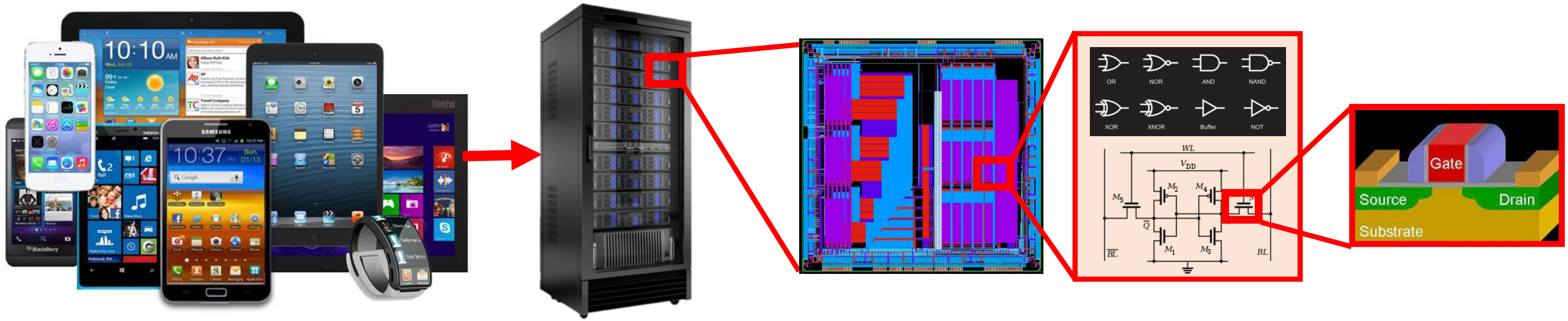
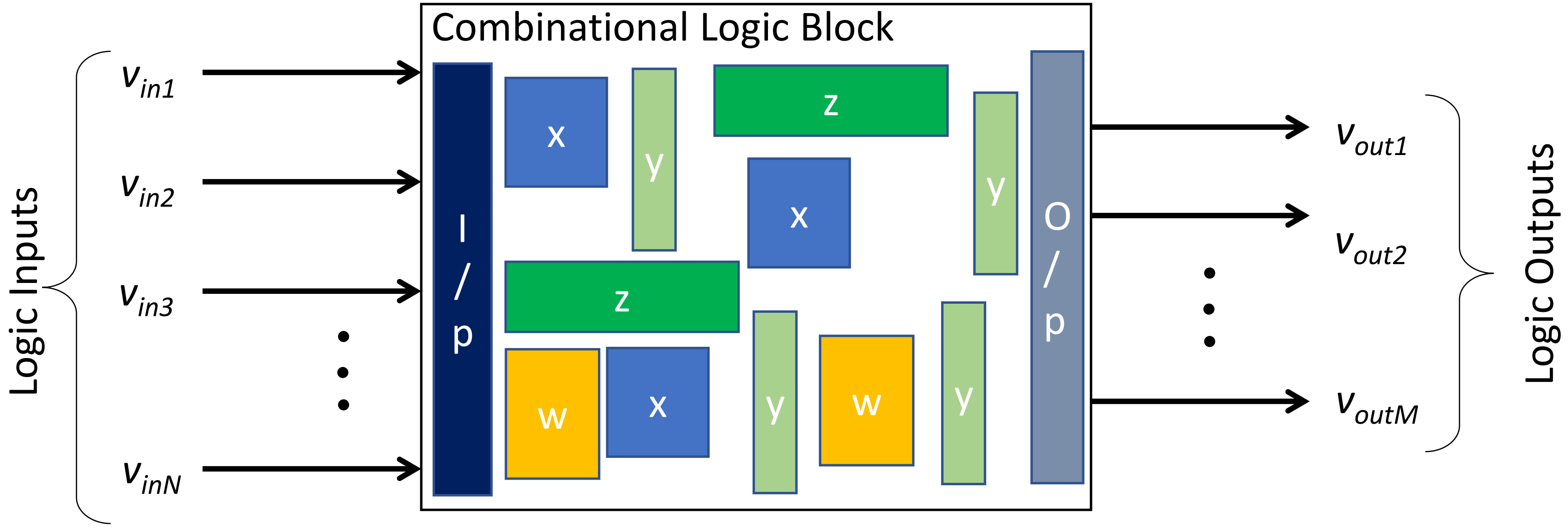


ESC201: Introduction to Electronics

Module 6: Digital Circuits

Dr. Shubham Sahay,
Associate Professor,
Department of Electrical Engineering,
IIT Kanpur

Useful Circuit Blocks

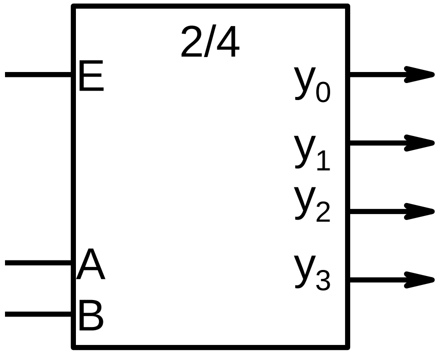


Build the larger circuit by using many smaller combinational logic functional blocks.

MUX, DeMUX, Encoder, Decoder, … adder, subtractor, multiplier, … and so on.

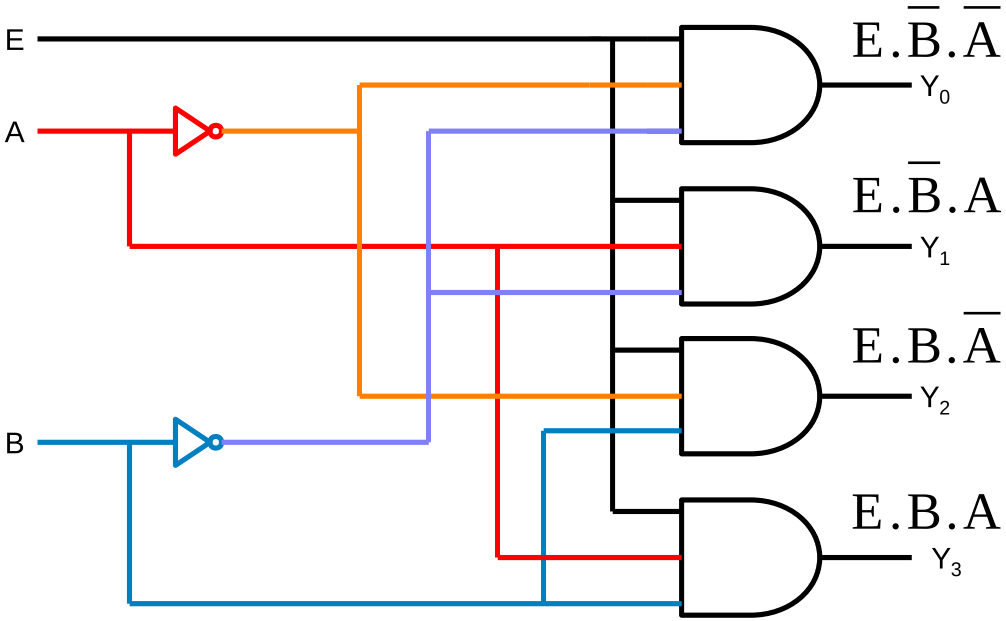
Only in special situations will one want to re-design to optimise the integrated block.

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

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

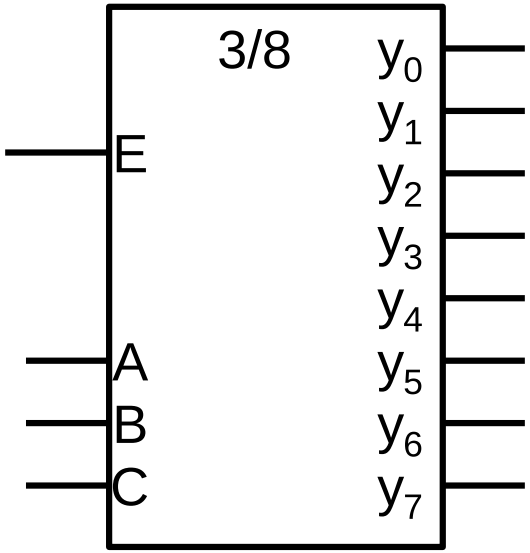


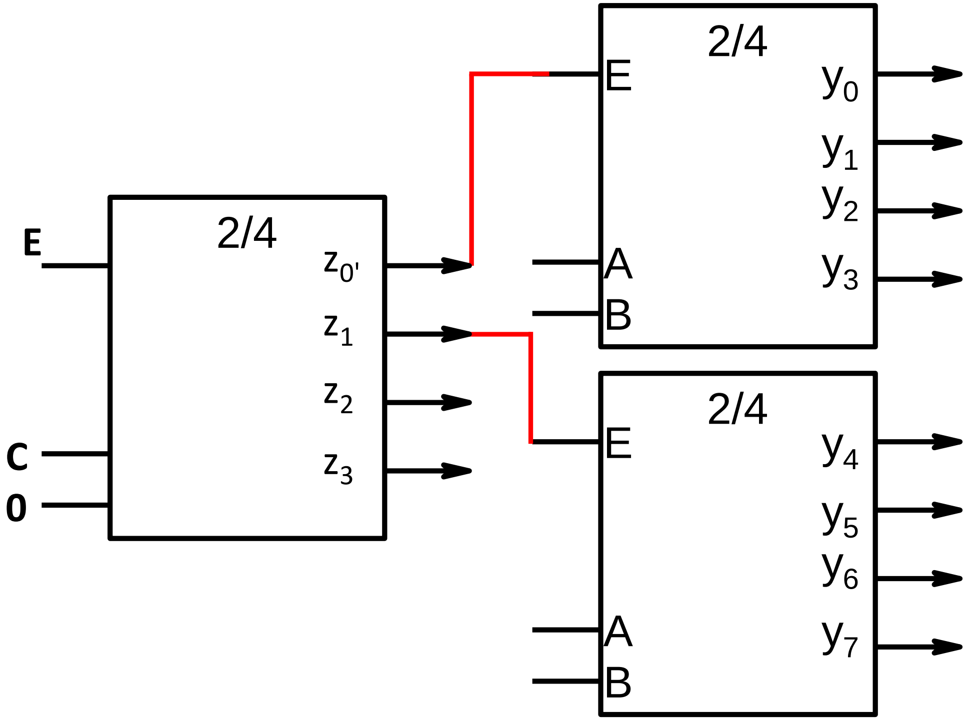
A *n* to 2^n decoder is a minterm generator

By selecting the min-terms, one may implement a truth table function!

Implementing Larger Decoders Using Smaller Units

3/8 decoder using 2/4 decoders

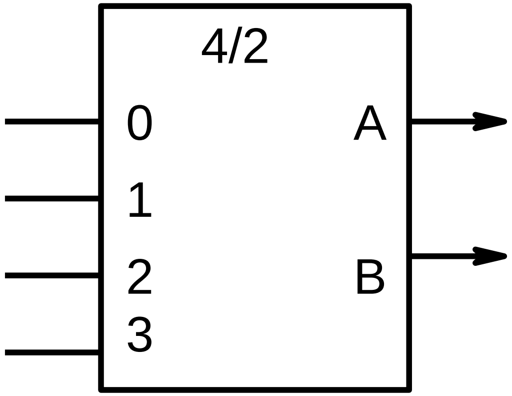




How many 2/4 decoders are required to implement a 4/16 decoder ?

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

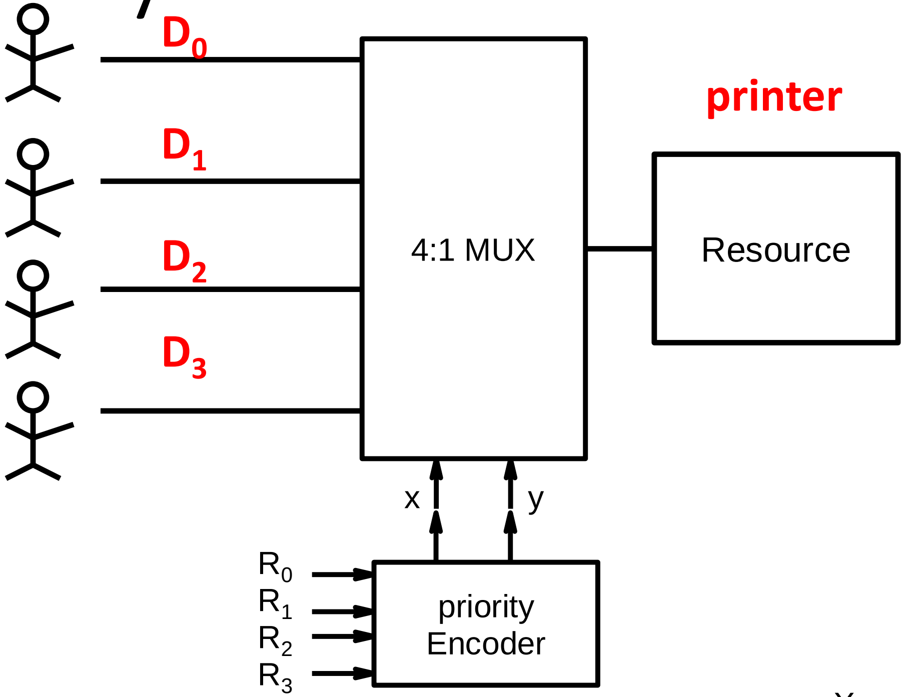
d_3d_2 \ d_1d_0	00	01	11	10
00	x	0	x	1
01	0	x	x	x
11	x	x	x	x
10	1	x	x	x

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

d_3d_2 \ d_1d_0	00	01	11	10
00	x	0	x	0
01	1	x	x	x
11	x	x	x	x
10	1	x	x	x

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

Priority Encoders



Priority is 3,2,1,0 with user 3 having the highest priority

X, Y have to be determined based on this priority order and the requests to use the resource.

R_0	R_1	R_2	R_3	x	y
0	0	0	0	x	x
1	0	0	0	0	0
x	1	0	0	0	1
x	x	1	0	1	0
x	x	x	1	1	1

X

R_3R_2 \ R_1R_0	00	01	11	10
00	0	x	0	0
01	1	1	1	1
11	1	1	1	1
10	1	1	1	1

$$x = R_2 + R_3$$

Y

R_3R_2 \ R_1R_0	00	01	11	10
00	x	0	1	1
01	0	0	0	0
11	1	1	1	1
10	1	1	1	1

$$y = R_1\overline{R_2} + R_3$$

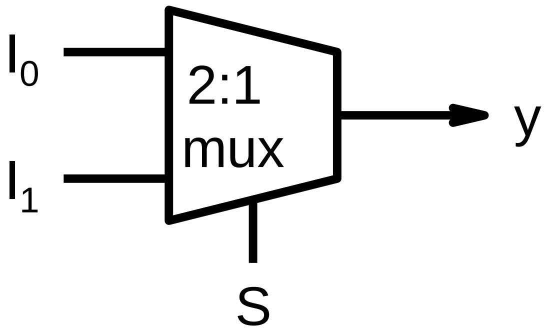
Parity

Extra bits are added to aid in error detection and correction

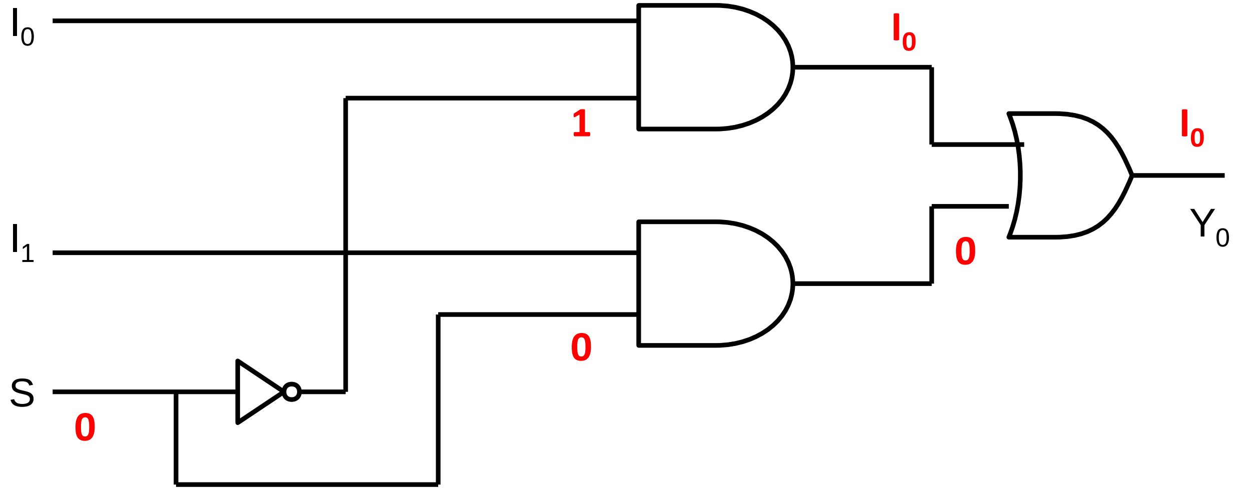
decimal	Binary	Even parity	Odd parity
0	000	0000	0001
1	001	0011	0010
2	010	0101	0100
3	011	0110	0111
4	100	1001	1000
5	101	1010	1011
6	110	1100	1101
7	111	1111	1110

A 1-bit error changes the parity and thus can be detected

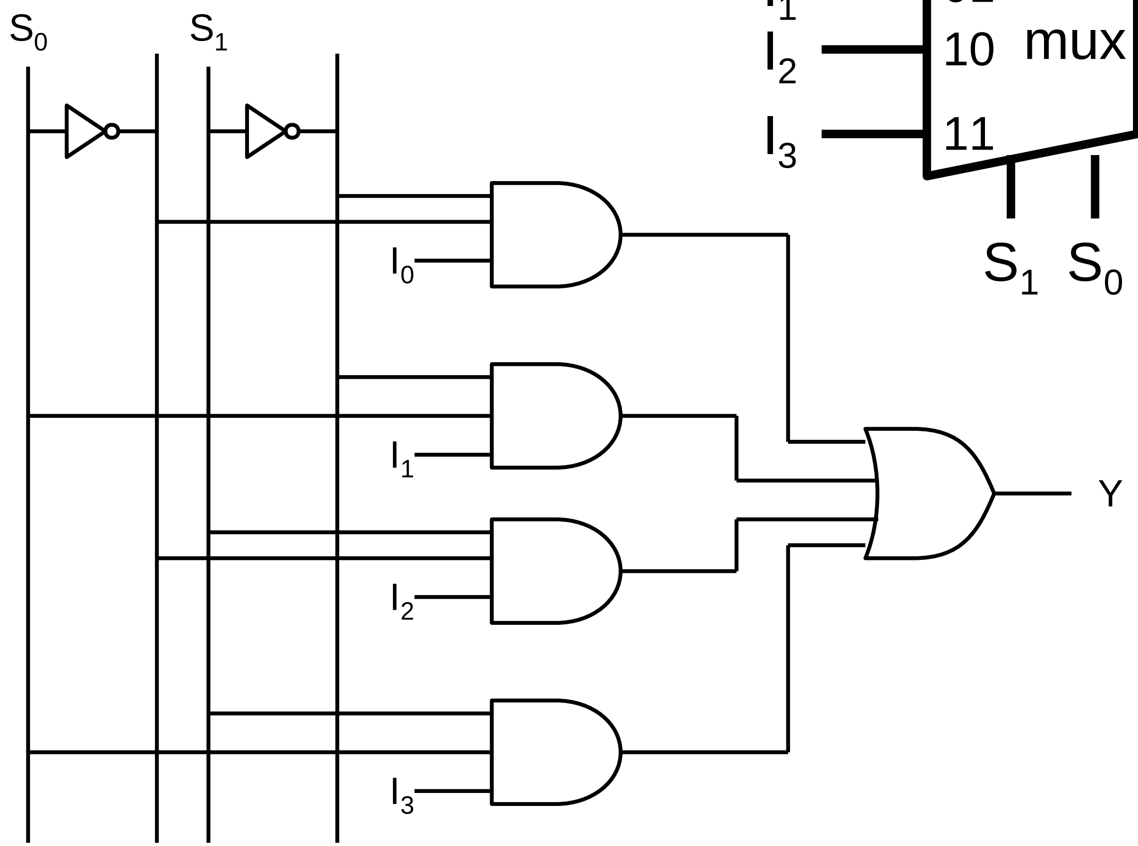
2:1 Multiplexers Gate Implementation

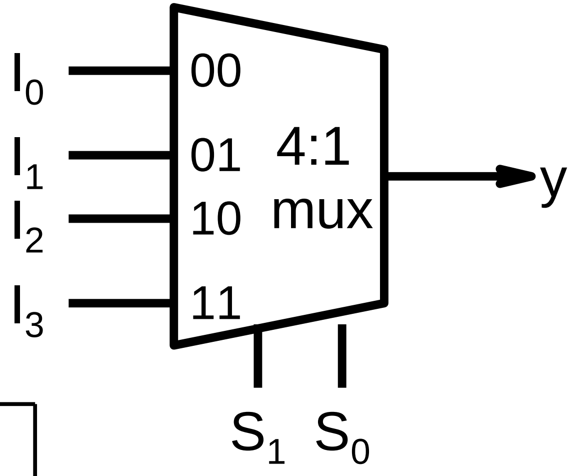


S	y
0	I_0
1	I_1



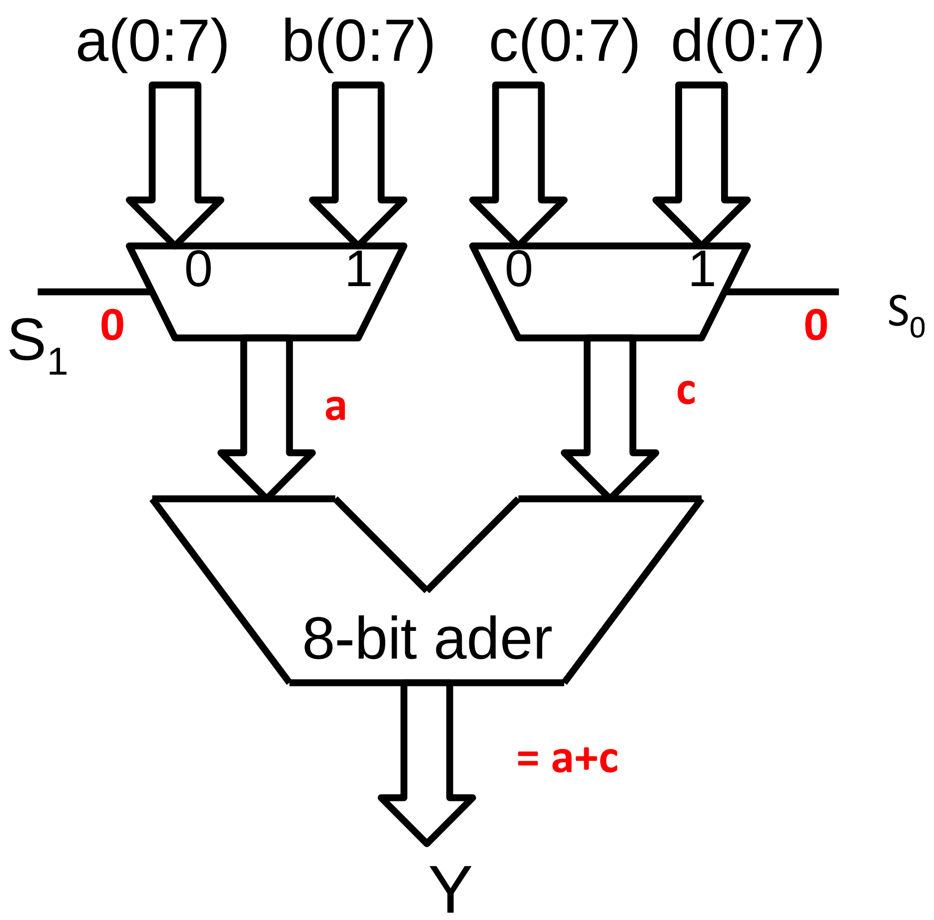
4:1 MUX Gate Implementation





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

Sharing Hardware With MUX



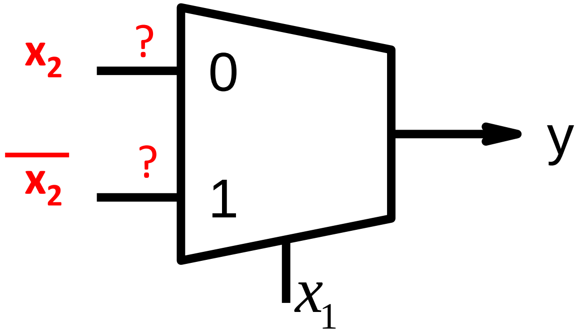
S_1	S_0	$y =$
0	0	a+c
0	1	a+d
1	0	b+c
1	1	b+d

Mux is often used when resources have to be shared

Boolean Functions with MUX

Implementing Boolean expressions using Multiplexers

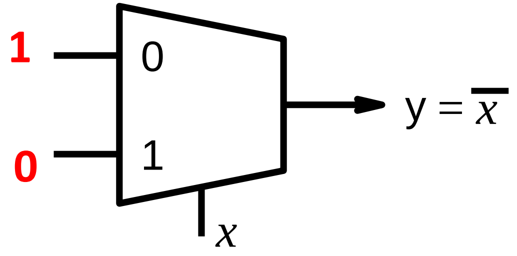
$$y = x_1 \overline{x_2} + \overline{x_1} x_2$$

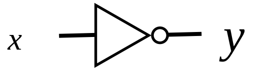


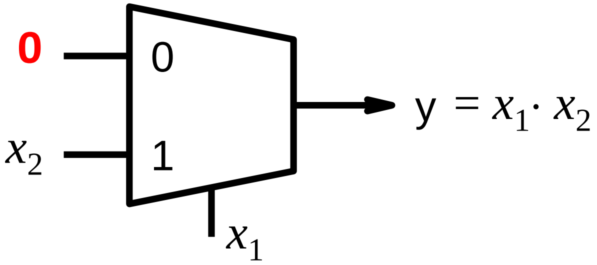
x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

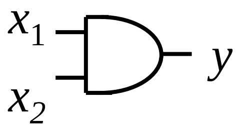
$y = x_2$ when $x_1 = 0$

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









AND combined with NOT → NAND

A universal gate!