ESc201 : Introduction to Electronics

Combinational Circuit Design

Dr. Y. S. Chauhan
Dept. of Electrical Engineering
IIT Kanpur

Design Flow

System Description

↓

Truth Table

↓

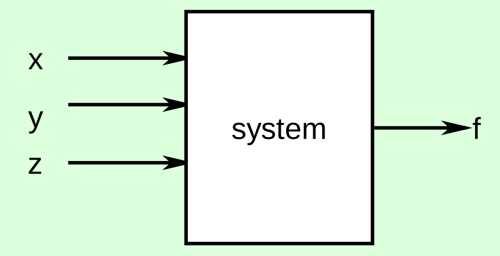
Boolean Expression

↓

Minimized Boolean Expression

↓

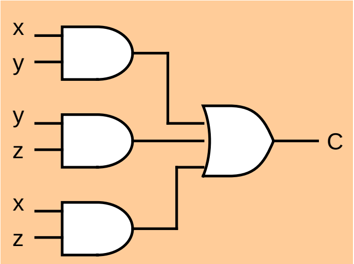
Gate Netlist



x	y	z	f
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

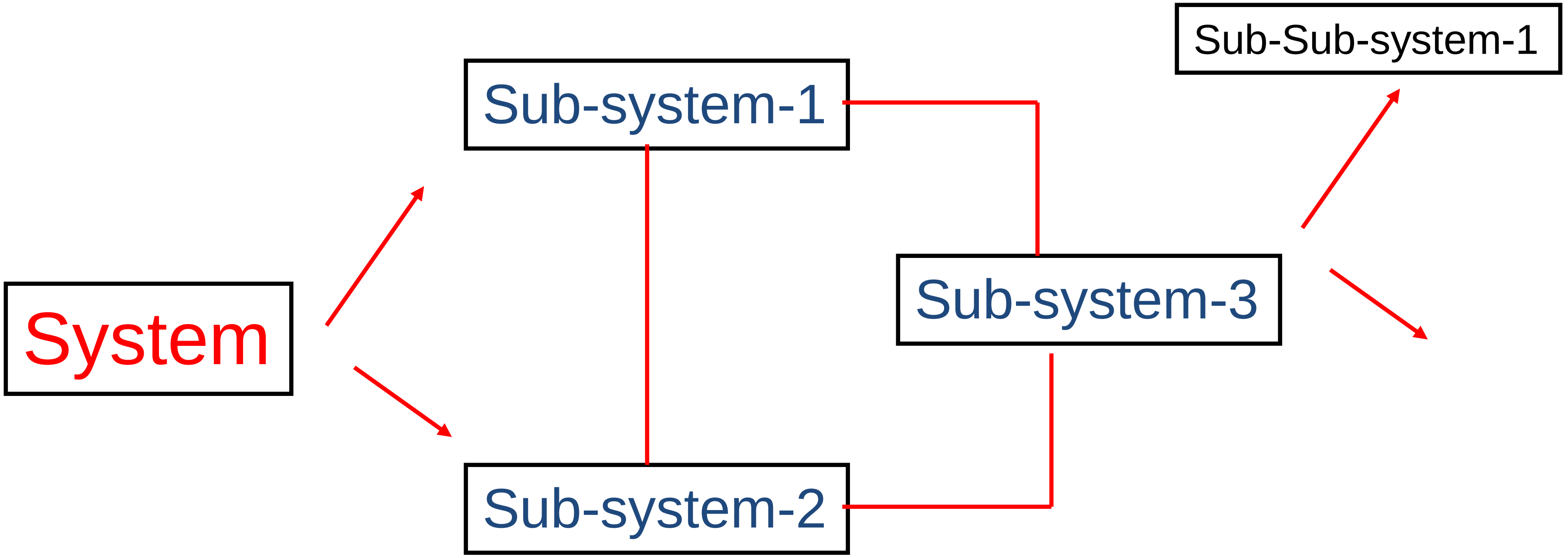
$$\mathrm{f} = \overline{x}.\overline{y}.z + \overline{x}.y.z + x.\overline{y}.z + x.y.z$$

$$\Rightarrow \mathrm{f} = \overline{x}\,.\overline{z} + x.z$$



This design approach becomes difficult to use

General Approach

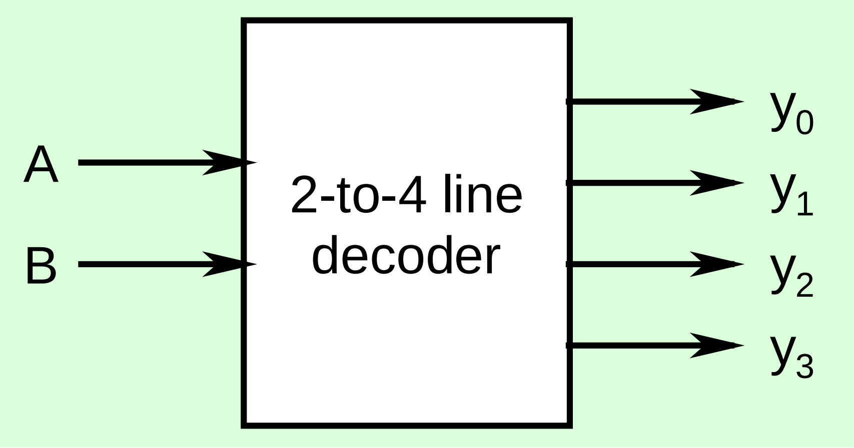


There are certain sub-systems or blocks that are used quite often such as :

1. **Decoders, Encoders**
2. **Multiplexers**
3. **Adder/Subtractors, Multipliers**
4. **Comparators**
5. **Parity Generators**
6. **……..**

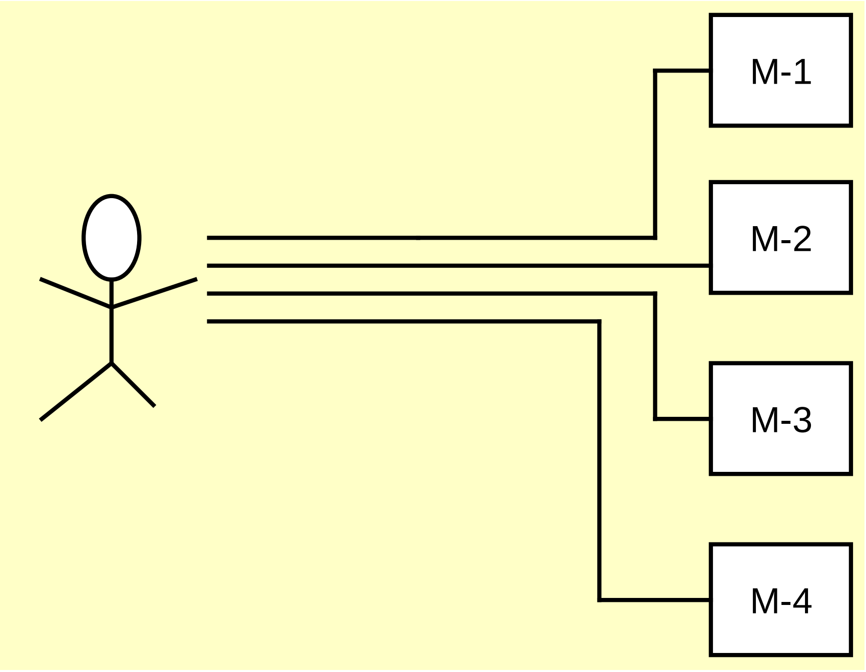
Decoders

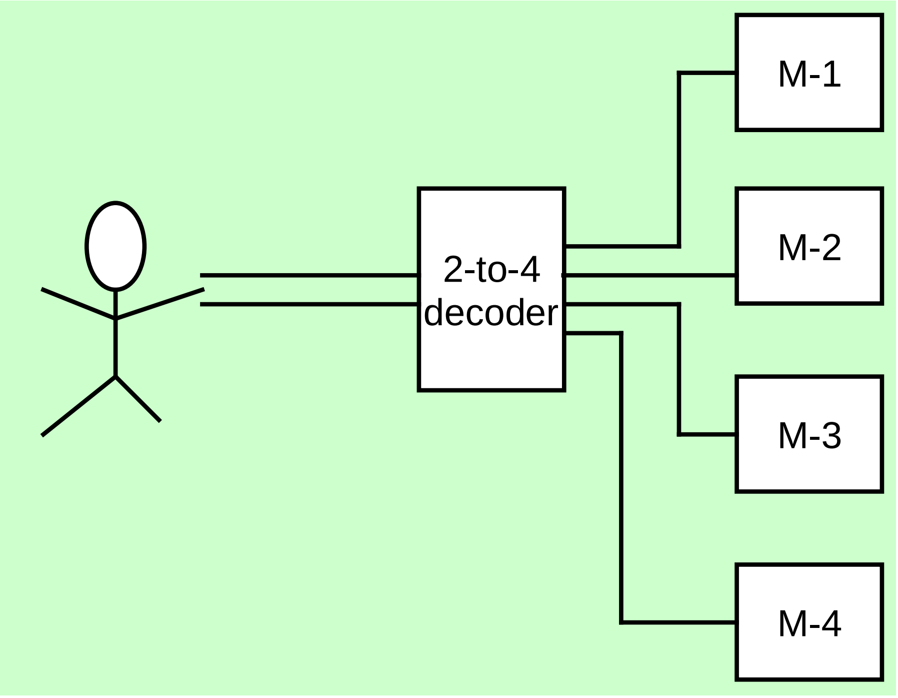
In general maps a smaller number of inputs to a larger set of outputs



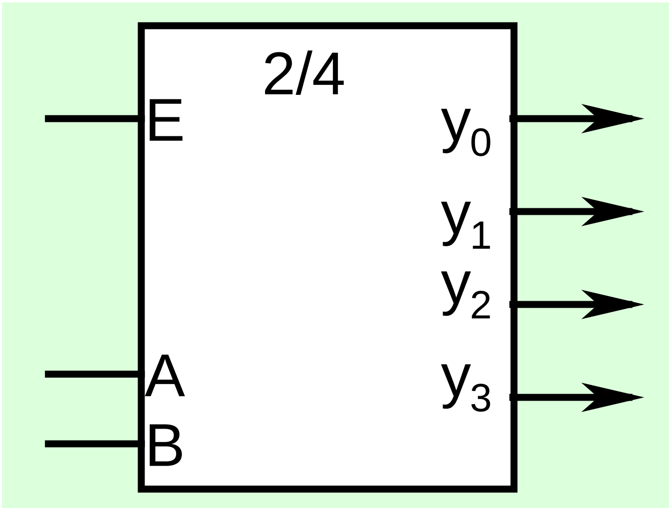
B	A	Y_0	Y_1	Y_2	Y_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

Example

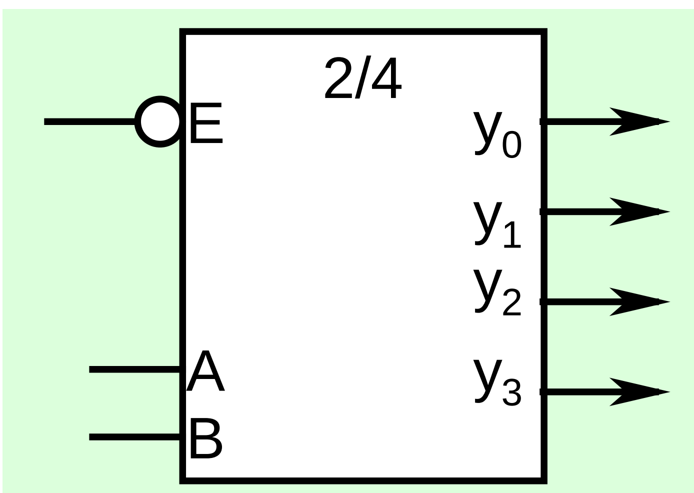




Decoder with Enable Input

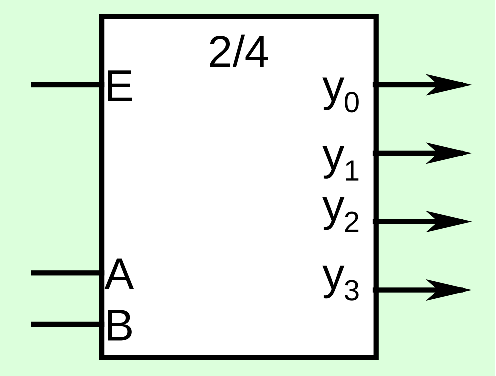


E	B	A	Y_0	Y_1	Y_2	Y_3
0	x	x	0	0	0	0
1	0	0	1	0	0	0
1	0	1	0	1	0	0
1	1	0	0	0	1	0
1	1	1	0	0	0	1



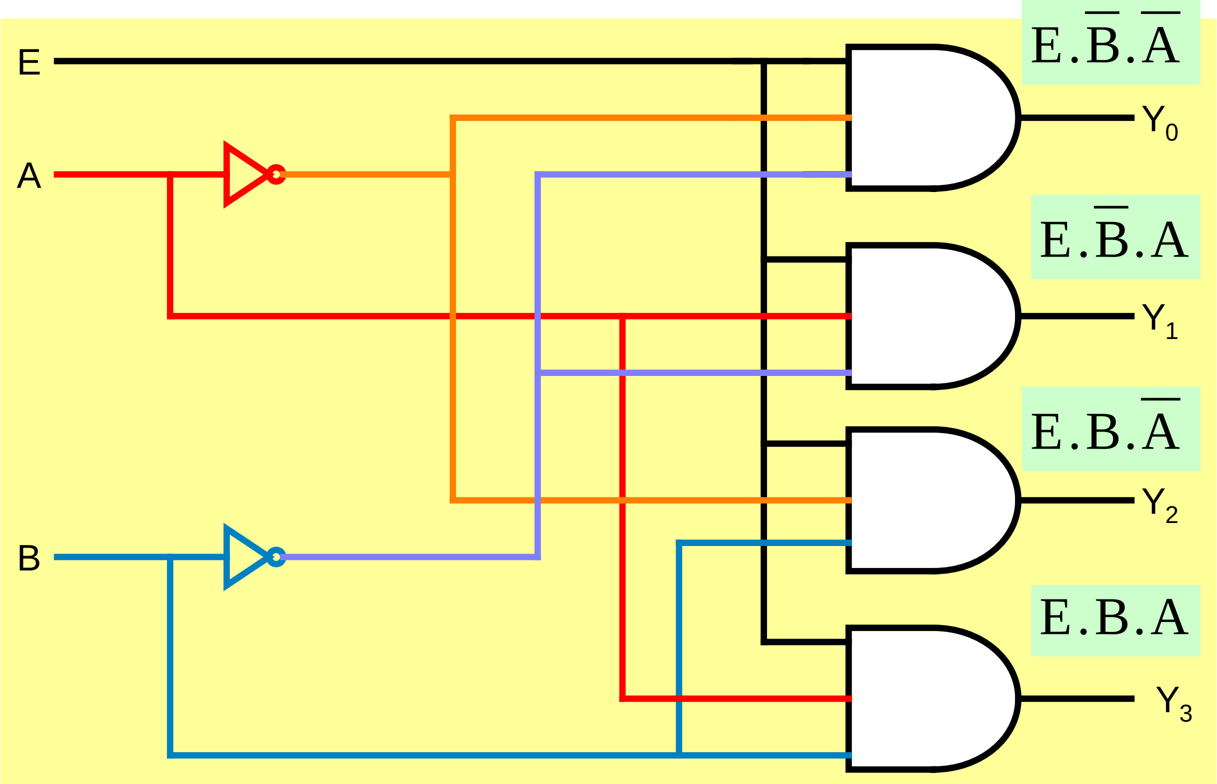
E	B	A	Y_0	Y_1	Y_2	Y_3
1	x	x	0	0	0	0
0	0	0	1	0	0	0
0	0	1	0	1	0	0
0	1	0	0	0	1	0
0	1	1	0	0	0	1

Decoder: gate Implementation



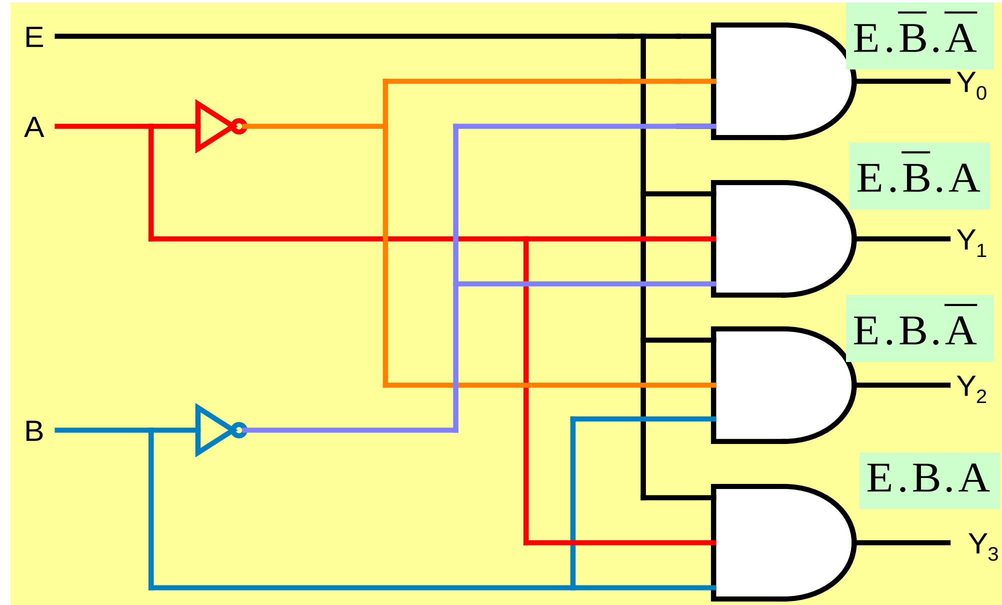
E	B	A	Y_0	Y_1	Y_2	Y_3
0	x	x	0	0	0	0
1	0	0	1	0	0	0
1	0	1	0	1	0	0
1	1	0	0	0	1	0
1	1	1	0	0	0	1

$$Y_0 = E.\overline{B}.\overline{A}\ ;\ Y_1 = E.\overline{B}.A\ ;\ Y_2 = E.B.\overline{A}\ ;\ Y_3 = E.B.A$$



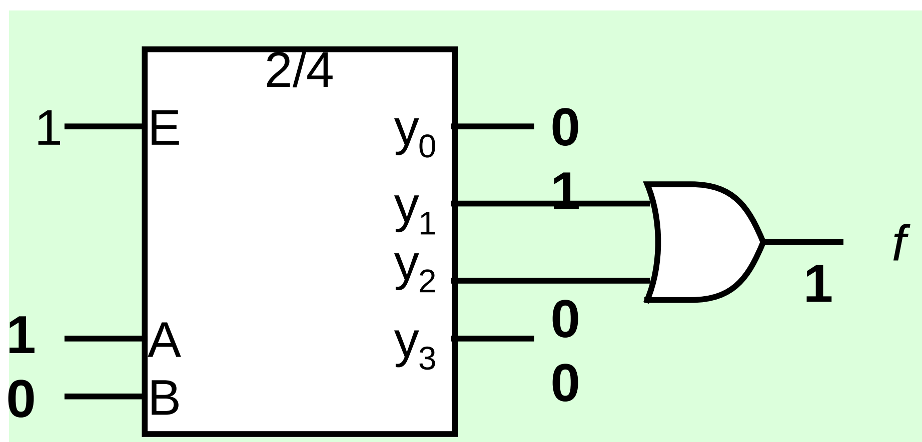
A n to 2^n decoder is a minterm generator

x	y	min term
0	0	$\overline{x} . \overline{y}$ m0
0	1	$\overline{x} . y$ m1
1	0	$x . \overline{y}$ m2
1	1	$x . y$ m3



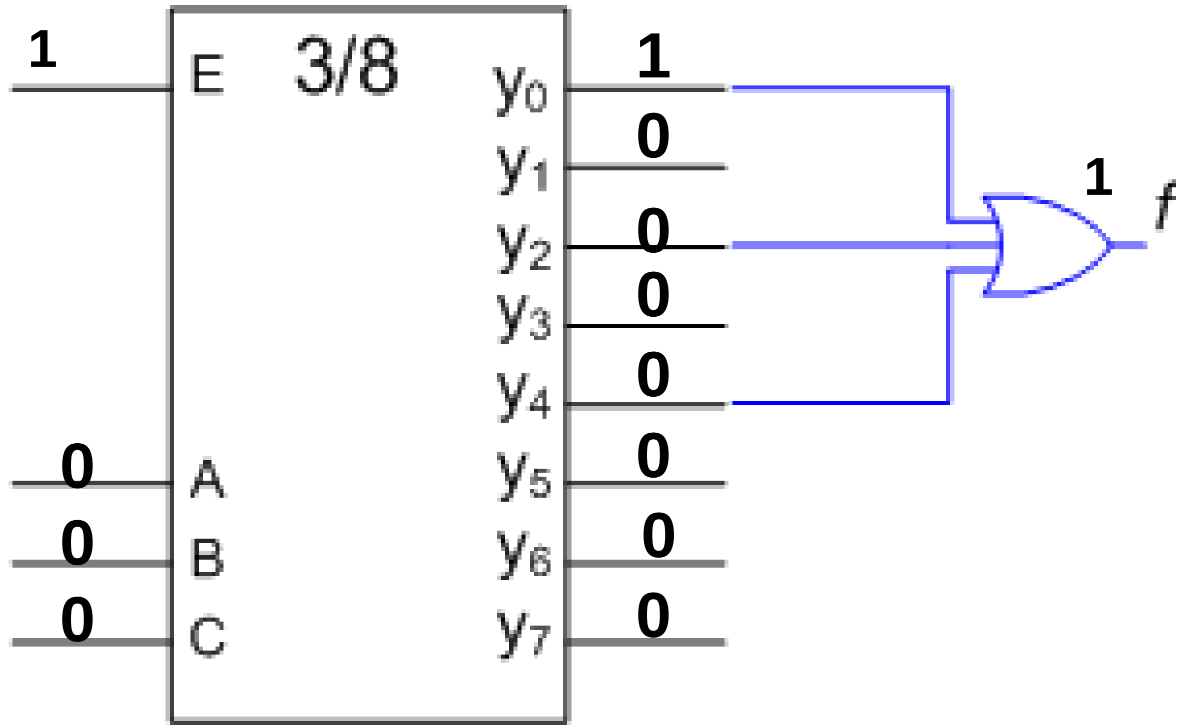
It can be used to implement any combinational circuit

B	A	f_1
0	0	0
0	1	1
1	0	1
1	1	0



Implementation of a 3-variable function with a 3-to-8 decoder

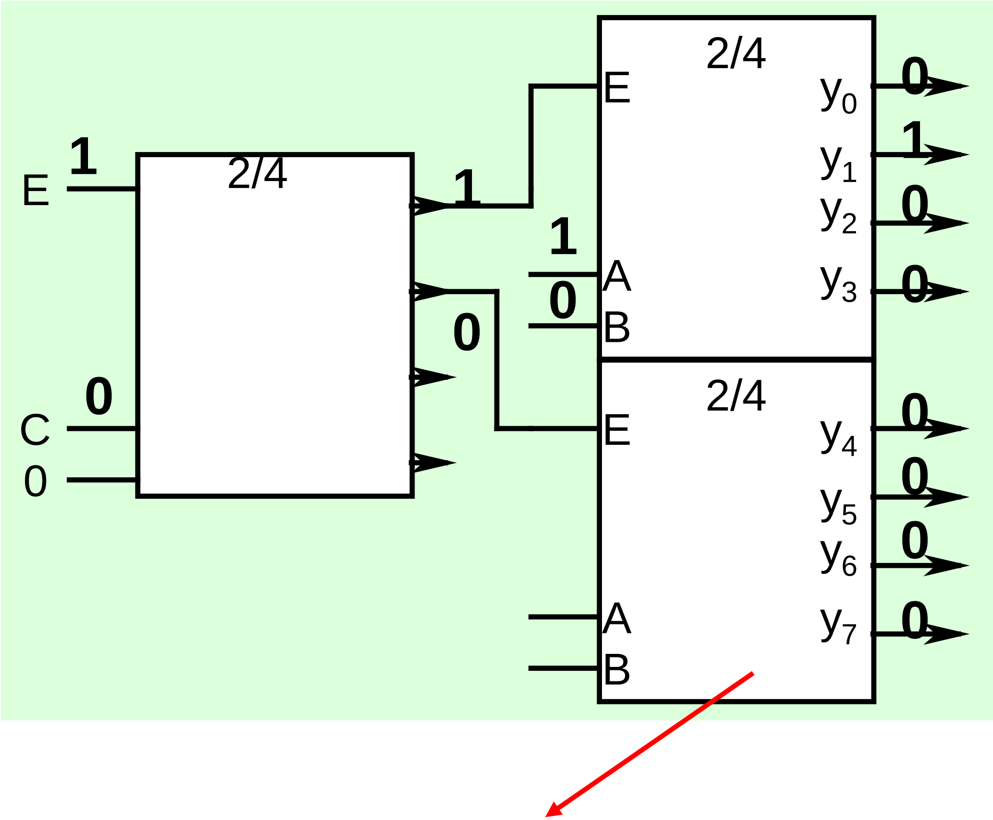
C	B	A	*f*
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0



Although it is easy to implement any combinational circuit with this method , it is often very inefficient in terms of gate utilization. Note that this method does not require any minimization.

3/8 decoder using 2/4 decoders

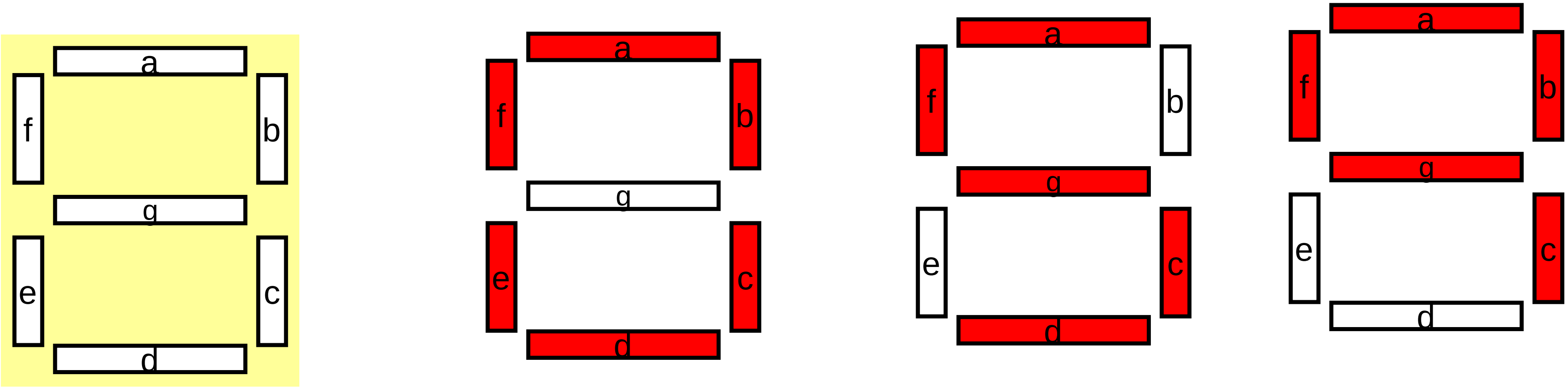
E	C	B	A	y_0	y_1	y_2	y_3	y_4	y_5	y_6	y_7
0	x	x	x	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	0	0	0	0
1	0	0	1	0	1	0	0	0	0	0	0
1	0	1	0	0	0	1	0	0	0	0	0
1	0	1	1	0	0	0	1	0	0	0	0
1	1	0	0	0	0	0	0	1	0	0	0
1	1	0	1	0	0	0	0	0	1	0	0
1	1	1	0	0	0	0	0	0	0	1	0
1	1	1	1	0	0	0	0	0	0	0	1

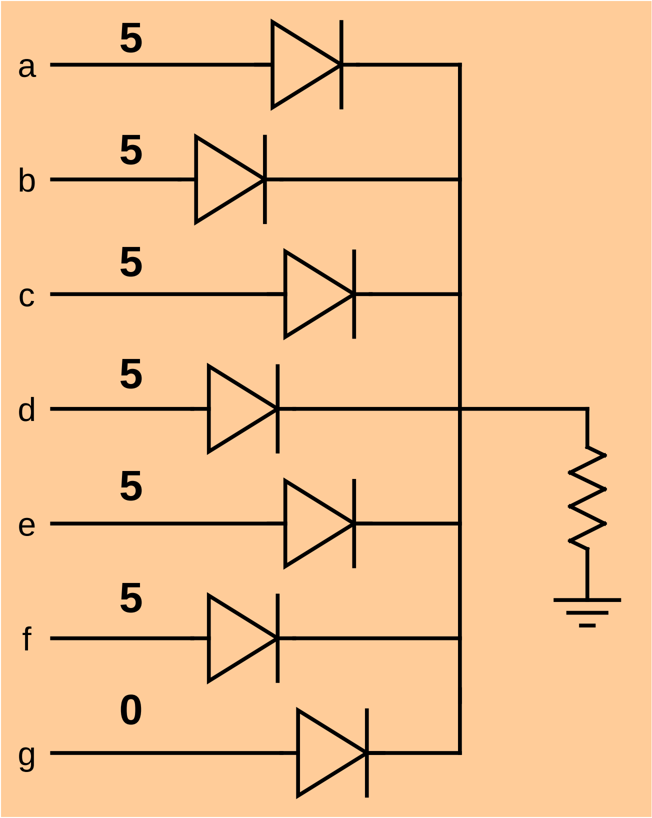


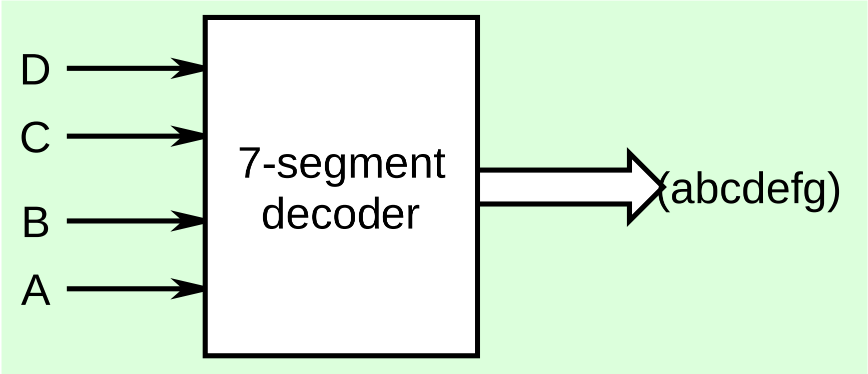
E	B	A	Y_0	Y_1	Y_2	Y_3
0	x	x	0	0	0	0
1	0	0	1	0	0	0
1	0	1	0	1	0	0
1	1	0	0	0	1	0
1	1	1	0	0	0	1

How many 2/4 decoders are required to implement a 4/16 decoder ?

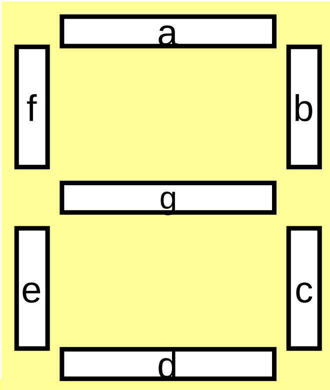
Seven segment decoder

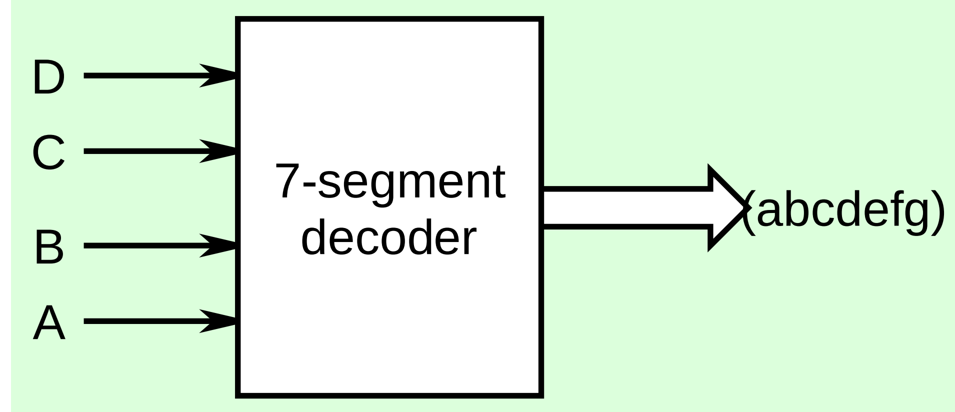






Seven segment decoder



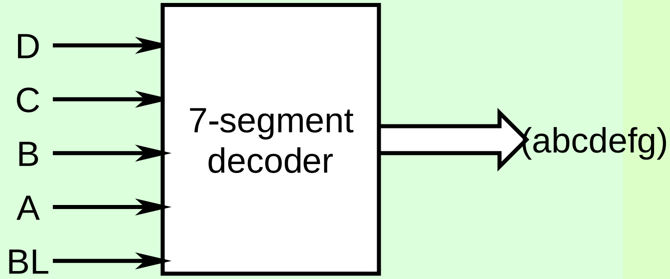


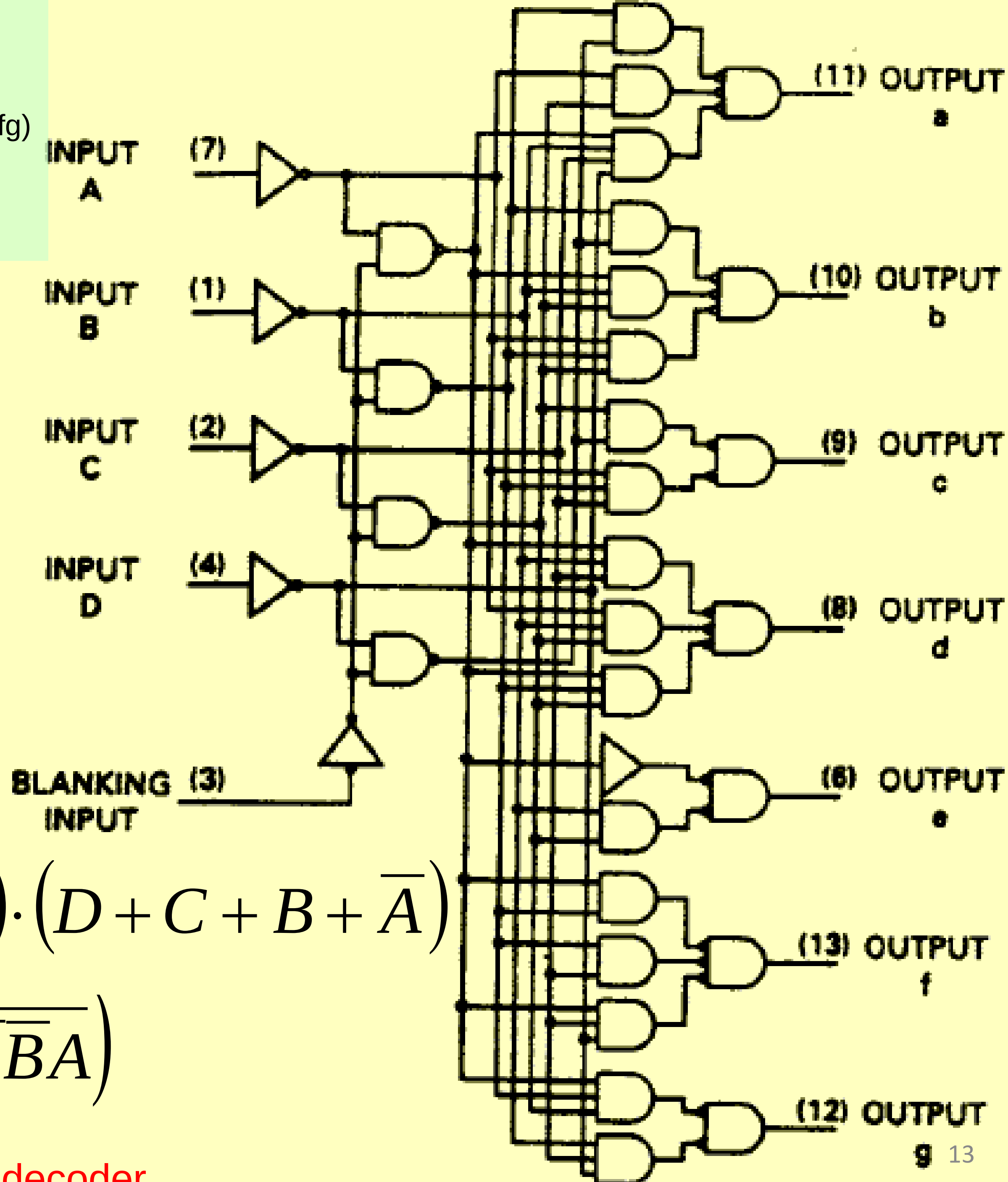
Dec or Function	Input					Output						
	D	C	B	A	BI	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	1	0
1	0	0	0	1	1	0	1	1	0	0	0	0
2	0	0	1	0	1	1	1	0	1	1	0	1
3	0	0	1	1	1	1	1	1	1	0	0	1
4	0	1	0	0	1	0	1	1	0	0	1	1
5	0	1	0	1	1	1	0	1	1	0	1	1
6	0	1	1	0	1	0	0	1	1	1	1	1
7	0	1	1	1	1	1	1	1	0	0	0	0
8	1	0	0	0	1	1	1	1	1	1	1	1
9	1	0	0	1	1	1	1	1	0	0	1	1
10	1	0	1	0	1	0	0	0	1	1	0	1
11	1	0	1	1	1	0	0	1	1	0	0	1
12	1	1	0	0	1	0	1	0	0	0	1	1
13	1	1	0	1	1	1	0	0	1	0	1	1
14	1	1	1	0	1	0	0	0	1	1	1	1
15	1	1	1	1	1	0	0	0	0	0	0	0
BI	x	x	x	x	0	0	0	0	0	0	0	0

output: a

DC \ BA	00	01	11	10
00	1	0	1	1
01	0	1	1	0
11	0	1	0	0
10	1	1	0	0

Please determine the simplified POS





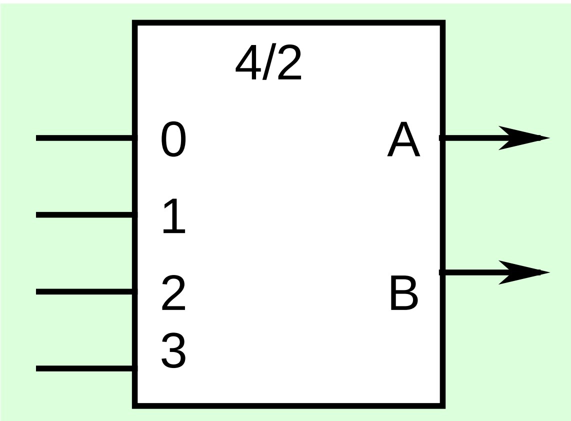
$$a = \left(\overline{D} + \overline{B}\right) \cdot \left(\overline{C} + A\right) \cdot \left(D + C + B + \overline{A}\right)$$

$$a = \left(\overline{DB}\right) \cdot \left(\overline{C\overline{A}}\right) \cdot \left(\overline{\overline{D}\,\overline{C}\,\overline{B}A}\right)$$

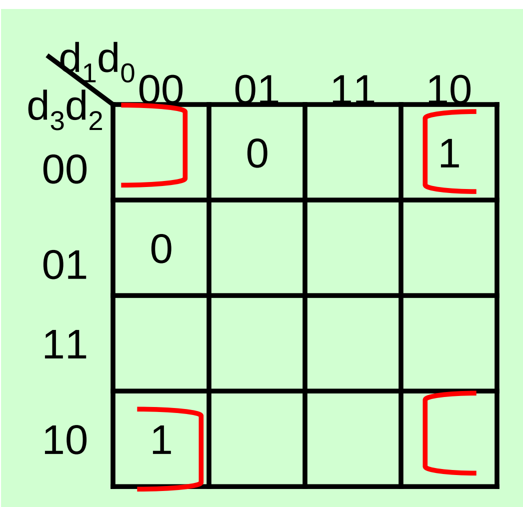
7449 BCD to seven segment decoder

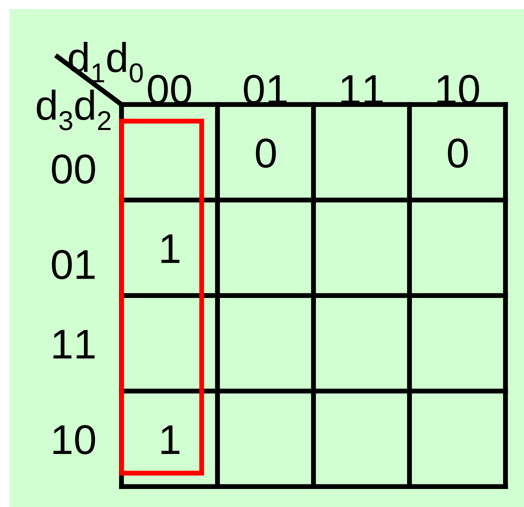
Encoders

An encoder performs the inverse operation of a decoder.



d_3	d_2	d_1	d_0	B	A
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1



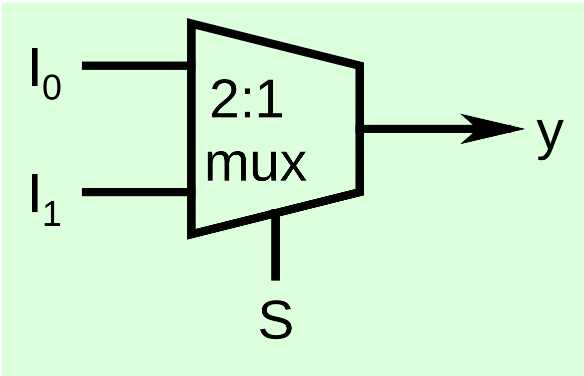


$$B = \overline{d_1}\ \overline{d_0}$$

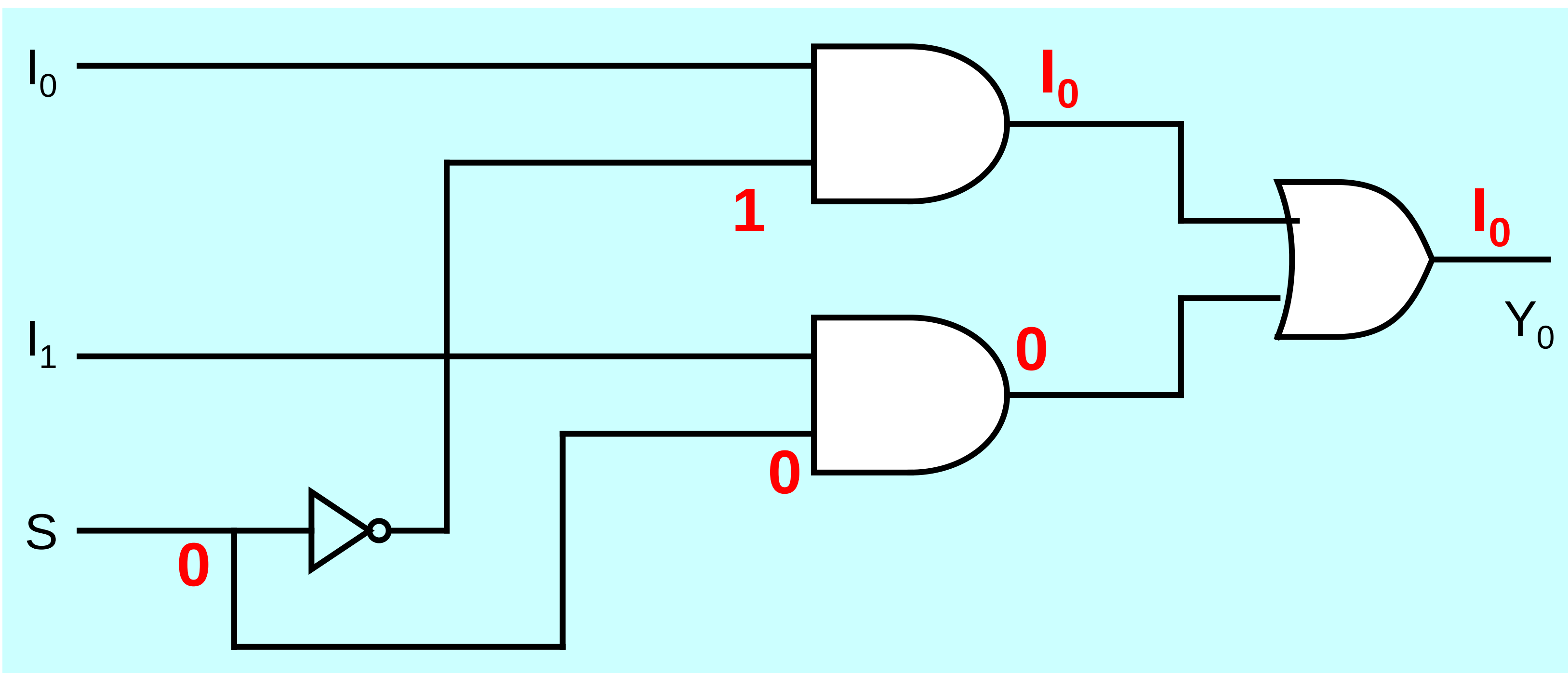
$$A = \overline{d_2}\ \overline{d_0}$$

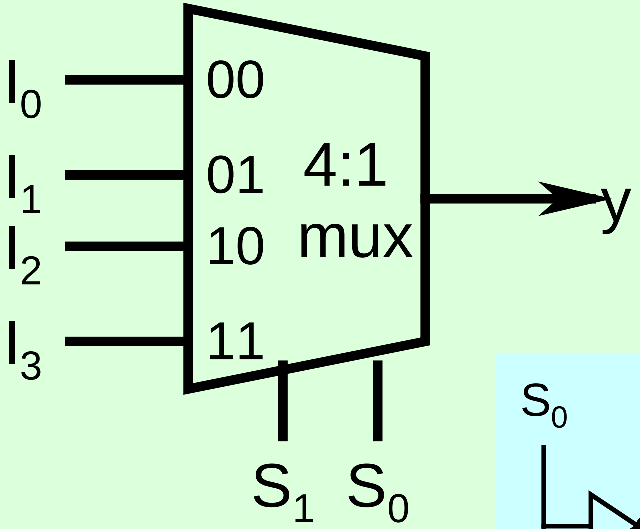
Vacants are don’t care

Multiplexers

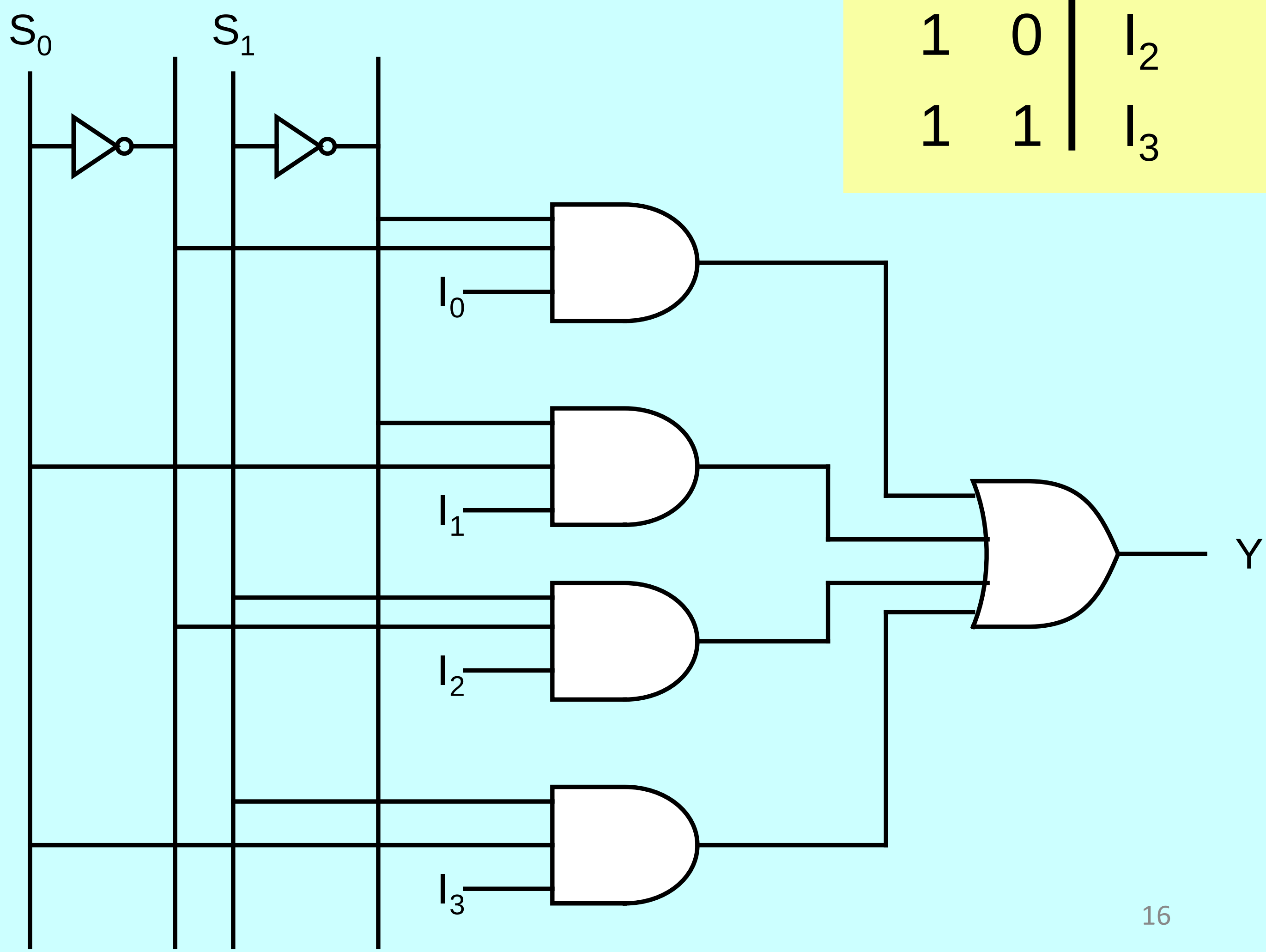


S	y
0	I_0
1	I_1



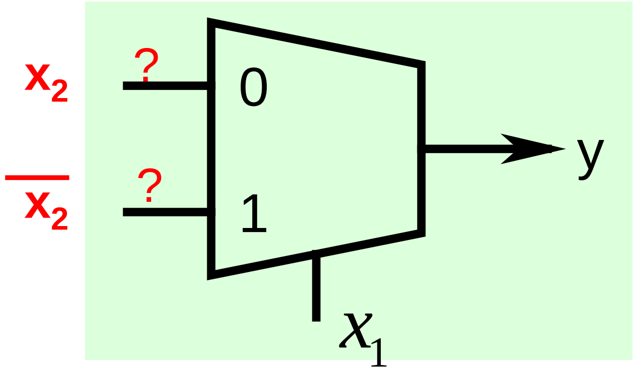


S_1	S_0	y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3



Implementing Boolean expressions using Multiplexers

$$y = x_1 \overline{x_2} + \overline{x_1} x_2$$



x_1	x_2	y
0	**0**	**0**
0	**1**	**1**
1	**0**	**1**
1	**1**	**0**

$y = x_2$ when $x_1 = 0$

$y = \overline{x_2}$ when $x_1 = 1$