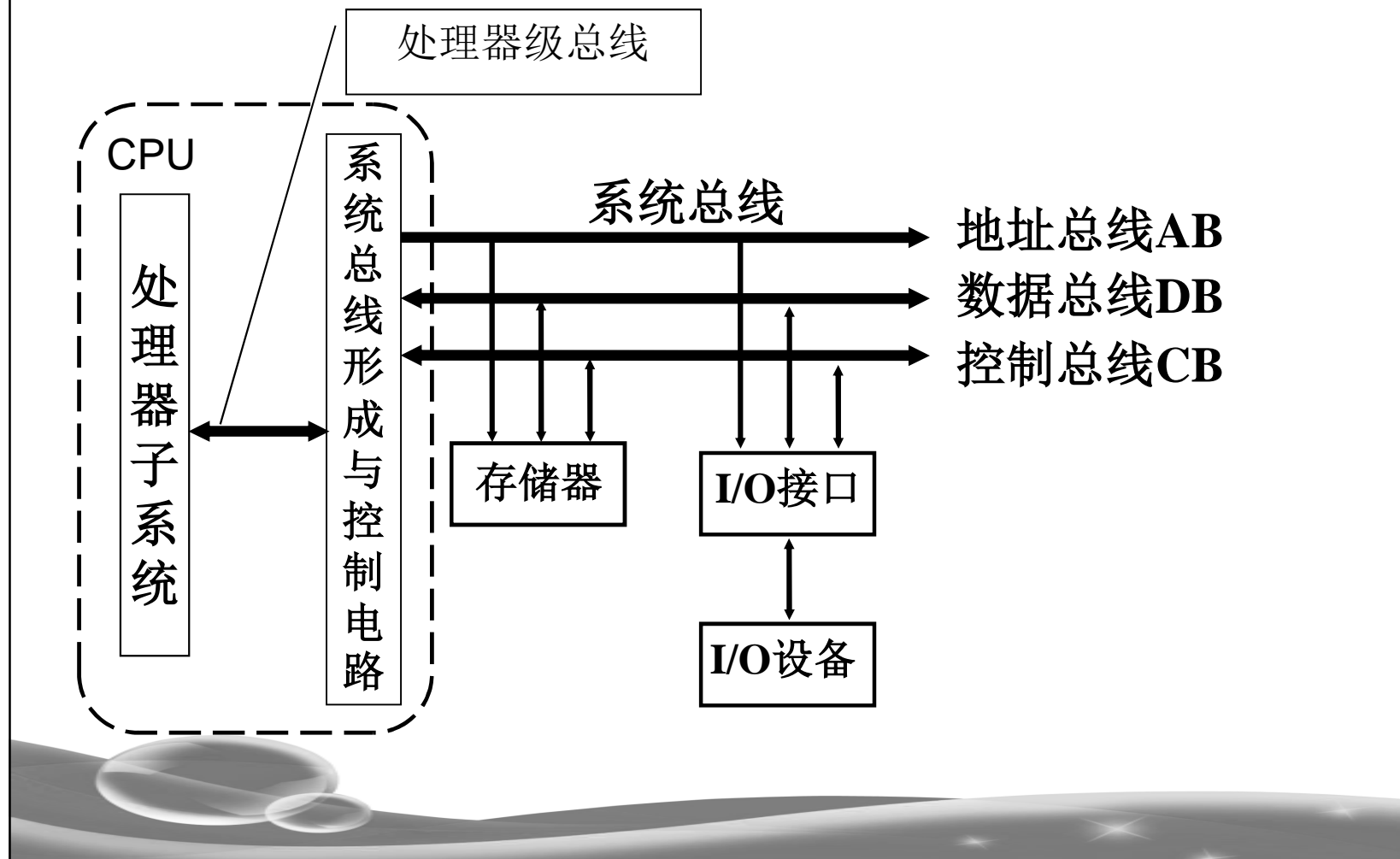
I ALE(25)地址锁存信号(出, 三态)

n 当A_{19~0}出现地址, 产生正脉冲。

n 利用该脉冲锁存地址

地	1	40	Vcc(+5V)
A14	2	39	A15
A13	3	38	A16 / S3
A12	4	37	A17 / S4
A11	5	36	A18 / S5
A10	6	35	A19 / S6
A9	7	34	SS0 (HIGH)
A8	8	33	MN / \overline{MX}
AD7	9	32	RD
AD6	10	31	HOLD ($\overline{RQ/GT0}$)
AD5	11	30	HLDA ($\overline{RQ/GT1}$)
AD4	12	29	WR (LOCK)
AD3	13	28	M / IO ($\overline{S2}$)
AD2	14	27	DT / R (S1)
AD1	15	26	\overline{DEN} (S0)
AD0	16	25	ALE (QS0)
NMI	17	24	\overline{INTA} (QS1)
INTR	18	23	TEST
CLK	19	22	READY
地	20	21	RESET

8088系统总线的形成



8088系统总线的形成（最小系统）

I 主要解决

n 实现地址总线，数据总线和控制总线

u 地址与数据的分离

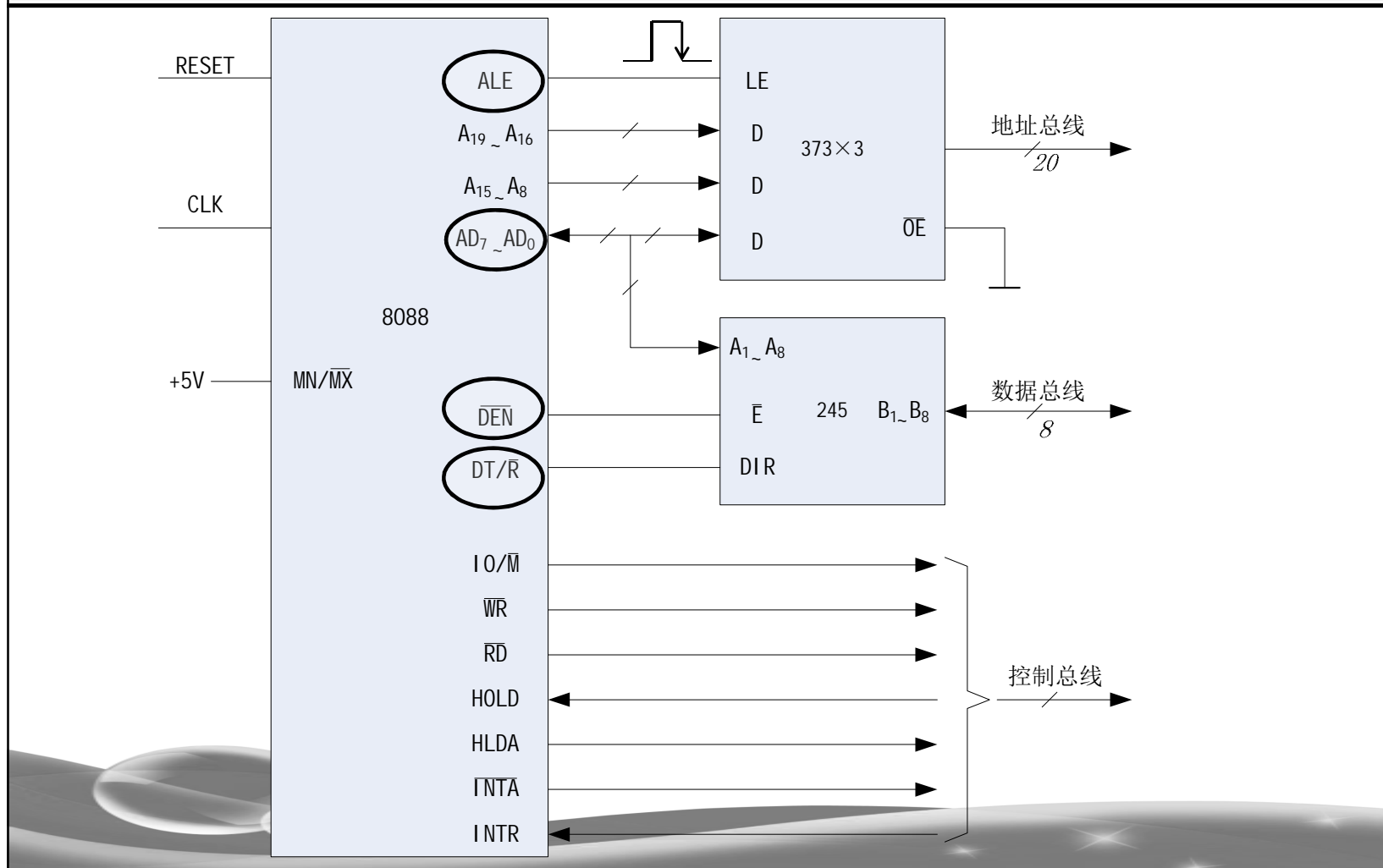
u 地址锁存

8088系统总线的形成（最小系统）

I 电路实现方案

- n 用3片8位锁存器（例74LS373）实现地址锁存。ALE为锁存控制信号， $\overline{OE}=0$ 输出地址；
- n 用1片双向缓冲器（三态门, 例如74LS245）用作数据总线隔离，DT/ \overline{R} 控制方向， \overline{DEN} 作为使能信号；
- n 控制信号由8088直接产生。

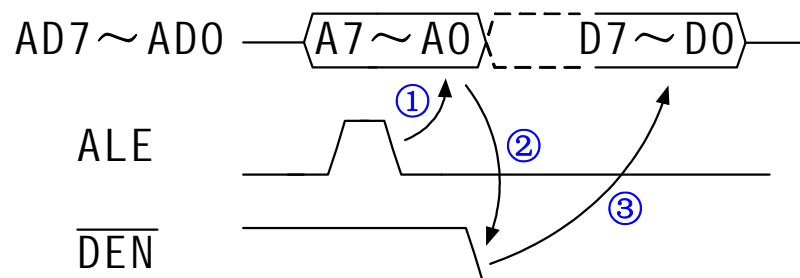
8088最小系统的典型实现电路



8088最小系统的典型实现电路

I AD_{7~0}解复用和地址锁存

n先传送地址，后传送数据



①总线出现地址后，在ALE下降沿，地址被373锁存；

②总线出现地址时， $\overline{\text{DEN}}$ 高电平，245失能，数据总线被禁止。

③总线出现数据时， $\overline{\text{DEN}}$ 变低，245使能，数据总线开通。

I 数据传送方向的确定

n数据输出时DT/ $\overline{\text{R}}$ 高电平，245的DIR高电平：A \longrightarrow B

n数据输出时DT/ $\overline{\text{R}}$ 高电平，245的DIR高电平：B \longleftarrow A

8088的外部引脚

- I INTR(18, 中断请求信号, 入)
 - n 高电平有效
 - n 外设发来的可屏蔽中断请求信号。
- I $\overline{\text{INTA}}$ (24, 中断请应答信号, 出)
 - n 低电平有效
 - n 向外设回应的中断应答信号。
 - n 连续2个负脉冲
- I NMI (17, 非屏蔽中断请求信号, 入)
 - n 非屏蔽中断请求信号
 - n 高电平有效

地	1	40	Vcc(+5V)
A14	2	39	A15
A13	3	38	A16 / S3
A12	4	37	A17 / S4
A11	5	36	A18 / S5
A10	6	35	A19 / S6
A9	7	34	$\overline{\text{SS0}}$ (HIGH)
A8	8	33	$\overline{\text{MN}} / \overline{\text{MX}}$
AD7	9	32	RD
AD6	10	31	HOLD ($\overline{\text{RQ}}/\text{GT0}$)
AD5	11	30	HLDA ($\overline{\text{RQ}}/\text{GT1}$)
AD4	12	29	$\overline{\text{WR}}$ (LOCK)
AD3	13	28	$\overline{\text{M}} / \overline{\text{IO}} (\overline{\text{S2}})$
AD2	14	27	DT / R (S1)
AD1	15	26	$\overline{\text{DEN}}$ (S0)
AD0	16	25	ALE (QS0)
NMI	17	24	$\overline{\text{INTA}}$ (QS1)
INTR	18	23	TEST
CLK	19	22	READY
地	20	21	RESET

8088的外部引脚

I READY(22, 准备就绪信号, 入)

n 高电平有效

n 存储器或I/O口准备好时将该信号置为高电平。

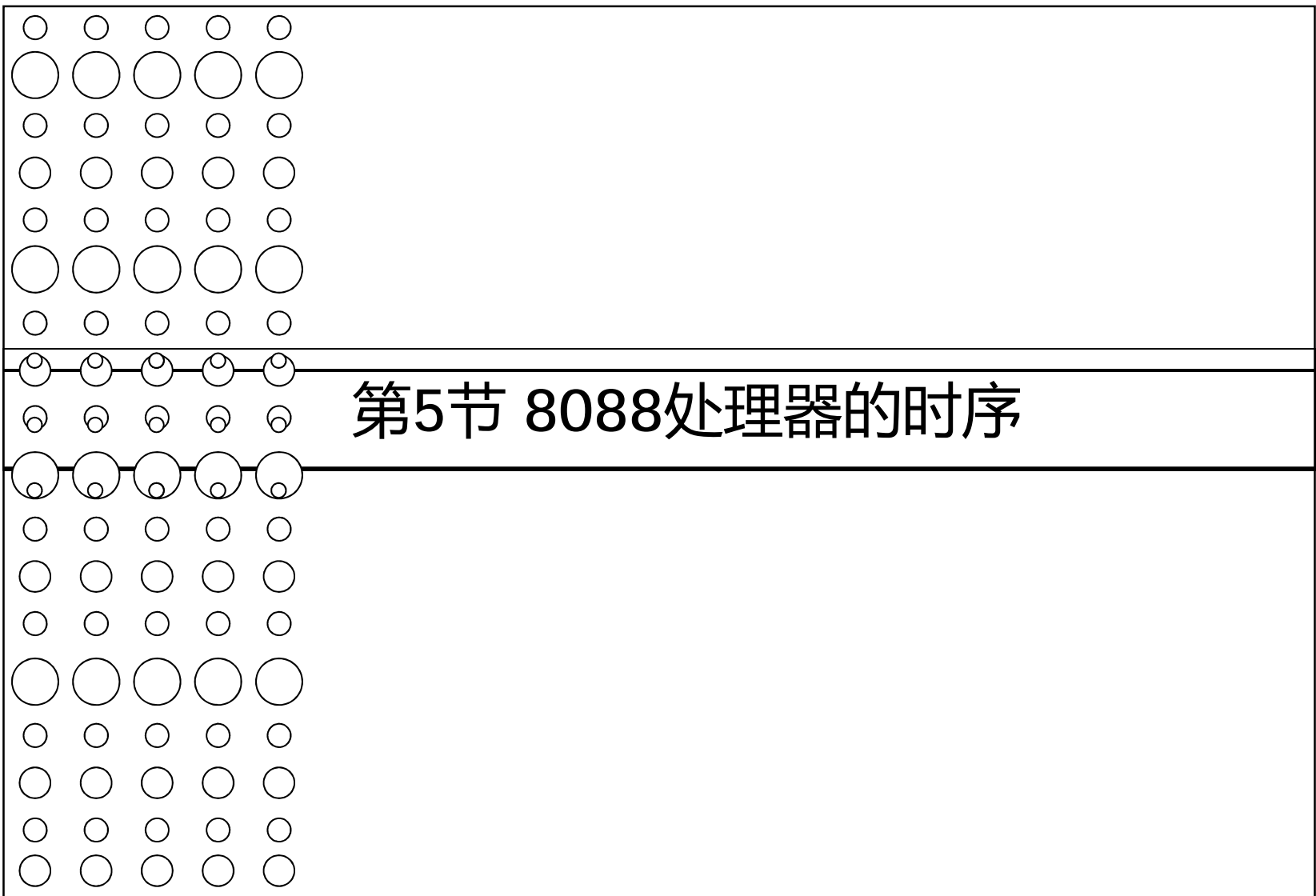
I TEST(23, 测试信号, 入)

n 低电平有效

n 执行WAIT指令时CPU监视TEST端，为低电平则执行WAIT后面的指令；为高时CPU空转。

n 用来与外设同步。

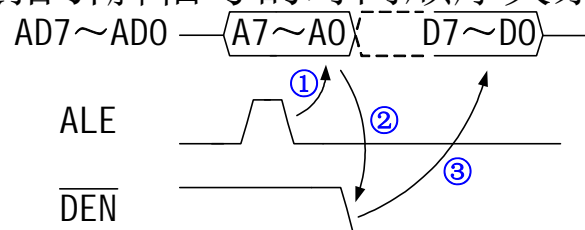
地	1	40	Vcc(+5V)
A14	2	39	A15
A13	3	38	A16 / S3
A12	4	37	A17 / S4
A11	5	36	A18 / S5
A10	6	35	A19 / S6
A9	7	34	SS0 (HIGH)
A8	8	33	MN / MX
AD7	9	32	RD
AD6	10	31	HOLD (RQ/GT0)
AD5	11	30	HLDA (RQ/GT1)
AD4	12	29	WR (LOCK)
AD3	13	28	M / IO (S2)
AD2	14	27	DT / R (S1)
AD1	15	26	DEN (S0)
AD0	16	25	ALE (QS0)
NMI	17	24	INTA (QS1)
INTR	18	23	TEST
CLK	19	22	READY
地	20	21	RESET



8088处理器的时序

I 时序概念 (Timing)

n 时序是指引脚信号的时间顺序关系。



n 时序描述CPU如何通过总线对外完成各种操作（总线操作）：

- u 存储器读操作
- u I/O读操作
- u 存储器写操作
- u I/O写操作
- u 中断响应操作
- u 总线请求及响应操作

和时序相关的几个概念

I 周期

n时钟周期 | 总线周期 | 指令周期

I 时钟周期 (Clock Cycle)

n时钟的周期，时钟频率的倒数。8088: 4.77MHz

I 总线周期 (Bus Cycle)

nCPU通过总线与外部进行基本操作（一次数据交换）的过程

nI/O读或写总线周期，存储器读或写总线周期,...

I 指令周期 (Instruction Cycle)

n指令经取指、译码、读写操作数到执行完成的过程

指令周期 > 总线周期 > 时钟周期

I 总线周期（即总线操作）的产生

n指令取指阶段：存储器读总线周期(读取指令代码)

n源操作数是存储单元的指令：存储器读总线周期

n目的操作数是存储单元的指令：存储器写总线周期

n执行IN指令：I/O读总线周期

n执行OUT指令：I/O写总线周期

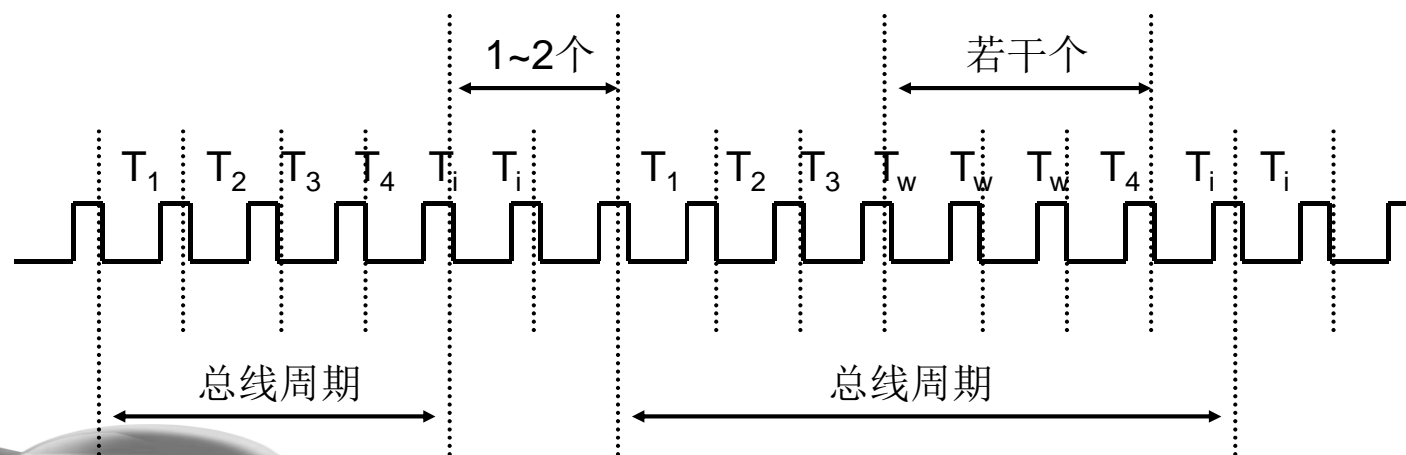
nCPU响应可屏蔽中断：中断响应总线周期

I 空闲总线周期

nCPU不执行任何存储单元或I/O操作，则执行空闲周期 T_i (Idle)

总线周期的构成

- I (基本)总线周期需要4个时钟周期: T_1 、 T_2 、 T_3 和 T_4
 - n时钟周期也被称作“T状态” (T State)
 - n空闲时钟周期 T_i , 在两个总线周期之间插入。
 - n当要延长总线周期时要插入等待状态 T_w (Wait)
 - u在 T_3 和 T_4 之间插入 T_w



CPU和外设操作的同步：READY,等待状态Tw

I READY(22, 准备就绪信号,入)

n 高电平有效

n MEM或I/O准备好时将该信号置高

n CPU在T3期间采样READY信号

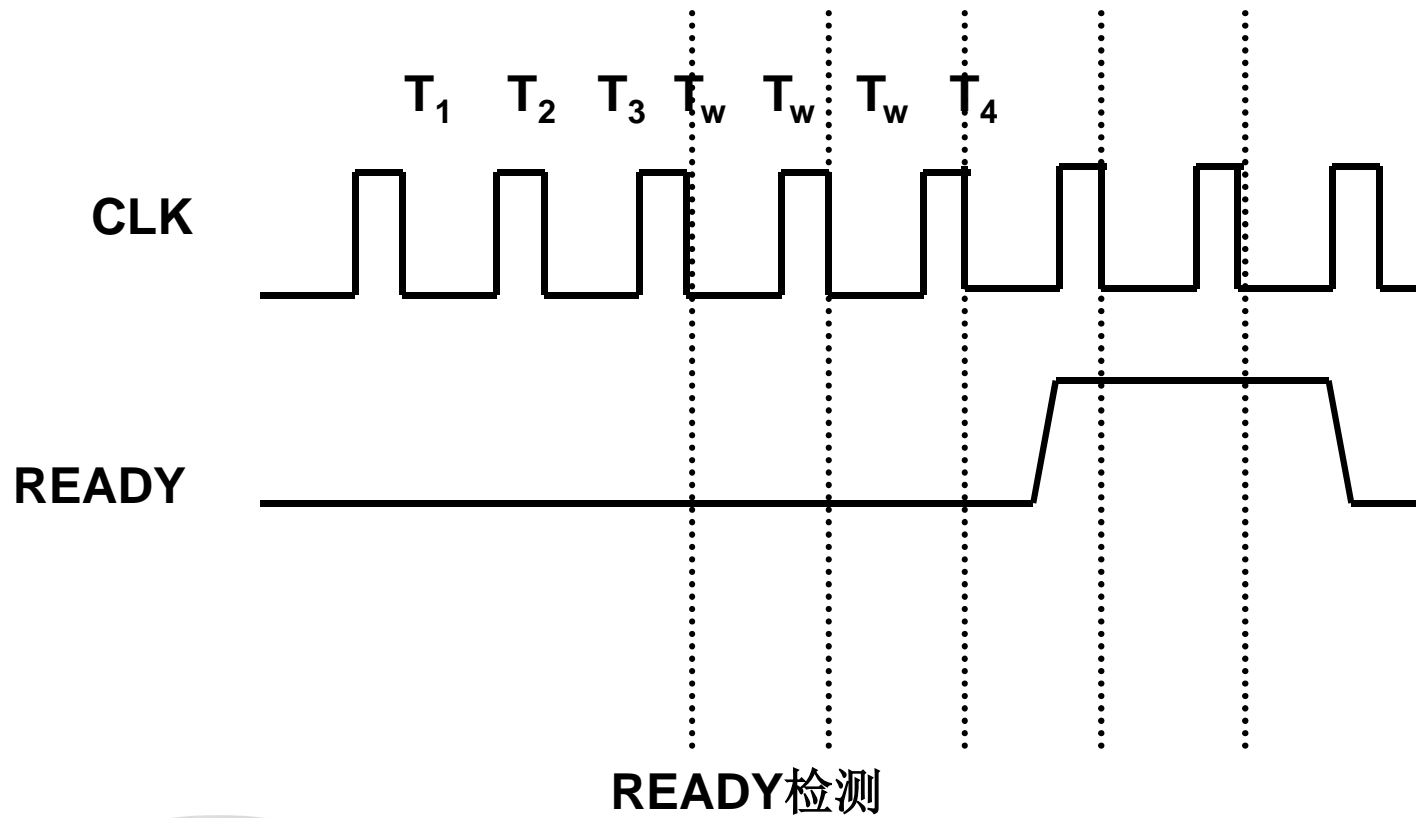
u1. T3期间检测READY是否有效?

u2. 如果READY无效, 在T3和T4间
插一个等效T3的Tw状态, 转1

u3. 如果READY有效, 执行完T3后,
进入T4。

地	1	40	Vcc(+5V)
A14	2	39	A15
A13	3	38	A16 / S3
A12	4	37	A17 / S4
A11	5	36	A18 / S5
A10	6	35	A19 / S6
A9	7	34	SS0 (HIGH)
A8	8	33	MN / MX
AD7	9	32	RD
AD6	10	31	HOLD (RQ/GT0)
AD5	11	30	HLDA (RQ/GT1)
AD4	12	29	WR (LOCK)
AD3	13	28	M / IO (S2)
AD2	14	27	DT / R (S1)
AD1	15	26	DEN (S0)
AD0	16	25	ALE (QS0)
NMI	17	24	INTA (QS1)
INTR	18	23	TEST
CLK	19	22	READY
地	20	21	RESET

同步：T3和T4间插入等待状态 T_w



I 基本的总线周期

n存储器读

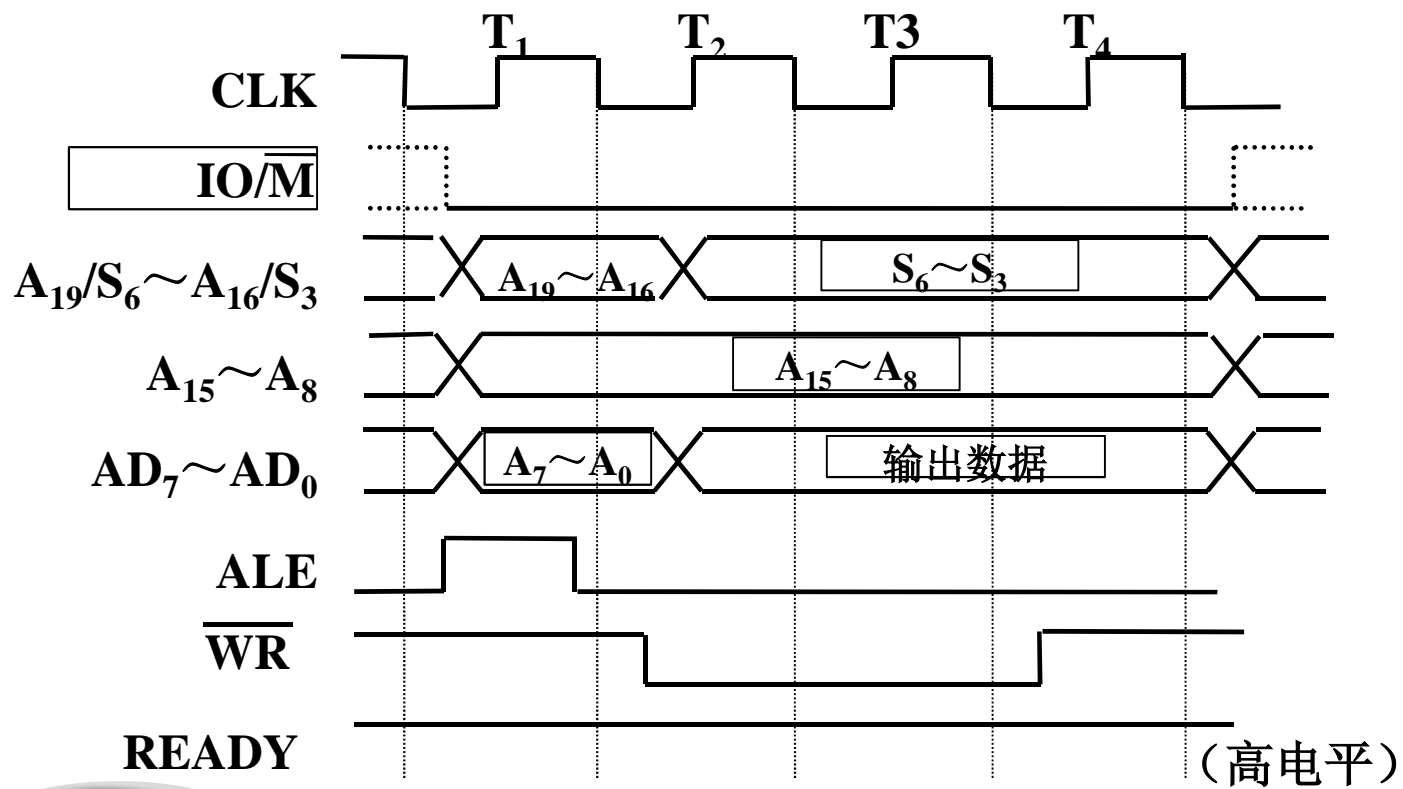
n存储器写

nI/O端口读

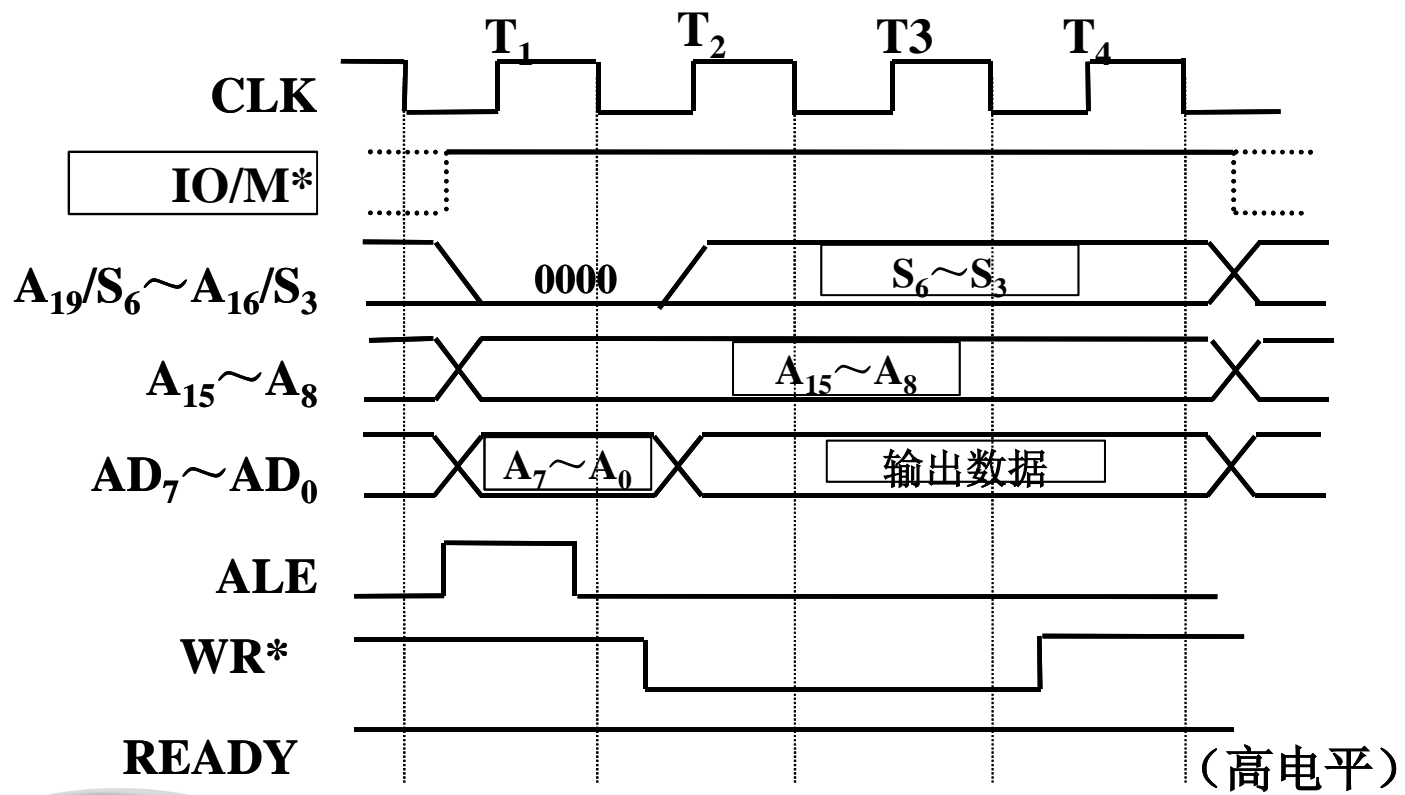
nI/O端口写

n中断响应

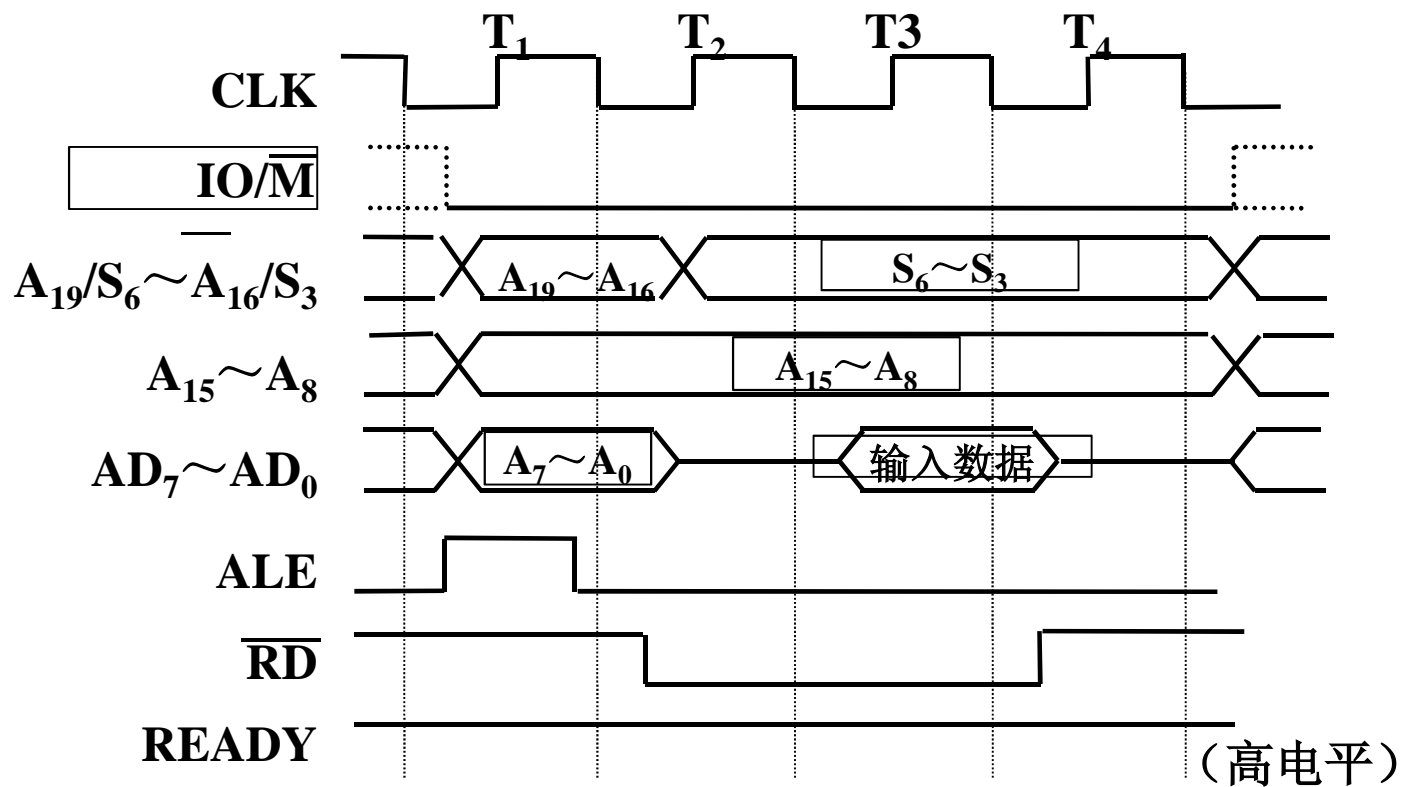
存储器写总线周期



I/O写总线周期



存储器读总线周期



I/O读总线周期

