Computer Science CSCI 355

Digital Logic and Computer Organization

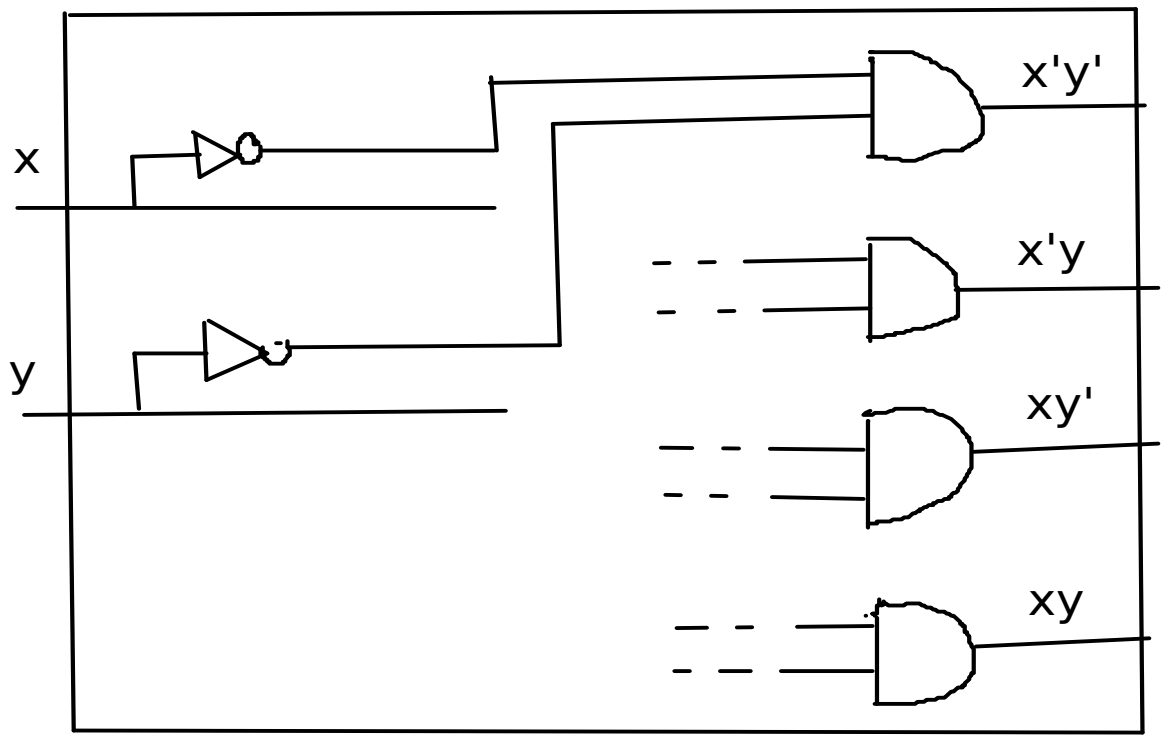
Dr. Peter Walsh
Department of Computer Science
Vancouver Island University
peter.walsh@viu.ca

Bigger Building Blocks

- MSI
  - Decoders
  - Encoders
  - Multiplexors
  - Demultiplexors

Decoders

- e.g. 2 x 4 decoder



Decoders cont.

◯ e.g. Implement $f(a, b, c) = ab + c$ using 3 x 8 decoder

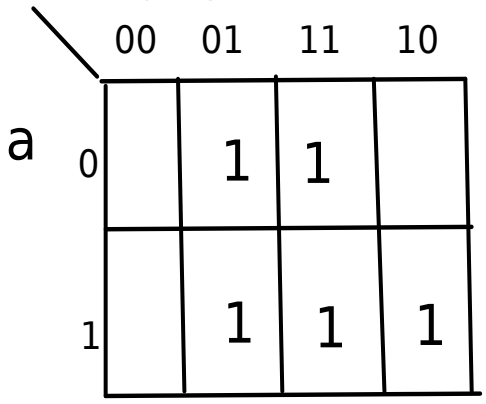
ab-

110
111

--c

001
011
101
111

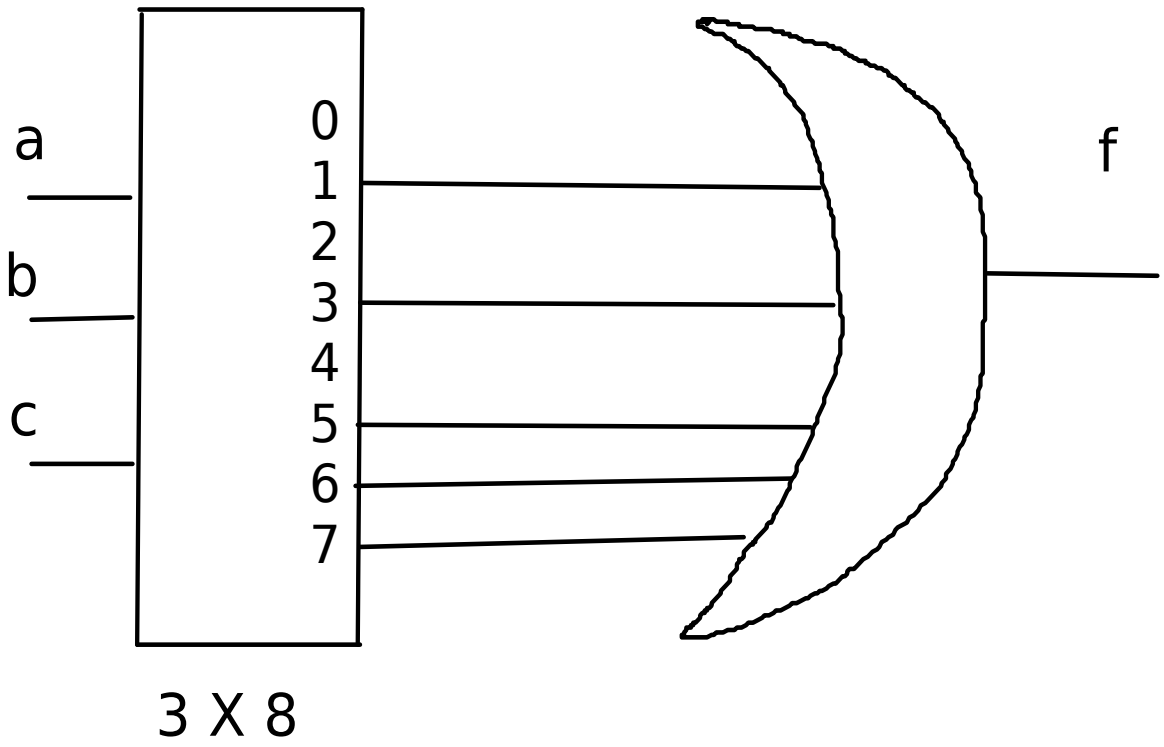
f



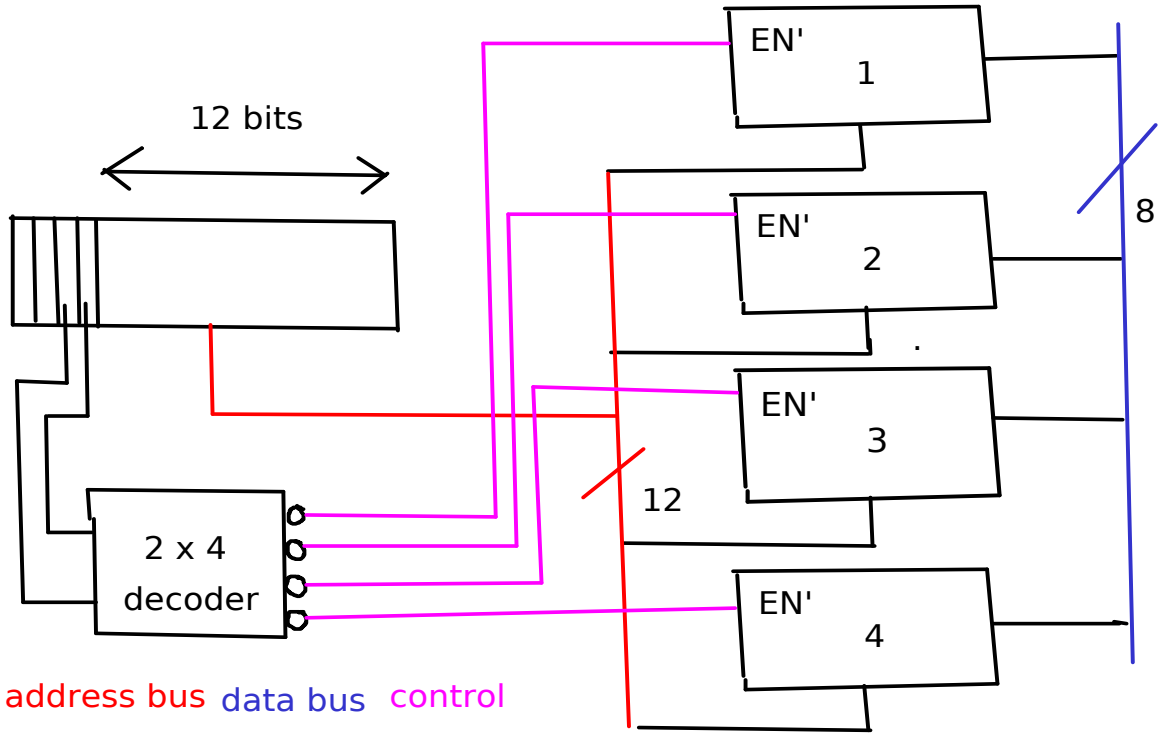
m (1, 3, 5, 6, 7)

Decoders cont.

- $f = \Sigma m(1, 3, 5, 6, 7)$

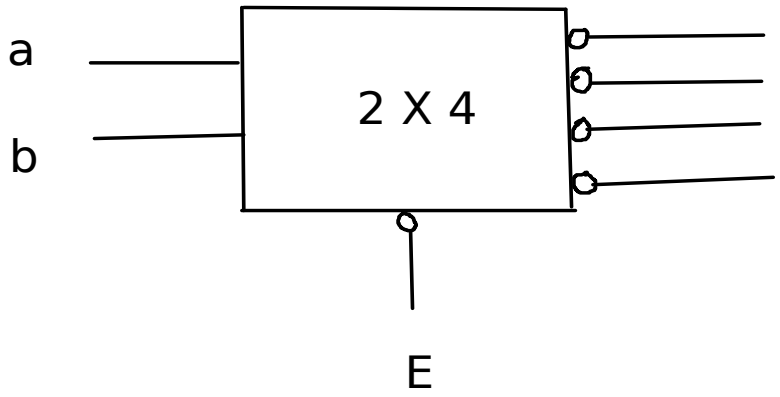


RAM/ROM (14 bit address - 8 bit data)



Decoders cont.

- active high inputs
- active low outputs
- active low enable



Encoder

D0	D1	D2	D3	x	y
1	0	0	0	0	0
0	1	0	0	0	1
0	0	1	0	1	0
0	0	1	1	1	1

0 1 2

x

D0 D1 / D2 D3	00	01	11	10
00	-	1	-	1
01	0	-	-	-
11	-	-	-	-
10	0	-	-	-

y

	00	01	11	10
00	-	1	-	0
01	1	-	-	-
11	-	-	-	-
10	0	-	-	-

x = D2 + D3
y = D1 + D3

Priority Encoder

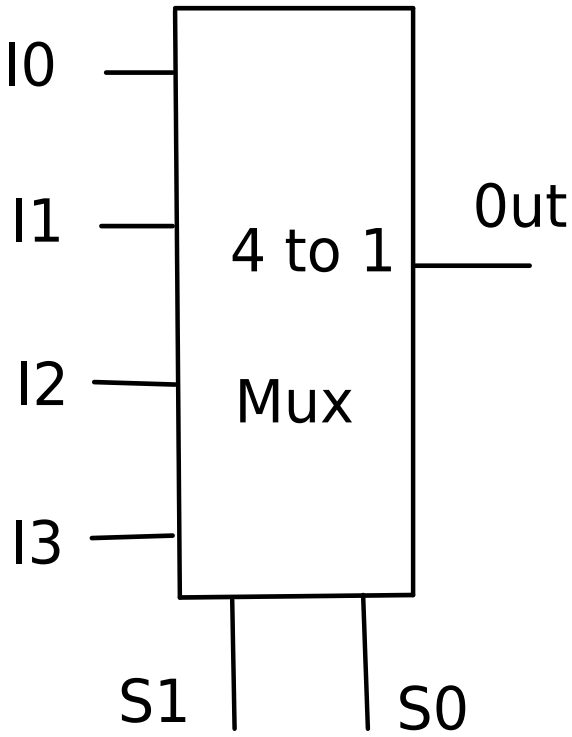
D0	D1	D2	D3	x	y	z
0	0	0	0	-	-	0
1	0	0	0	0	0	1
-	1	0	0	0	1	1
-	-	1	0	1	0	1
-	-	-	1	1	1	1

0 1 2

2

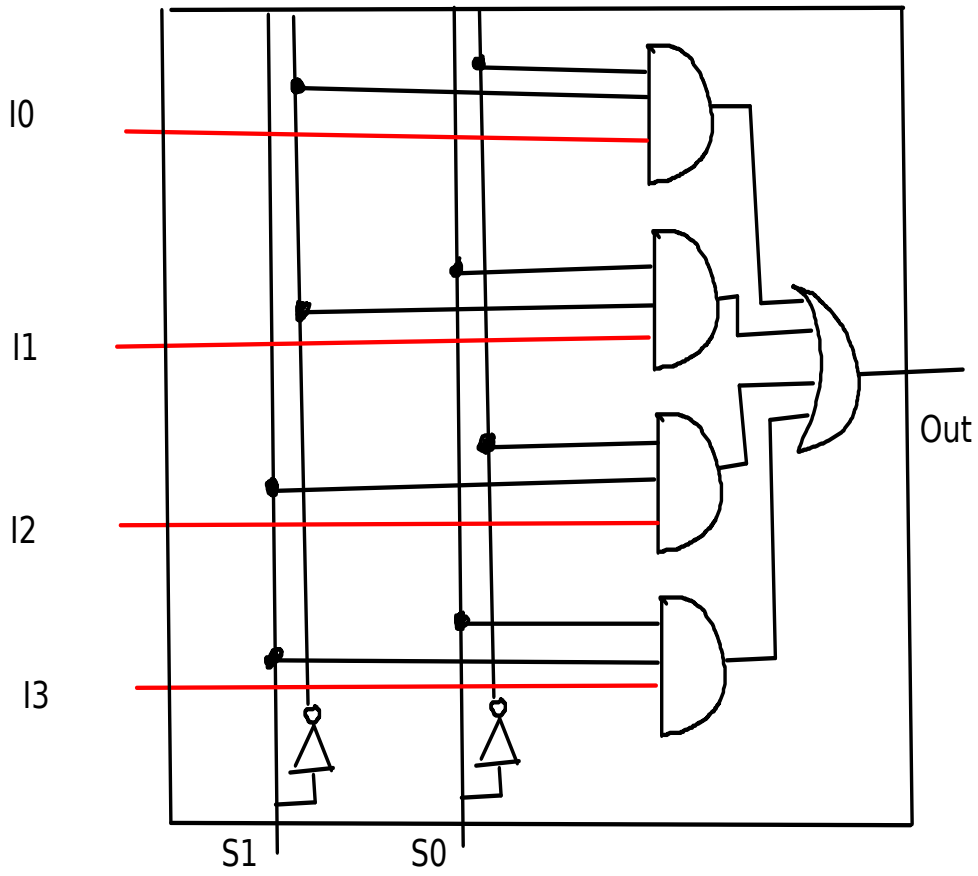
Multiplexors

- e.g. 4 to 1 multiplexor



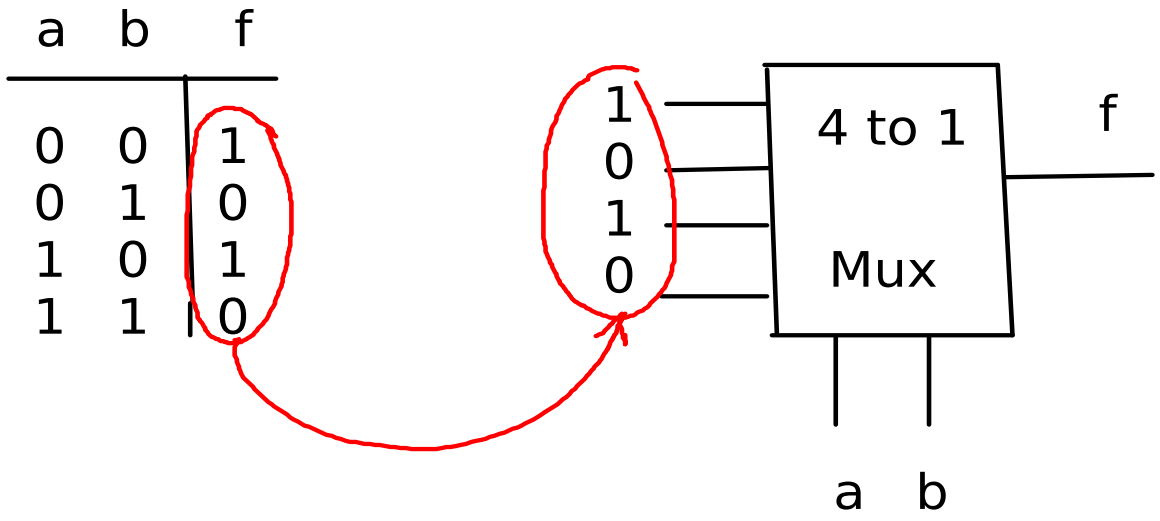
S1	S0	0ut
0	0	I0
0	1	I1
1	0	I2
1	1	I3

Multiplexors cont.



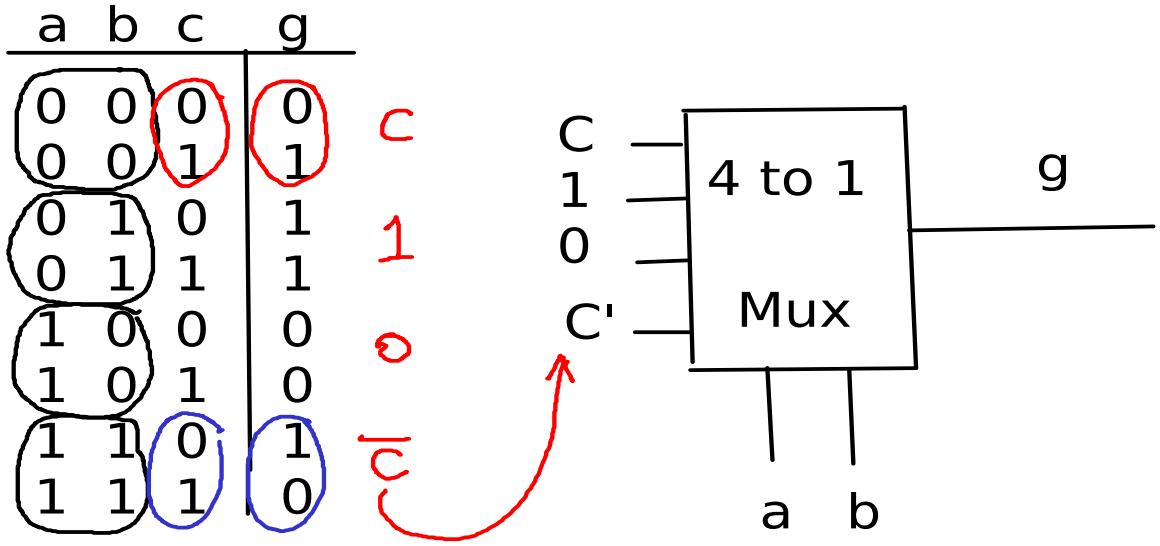
Multiplexors cont.

◯ e.g. Implement f using 4 to 1 multiplexor



Multiplexors cont.

- e.g. Implement g using 4 to 1 multiplexor

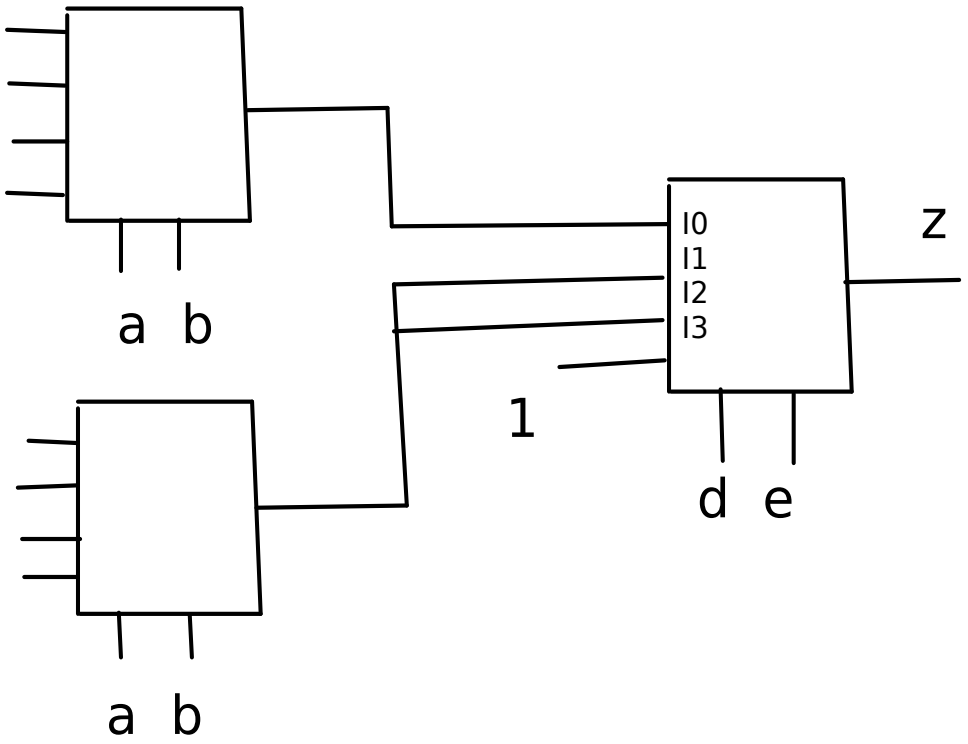


Multiplexors cont.

◯ e.g. Implement $z(a, b, c, d, e)$ using 3 4-to-1 multiplexors

			d e				
			I0	I1	I2	I3	
a	b	c	00	01	10	11	
0	0	0	0	0	0	1	1
0	0	1	0	1	1	1	
0	1	0	0	1	1	1	
0	1	1	1	1	1	1	
1	0	0	0	1	1	1	
1	0	1	1	1	1	1	
1	1	0	1	1	1	1	
1	1	1	1	1	1	1	

Multiplexors cont.



Multiplexors cont.

	d e			
	I0	I1	I2	I3
a b c	00	01	10	11
0 0 0	0	0	0	1
0 0 1	0	1	1	1
0 1 0	0	1	1	1
0 1 1	1	1	1	1
1 0 0	0	1	1	1
1 0 1	1	1	1	1
1 1 0	1	1	1	1
1 1 1	1	1	1	1

Multiplexors cont.