《数字逻辑》
Digital Logic

组合逻辑

北京工业大学软件学院
王晓懿

组合逻辑引言

- 组合逻辑的概念

  组合逻辑函数的输出状态取决于所有输入的状态“逻辑组合”。

  如与非、与或逻辑等。

- 组合逻辑电路的特点：

1）电路的输出只是和输入的当前状态有关，和过去的状态无关。

2）区别于时序电路：和过去的状态有关。

组合逻辑学习目标

- 掌握组合逻辑的设计方法
- 掌握组合逻辑的分析方法
- 掌握使用HDL描述组合逻辑的方法

组合电路设计综合方法回顾

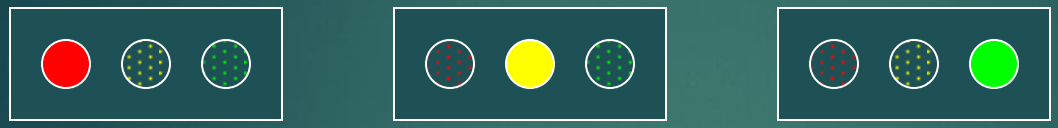
根据给出的实际问题，

求出实现这一逻辑功能的电路。

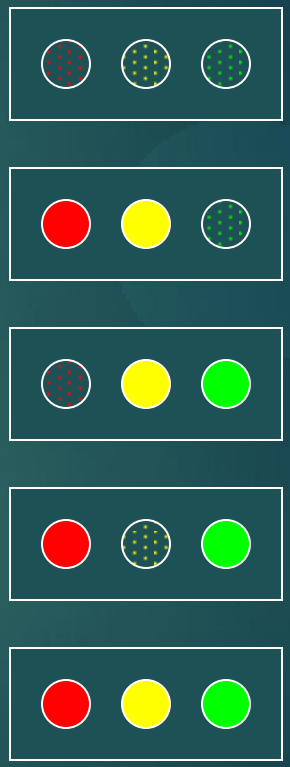
- 进行逻辑抽象，得到真值表或逻辑函数式
- 选择器件的类型
- 逻辑化简或变换成适当的形式
- 电路处理，得到电路图

例：设计一个监视交通信号灯工作状态的逻辑电路

正常工作状态



故障状态



1、进行逻辑抽象：

输入变量：红R 黄Y 绿G 三盏灯的状态

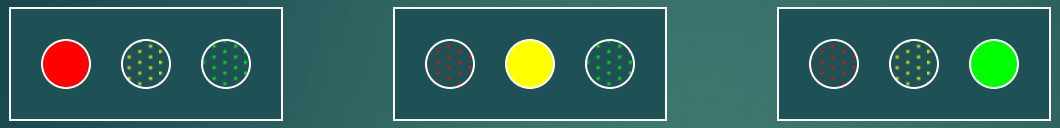
灯亮为1，不亮为0

输出变量：故障信号F

正常工作为0，发生故障为1

例：设计一个监视交通信号灯工作状态的逻辑电路

正常工作状态



1、进行逻辑抽象：

输入变量：红R 黄Y 绿G 三盏灯的状态

灯亮为1，不亮为0

输出变量：故障信号F

正常工作为0，发生故障为1

真 值 表

R	Y	G	F
0	0	0	1
0	0	1	
0	1	0	
0	1	1	1
1	0	0	
1	0	1	1
1	1	0	1
1	1	1	1

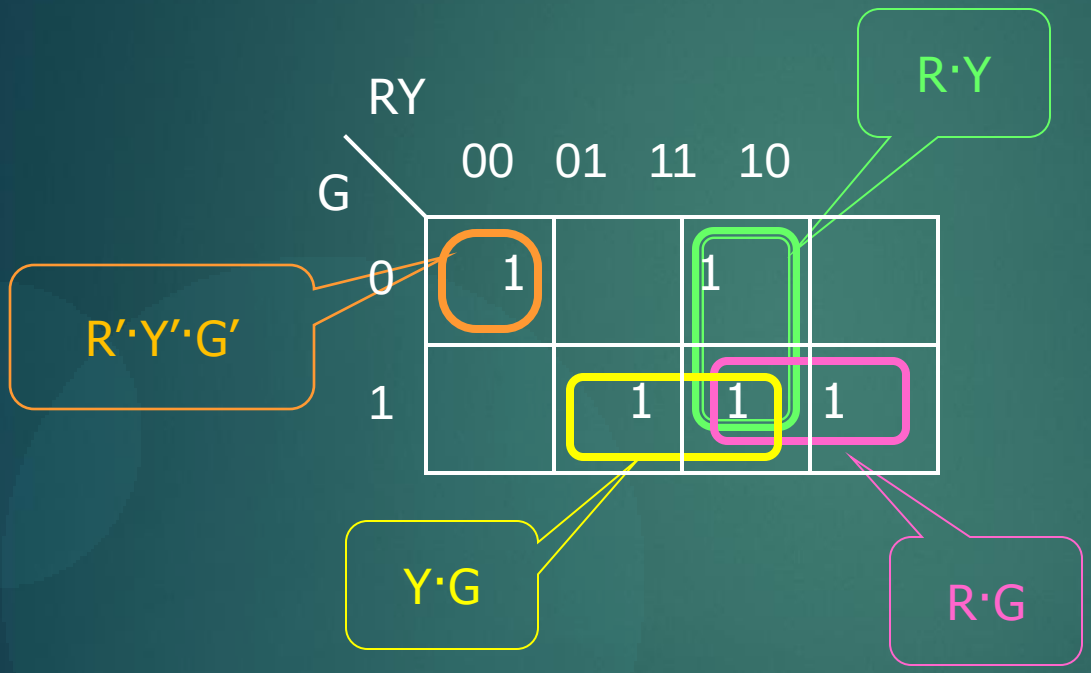
1、逻辑抽象

真 值 表

R	Y	G	F
0	0	0	1
0	0	1	
0	1	0	
0	1	1	1
1	0	0	
1	0	1	1
1	1	0	1
1	1	1	1

2、用门电路设计

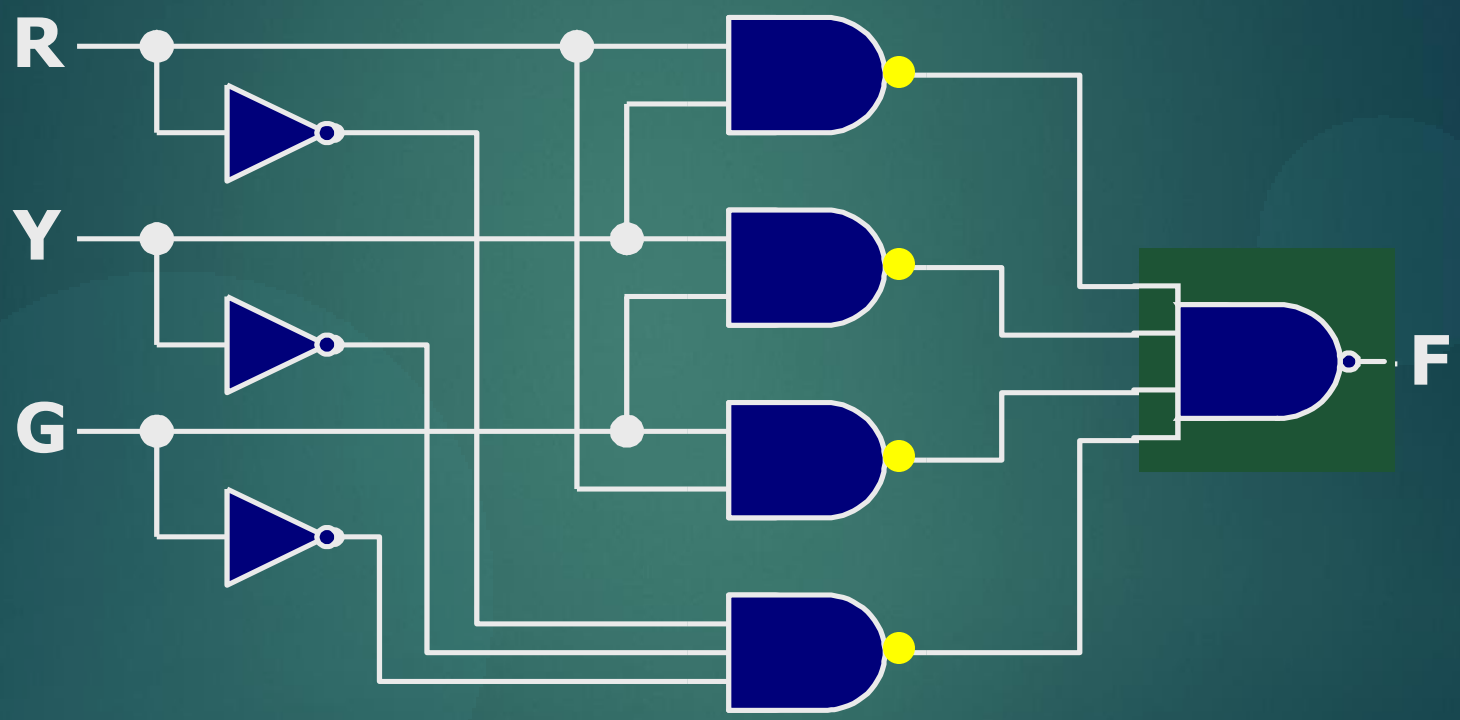
写出逻辑函数式并化简



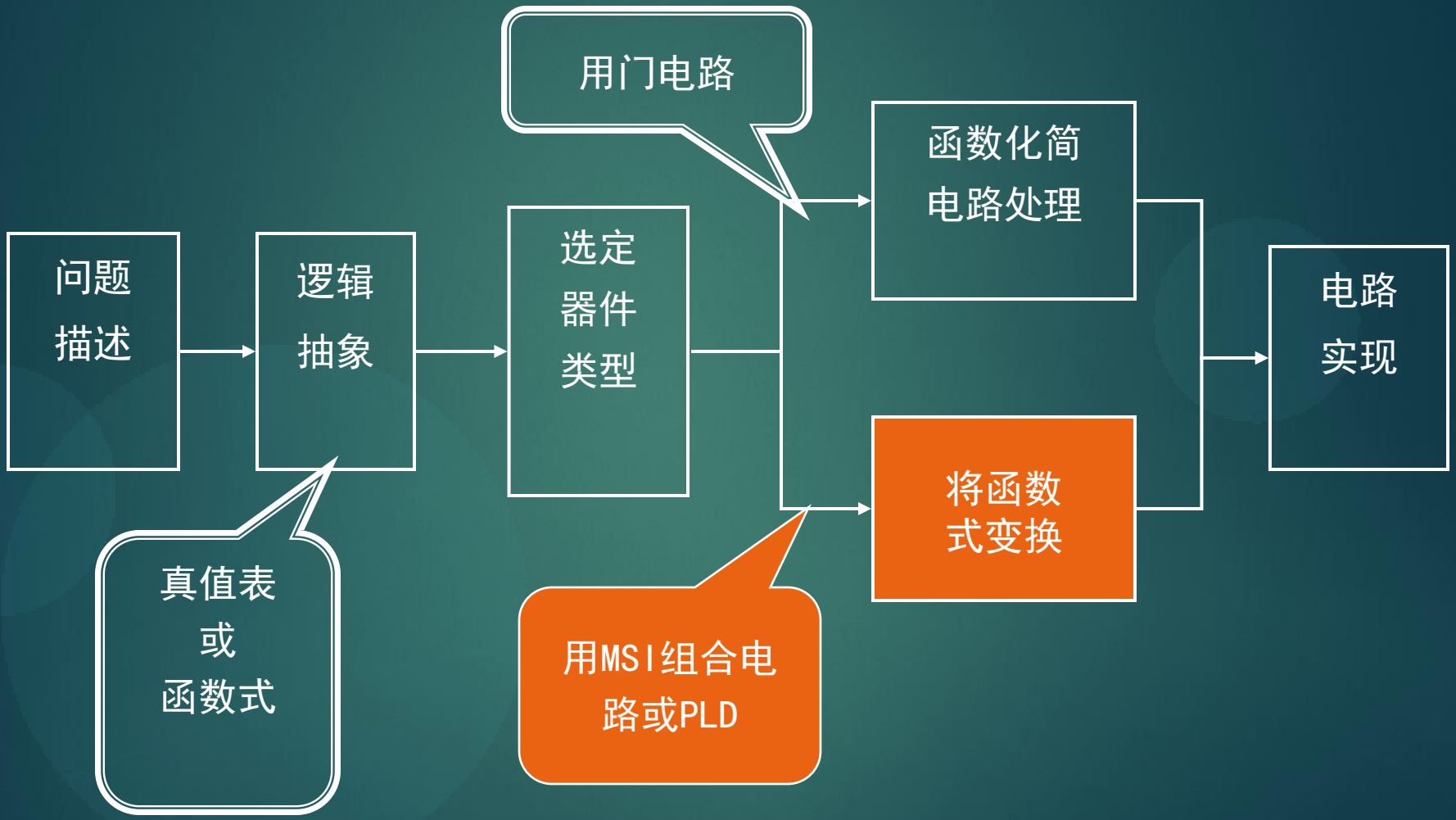
F = R′·Y′·G′ + R·Y + R·G + Y·G

3、电路处理

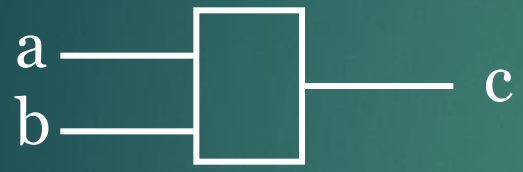
$F = R'\cdot Y'\cdot G' + R\cdot Y + R\cdot G + Y\cdot G$



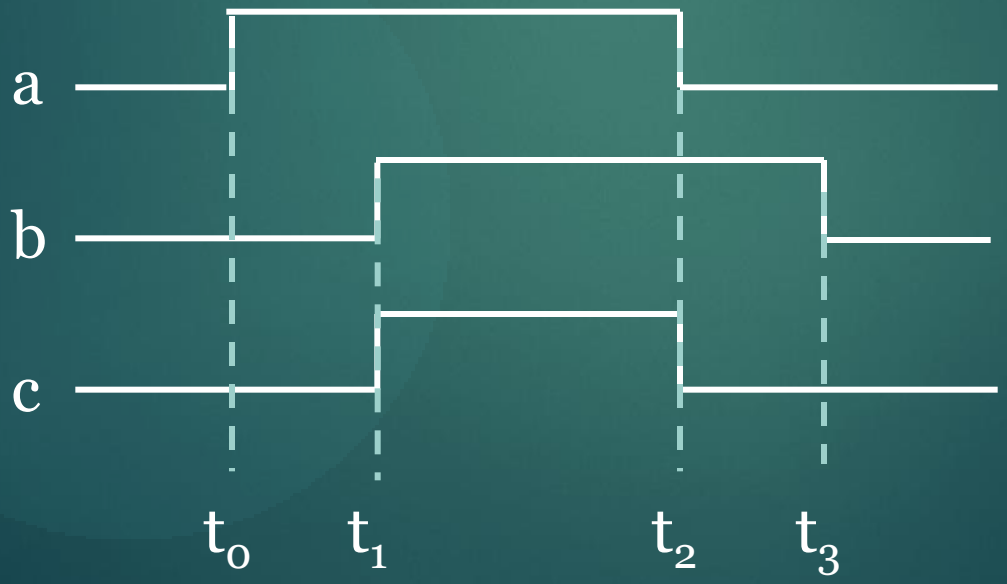
组合电路的综合


问题
描述
逻辑
抽象
选定
器件
类型
用门电路
函数化简
电路处理
将函数
式变换
电路
实现
真值表
或
函数式
用MSI组合电
路或PLD

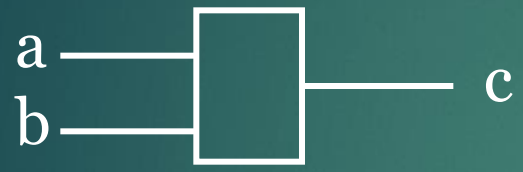
电路延迟



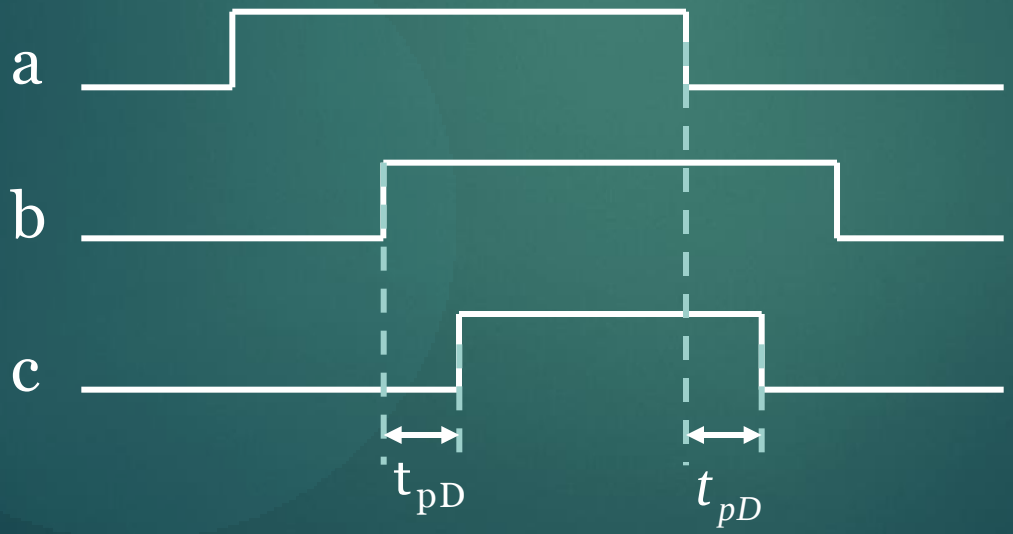
理想情况：门电路没有延迟



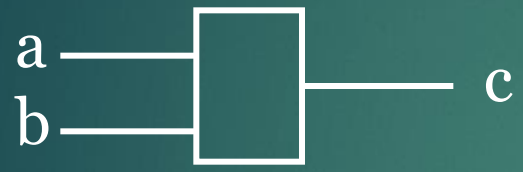
电路延迟



实际情况：门电路存在延迟 t_{pD}

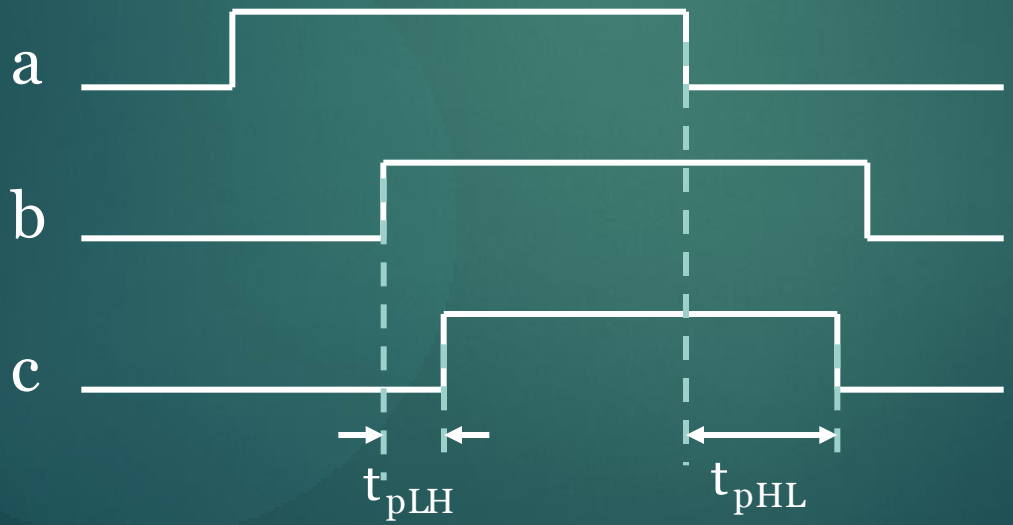


电路延迟



实际情况：门电路存在延迟

前沿延迟与后沿延迟不相等



典型的组合逻辑电路

（1）门电路　　（Gates）
（2）译码电路　　（Decoders）
（3）数据选择电路　（Multiplexer）（多路开关）
（4）加法器　　（Adders）
　　减法器
（5）编码电路　　（Encoders）
（6）比较器
（7）乘法器

常用组合逻辑电路

- 译码器（Decoder)

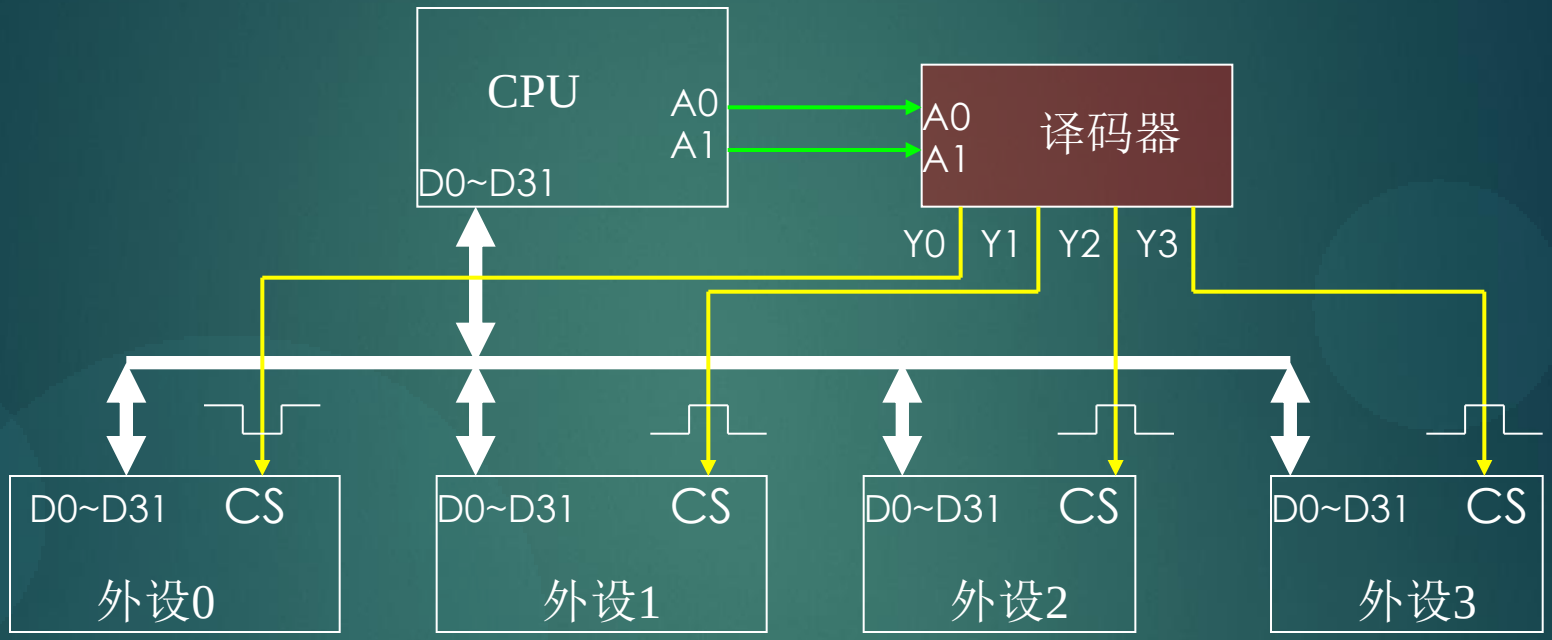
译码器的功能分类

1. 用来表示输入变量状态全部组合的，称变量译码器
   N位输入，2^N输出。
   常见的集成化译码器有2-4、3-8、4-16

2. 码制译码器：如8421码变换为循环码等

3. 显示译码器：控制数码管显示

译码器的设计需求(设计步骤一)



功能级设计要求：

A0=0，A1=0时，外设0工作

A0=1，A1=0时，外设1工作

A0=0，A1=1时，外设2工作

A0=1，A1=1时，外设3工作

信号级设计要求：

A0=0，A1=0时，Y0=0，Y1,Y2,Y3=1

A0=1，A1=0时，Y1=0，Y0,Y2,Y3=1

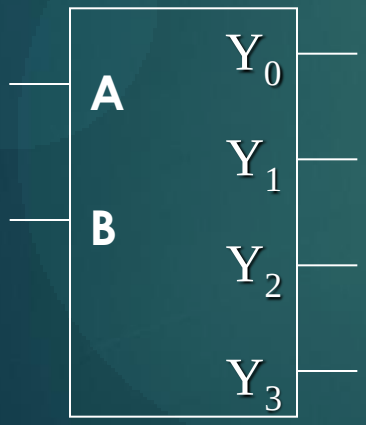
A0=0，A1=1时，Y2=0，Y0,Y1,Y3=1

A0=1，A1=1时，Y3=0，Y0,Y1,Y2=1

2-4译码器真值表和输出表达式（设计步骤二）

定义：2－4译码器是指2输入－4输出的变量译码器。2输入,4输出.对应输入的每一种组合，唯一只有一个输出为“0”.

逻辑示意图



真值表

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

输出表达式

$$\begin{cases} Y_0 = \overline{\overline{A}\,\overline{B}} \\ Y_1 = \overline{\overline{A}B} \\ Y_2 = \overline{A\overline{B}} \\ Y_3 = \overline{AB} \end{cases}$$

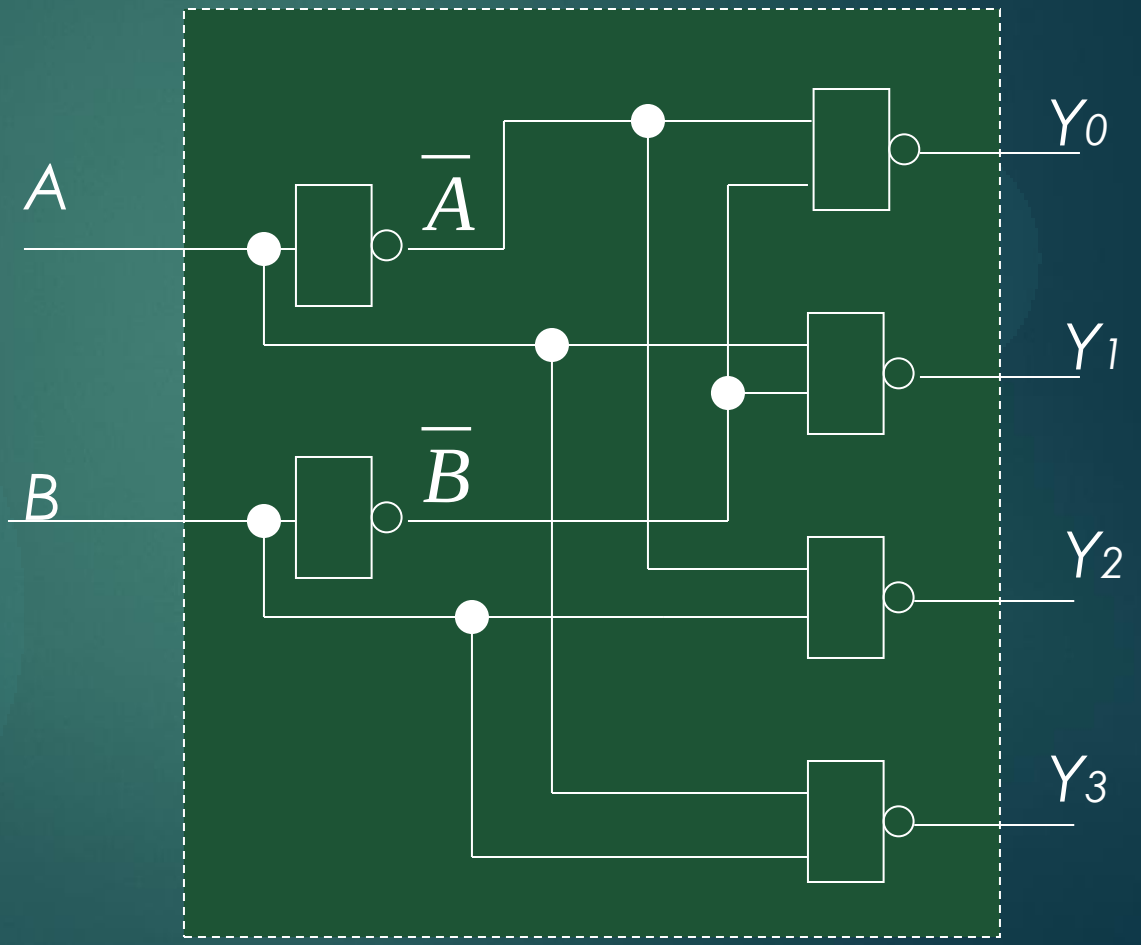
只用与非门实现的输出表达式

按照输出表达式画出逻辑图（步骤三）

输出表达式

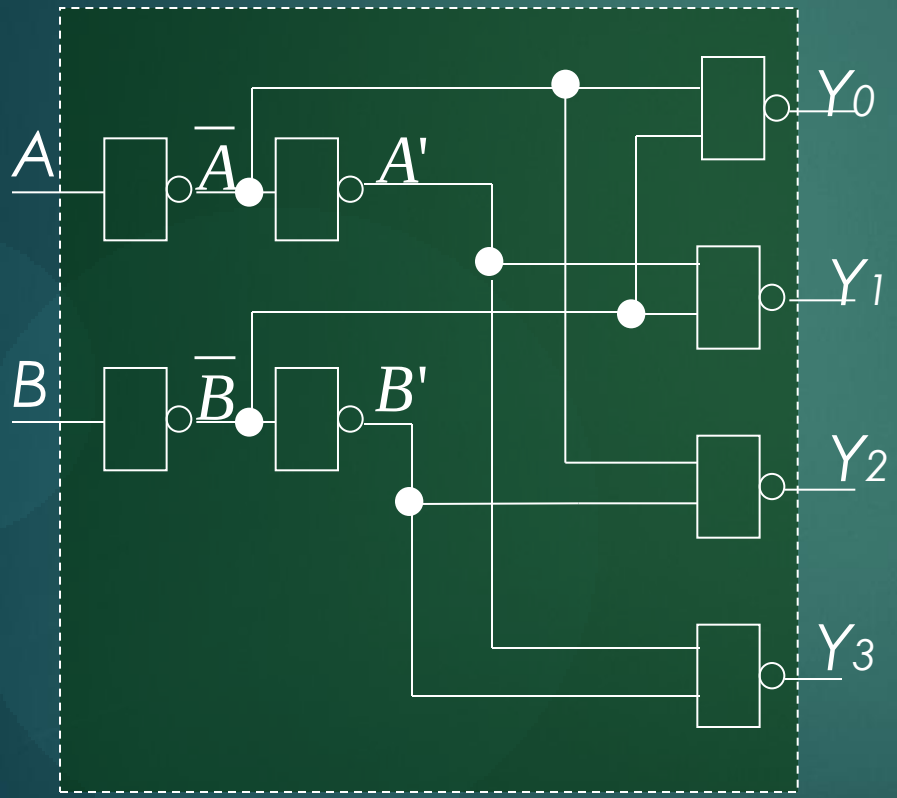
$$\begin{cases} Y_0 = \overline{\overline{A}\,\overline{B}} \\ Y_1 = \overline{A\overline{B}} \\ Y_2 = \overline{\overline{A}B} \\ Y_3 = \overline{AB} \end{cases}$$

有没有什么问题？



2-4检查可能出现的问题（步骤四）

2输入－4输出译码器



输入缓冲部分　　译　码　部　分

电路由输入缓冲部分和译码部分组成。

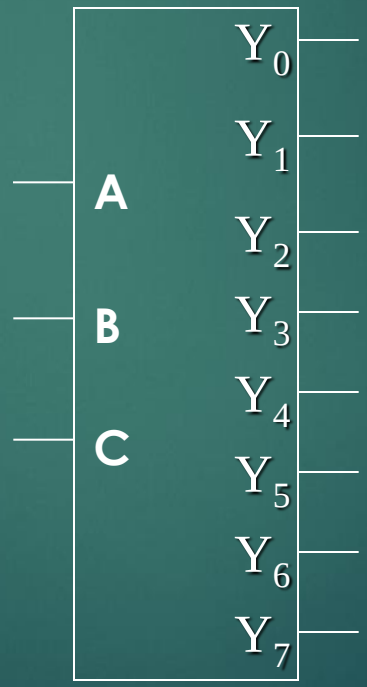
输入缓冲部分使得对外负载只有一个，减轻前面电路的负担。

3-8译码器定义，逻辑示意图

定义：3—8译码器是指3输入—8输出的变量译码器。

3—8译码器
逻辑示意图

C为最高位，
A为最低位



3-8译码器真值表和逻辑表达式

真　值　表

输　入			输　出							
A	B	C	Y_0	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7
0	0	0	0	1	1	1	1	1	1	1
1	0	0	1	0	1	1	1	1	1	1
0	1	0	1	1	0	1	1	1	1	1
1	1	0	1	1	1	0	1	1	1	1
0	0	1	1	1	1	1	0	1	1	1
1	0	1	1	1	1	1	1	0	1	1
0	1	1	1	1	1	1	1	1	0	1
1	1	1	1	1	1	1	1	1	1	0

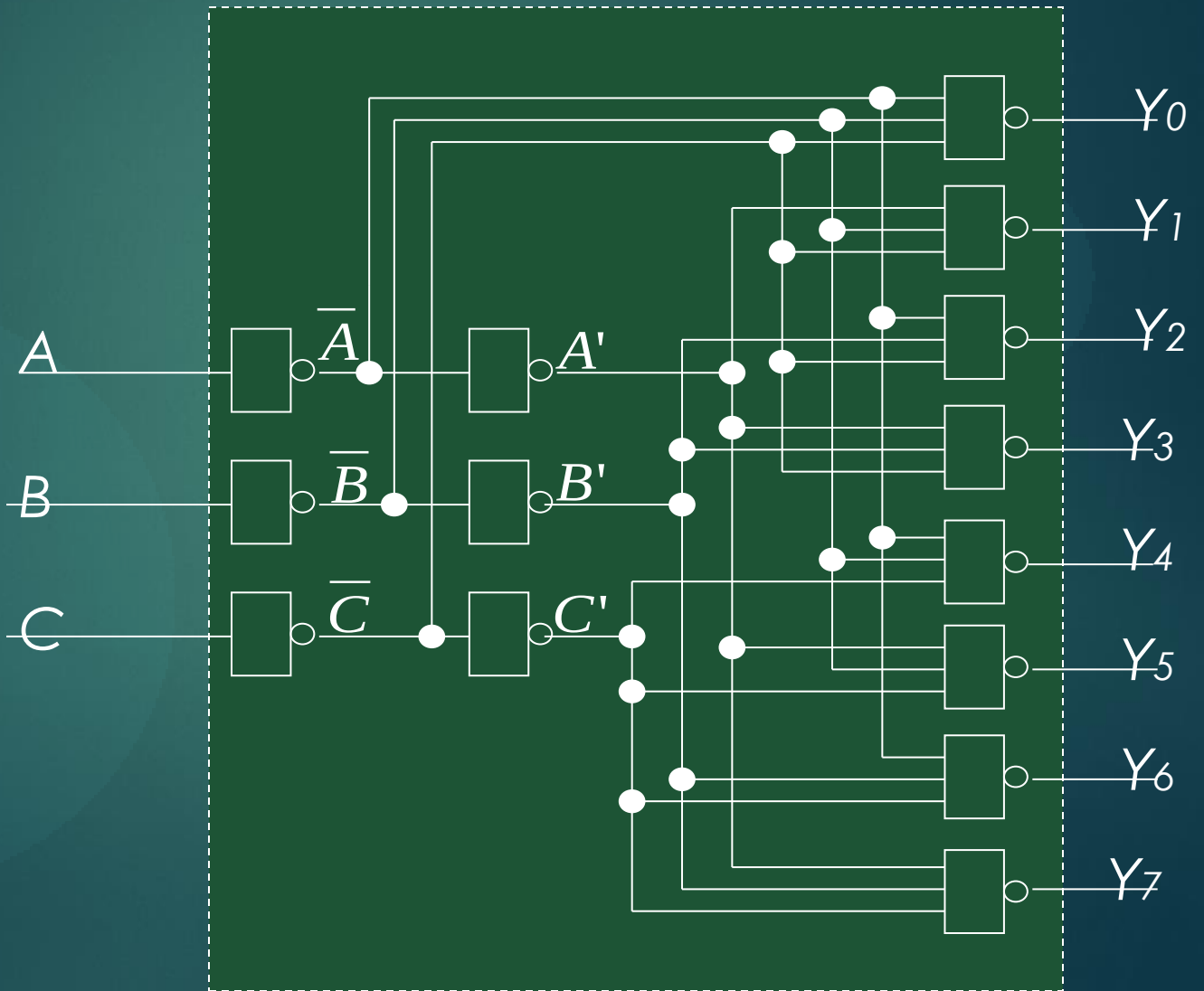
输出表达式

$$
\begin{cases}
Y_0 = \overline{\overline{A}\,\overline{B}\,\overline{C}} \\
Y_1 = \overline{\overline{A}\,\overline{B}C} \\
Y_2 = \overline{\overline{A}B\overline{C}} \\
Y_3 = \overline{\overline{A}BC} \\
Y_4 = \overline{A\overline{B}\,\overline{C}} \\
Y_5 = \overline{A\overline{B}C} \\
Y_6 = \overline{AB\overline{C}} \\
Y_7 = \overline{ABC}
\end{cases}
$$

只用与非门实现的输出表达式

按照输出表达式画出3—8译码器的逻辑图

$$
\begin{cases}
Y_0 = \overline{\overline{A}\,\overline{B}\,\overline{C}} \\
Y_1 = \overline{\overline{A}\,\overline{B}C} \\
Y_2 = \overline{\overline{A}B\overline{C}} \\
Y_3 = \overline{\overline{A}BC} \\
Y_4 = \overline{A\overline{B}\,\overline{C}} \\
Y_5 = \overline{A\overline{B}C} \\
Y_6 = \overline{AB\overline{C}} \\
Y_7 = \overline{ABC}
\end{cases}
$$



有使能端$\overline{E}$的2-4译码器

功能表

$\overline{E}$	A	B	Y_0	Y_1	Y_2	Y_3
1	X	X	1	1	1	1
0	0	0	0	1	1	1
0	1	0	1	0	1	1
0	0	1	1	1	0	1
0	1	1	1	1	1	0

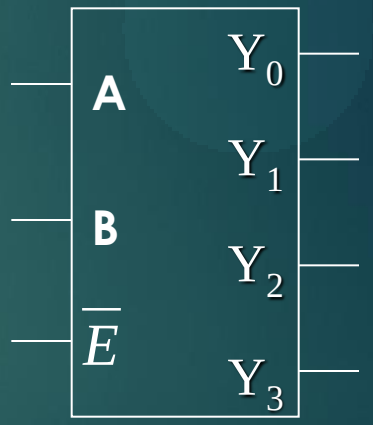
设置使能端（Enable）$\overline{E}$

当$\overline{E}$=0，译码器使能

当$\overline{E}$=1，译码器禁止

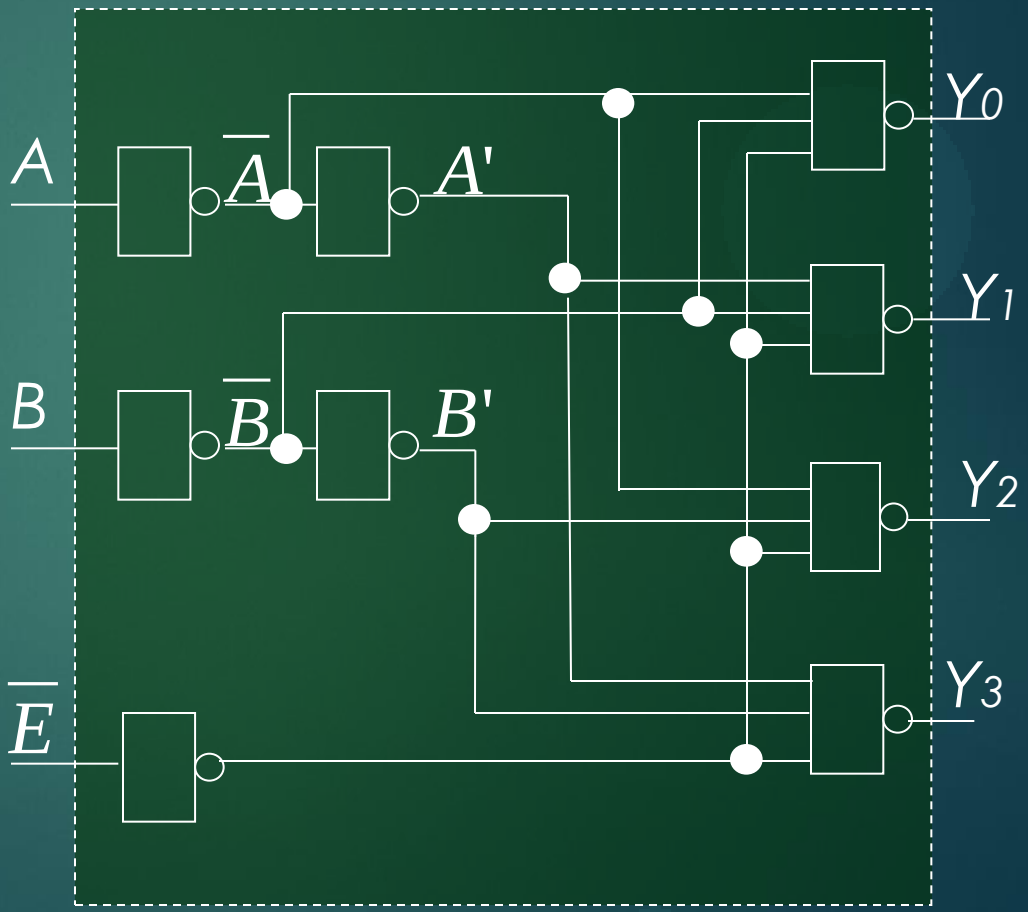
$$\begin{cases} Y_0 = \overline{E}\overline{A}\overline{B} \\ Y_1 = \overline{E}\overline{A}B \\ Y_2 = \overline{E}A\overline{B} \\ Y_3 = \overline{E}AB \end{cases}$$

逻辑示意图



有使能端$\overline{E}$的2-4译码器

$$\begin{cases} Y_0 = \overline{\overline{E}AB} \\ Y_1 = \overline{\overline{E}A\overline{B}} \\ Y_2 = \overline{\overline{E}\overline{A}B} \\ Y_3 = \overline{\overline{E}\overline{A}\overline{B}} \end{cases}$$



译码器使能端$\overline{E}$的作用

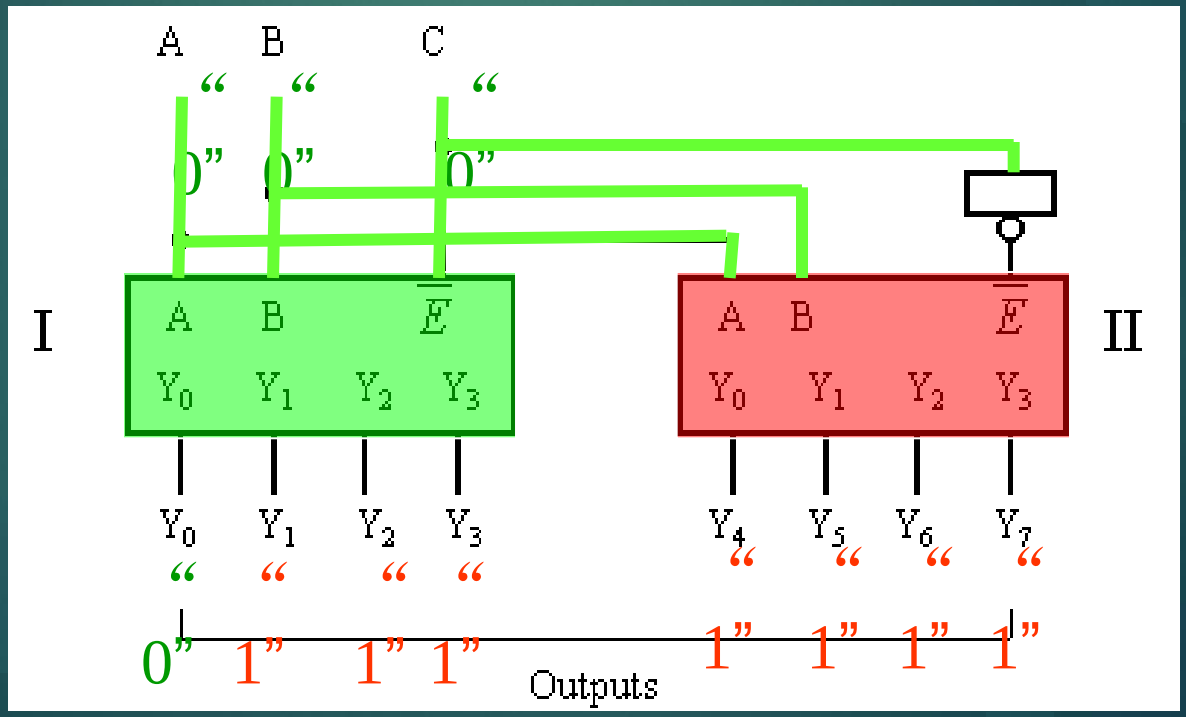
- 在集成电路中增加控制使能(Enable)端$\overline{E}$，是电路设计中常用的技术，使得集成电路更加灵活、可靠。

  一、灵活：用于扩展

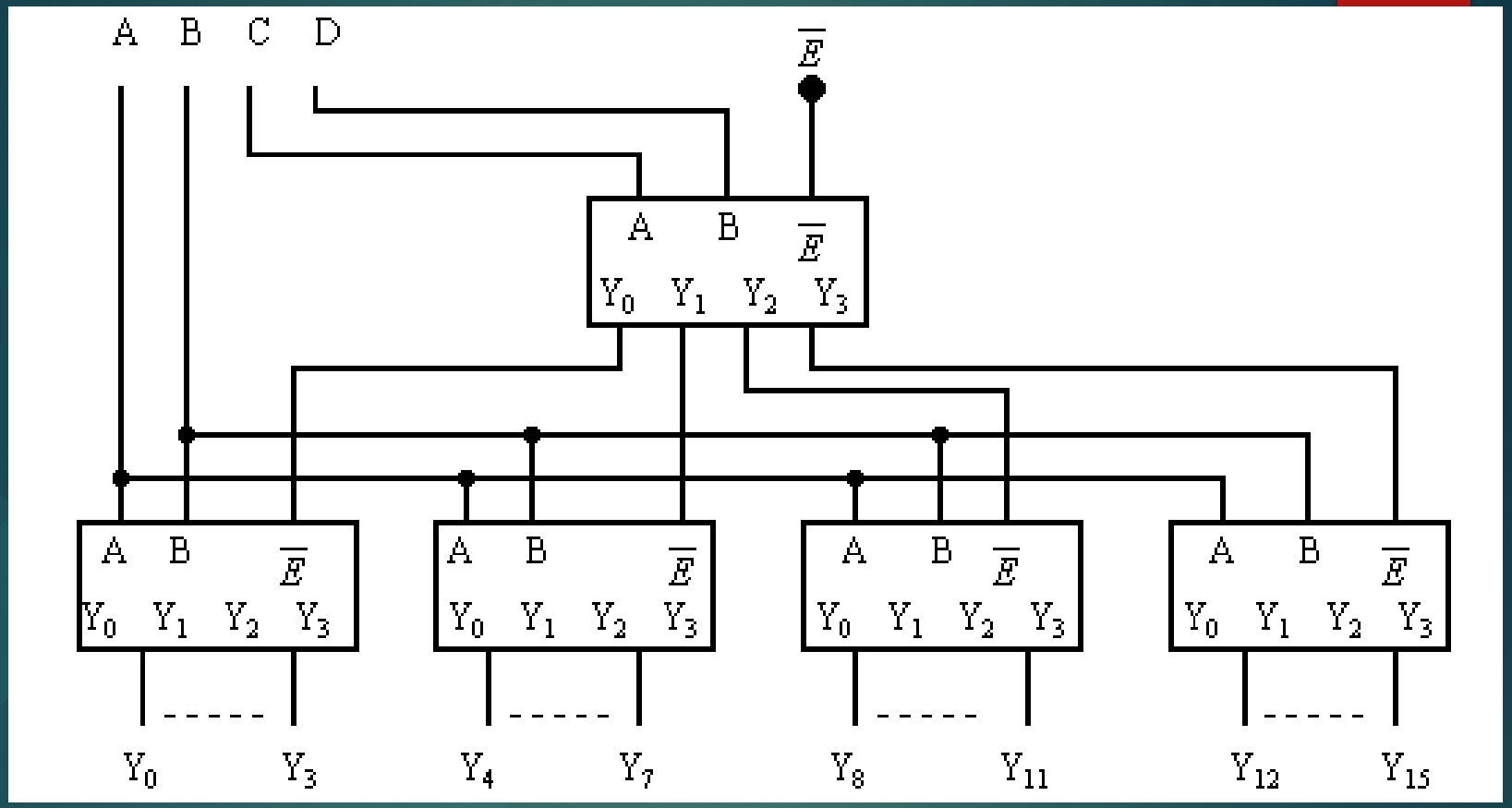
  二、可靠：用于选通

$\overline{E}$用作扩展（作用一）

用两片2-4译码器组成3-8译码器:
高位输入C用作选片，A、B用于选中片内译码。
C＝0选中片I，C＝1选中片II。



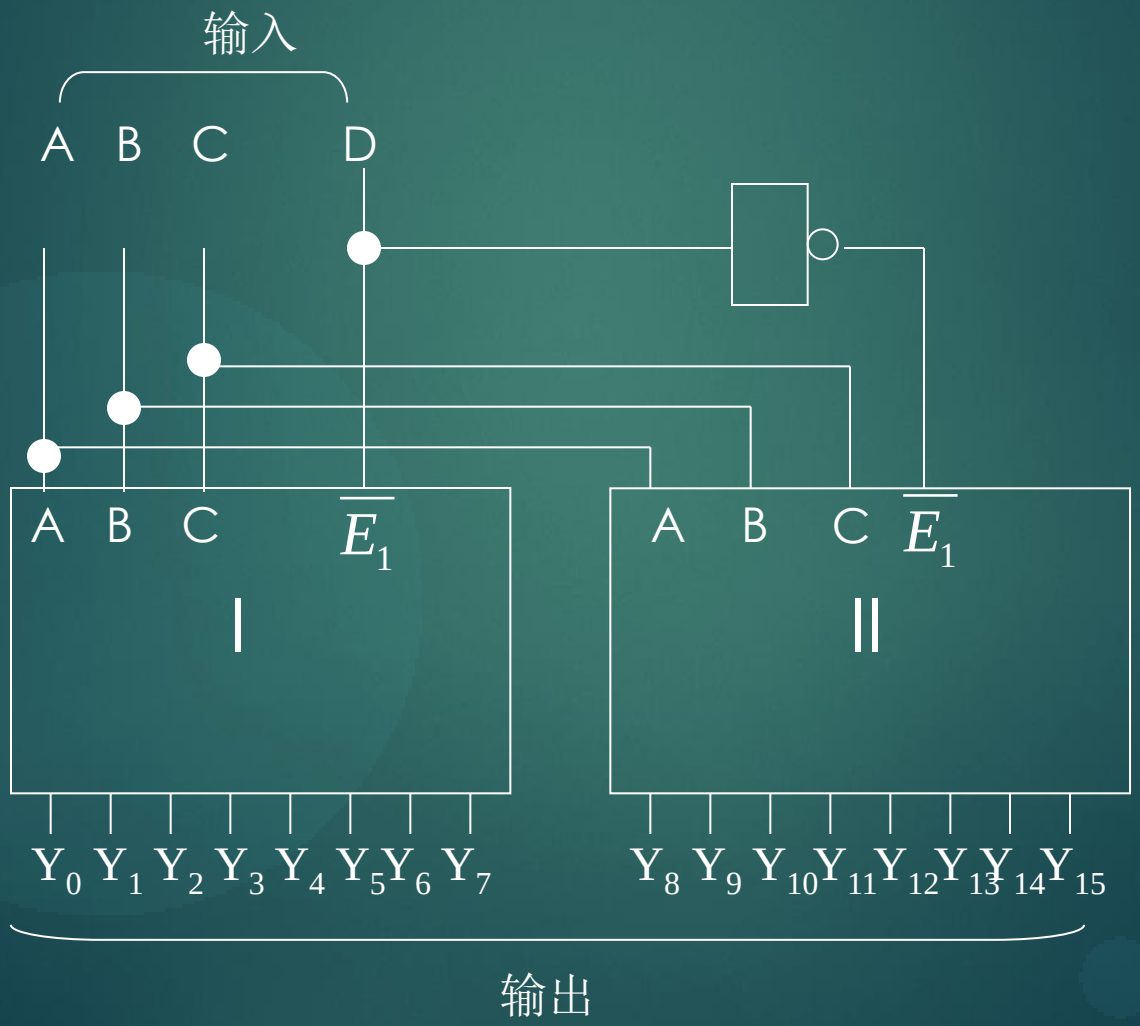
$\overline{E}$用作扩展（续）



5片2－4译码器构成4－16译码器。第一层的一个译码器用作选片。$\overline{E}$=0时，C D＝00时选中左边一片，译出Y_0…Y_3;依此类推。

$\overline{E}$用作扩展（续）

3-8译码器扩展为4-16译码器

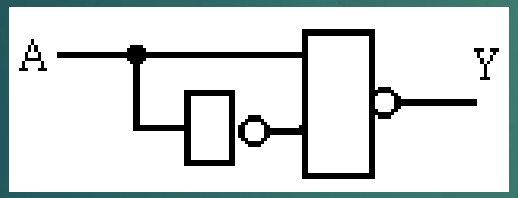


$\overline{E}$ 用作选通（作用二）

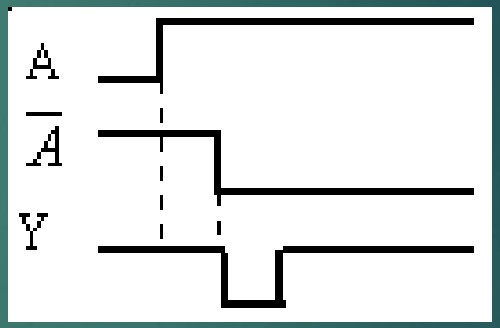
为什么需要选通？

针对门电路的传输延迟造成的竞争、冒险问题提出的。

二输入AND门（OR门）的输入为A和 $\overline{A}$时，$\overline{A}$滞后于A，则Y会出现尖峰信号。



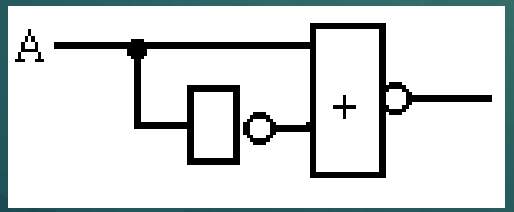
与非门上升沿有尖峰



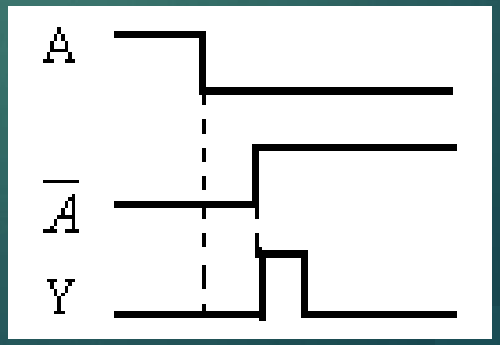
理想情况：

$Y = A\overline{A} = 1$

负向尖峰

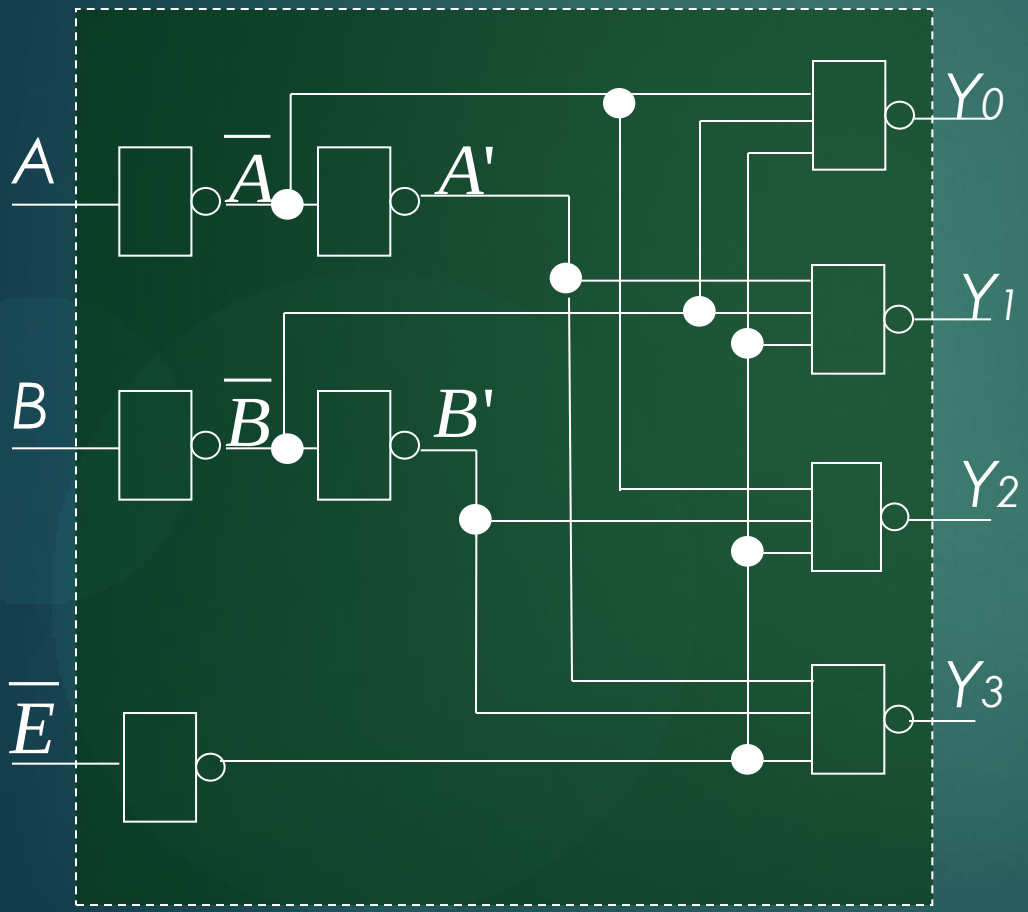


或非门下降沿有尖峰



正向尖峰

$\overline{E}$ 端用于选通（续）

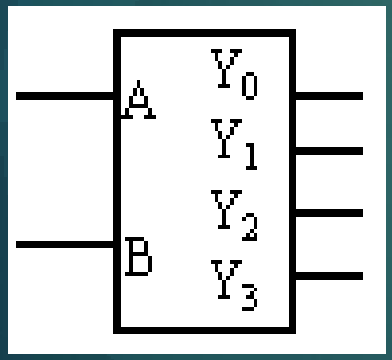


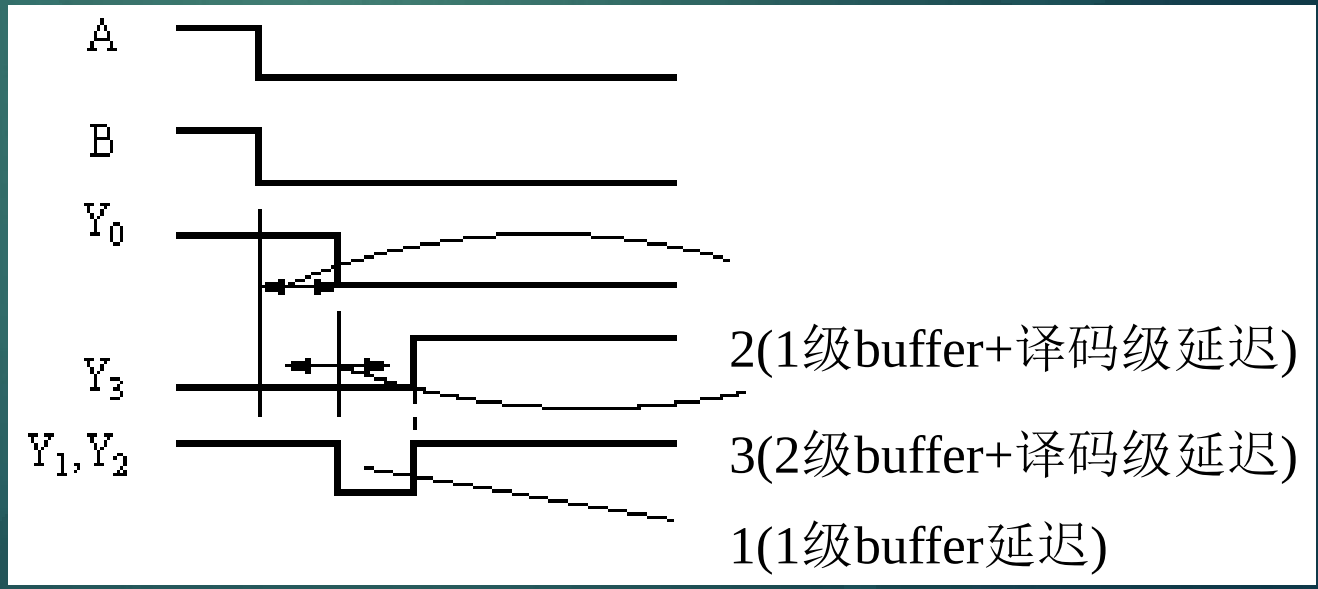
译码器中设置二级缓冲，目的是均衡负载，但是由于信号传输的延迟，会在输出端产生“0”重叠(Overlap)和尖峰信号(有些书中称为毛刺，英文词为：Spike, Glitch)。

为消除尖峰和重叠，增加了$\overline{E}$。

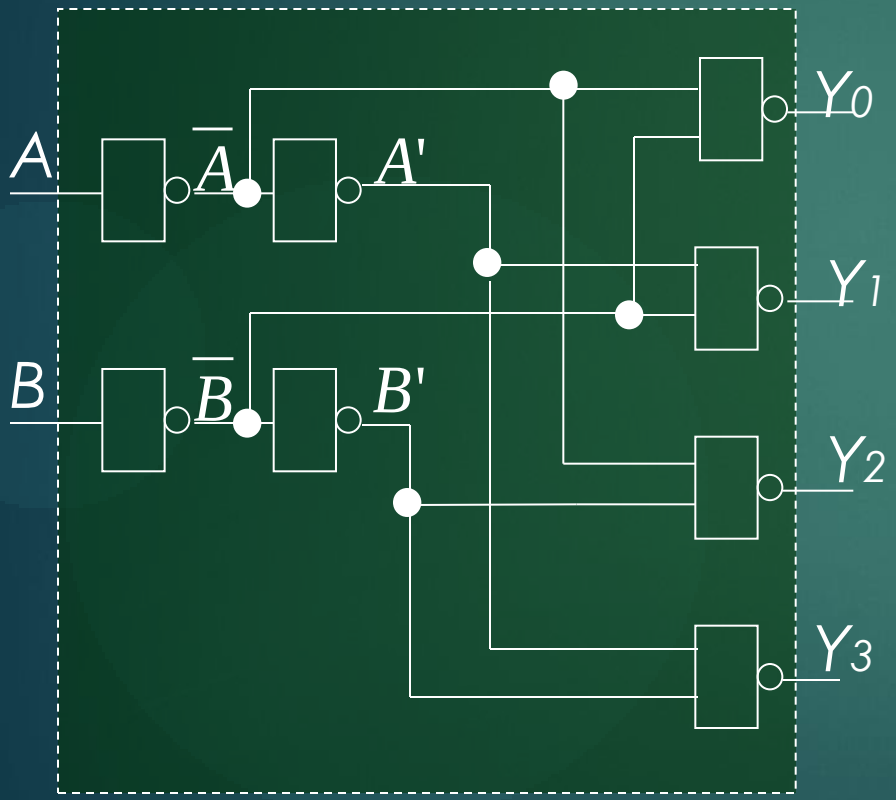
延迟产生尖峰

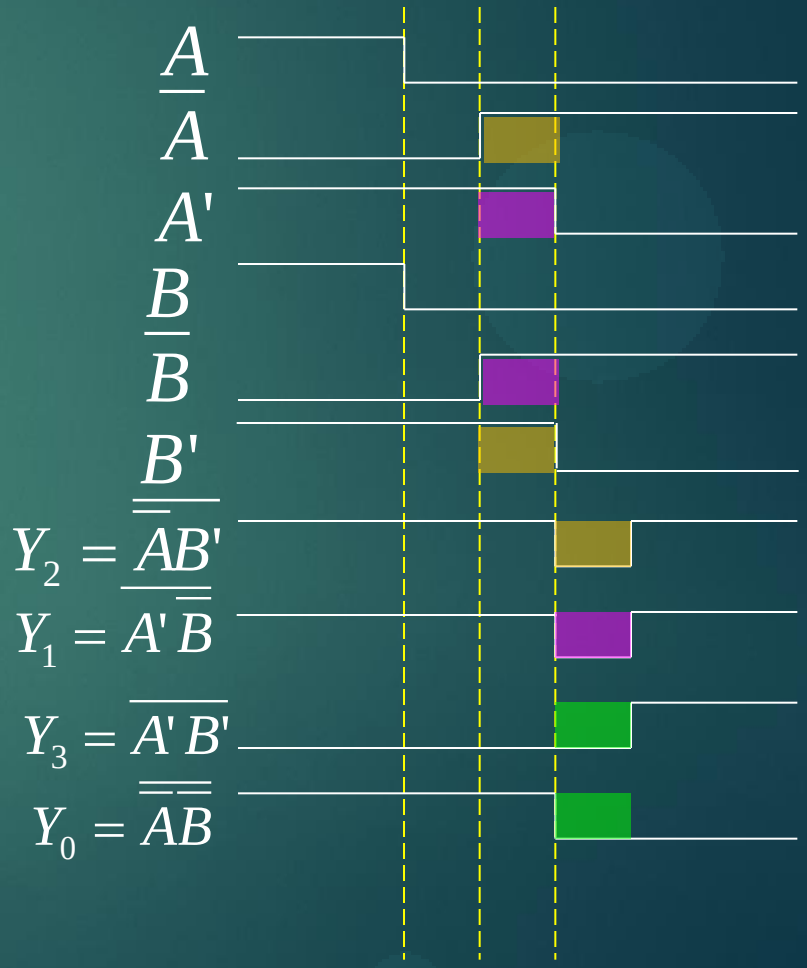
若A B同时到来(无偏移Skew)。从功能表上分析，A B从“11”变到“00”时，输出应从Y_3=0变成Y_0=0，Y_1Y_2保持为“1”。但是，由于门的传输延迟，造成Y_1，Y_2上出现了尖峰，同时，Y_3,Y_0有一段时间同时为“0”，即零重叠。





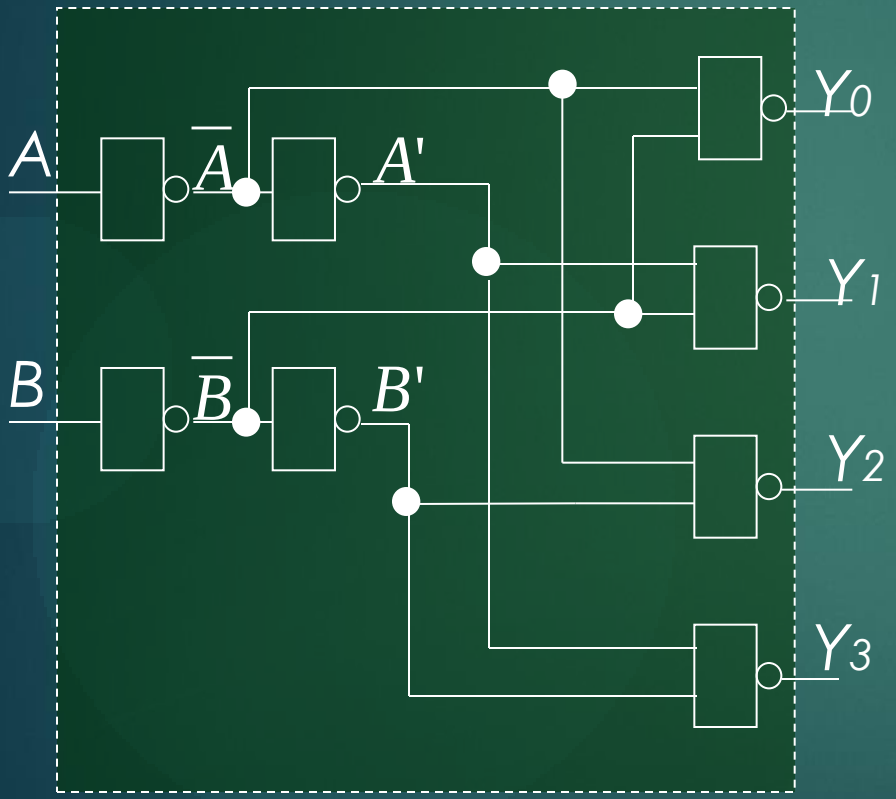
延迟产生尖峰

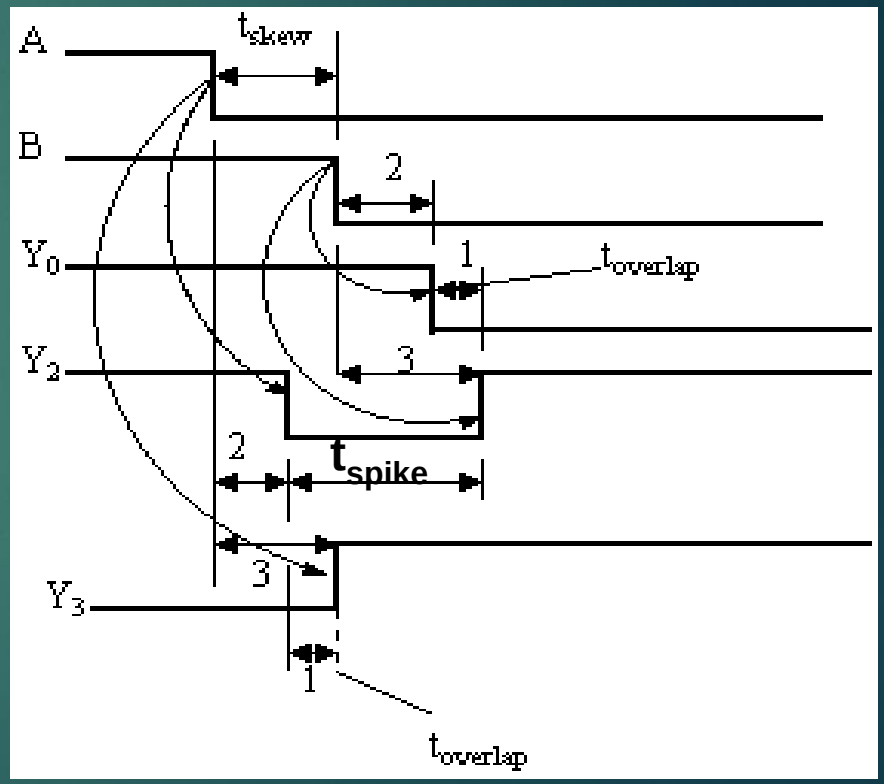

A
$\overline{A}$
A'
B
$\overline{B}$
B'
Y_0
Y_1
Y_2
Y_3


$\overline{A}$
A
A'
$\overline{B}$
B
B'
$Y_2 = \overline{AB'}$
$Y_1 = \overline{A'}\,\overline{B}$
$Y_3 = \overline{A'\,B'}$
$Y_0 = \overline{\overline{A}\,\overline{B}}$

延迟产生尖峰

当A B从“11”变到“00”时，输出应从Y_3=0变成Y_0=0。
假设A B不能同时到来，存在偏移(Skew)，导致尖峰信号更宽。



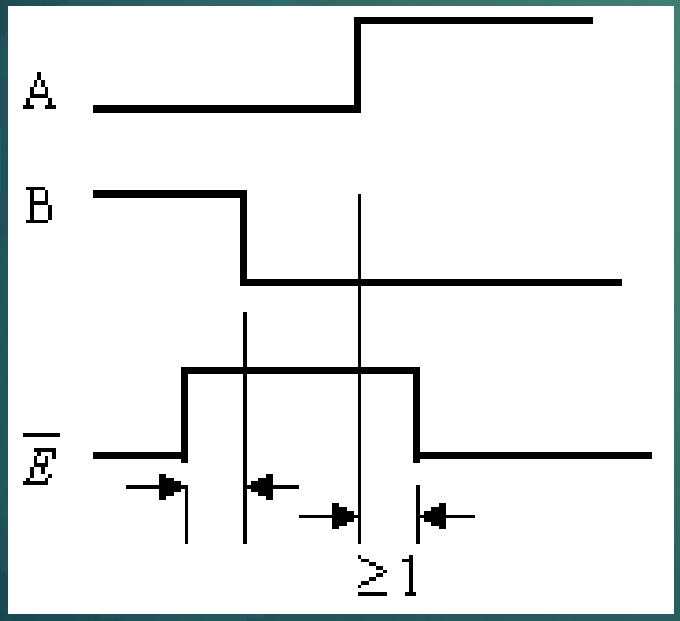


$t_{overlap}$=1级延迟

t_{spike} = t_{skew}+1级延迟
t_{spike}加宽、两处出现零重叠

$\overline{E}$ 端覆盖输入的变化

在A B变化期间，输出是不稳定的，可能会出现尖峰信号。加一个能覆盖输入变化的正脉冲($\overline{E}$＝1)，使得A B变化期间强制Y_0-Y_3=1，既可消除输出端的干扰。

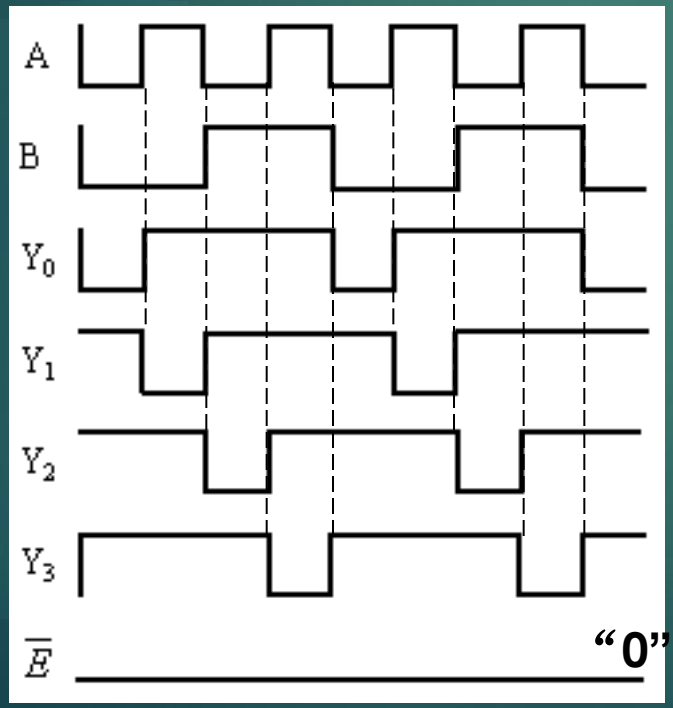


抑制尖峰和零重叠的正信号应提前（或同时）于译码器的变量输入变化前到来，正信号撤除应滞后于变量输入的变化(至少滞后1级缓冲的延迟)。
$\overline{E}$也不能太宽，否则速度会慢。

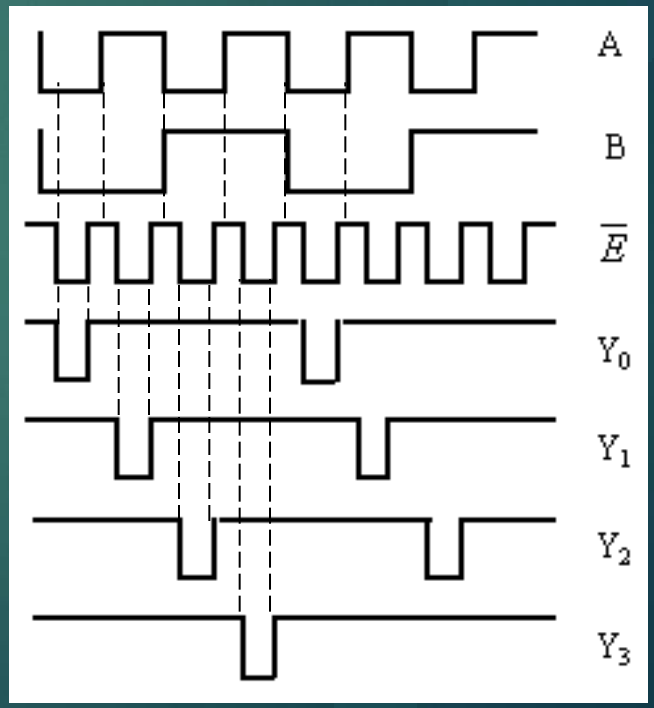
$\overline{E}$ 端功能

使用$\overline{E}$来抑制零重叠和尖峰，译码器的输出波形变窄了.

不使用$\overline{E}$



使用$\overline{E}$



有多个使能端的译码器件

器件一、编号：74LS(HCT，HC）138：

功能： 3-8 译码器（3个使能端）

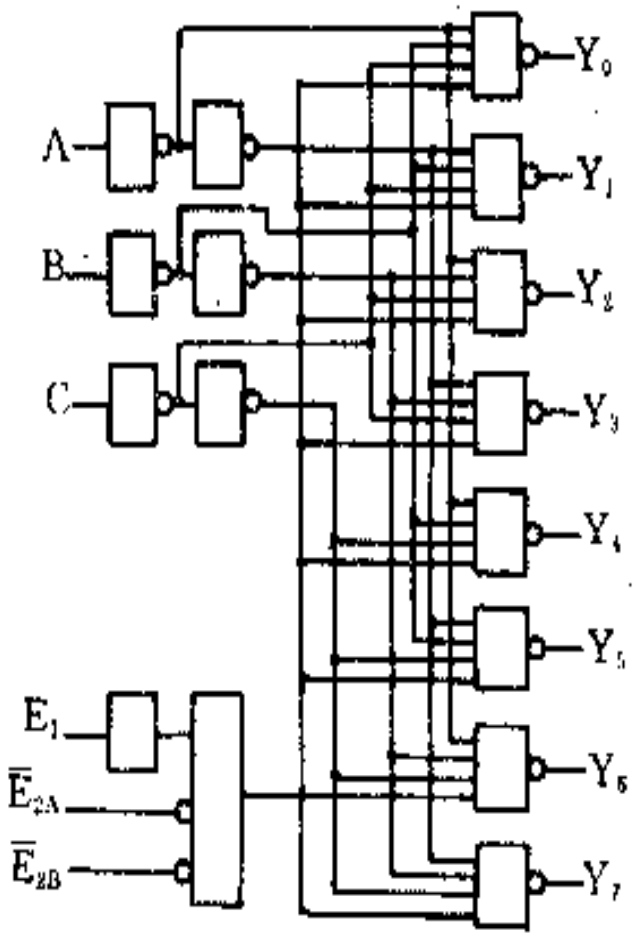
器件二、编号：74 LS (HCT，HC）154

功能4-16 译码器（2个使能端）

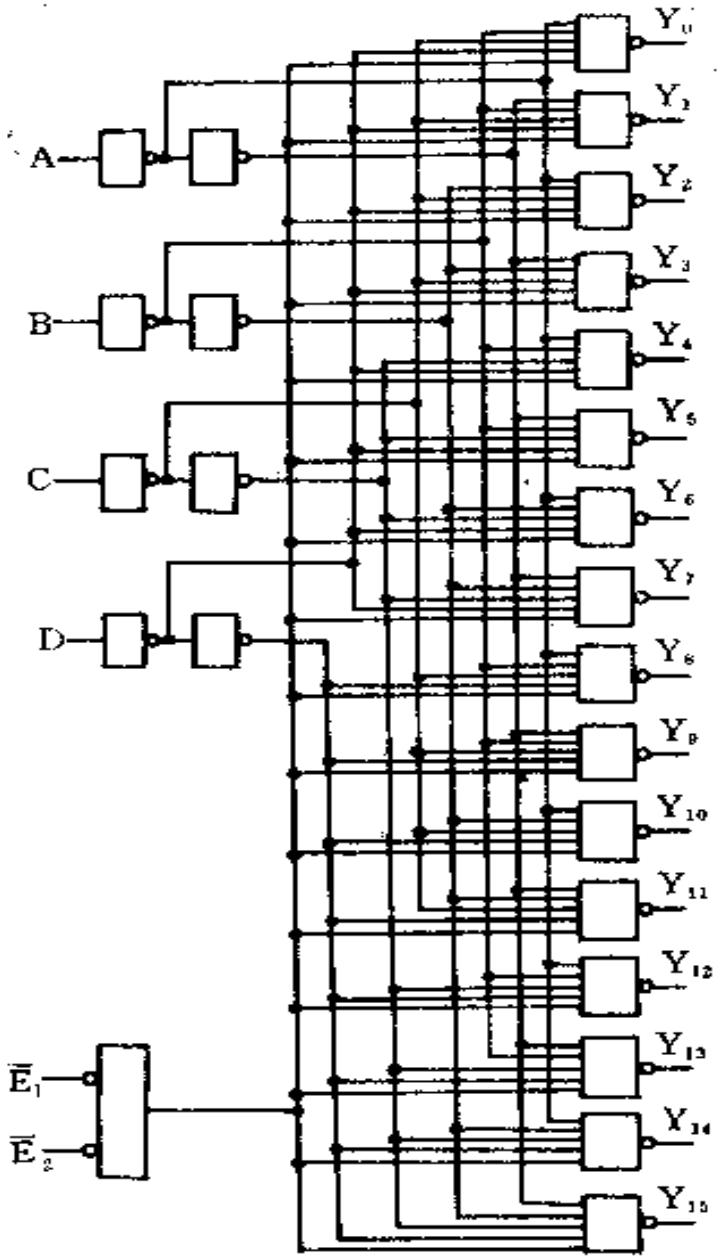
(前面介绍的器件型号为:74 x x 139 双2-4译码器)

功能表

输入					输出							
E_1	$\overline{E}_{2A}+\overline{E}_{2B}$	A	B	C	Y_0	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7
×	1	×	×	×	1	1	1	1	1	1	1	1
0	×	×	×	×	1	1	1	1	1	1	1	1
1	0	0	0	0	0	1	1	1	1	1	1	1
1	0	1	0	0	1	0	1	1	1	1	1	1
1	0	0	1	0	1	1	0	1	1	1	1	1
1	0	1	1	0	1	1	1	0	1	1	1	1
1	0	0	0	1	1	1	1	1	0	1	1	1
1	0	1	0	1	1	1	1	1	1	0	1	1
1	0	0	1	1	1	1	1	1	1	1	0	1
1	0	1	1	1	1	1	1	1	1	1	1	0



4-2　三输入变量译码器的逻辑图



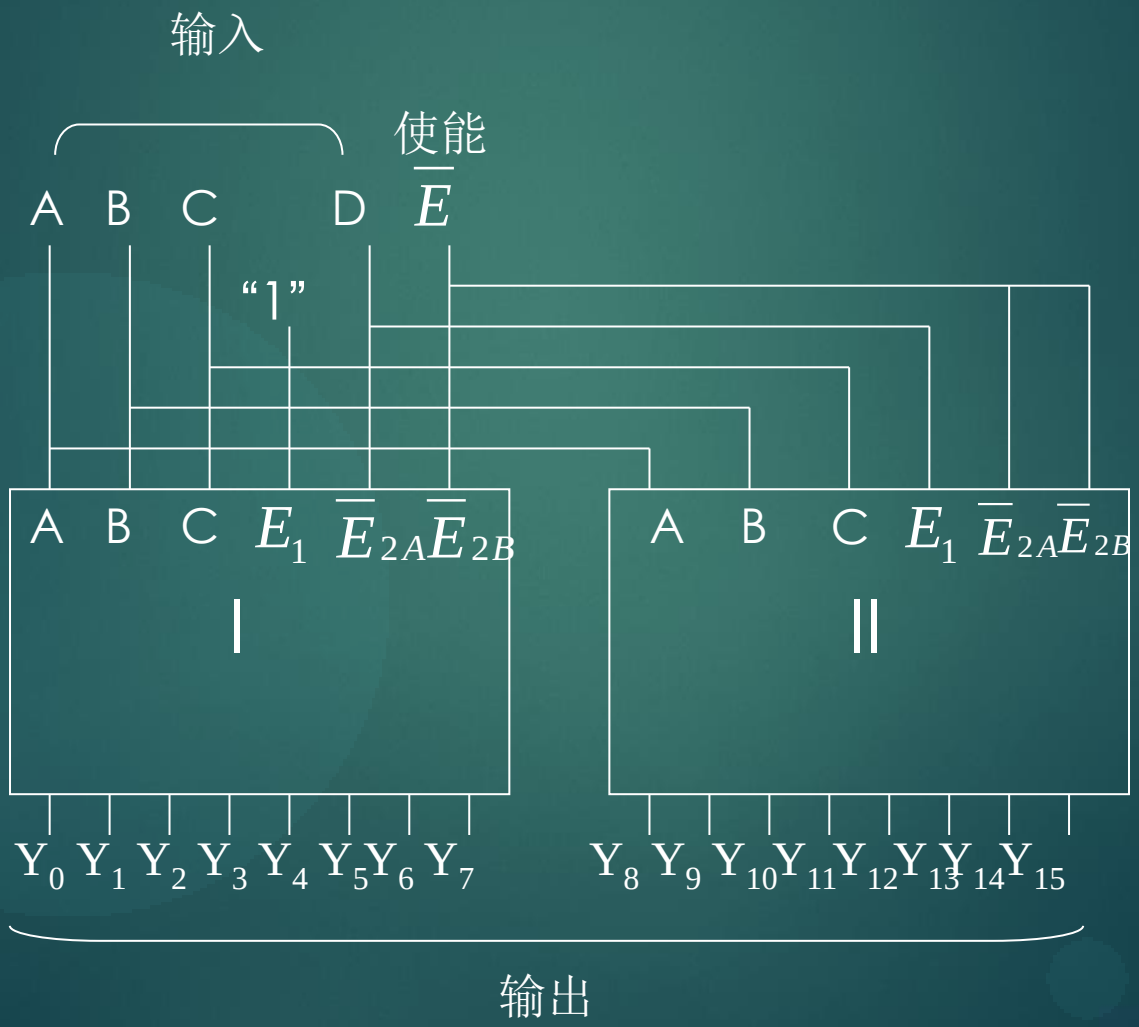
功　　能　　表

$\overline{E}_1$	$\overline{E}_2$	A	B	C	D	Y_0	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7	Y_8	Y_9	Y_{10}	Y_{11}	Y_{12}	Y_{13}	Y_{14}	Y_{15}
0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0	0	1	0	0	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0	0	0	1	0	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1
0	0	1	1	0	0	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1
0	0	0	0	1	0	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1
0	0	1	0	1	0	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1
0	0	0	1	1	0	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1
0	0	1	1	1	0	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1
0	0	0	0	0	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1
0	0	1	0	0	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1
0	0	0	1	0	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1
0	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1
0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1
0	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1
0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1
0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	×	×	×	×	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	0	×	×	×	×	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	×	×	×	×	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

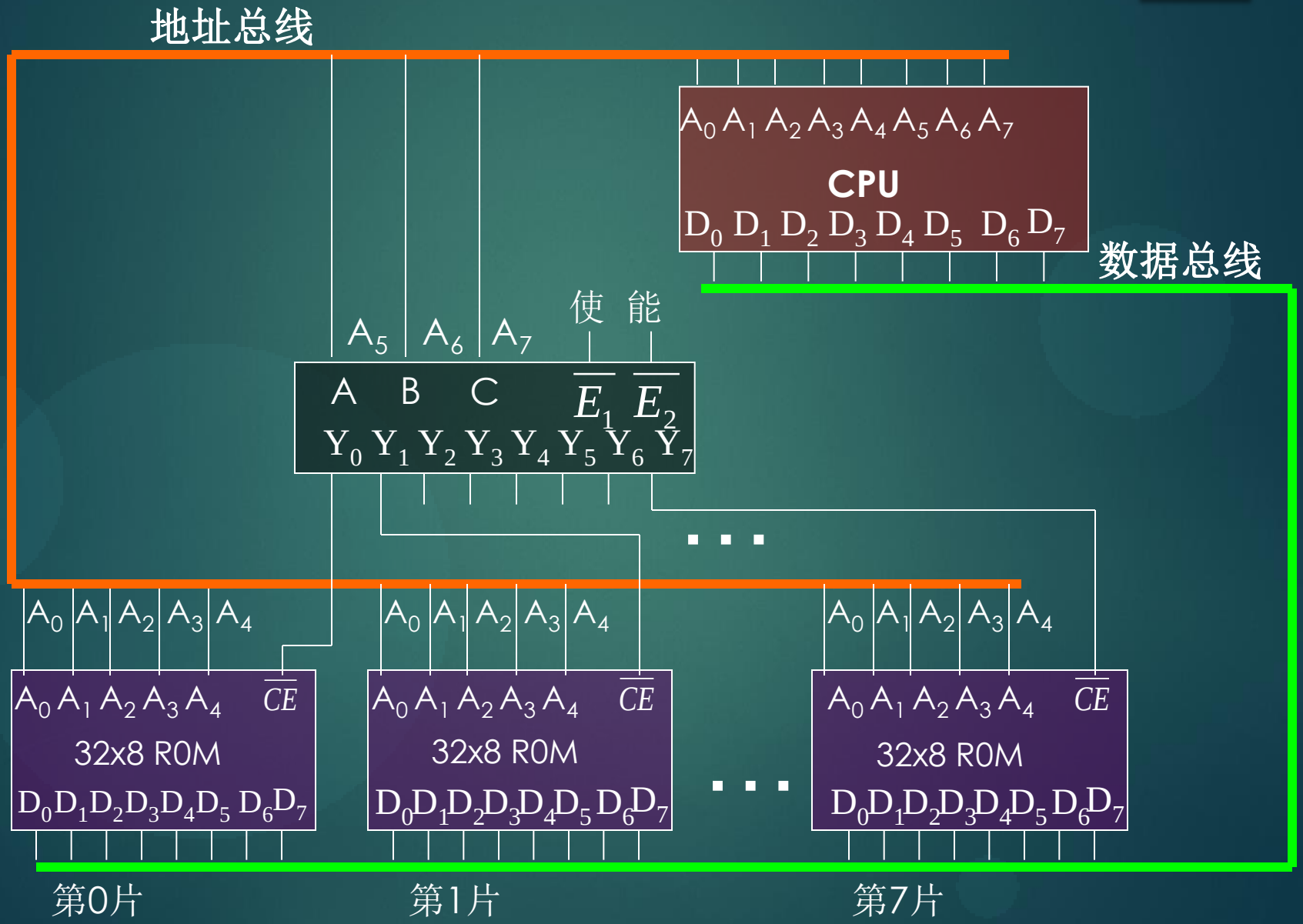
图 4-3　四输入变量译码器逻辑图

$\overline{E}$用作扩展（续）

具有多个使能端的3-8译码器扩展为4-16译码器



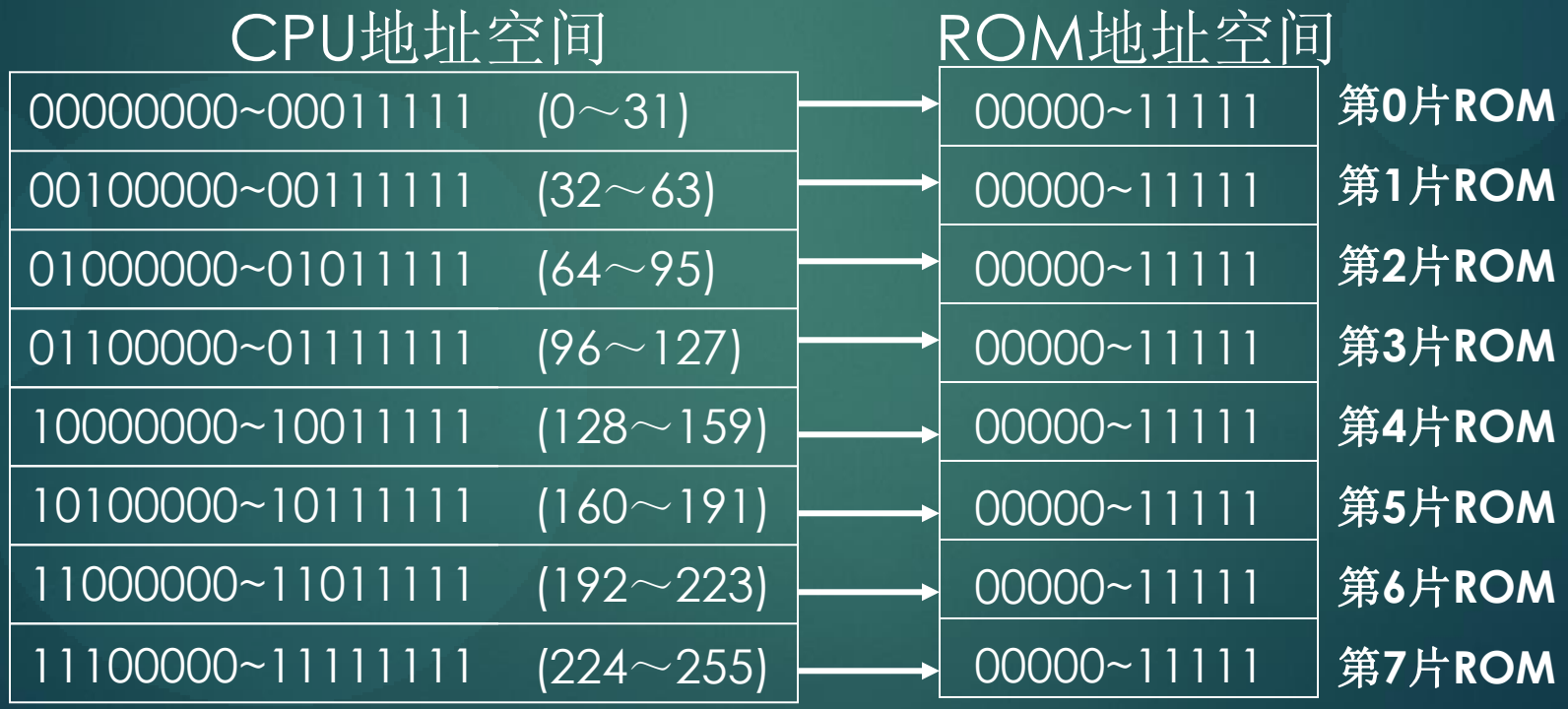
用3—8译码器分配地址区



用3—8译码器分配地址区（续）

- CPU的地址空间：A_7~A_0 共有256个地址空间
- 每个ROM有32个地址空间

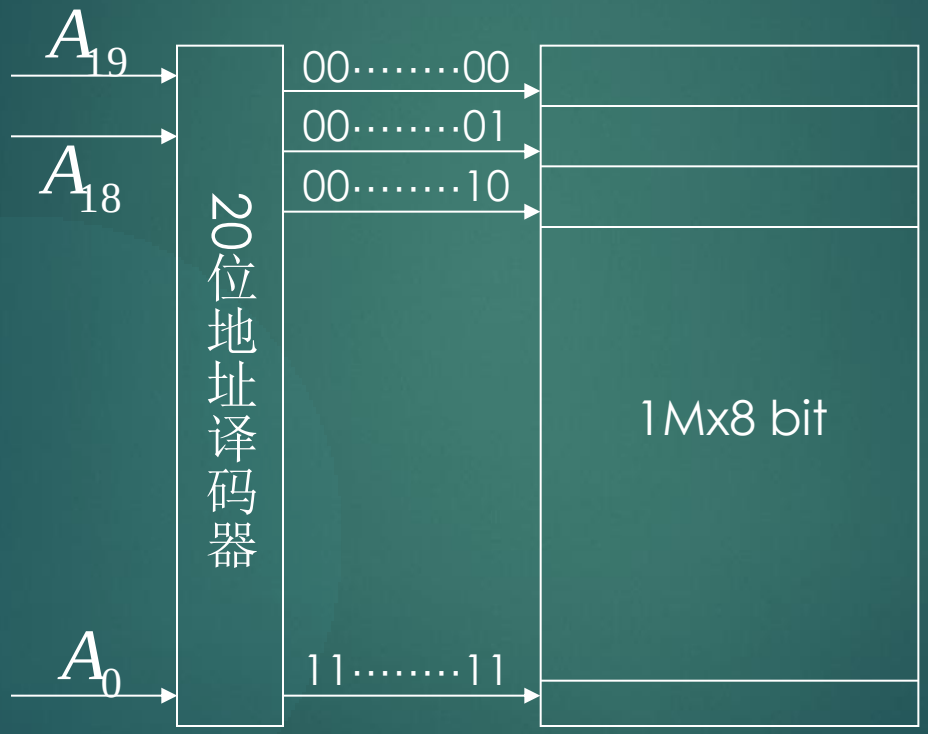
地址空间的对应关系如图：



用译码器完成地址分配

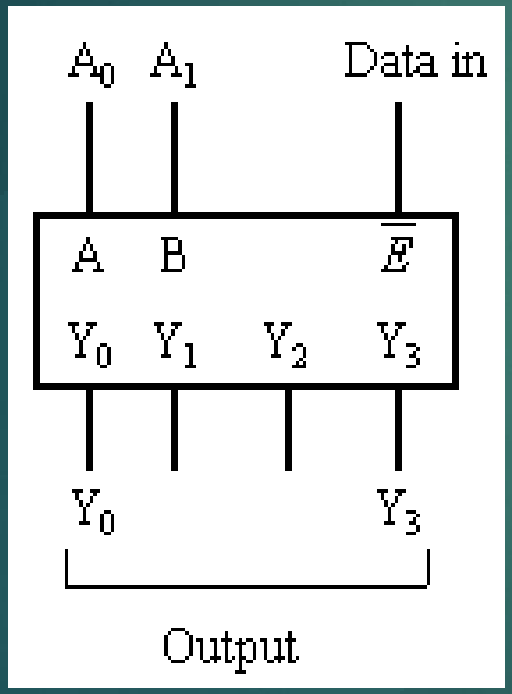
地址线有10位，可以表示2^{10}＝1K个地址空间；
地址线有20位，可以表示2^{20}＝1M个地址空间；
地址线有30位，可以表示2^{30}＝1G个地址空间；
32位地址可以表示4G地址；
16M存储器需要24位地址。

1Mx8存储器的地址译码结构



译码器的其他应用

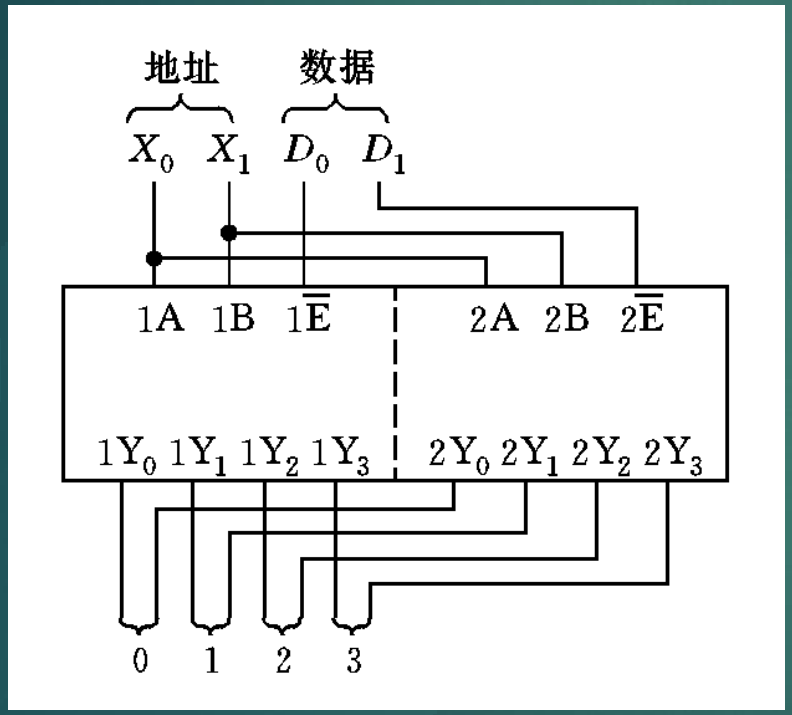
2-4译码器用作数据分配器（Demultiplexer）



数据分配：将输入数据在地址控制下连接到多个输出通道。

$\overline{E}$	A	B	Y_0	Y_1	Y_2	Y_3
0/1	0	0	0/1	1	1	1
0/1	1	0	1	0/1	1	1
0/1	0	1	1	1	0/1	1
0/1	1	1	1	1	1	0/1

两位数据分配器



地址		输出							
X_0	X_1	$1Y_0$	$2Y_0$	$1Y_1$	$2Y_1$	Y_2	$2Y_2$	$1Y_3$	$2Y_3$
0	0	D_0	D_1	1	1	1	1	1	1
1	0	1	1	D_0	D_1	1	1	1	1
0	1	1	1	1	1	D_0	D_1	1	1
1	1	1	1	1	1	1	1	D_0	D_1