3.4 二进制译码器的应用

典型的应用有以下几种：

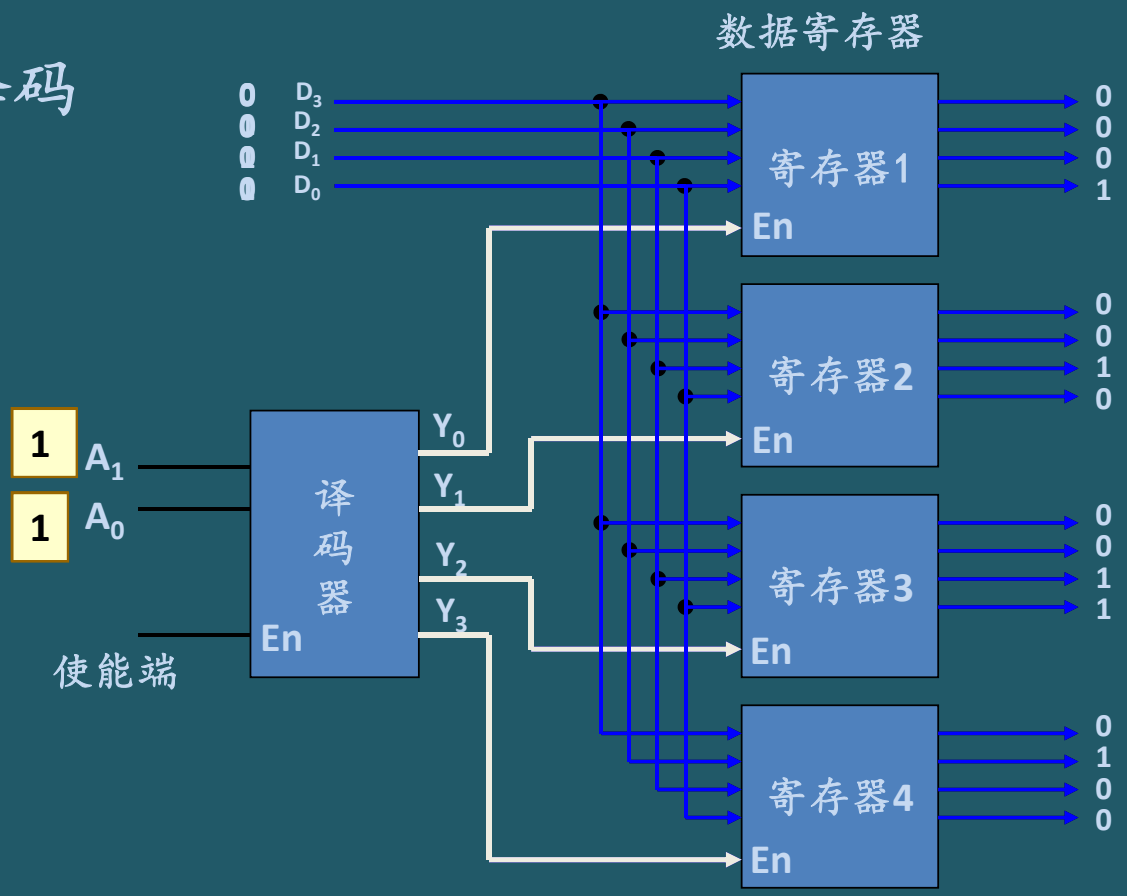
① 实现存储系统的地址译码；

② 实现逻辑函数；

③ 带使能端的译码器可用作数据分配器或脉冲分配器。

3.4.1 典型应用之一：

实现存储系统的地址译码

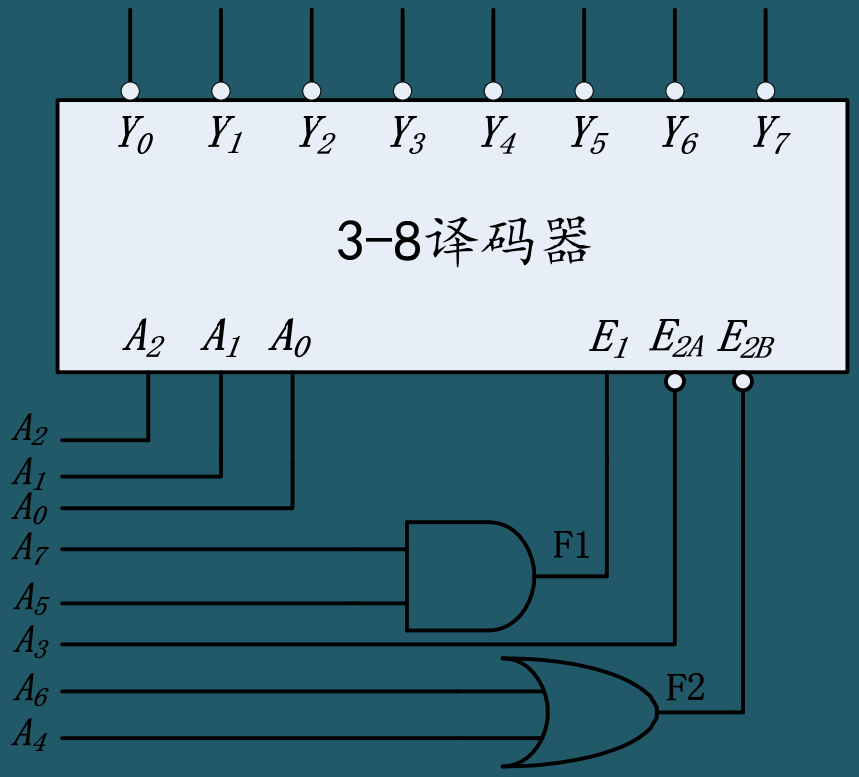


典型应用之一：

西安电子科技大学 XIDIAN UNIVERSITY

实现存储系统的地址译码

例1：分析下图所示Y0-Y7的译码地址。



$F1=1 \rightarrow A_5A_7=1 \rightarrow A_5=1 \quad A_7=1$

$F2=0 \rightarrow A_4+A_6=0 \rightarrow A_4=0 \quad A_6=0$

$A_3=0$

A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0
1	0	1	0	0	x	x	x

地址为：$(10100000)_2$-$(10100111)_2$

(A0H - A7H)

思考题：请用74138设计一地址译码电路，实现2E0-2E7的地址译码。

提示：

2E0H-2E7H，对应的地址线为：

A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0
1	0	1	1	1	0	0	0	0	0
1	0	1	1	1	0	0	0	0	1
1	0	1	1	1	0	0	0	1	0
1	0	1	1	1	0	0	0	1	1
1	0	1	1	1	0	0	1	0	0
1	0	1	1	1	0	0	1	0	1
1	0	1	1	1	0	0	1	1	0
1	0	1	1	1	0	0	1	1	1

3.4.2典型应用之二：实现组合逻辑函数

【例 2】 试用3-8译码器实现函数：

$$F_1 = \sum m(0,4,7)$$

$$F_2 = \sum m(1,2,3,5,6,7)$$

分析：因为当译码器的使能端有效时，每个输出 $Y_i = \overline{m}_i = M_i$ ，因此只要将函数的输入变量加至译码器的地址输入端，并在输出端辅以少量的门电路，便可以实现逻辑函数。

$$F_1 = \sum m(0,4,7)$$

$$F_2 = \sum m(1,2,3,5,6,7)$$

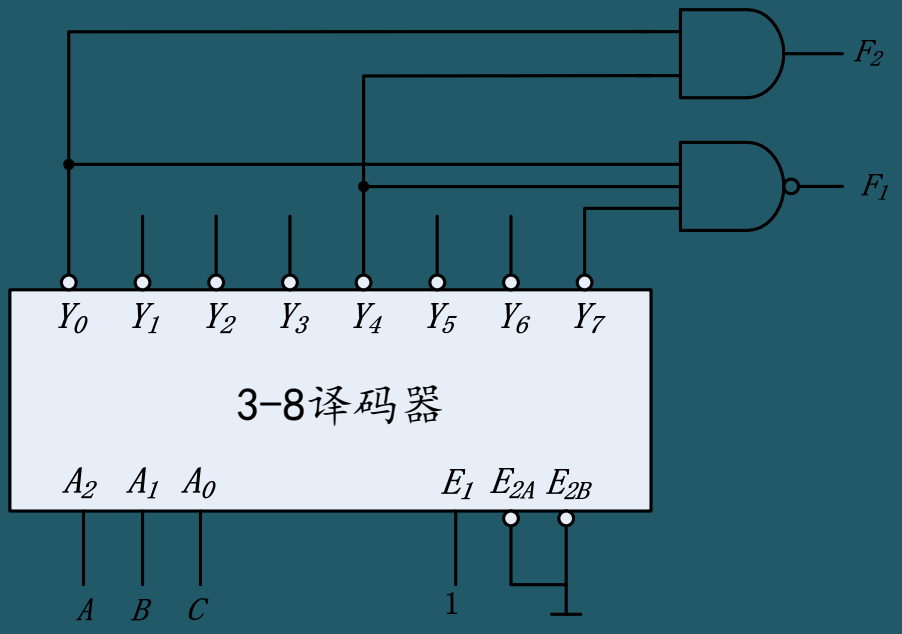
$$F_1 = m_0 + m_4 + m_7 = \overline{\overline{m}_0 \cdot \overline{m}_4 \cdot \overline{m}_7} = \overline{Y_0 \cdot Y_4 \cdot Y_7}$$

$$F_2 = m_1 + m_2 + m_3 + m_5 + m_6 + m_7$$

$$= M_0 \cdot M_4 = Y_0 \cdot Y_4$$

C \ AB	00	01	11	10
0	0	1	1	0
1	1	1	1	1

F_2



思考题1：用74138和门电路实现下面的逻辑函数。

$$f_1(a,b,c)=\sum m(1,2,4,5)$$

$$f_2(a,b,c)=\prod M(2,3,6,7)$$

思考题2：用74138和门电路设计一个一位二进制全加器。

3.4.3 二进制译码器的扩展

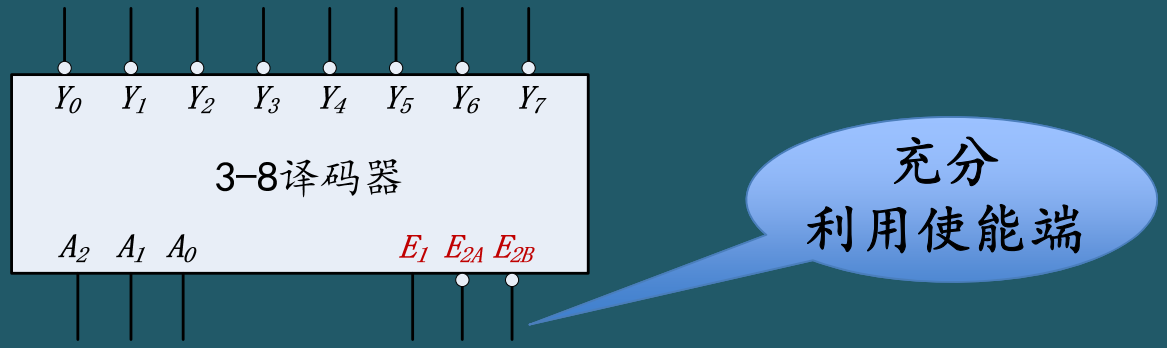
通常用译码器的使能端来实现二进制译码器的扩展

例1. 用3-8译码器实现4-16译码器。

分析：

3-8译码器：3个输入端　　8个输出端

4-16译码器：4个输入端　　16个输出端



二进制译码器的扩展

a	b	c	d	Y
0	0	0	0	Y_0
0	0	0	1	Y_1
0	0	1	0	Y_2
0	0	1	1	Y_3
0	1	0	0	Y_4
0	1	0	1	Y_5
0	1	1	0	Y_6
0	1	1	1	Y_7
1	0	0	0	Y_8
1	0	0	1	Y_9
1	0	1	0	Y_{10}
1	0	1	1	Y_{11}
1	1	0	0	Y_{12}
1	1	0	1	Y_{13}
1	1	1	0	Y_{14}
1	1	1	1	Y_{15}