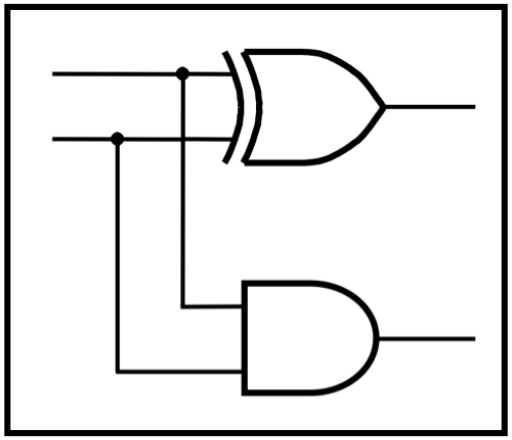


CprE 281: Digital Logic

Instructor: Alexander Stoytchev

http://www.ece.iastate.edu/~alexs/classes/

Decoders and Encoders

CprE 281: Digital Logic
Iowa State University, Ames, IA

Administrative Stuff

- **HW6 is due today**
- **HW7 is out. It is due on Monday Oct 21 @ 10pm.**

HW6, Problem 2

```
+ 110010        + 001011        + 101100
  010100          011001          011110
 ________        ________        ________


- 100101        - 011110        - 101011
  110011          001100          010111
 ________        ________        ________
```

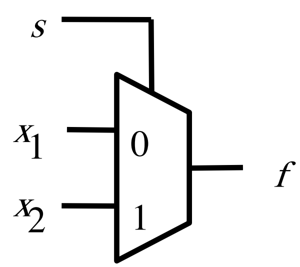
HW6, Problem 2

```
  110010        001011        101100
+ 010100      + 011001      + 011110
--------      --------      --------

  100101        011110        101011
- 110011      - 001100      - 010111
--------      --------      --------
```

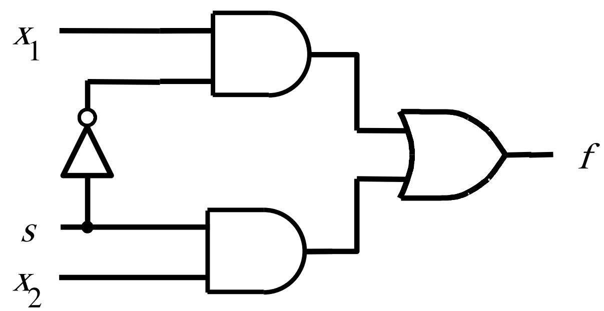
Quick Review

Graphical Symbol for a 2-1 Multiplexer

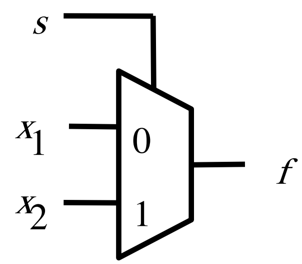


[Figure 2.33c from the textbook]

Circuit for 2-1 Multiplexer



(b) Circuit

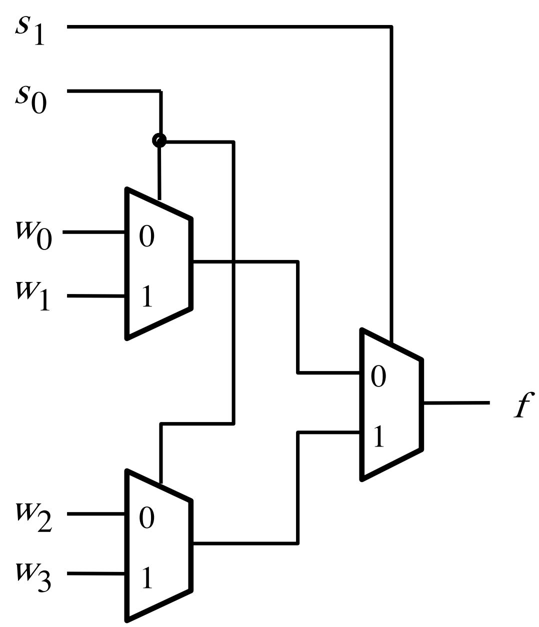


(c) Graphical symbol

$$f(s, x_1, x_2) = \overline{s}\, x_1 + s\, x_2$$

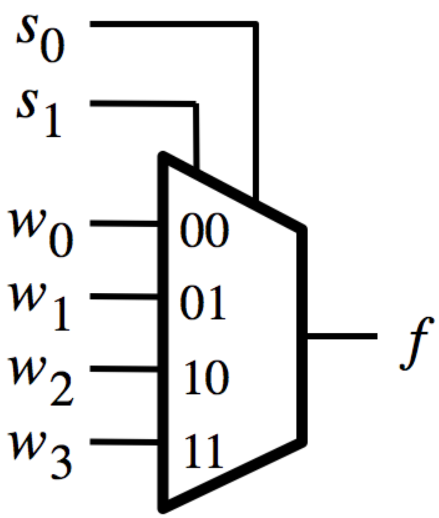
[Figure 2.33b-c from the textbook]

Using three 2-to-1 multiplexers to build one 4-to-1 multiplexer



[Figure 4.3 from the textbook]

Graphical Symbol and Truth Table



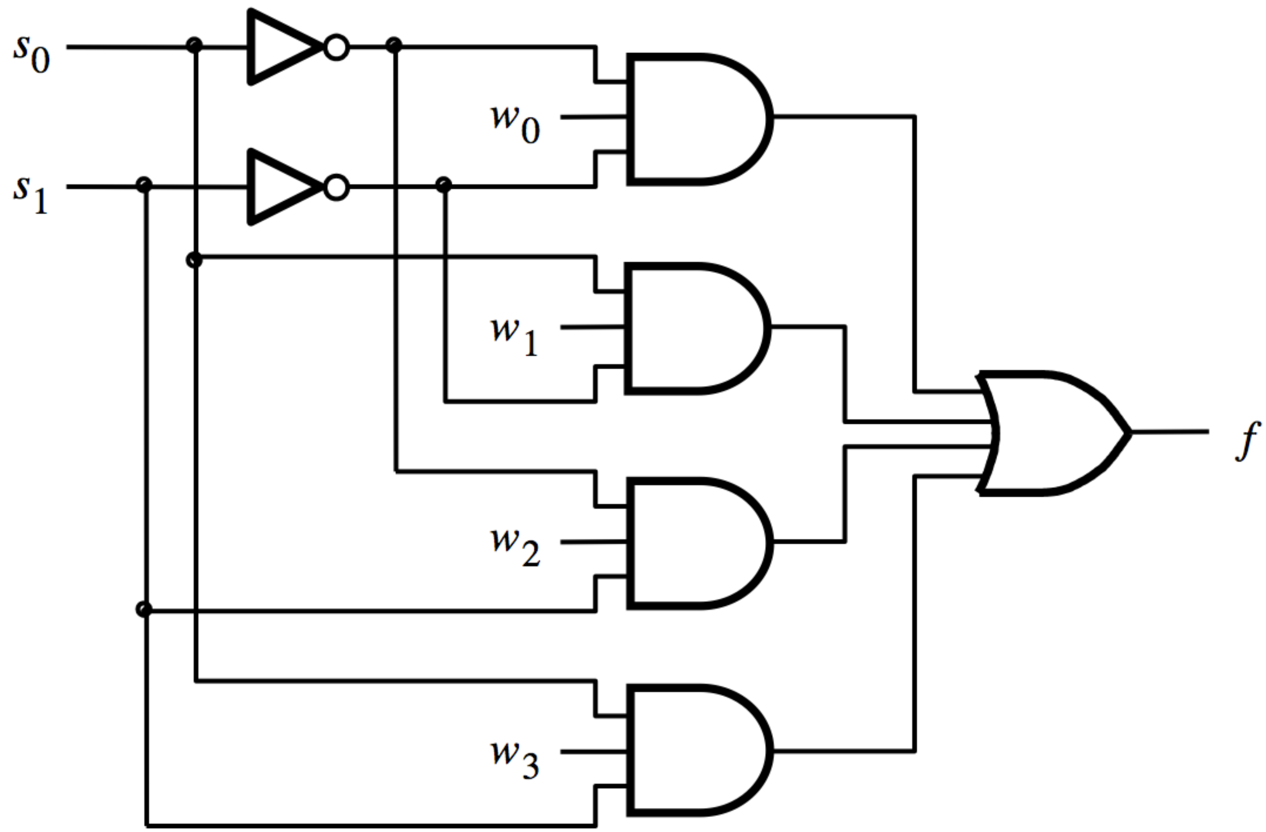
(a) Graphic symbol

s_1	s_0	f
0	0	w_0
0	1	w_1
1	0	w_2
1	1	w_3

(b) Truth table

[Figure 4.2a-b from the textbook]

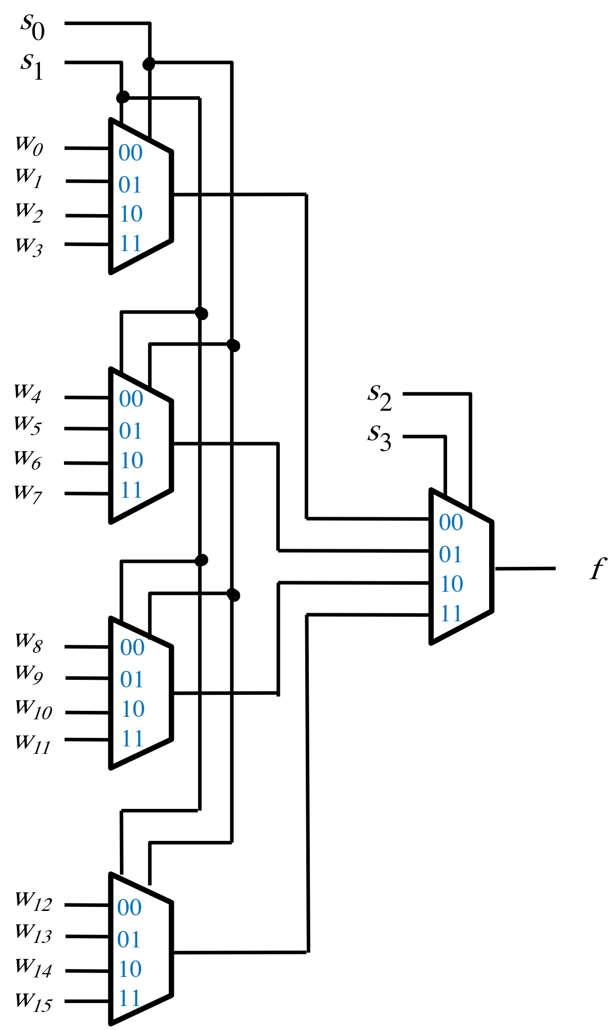
4-to-1 Multiplexer (SOP circuit)



$$f = \overline{s_1}\,\overline{s_0}\,w_0 + \overline{s_1}\,s_0\,w_1 + s_1\,\overline{s_0}\,w_2 + s_1\,s_0\,w_3$$

[Figure 4.2c from the textbook]

16-1 Multiplexer with 4-to-1 Multiplexers



Circuit Synthesis with Multiplexers Using Shannon's Expansion

Three-input majority function

w_1	w_2	w_3	f
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

[Figure 4.10a from the textbook]

Three-input majority function

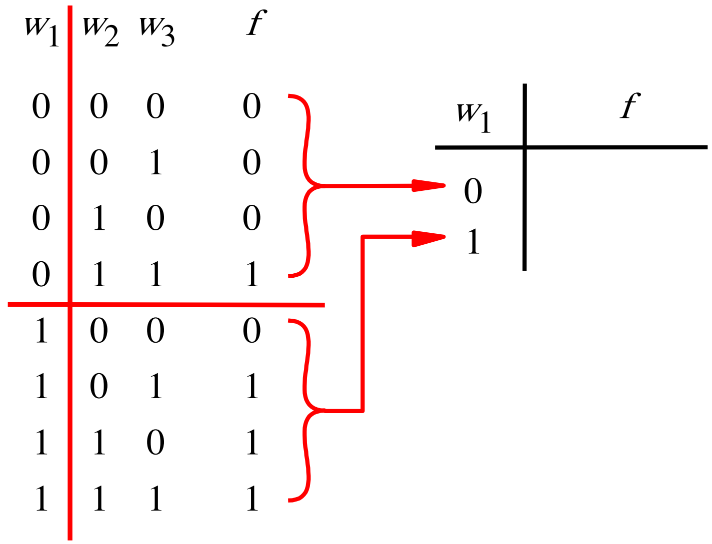
w_1	w_2	w_3	f
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

SOP expression for f :

$$f = \overline{w}_1 w_2 w_3 + w_1 \overline{w}_2 w_3 + w_1 w_2 \overline{w}_3 + w_1 w_2 w_3$$

[Figure 4.10a from the textbook]

Three-input majority function



Divide-and-conquer method (a.k.a., Shannon’s expansion)

[Figure 4.10a from the textbook]

Three-input majority function

w_1	w_2	w_3	f
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

w_1	f
0	$w_2 w_3$
1	$w_2 + w_3$

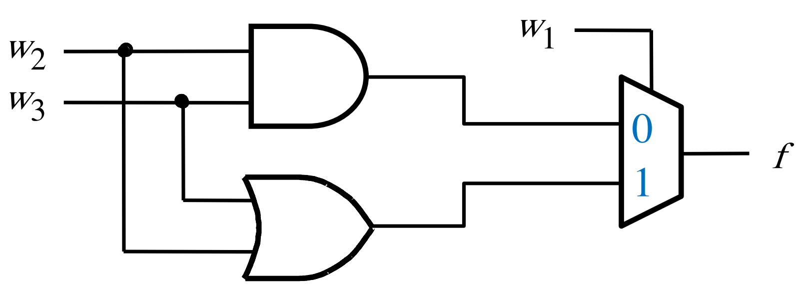
[Figure 4.10a from the textbook]

Three-input majority function

w_1	w_2	w_3	f
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

w_1	f
0	$w_2 w_3$
1	$w_2 + w_3$

(b) Truth table



(b) Circuit

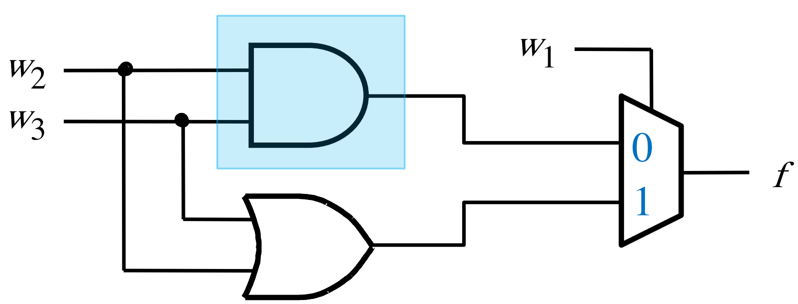
[Figure 4.10a from the textbook]

Three-input majority function

w_1	w_2	w_3	f
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

w_1	f
0	$w_2 w_3$
1	$w_2 + w_3$

(b) Truth table



(b) Circuit

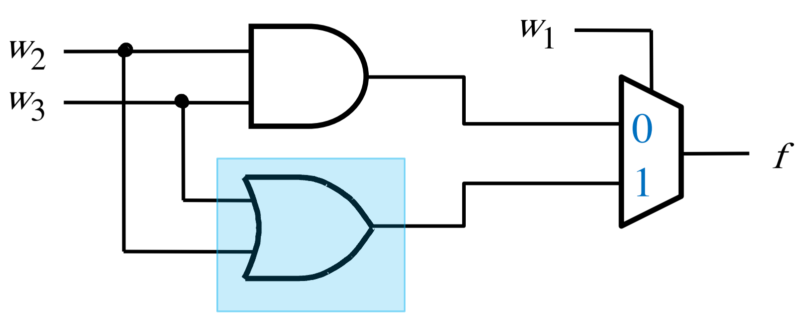
[Figure 4.10a from the textbook]

Three-input majority function

w_1	w_2	w_3	f
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

w_1	f
0	$w_2 w_3$
1	$w_2 + w_3$

(b) Truth table



(b) Circuit

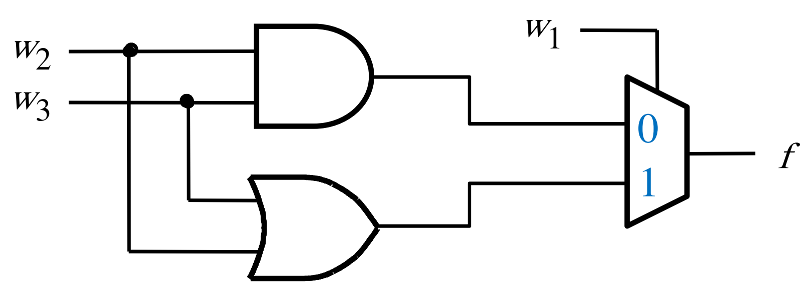
[Figure 4.10a from the textbook]

Three-input majority function

$$f = \overline{w}_1 w_2 w_3 + w_1 \overline{w}_2 w_3 + w_1 w_2 \overline{w}_3 + w_1 w_2 w_3$$

$$f = \overline{w}_1 (w_2 w_3) + w_1 (\overline{w}_2 w_3 + w_2 \overline{w}_3 + w_2 w_3)$$

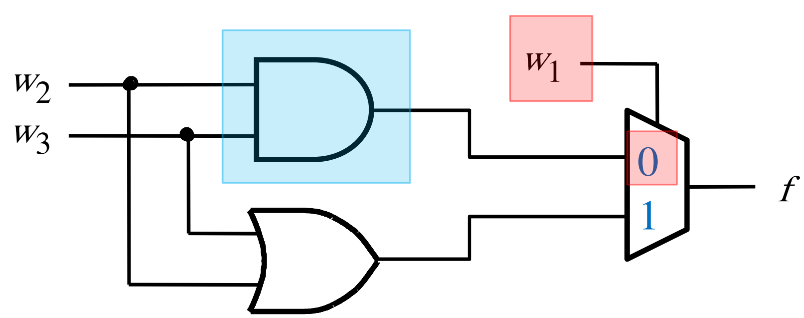
$$= \overline{w}_1 (w_2 w_3) + w_1 (w_2 + w_3)$$



Three-input majority function

$$f = \overline{w}_1 w_2 w_3 + w_1 \overline{w}_2 w_3 + w_1 w_2 \overline{w}_3 + w_1 w_2 w_3$$

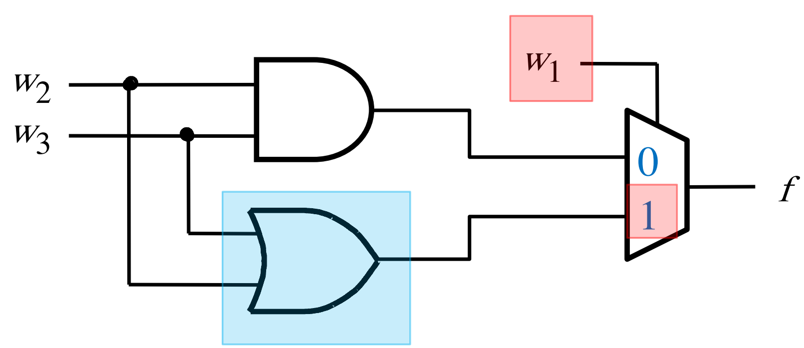
$$f = \overline{w}_1 (w_2 w_3) + w_1 (\overline{w}_2 w_3 + w_2 \overline{w}_3 + w_2 w_3)$$
$$= \overline{w}_1 (w_2 w_3) + w_1 (w_2 + w_3)$$



Three-input majority function

$$f = \overline{w}_1 w_2 w_3 + w_1 \overline{w}_2 w_3 + w_1 w_2 \overline{w}_3 + w_1 w_2 w_3$$

$$f = \overline{w}_1 (w_2 w_3) + w_1 (\overline{w}_2 w_3 + w_2 \overline{w}_3 + w_2 w_3)$$
$$= \overline{w}_1 (w_2 w_3) + w_1 (w_2 + w_3)$$



Shannon's Expansion Theorem
(general case with one select variable)

Shannon's Expansion Theorem

Any Boolean function $f(w_1, \ldots, w_n)$ can be rewritten in the form:

$$f(w_1, w_2, \ldots, w_n) = \overline{w}_1 \cdot f(0, w_2, \ldots, w_n) + w_1 \cdot f(1, w_2, \ldots, w_n)$$

This form is suitable for implementation with a 2-to-1 multiplexer.

Shannon's Expansion Theorem

Any Boolean function $f(w_1, \ldots, w_n)$ can be rewritten in the form:

$$f(w_1, w_2, \ldots, w_n) = \overline{w}_1 \cdot f(0, w_2, \ldots, w_n) + w_1 \cdot f(1, w_2, \ldots, w_n)$$

w_1 set to 0

Shannon's Expansion Theorem

Any Boolean function $f(w_1, \ldots, w_n)$ can be rewritten in the form:

$$f(w_1, w_2, \ldots, w_n) = \overline{w}_1 \cdot f(0, w_2, \ldots, w_n) + w_1 \cdot f(1, w_2, \ldots, w_n)$$

w_1 set to 1

Shannon's Expansion Theorem

Any Boolean function $f(w_1, \ldots, w_n)$ can be rewritten in the form:

$$f(w_1, w_2, \ldots, w_n) = \overline{w}_1 \cdot f(0, w_2, \ldots, w_n) + w_1 \cdot f(1, w_2, \ldots, w_n)$$

Shannon's Expansion Theorem

Any Boolean function $f(w_1, \ldots, w_n)$ can be rewritten in the form:

$$f(w_1, w_2, \ldots, w_n) = \overline{w}_1 \cdot f(0, w_2, \ldots, w_n) + w_1 \cdot f(1, w_2, \ldots, w_n)$$

$$f = \overline{w}_1 f_{\overline{w}_1} + w_1 f_{w_1}$$

Shannon's Expansion Theorem

Any Boolean function $f(w_1, \ldots, w_n)$ can be rewritten in the form:

$$f(w_1, w_2, \ldots, w_n) = \overline{w_1} \cdot f(0, w_2, \ldots, w_n) + w_1 \cdot f(1, w_2, \ldots, w_n)$$

$$f = \overline{w}_1 f_{\overline{w}_1} + w_1 f_{w_1}$$

select variable
(negated)

Shannon's Expansion Theorem

Any Boolean function $f(w_1, \ldots, w_n)$ can be rewritten in the form:

$$f(w_1, w_2, \ldots, w_n) = \overline{w}_1 \cdot f(0, w_2, \ldots, w_n) + w_1 \cdot f(1, w_2, \ldots, w_n)$$

$$f = \overline{w}_1 f_{\overline{w}_1} + w_1 f_{w_1}$$

select variable
(not negated)

Shannon's Expansion Theorem

Any Boolean function $f(w_1, \ldots, w_n)$ can be rewritten in the form:

$$f(w_1, w_2, \ldots, w_n) = \overline{w}_1 \cdot f(0, w_2, \ldots, w_n) + w_1 \cdot f(1, w_2, \ldots, w_n)$$

$$f = \overline{w}_1 f_{\overline{w}_1} + w_1 f_{w_1}$$

cofactor

Shannon's Expansion Theorem

Any Boolean function $f(w_1, \ldots, w_n)$ can be rewritten in the form:

$$f(w_1, w_2, \ldots, w_n) = \overline{w}_1 \cdot f(0, w_2, \ldots, w_n) + w_1 \cdot f(1, w_2, \ldots, w_n)$$

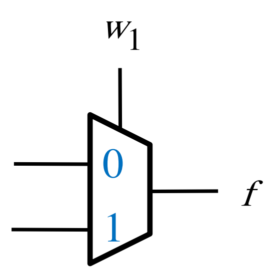
$$f = \overline{w}_1 f_{\overline{w}_1} + w_1 f_{w_1}$$

cofactor

Shannon's Expansion Theorem

Any Boolean function $f(w_1, \ldots, w_n)$ can be rewritten in the form:

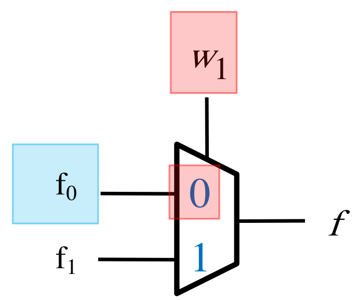
$$f(w_1, w_2, \ldots, w_n) = \overline{w}_1 \cdot f(0, w_2, \ldots, w_n) + w_1 \cdot f(1, w_2, \ldots, w_n)$$



Shannon's Expansion Theorem

Any Boolean function $f(w_1, \ldots, w_n)$ can be rewritten in the form:

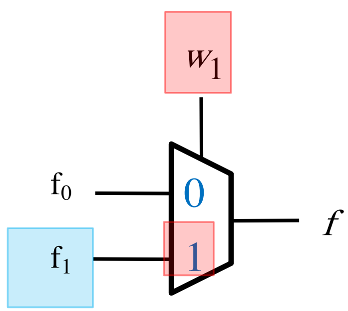
$$f(w_1, w_2, \ldots, w_n) = \overline{w_1} \cdot f(0, w_2, \ldots, w_n) + w_1 \cdot f(1, w_2, \ldots, w_n)$$



Shannon's Expansion Theorem

Any Boolean function $f(w_1, \ldots, w_n)$ can be rewritten in the form:

$$f(w_1, w_2, \ldots, w_n) = \overline{w}_1 \cdot f(0, w_2, \ldots, w_n) + w_1 \cdot f(1, w_2, \ldots, w_n)$$



Shannon's Expansion Theorem
(example with one select variable)

Shannon's Expansion Theorem (Example)

$$f(w_1, w_2, w_3) = w_1 w_2 + w_1 w_3 + w_2 w_3$$

Shannon's Expansion Theorem (Example)

$$f(w_1, w_2, w_3) = w_1 w_2 + w_1 w_3 + w_2 w_3$$

$$f(w_1, w_2, w_3) = w_1 w_2 + w_1 w_3 + w_2 w_3 \, (\overline{w_1} + w_1)$$

Shannon's Expansion Theorem (Example)

$$f(w_1, w_2, w_3) = w_1 w_2 + w_1 w_3 + w_2 w_3$$

$$f(w_1, w_2, w_3) = w_1 w_2 + w_1 w_3 + w_2 w_3 \; (\overline{w_1} + w_1)$$

$$\begin{aligned} f &= \overline{w}_1(0 \cdot w_2 + 0 \cdot w_3 + w_2 w_3) + w_1(1 \cdot w_2 + 1 \cdot w_3 + w_2 w_3) \\ &= \overline{w}_1(w_2 w_3) + w_1(w_2 + w_3) \end{aligned}$$

Shannon's Expansion Theorem
(general case with two select variables)

Shannon's Expansion Theorem
(In terms of more than one variable)

$$f(w_1, \ldots, w_n) = \overline{w}_1\overline{w}_2 \cdot f(0, 0, w_3, \ldots, w_n) + \overline{w}_1 w_2 \cdot f(0, 1, w_3, \ldots, w_n)$$
$$+ w_1\overline{w}_2 \cdot f(1, 0, w_3, \ldots, w_n) + w_1 w_2 \cdot f(1, 1, w_3, \ldots, w_n)$$

This form is suitable for implementation with a 4-to-1 multiplexer.

Shannon's Expansion Theorem
(In terms of more than one variable)

$$f(w_1, \ldots, w_n) = \overline{w}_1\overline{w}_2 \cdot f(0, 0, w_3, \ldots, w_n) + \overline{w}_1 w_2 \cdot f(0, 1, w_3, \ldots, w_n)$$
$$+ w_1\overline{w}_2 \cdot f(1, 0, w_3, \ldots, w_n) + w_1 w_2 \cdot f(1, 1, w_3, \ldots, w_n)$$

These are the four possible minterms with two variables.

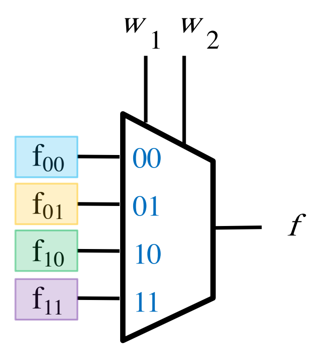
Shannon's Expansion Theorem (In terms of more than one variable)

$$f(w_1, \ldots, w_n) = \overline{w}_1\overline{w}_2 \cdot f(0, 0, w_3, \ldots, w_n) + \overline{w}_1 w_2 \cdot f(0, 1, w_3, \ldots, w_n) + w_1\overline{w}_2 \cdot f(1, 0, w_3, \ldots, w_n) + w_1 w_2 \cdot f(1, 1, w_3, \ldots, w_n)$$

These are the four cofactors, one for each of the minterms.

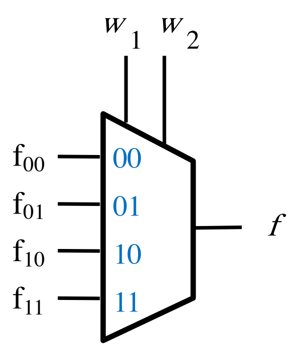
Shannon's Expansion Theorem
(In terms of more than one variable)

$$f(w_1, \ldots, w_n) = \overline{w}_1\overline{w}_2 \cdot f(0, 0, w_3, \ldots, w_n) + \overline{w}_1 w_2 \cdot f(0, 1, w_3, \ldots, w_n)$$
$$+ w_1\overline{w}_2 \cdot f(1, 0, w_3, \ldots, w_n) + w_1 w_2 \cdot f(1, 1, w_3, \ldots, w_n)$$



Shannon's Expansion Theorem
(In terms of more than one variable)

$$f(w_1, \ldots, w_n) = \overline{w}_1\overline{w}_2 \cdot f(0, 0, w_3, \ldots, w_n) + \overline{w}_1 w_2 \cdot f(0, 1, w_3, \ldots, w_n)$$
$$+ w_1\overline{w}_2 \cdot f(1, 0, w_3, \ldots, w_n) + w_1 w_2 \cdot f(1, 1, w_3, \ldots, w_n)$$



Shannon’s Expansion Theorem
(example with two select variables)

Factor and implement the following function with a 4-to-1 multiplexer

$$f = \overline{w}_1\overline{w}_3 + w_1w_2 + w_1w_3$$

Factor and implement the following function with a 4-to-1 multiplexer

$$f = \overline{w}_1\overline{w}_3 + w_1w_2 + w_1w_3$$

$$= \overline{w}_1 (\overline{w}_2 + w_2) \overline{w}_3 + w_1 w_2 + w_1 (\overline{w}_2 + w_2) w_3$$

Factor and implement the following function with a 4-to-1 multiplexer

$$f = \overline{w}_1 \overline{w}_3 + w_1 w_2 + w_1 w_3$$

$$= \overline{w_1}\,(\overline{w_2} + w_2)\,\overline{w_3} + w_1 w_2 + w_1\,(\overline{w_2} + w_2)\,w_3$$

$$= \overline{w_1}\,\overline{w_2}\,\overline{w_3} + \overline{w_1}\,w_2\,\overline{w_3} + w_1 w_2 + w_1\,\overline{w_2}\,w_3 + w_1 w_2 w_3$$

$$= \overline{w_1}\,\overline{w_2}\,\overline{w_3} + \overline{w_1}\,w_2\,\overline{w_3} + w_1\,\overline{w_2}\,w_3 + w_1 w_2\,(1 + w_3)$$

$$= \overline{w_1}\,\overline{w_2}\,(\,\overline{w_3}\,) + \overline{w_1}\,w_2\,(\,\overline{w_3}\,) + w_1\,\overline{w_2}\,(w_3) + w_1 w_2\,(1)$$

Factor and implement the following function with a 4-to-1 multiplexer

$$f = \overline{w}_1 \overline{w}_3 + w_1 w_2 + w_1 w_3$$

$$= \overline{w}_1 (\overline{w}_2 + w_2) \overline{w}_3 + w_1 w_2 + w_1 (\overline{w}_2 + w_2) w_3$$

$$= \overline{w}_1 \overline{w}_2 \overline{w}_3 + \overline{w}_1 w_2 \overline{w}_3 + w_1 w_2 + w_1 \overline{w}_2 w_3 + w_1 w_2 w_3$$

$$= \overline{w}_1 \overline{w}_2 \overline{w}_3 + \overline{w}_1 w_2 \overline{w}_3 + w_1 \overline{w}_2 w_3 + w_1 w_2 (1 + w_3)$$

$$= \overline{w}_1 \overline{w}_2 (\overline{w}_3) + \overline{w}_1 w_2 (\overline{w}_3) + w_1 \overline{w}_2 (w_3) + w_1 w_2 (1)$$

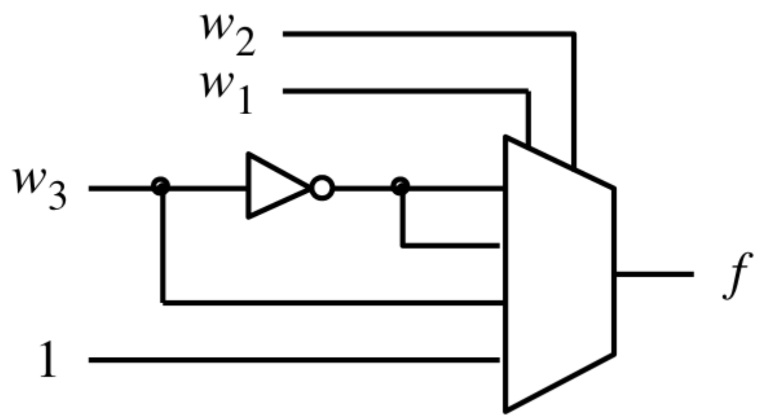
these are the 4 cofactors

Factor and implement the following function with a 4-to-1 multiplexer

$$f = \overline{w}_1\overline{w}_3 + w_1w_2 + w_1w_3$$

$$\begin{aligned} f &= \overline{w}_1\overline{w}_2 f_{\overline{w}_1\overline{w}_2} + \overline{w}_1 w_2 f_{\overline{w}_1 w_2} + w_1\overline{w}_2 f_{w_1\overline{w}_2} + w_1 w_2 f_{w_1 w_2} \\ &= \overline{w}_1\overline{w}_2(\overline{w}_3) + \overline{w}_1 w_2(\overline{w}_3) + w_1\overline{w}_2(w_3) + w_1 w_2(1) \end{aligned}$$

Factor and implement the following function with a 4-to-1 multiplexer



$$f = \overline{w}_1\overline{w}_2 f_{\overline{w}_1\overline{w}_2} + \overline{w}_1 w_2 f_{\overline{w}_1 w_2} + w_1\overline{w}_2 f_{w_1\overline{w}_2} + w_1 w_2 f_{w_1 w_2}$$
$$= \overline{w}_1\overline{w}_2(\overline{w}_3) + \overline{w}_1 w_2(\overline{w}_3) + w_1\overline{w}_2(w_3) + w_1 w_2(1)$$

[Figure 4.11b from the textbook]

Yet Another Example

Factor and implement the following function using <u>only</u> 2x1 multiplexers

$$f = w_1w_2 + w_1w_3 + w_2w_3$$

Factor and implement the following function using <u>only</u> 2x1 multiplexers

$$f = w_1 w_2 + w_1 w_3 + w_2 w_3$$

$$\begin{aligned} f &= \overline{w}_1 (w_2 w_3) + w_1 (w_2 + w_3 + w_2 w_3) \\ &= \overline{w}_1 (w_2 w_3) + w_1 (w_2 + w_3) \end{aligned}$$

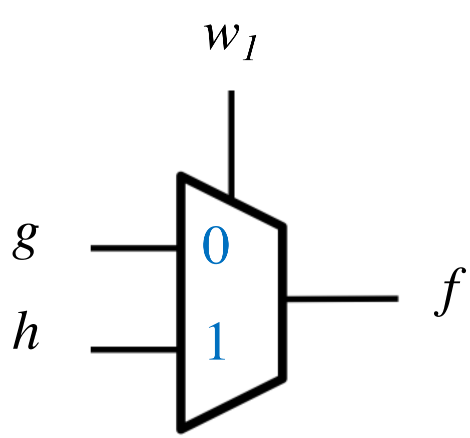
Factor and implement the following function using only 2x1 multiplexers

$$f = w_1 w_2 + w_1 w_3 + w_2 w_3$$

$$\begin{aligned} f &= \overline{w}_1 (w_2 w_3) + w_1 (w_2 + w_3 + w_2 w_3) \\ &= \overline{w}_1 (\underbrace{w_2 w_3}) + w_1 (\underbrace{w_2 + w_3}) \end{aligned}$$

$$g = w_2 w_3 \qquad h = w_2 + w_3$$

Factor and implement the following function using <u>only</u> 2-to-1 multiplexers



$$f = \overline{w}_1(w_2w_3) + w_1(w_2 + w_3 + w_2w_3)$$

$$= \overline{w}_1(w_2w_3) + w_1(w_2 + w_3)$$

$$g = w_2w_3 \qquad h = w_2 + w_3$$

Factor and implement the following function using only 2x1 multiplexers

$g = w_2 w_3$

$h = w_2 + w_3$

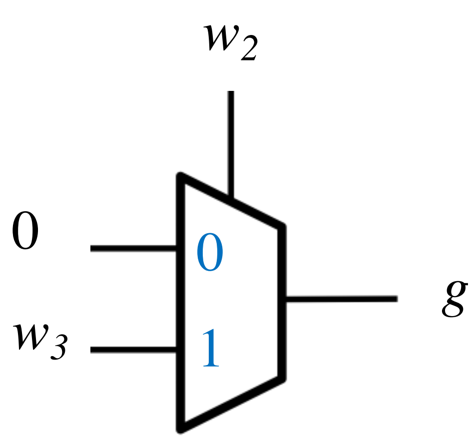
Factor and implement the following function using <u>only</u> 2x1 multiplexers

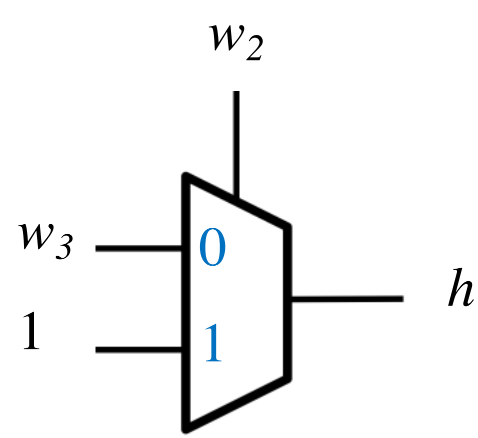
$$g = w_2 w_3 \qquad h = w_2 + w_3$$

↓ ↓

$$g = \overline{w}_2(0) + w_2(w_3) \qquad h = \overline{w}_2(w_3) + w_2(1)$$

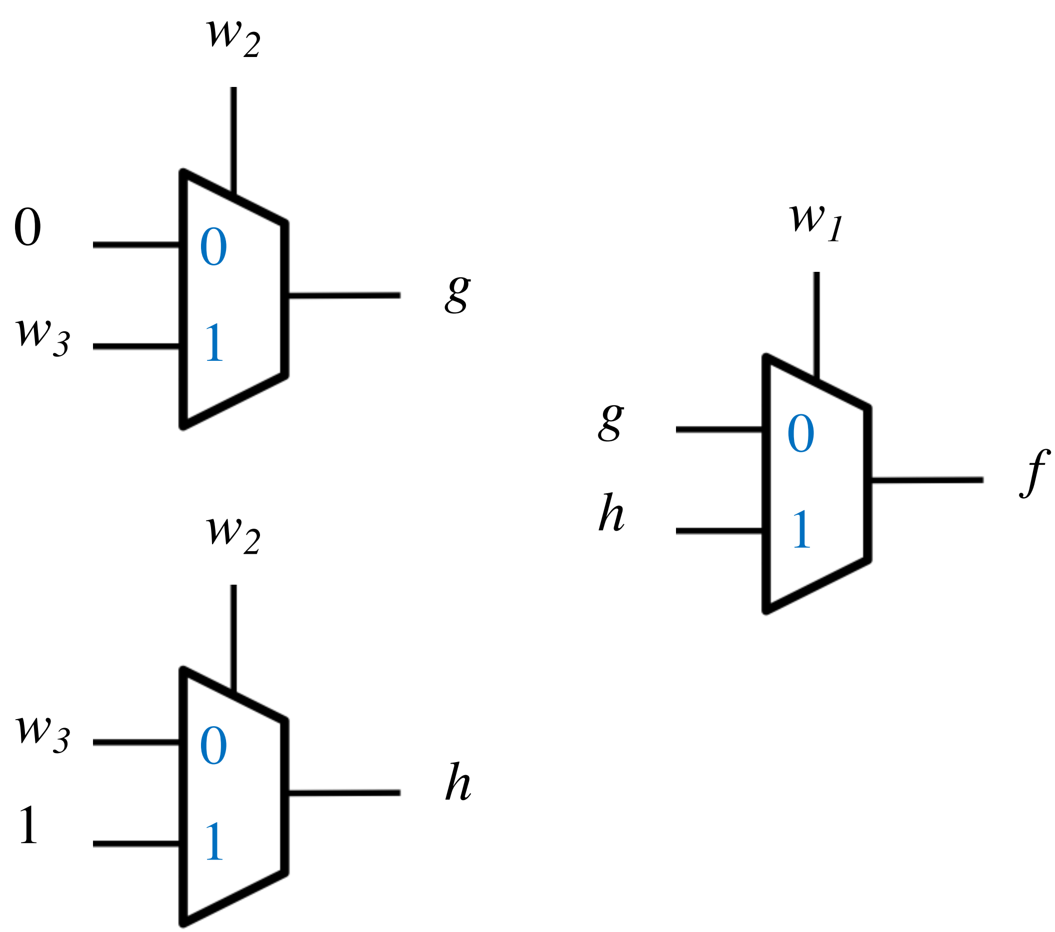
Factor and implement the following function using <u>only</u> 2-to-1 multiplexers



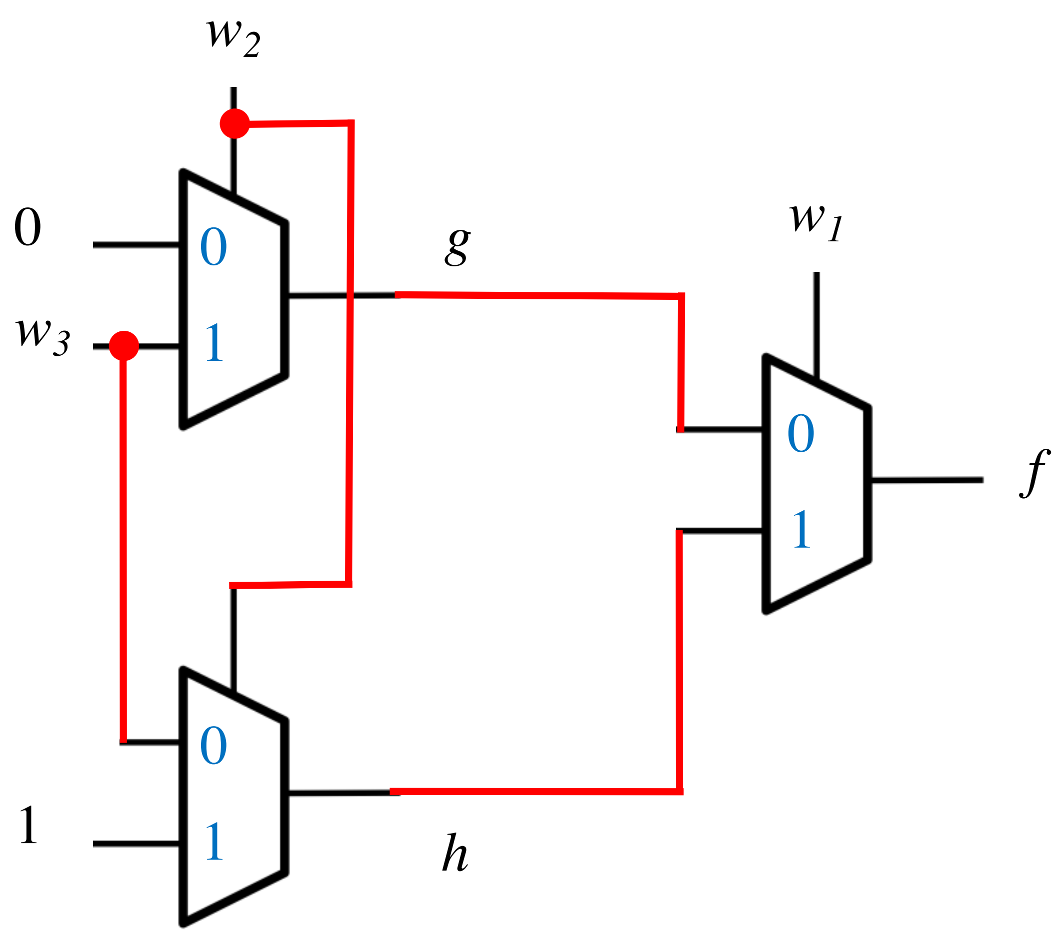


$$g = \overline{w}_2(0) + w_2(w_3) \qquad h = \overline{w}_2(w_3) + w_2(1)$$

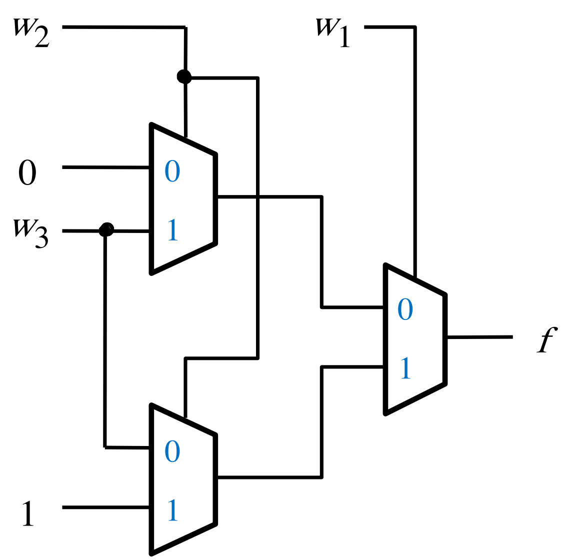
Finally, we are ready to draw the circuit



Finally, we are ready to draw the circuit



Finally, we are ready to draw the circuit



[Figure 4.12 from the textbook]

Decoders

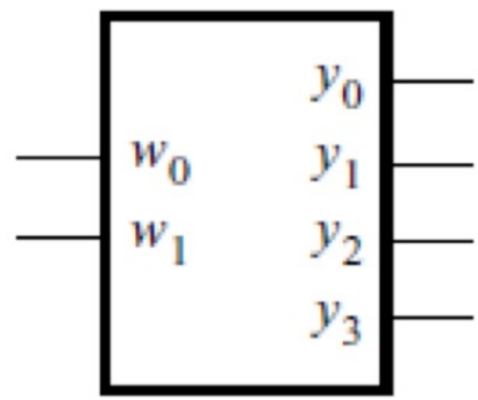
2-to-4 Decoder (Definition)

- Has two inputs: w_1 and w_0
- Has four outputs: y_0 , y_1 , y_2 , and y_3
- If w_1=0 and w_0=0, then the output y_0 is set to 1
- If w_1=0 and w_0=1, then the output y_1 is set to 1
- If w_1=1 and w_0=0, then the output y_2 is set to 1
- If w_1=1 and w_0=1, then the output y_3 is set to 1
- Only one output is set to 1. All others are set to 0.

Truth Table and Graphical Symbol for a 2-to-4 Decoder

w_1	w_0	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

(a) Truth table



(b) Graphical symbol

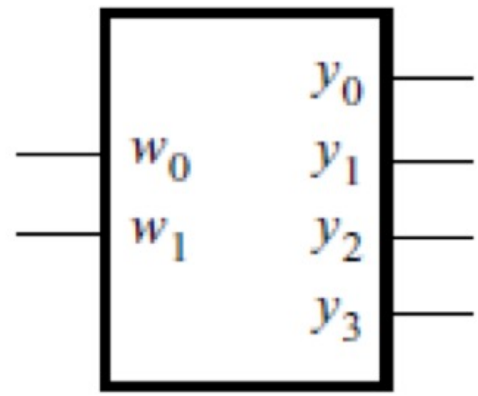
[Figure 4.13a-b from the textbook]

Truth Table and Graphical Symbol for a 2-to-4 Decoder

w_1	w_0	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

The outputs are "one-hot" encoded

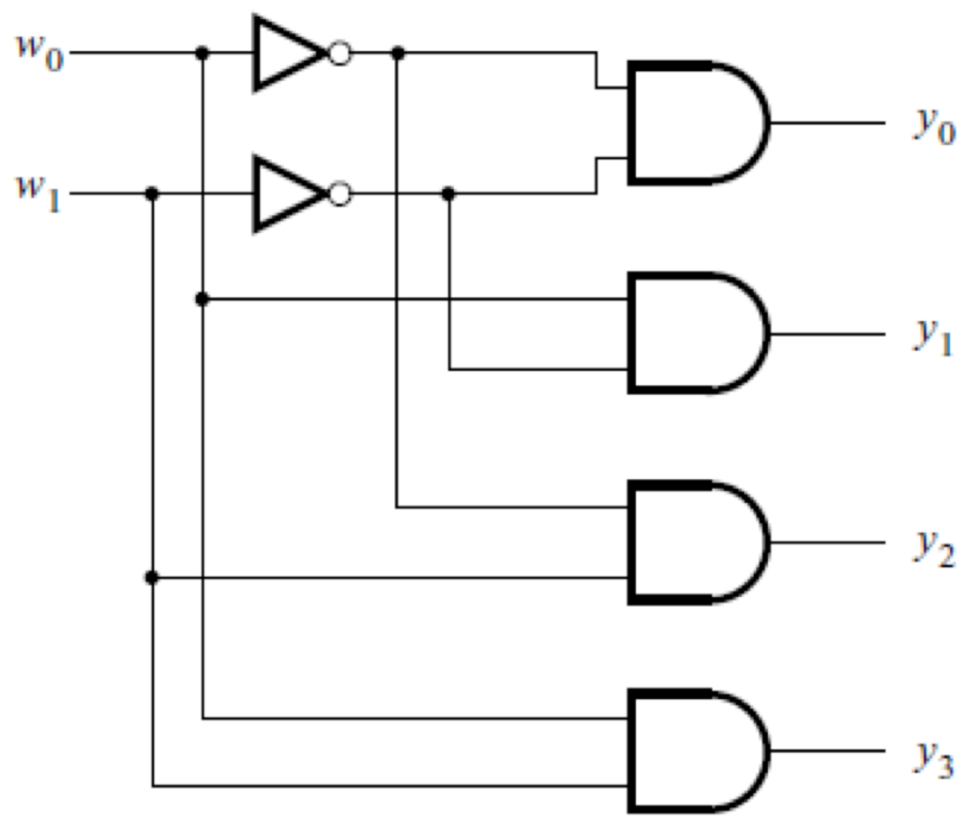
(a) Truth table



(b) Graphical symbol

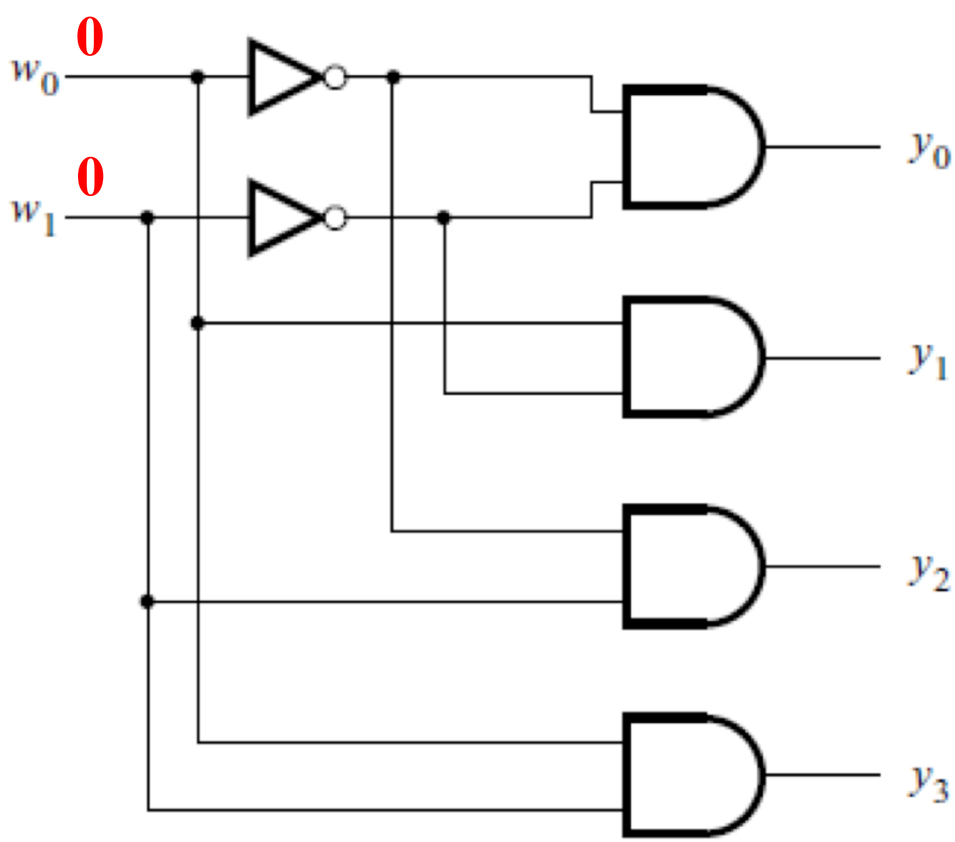
[Figure 4.13a-b from the textbook]

The Logic Circuit for a 2-to-4 Decoder



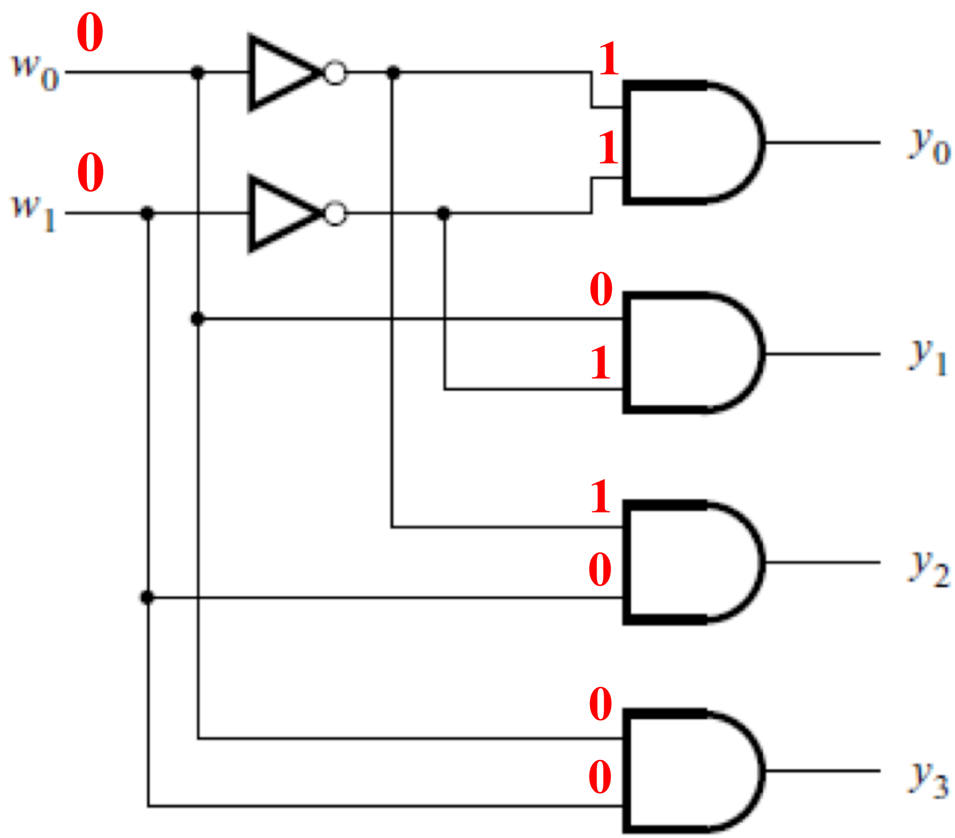
[Figure 4.13c from the textbook]

The Logic Circuit
for a 2-to-4 Decoder



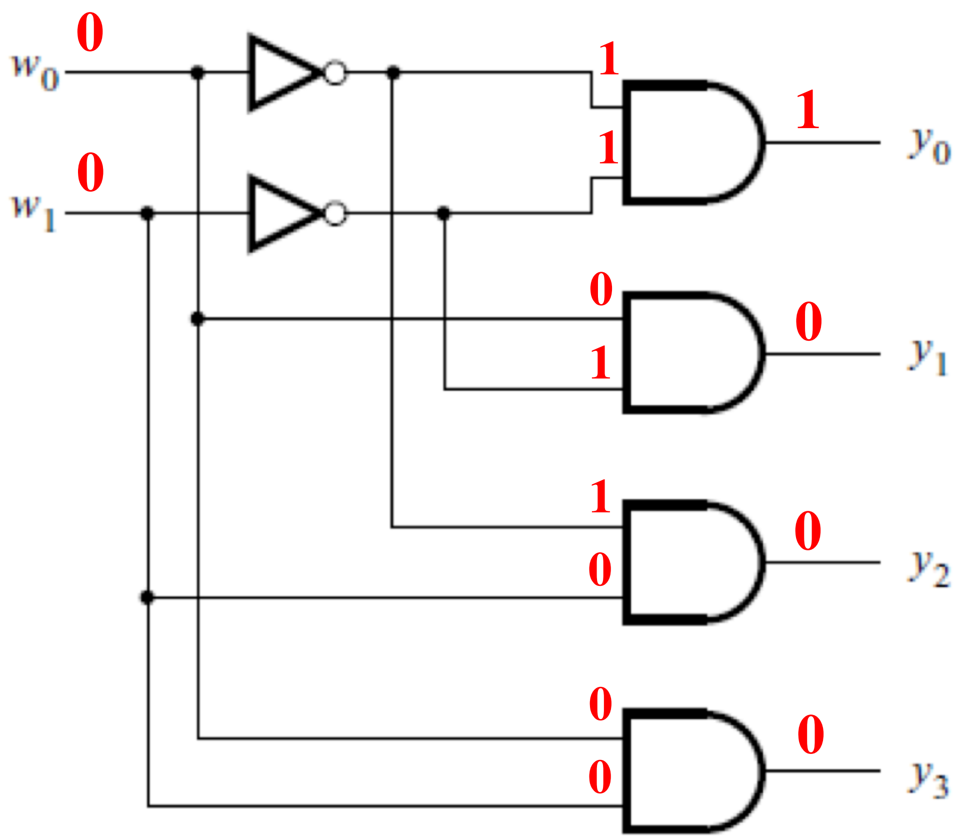
[Figure 4.13c from the textbook]

The Logic Circuit for a 2-to-4 Decoder



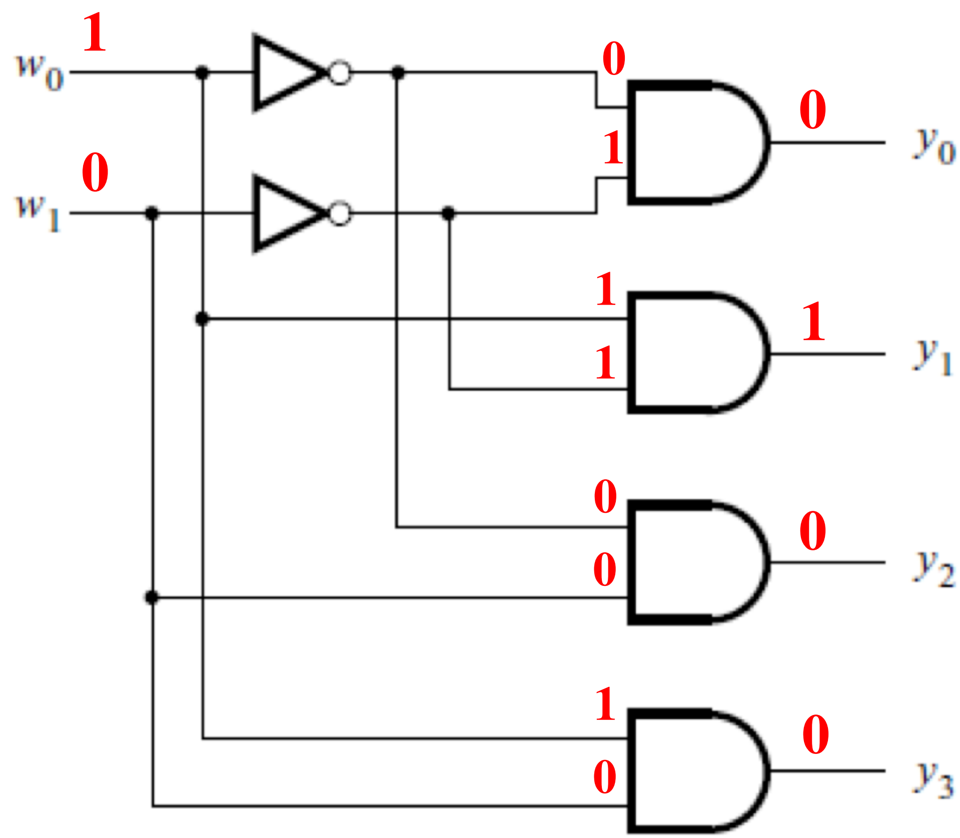
[Figure 4.13c from the textbook]

The Logic Circuit for a 2-to-4 Decoder



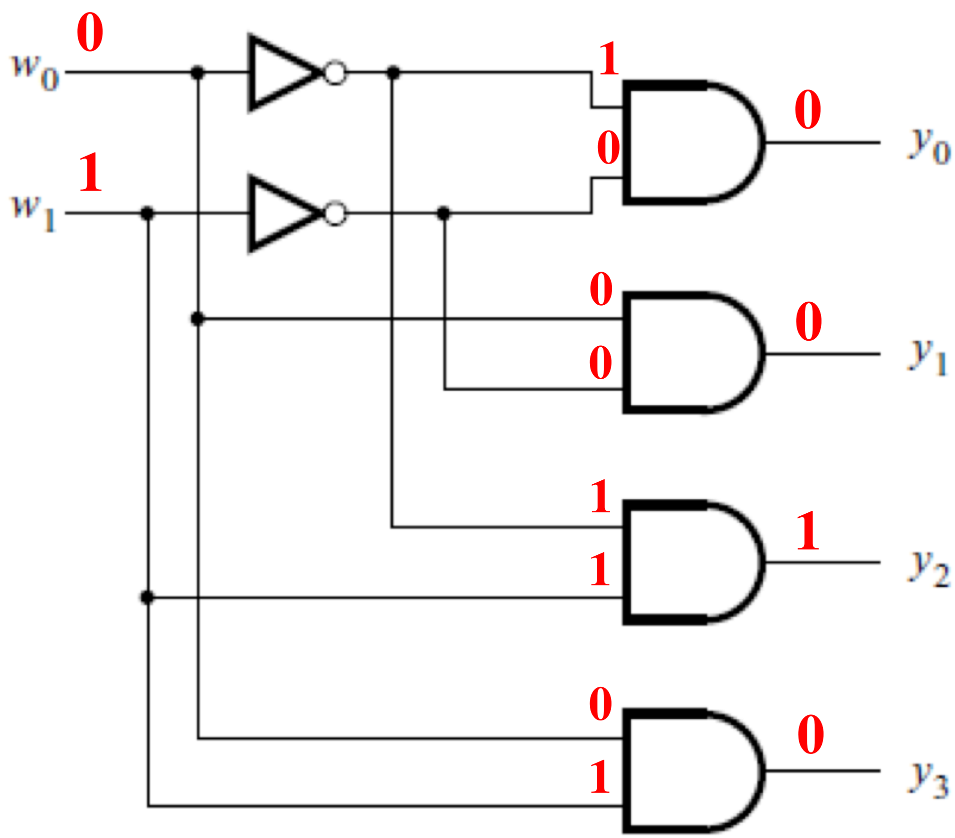
[Figure 4.13c from the textbook]

The Logic Circuit for a 2-to-4 Decoder



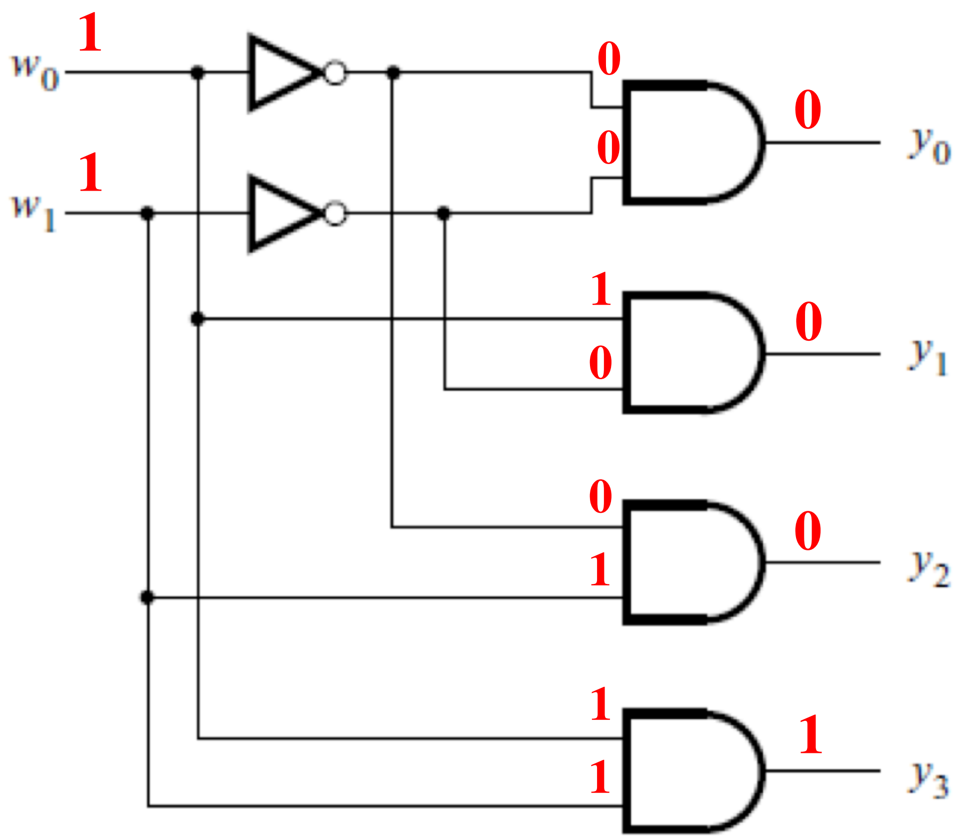
[Figure 4.13c from the textbook]

The Logic Circuit for a 2-to-4 Decoder



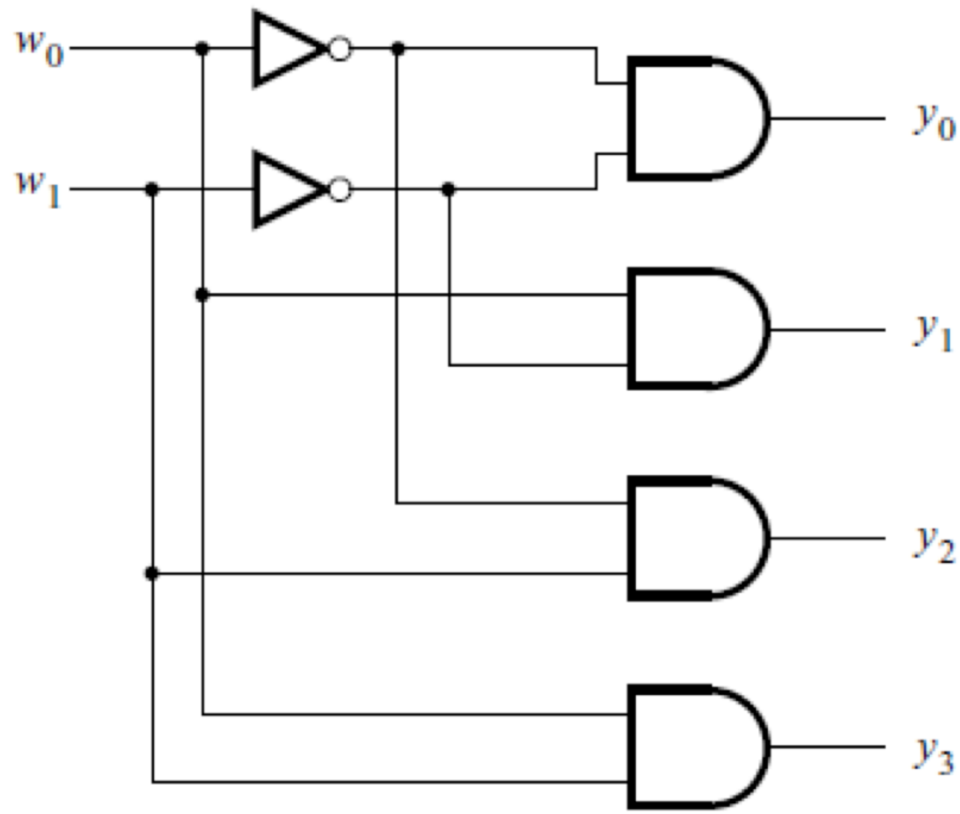
[Figure 4.13c from the textbook]

The Logic Circuit for a 2-to-4 Decoder



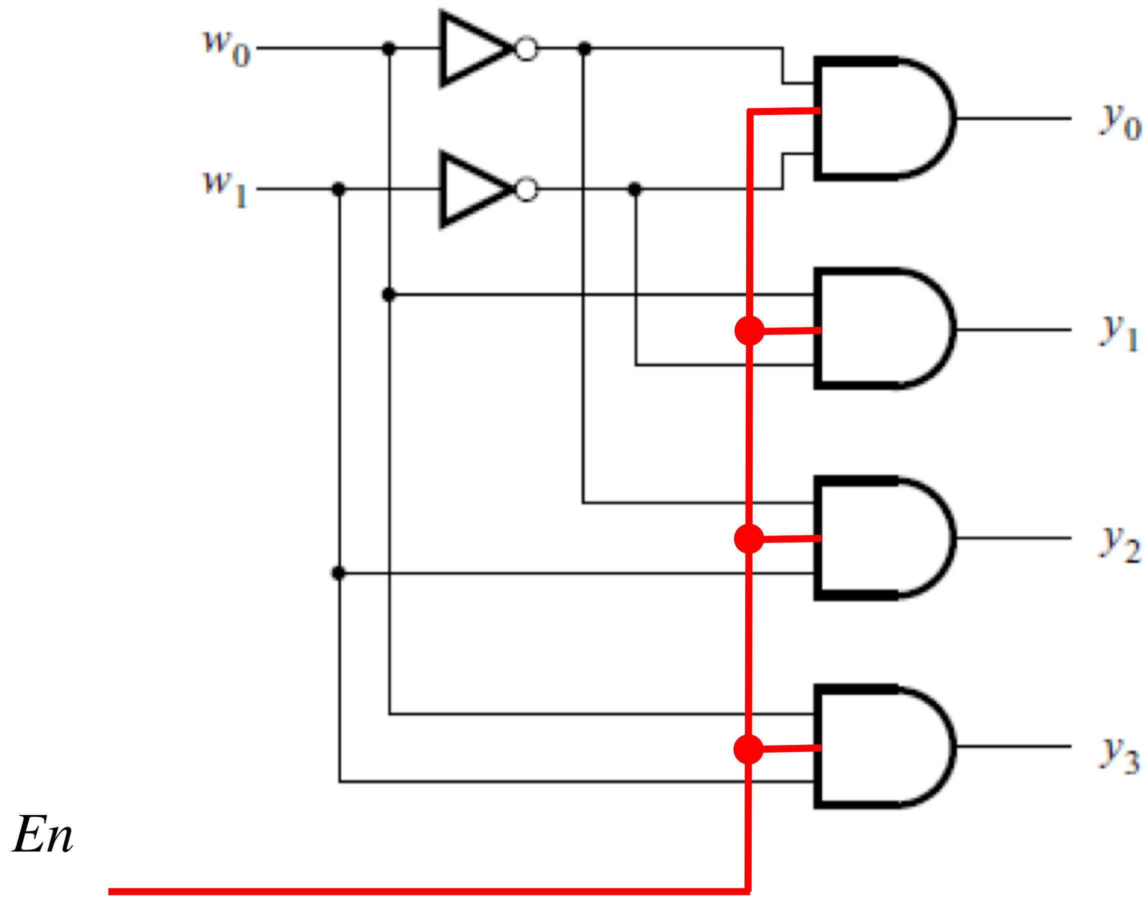
[Figure 4.13c from the textbook]

Adding an Enable Input



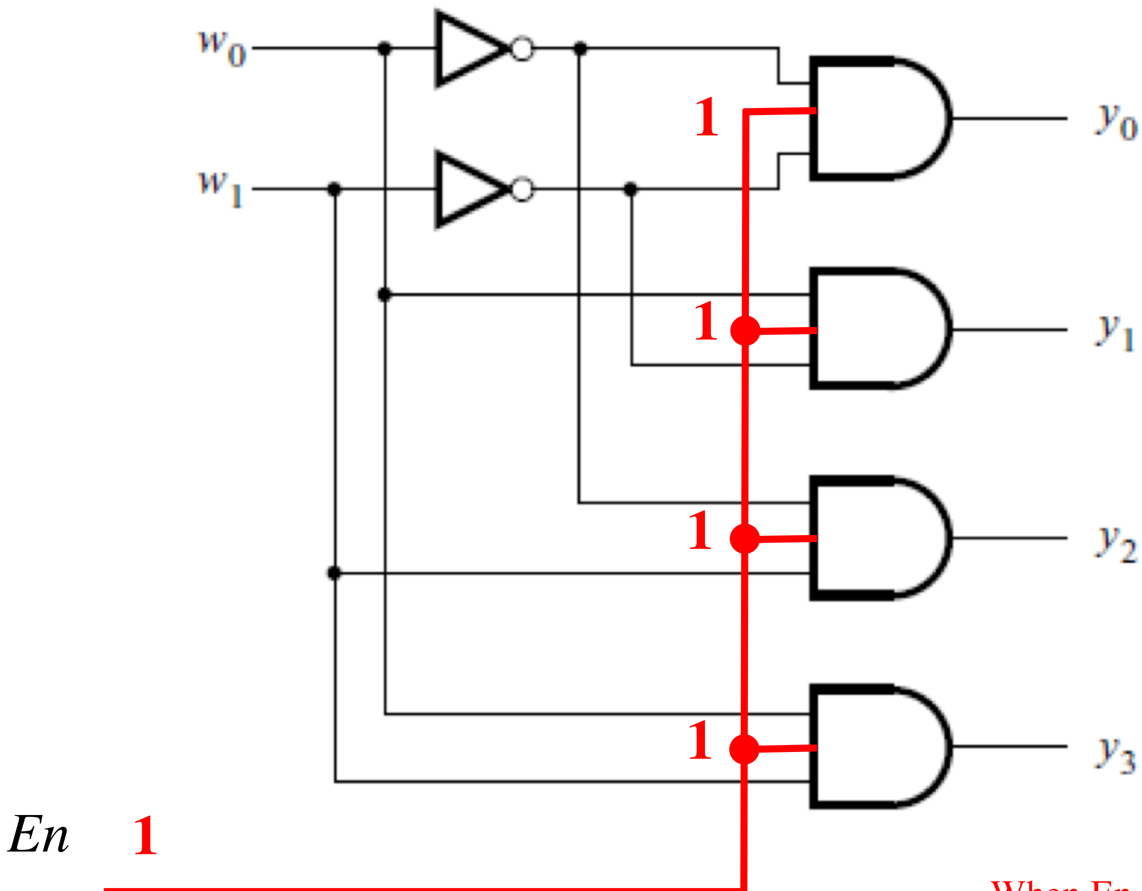
[Figure 4.13c from the textbook]

Adding an Enable Input



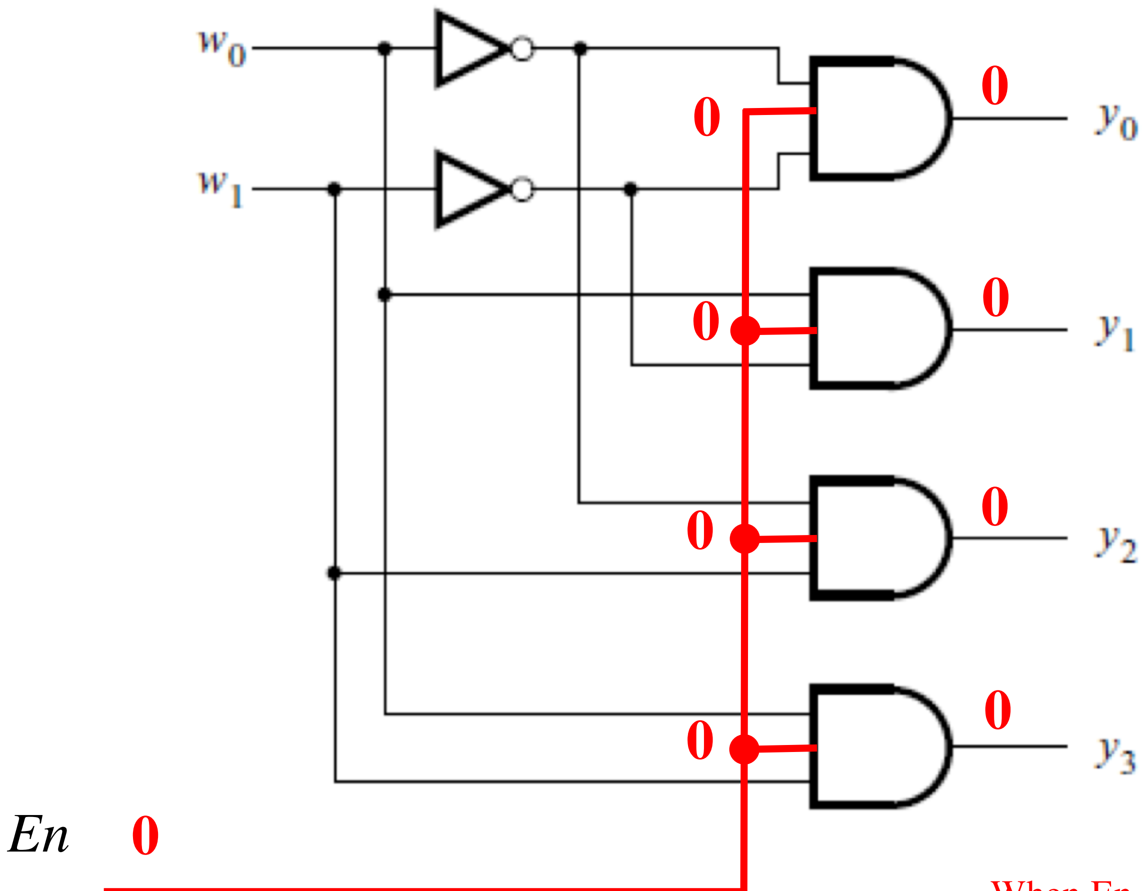
[Figure 4.14c from the textbook]

Adding an Enable Input



When En=1 this circuit behaves like the one without enable.

Adding an Enable Input

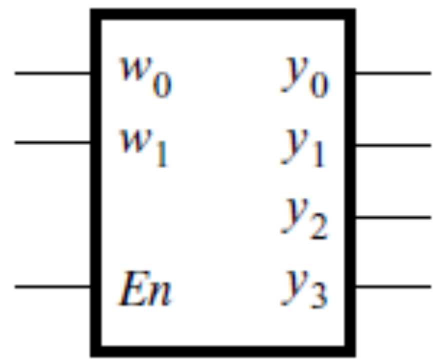


When En=0 all outputs are set to 0 and the decoder is disabled.

Truth Table and Graphical Symbol for a 2-to-4 Decoder with an Enable Input

En	w_1	w_0	y_0	y_1	y_2	y_3
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
0	x	x	0	0	0	0

(a) Truth table



(b) Graphical symbol

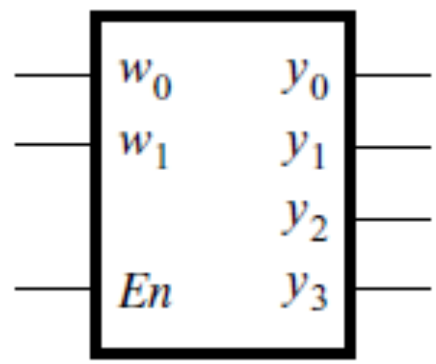
[Figure 4.14a-b from the textbook]

Truth Table and Graphical Symbol for a 2-to-4 Decoder with an Enable Input

En	w_1	w_0	y_0	y_1	y_2	y_3
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
0	x	x	0	0	0	0

(a) Truth table

x indicates that it does not matter what the value of this variable is for this row of the truth table

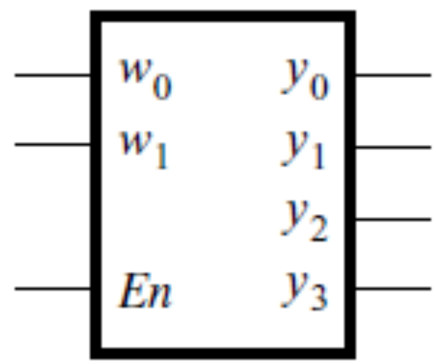


(b) Graphical symbol

[Figure 4.14a-b from the textbook]

Truth Table and Graphical Symbol for a 2-to-4 Decoder with an Enable Input

En	w_1	w_0	y_0	y_1	y_2	y_3
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
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	1	0	0	0	0	0
0	1	1	0	0	0	0

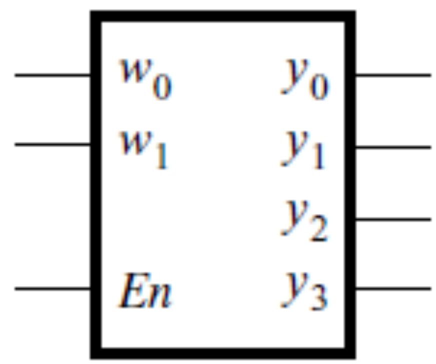


(b) Graphical symbol

[Figure 4.14a-b from the textbook]

Truth Table and Graphical Symbol for a 2-to-4 Decoder with an Enable Input

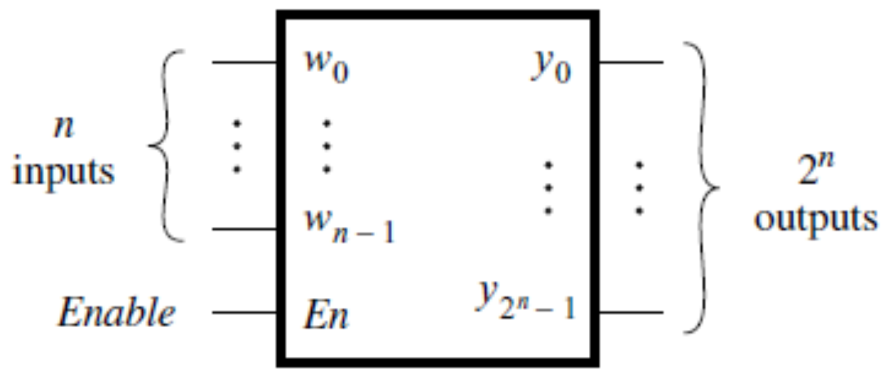
En	w_1	w_0	y_0	y_1	y_2	y_3
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	1	0	0	0	0	0
0	1	1	0	0	0	0
1	0	0	1	0	0	0
1	0	1	0	1	0	0
1	1	0	0	0	1	0
1	1	1	0	0	0	1



(b) Graphical symbol

[Figure 4.14a-b from the textbook]

Graphical Symbol for a Binary n-to-2^n Decoder with an Enable Input



(d) An n-to-2^n decoder

A binary decoder with n inputs has 2^n outputs

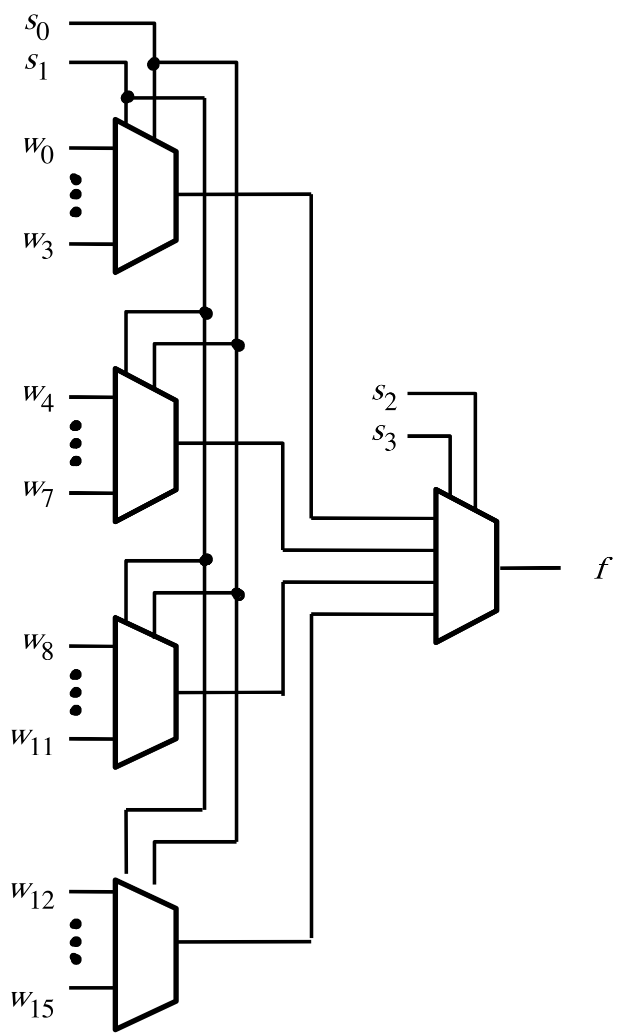
The outputs of an enabled binary decoder are "one-hot" encoded, meaning that only a single bit is set to 1, i.e., it is *hot*.

[Figure 4.14d from the textbook]

How can we build larger decoders?

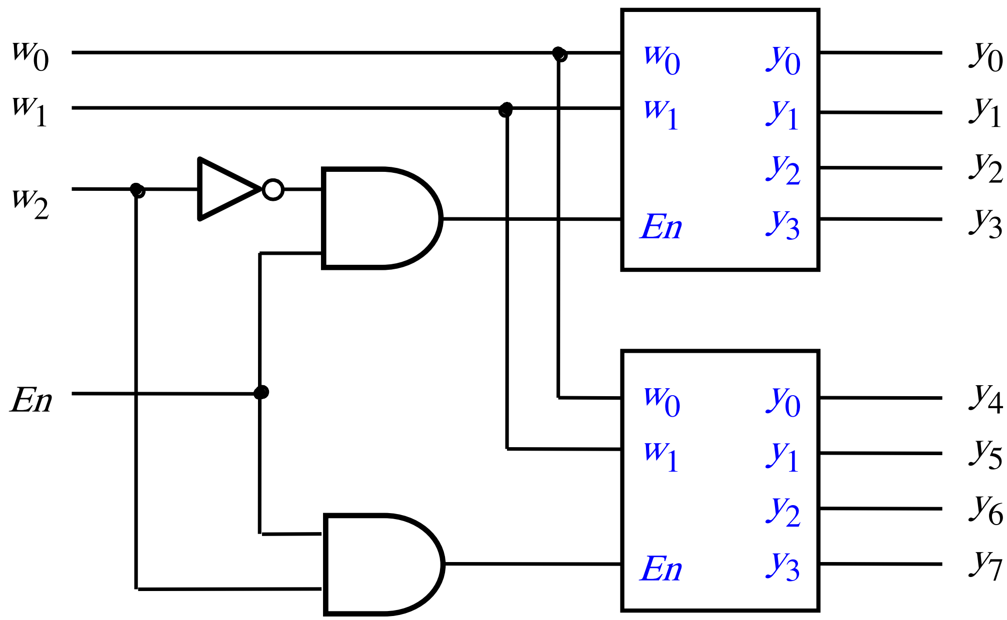
- 3-to-8 ?
- 4-to-16?
- 5-to-??

Hint: How did we build a 16-1 Multiplexer



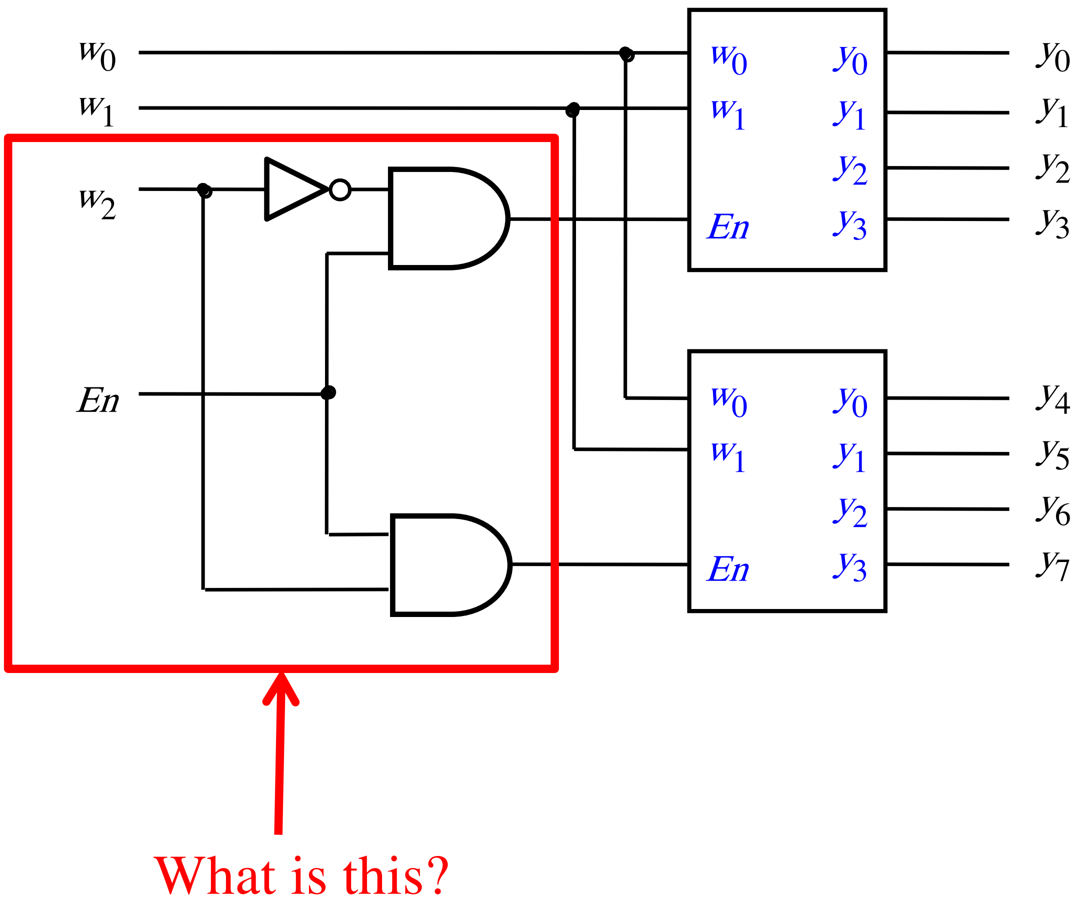
[Figure 4.4 from the textbook]

A 3-to-8 decoder using two 2-to-4 decoders



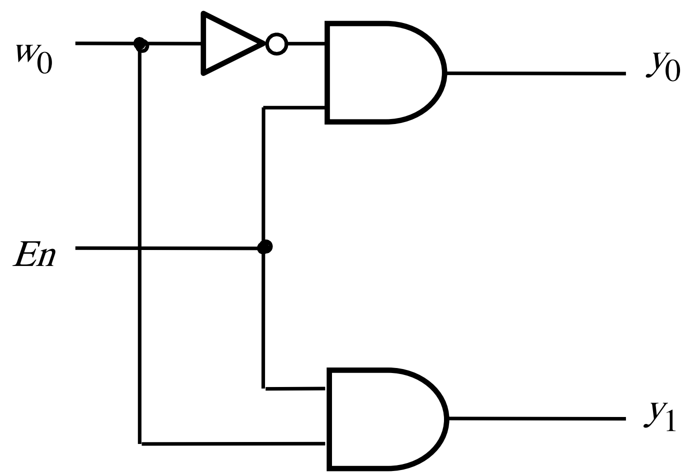
[Figure 4.15 from the textbook]

A 3-to-8 decoder using two 2-to-4 decoders

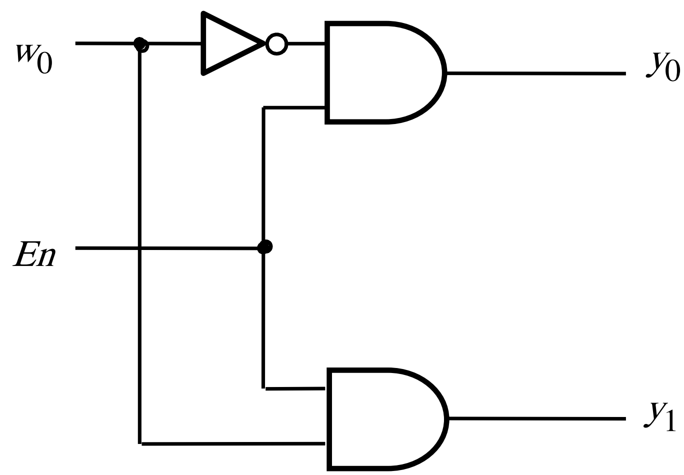


[Figure 4.15 from the textbook]

What is this?

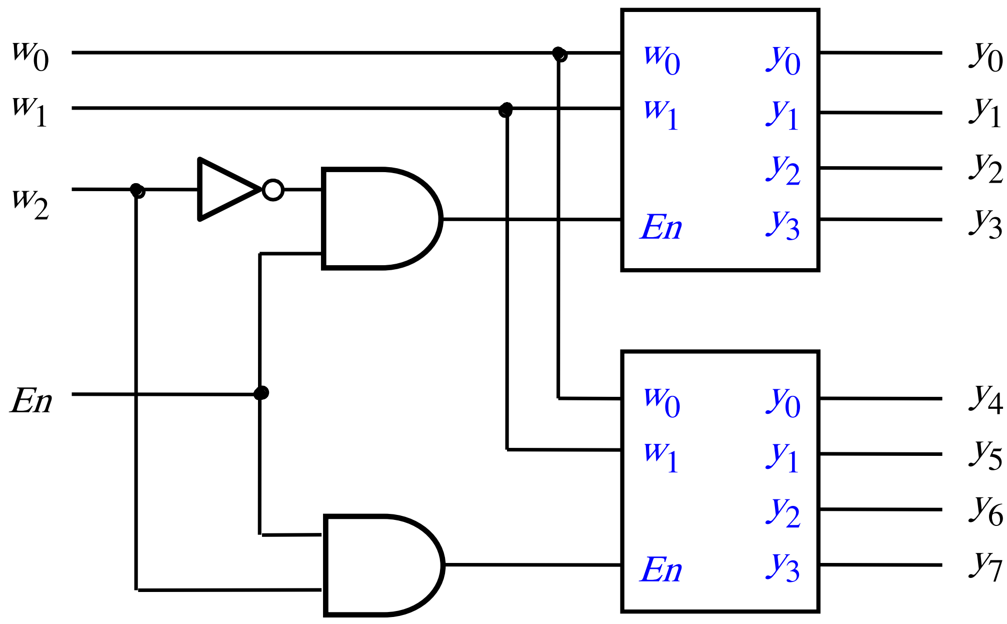


What is this?



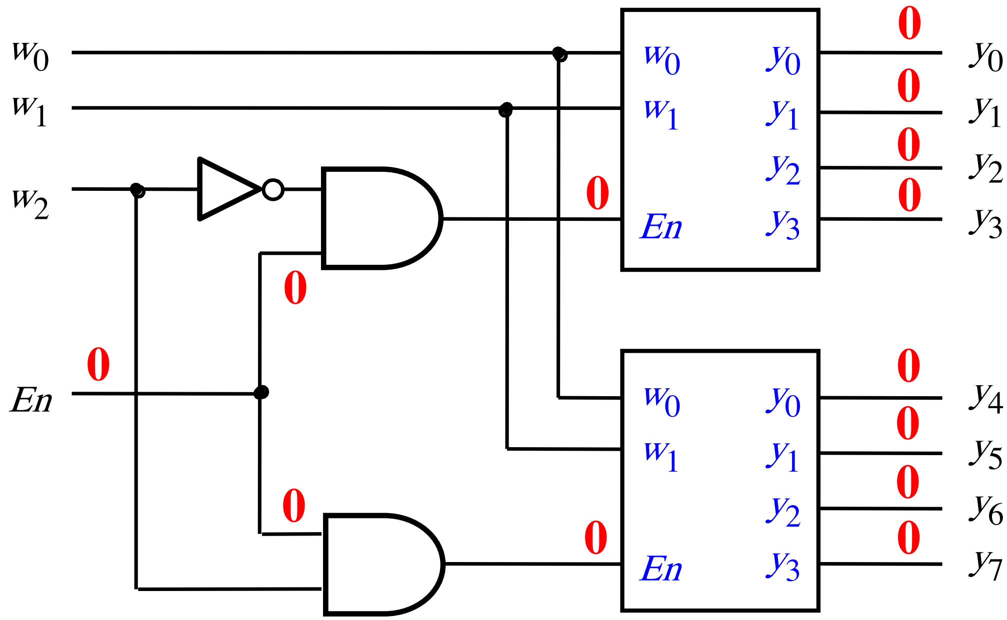
This is a 1-to-2 decoder with an enable input.

A 3-to-8 decoder using two 2-to-4 decoders



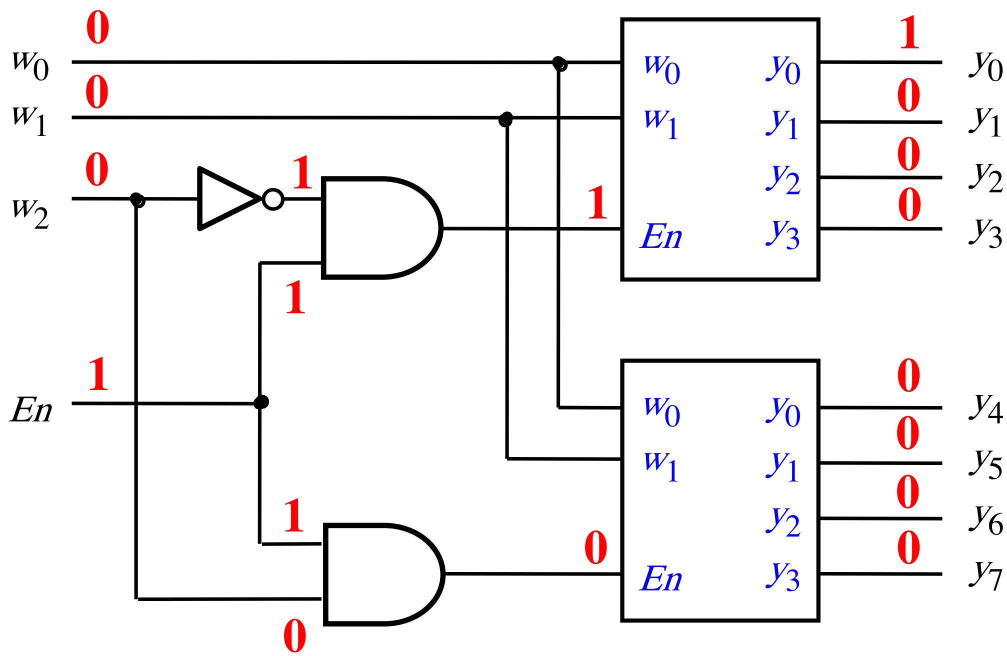
[Figure 4.15 from the textbook]

A 3-to-8 decoder using two 2-to-4 decoders



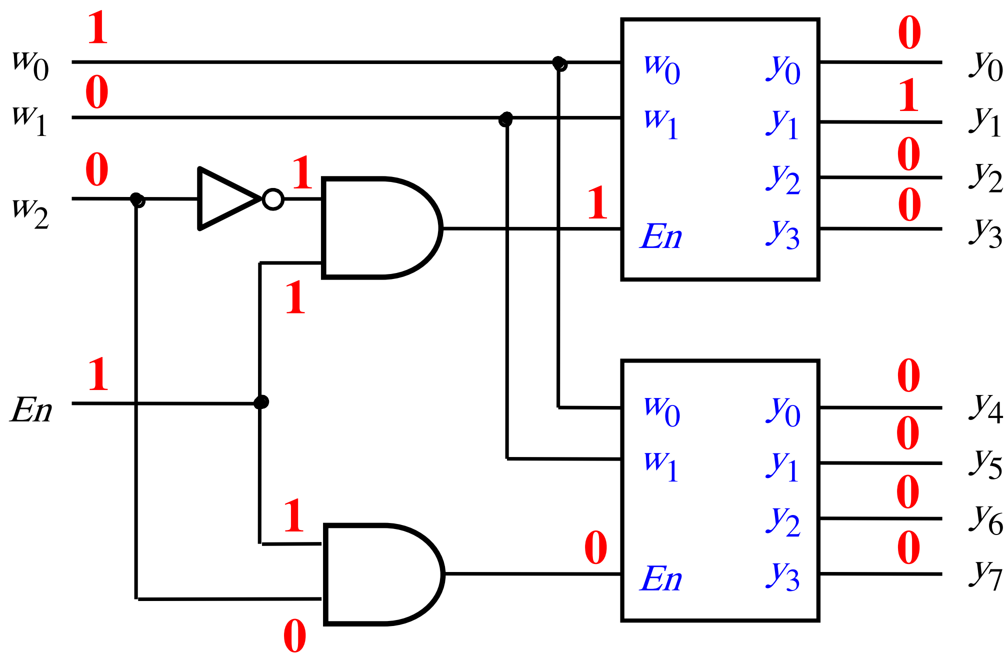
[Figure 4.15 from the textbook]

A 3-to-8 decoder using two 2-to-4 decoders



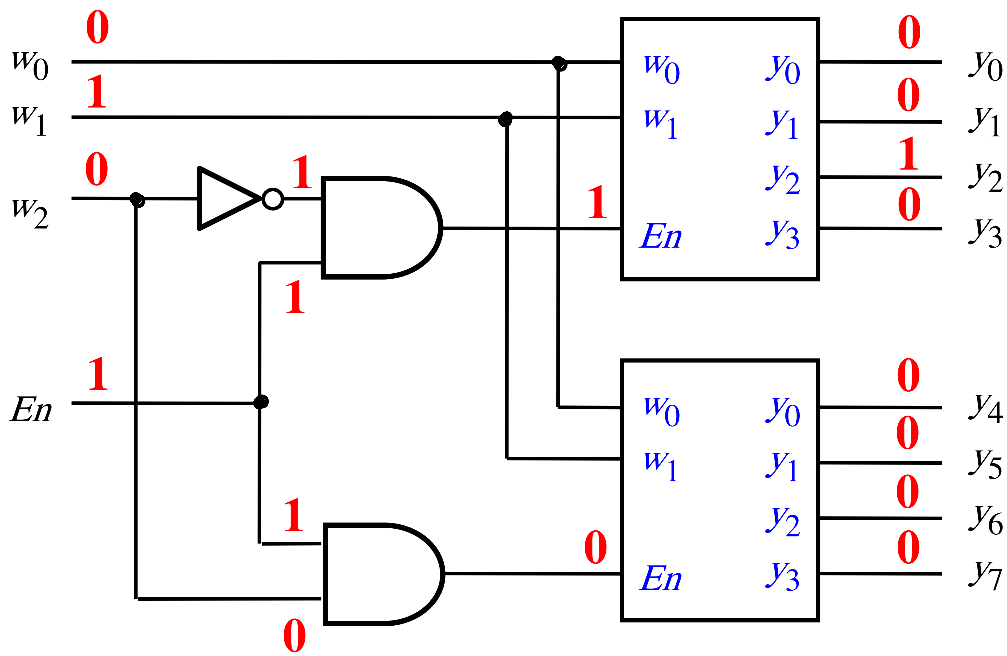
[Figure 4.15 from the textbook]

A 3-to-8 decoder using two 2-to-4 decoders



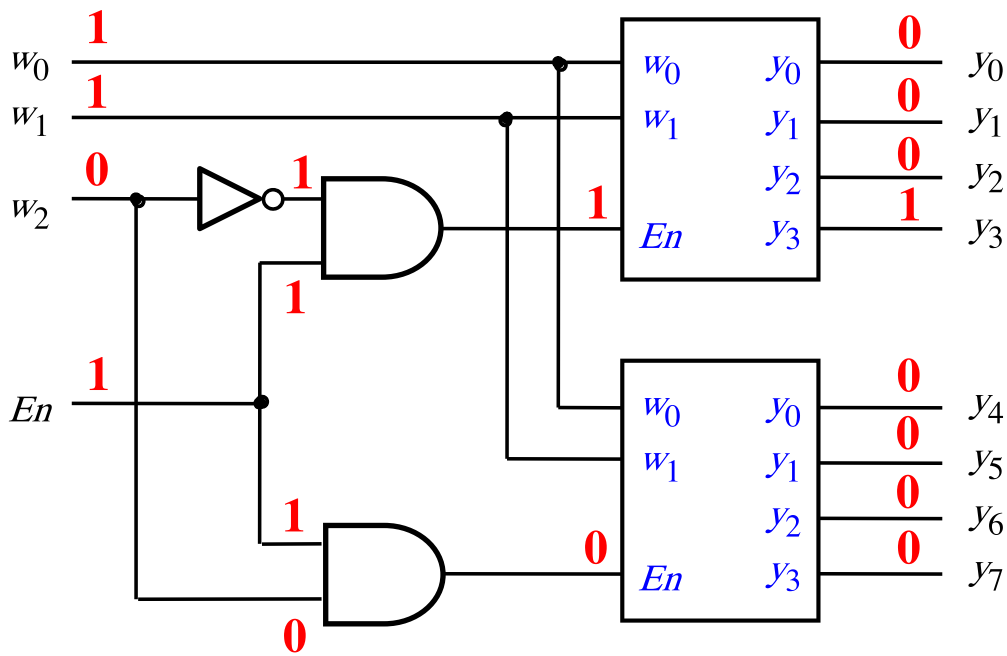
[Figure 4.15 from the textbook]

A 3-to-8 decoder using two 2-to-4 decoders



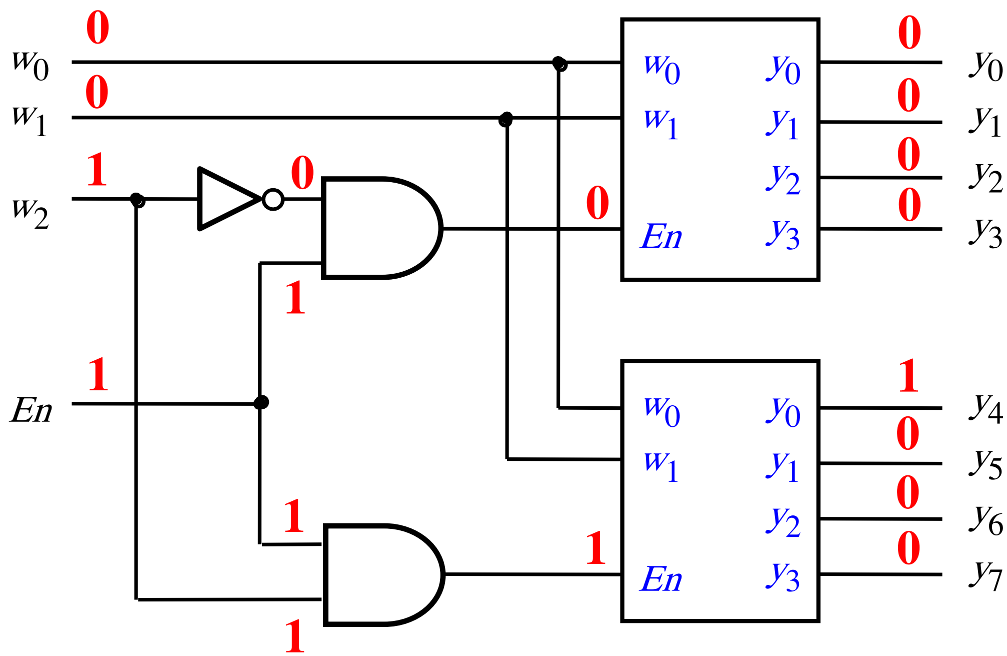
[Figure 4.15 from the textbook]

A 3-to-8 decoder using two 2-to-4 decoders



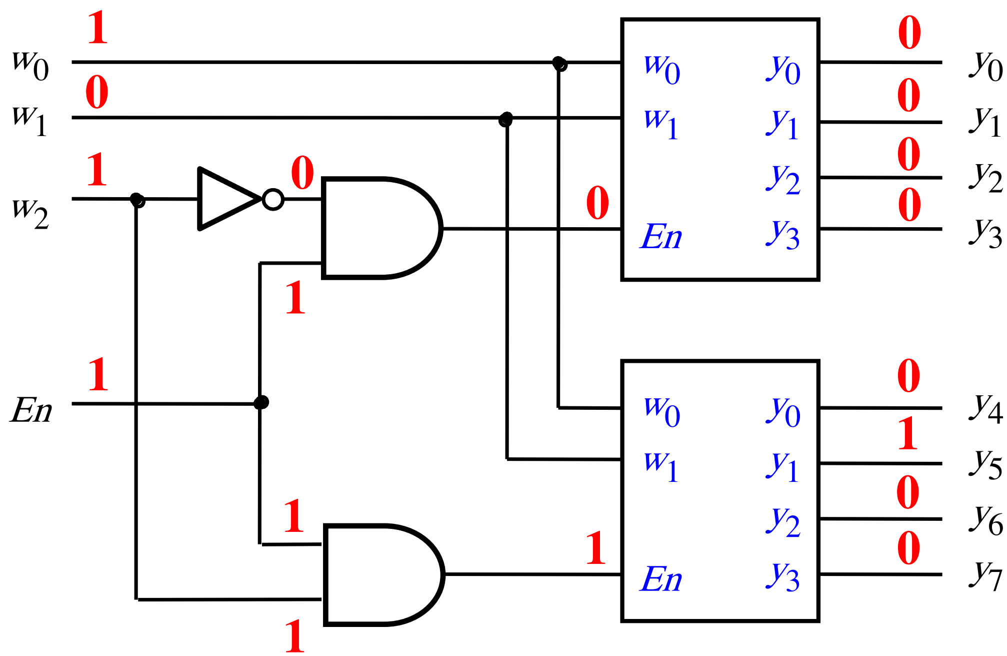
[Figure 4.15 from the textbook]

A 3-to-8 decoder using two 2-to-4 decoders



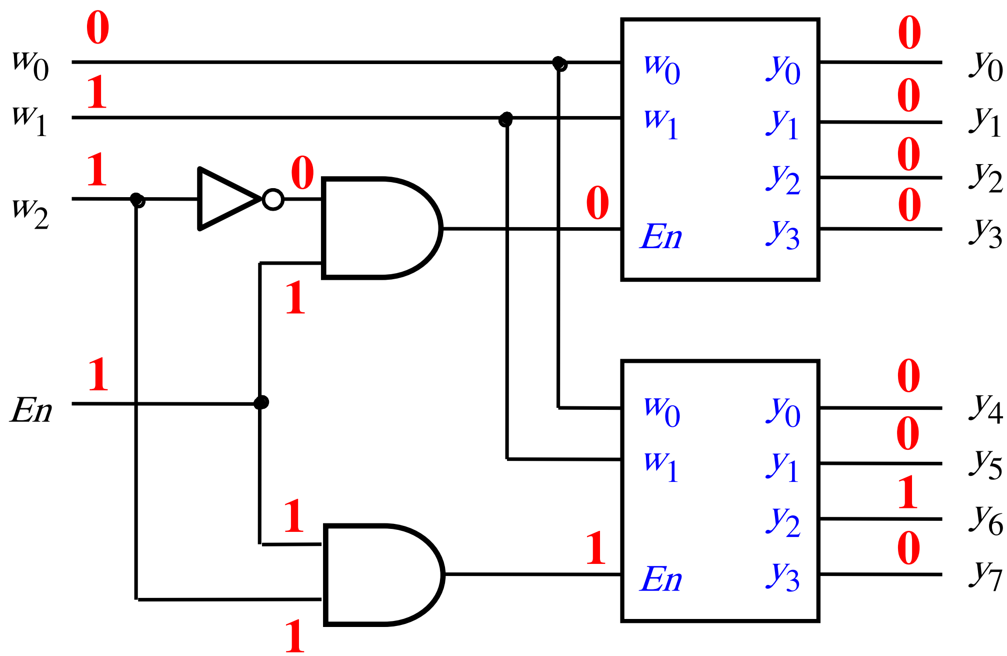
[Figure 4.15 from the textbook]

A 3-to-8 decoder using two 2-to-4 decoders



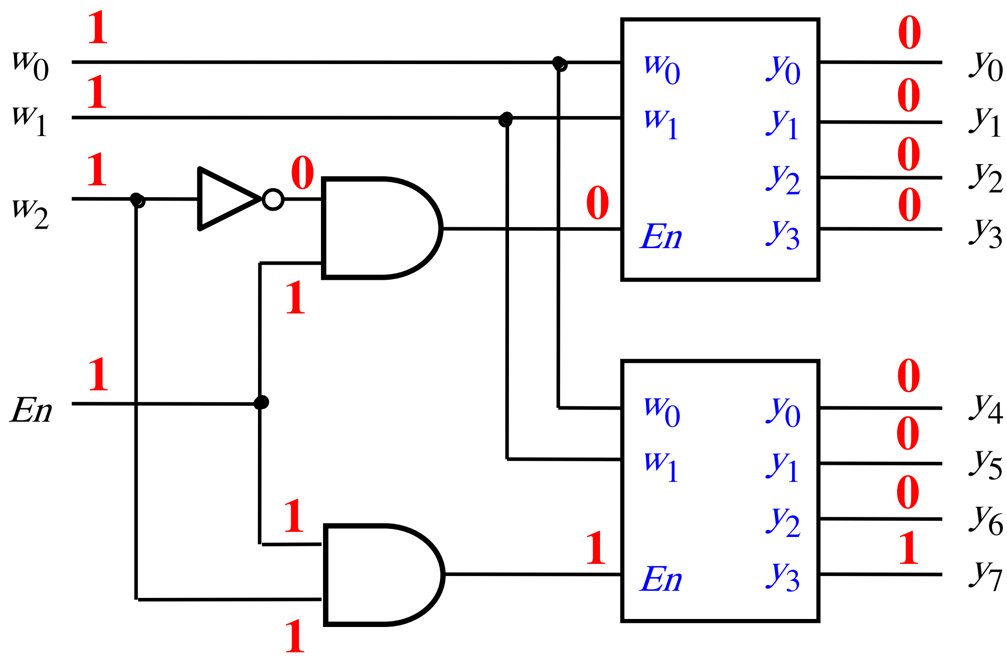
[Figure 4.15 from the textbook]

A 3-to-8 decoder using two 2-to-4 decoders



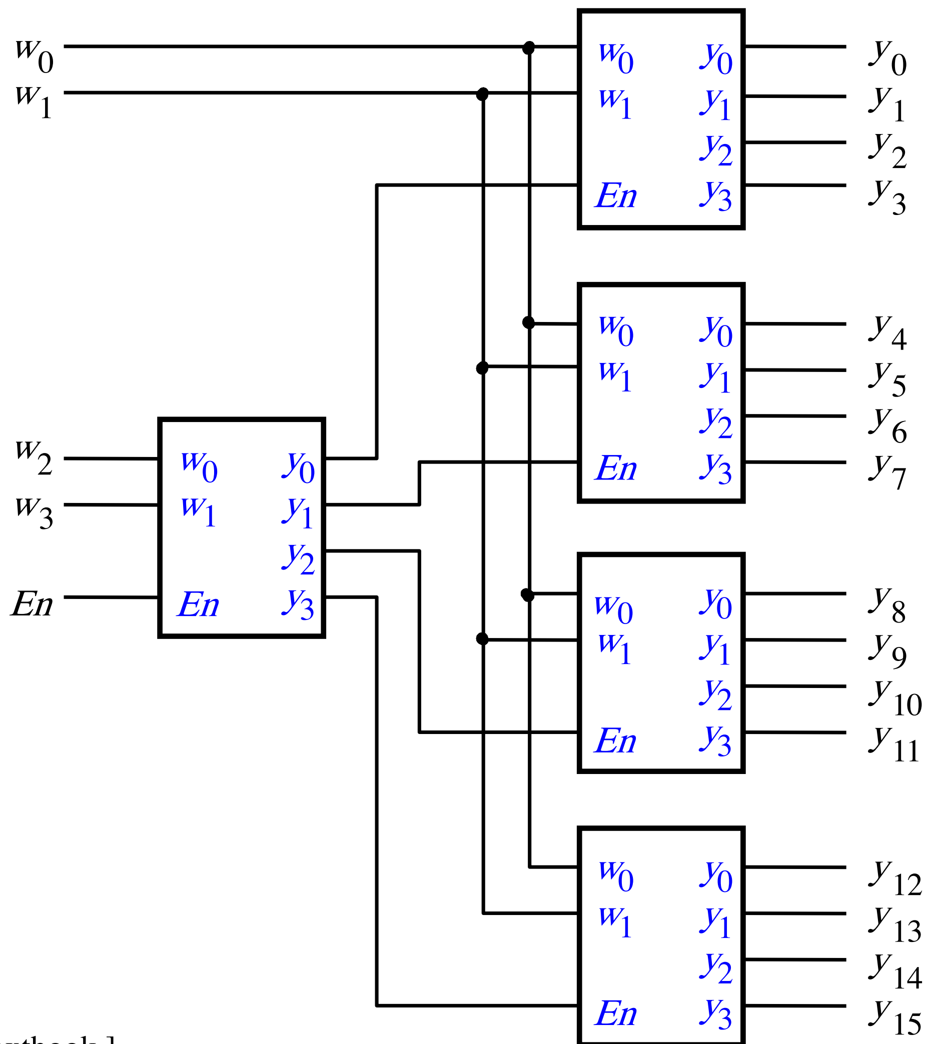
[Figure 4.15 from the textbook]

A 3-to-8 decoder using two 2-to-4 decoders



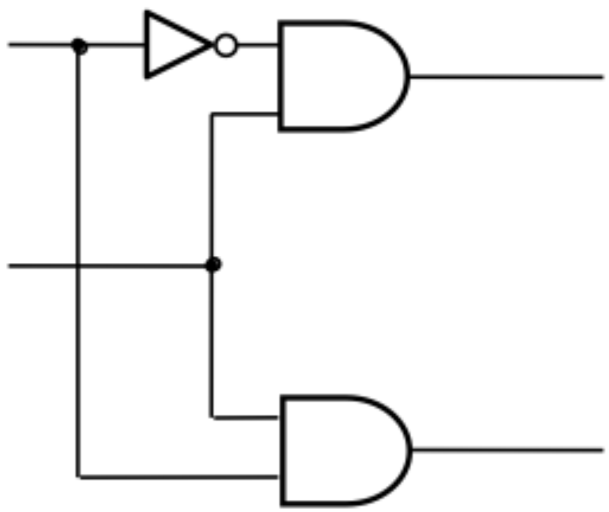
[Figure 4.15 from the textbook]

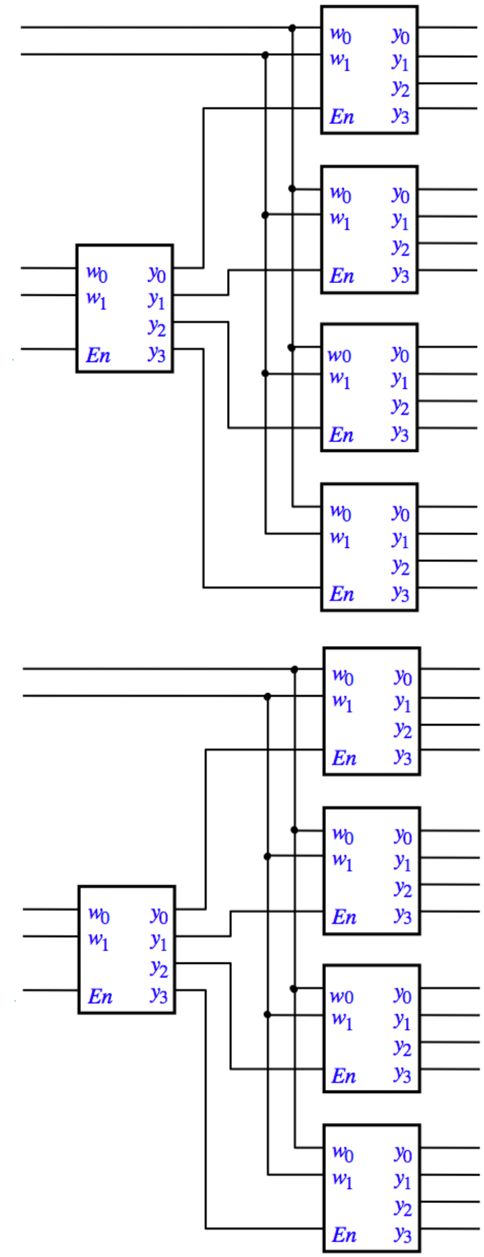
4-to-16 decoder built using a decoder tree



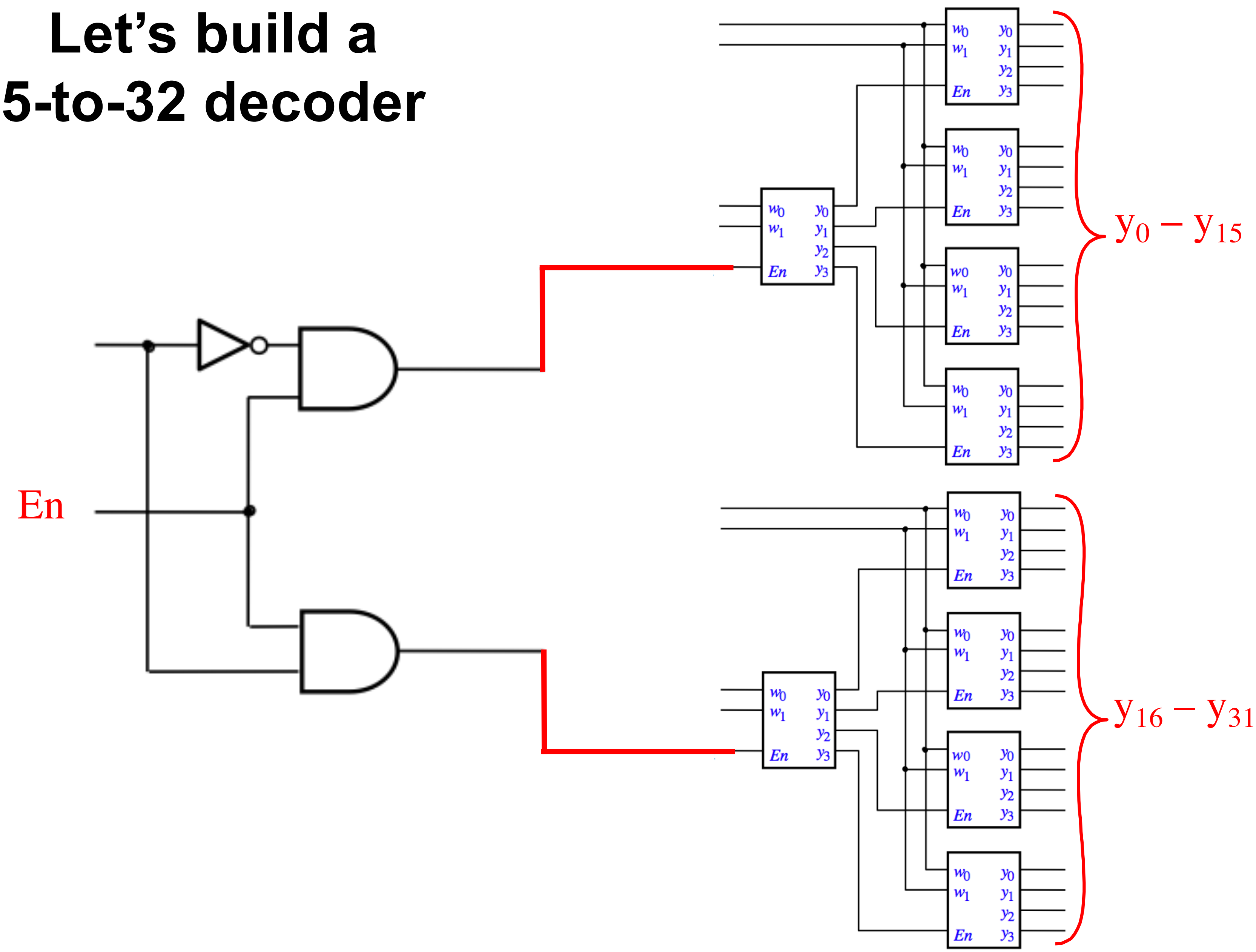
[Figure 4.16 from the textbook]

Let's build a 5-to-32 decoder

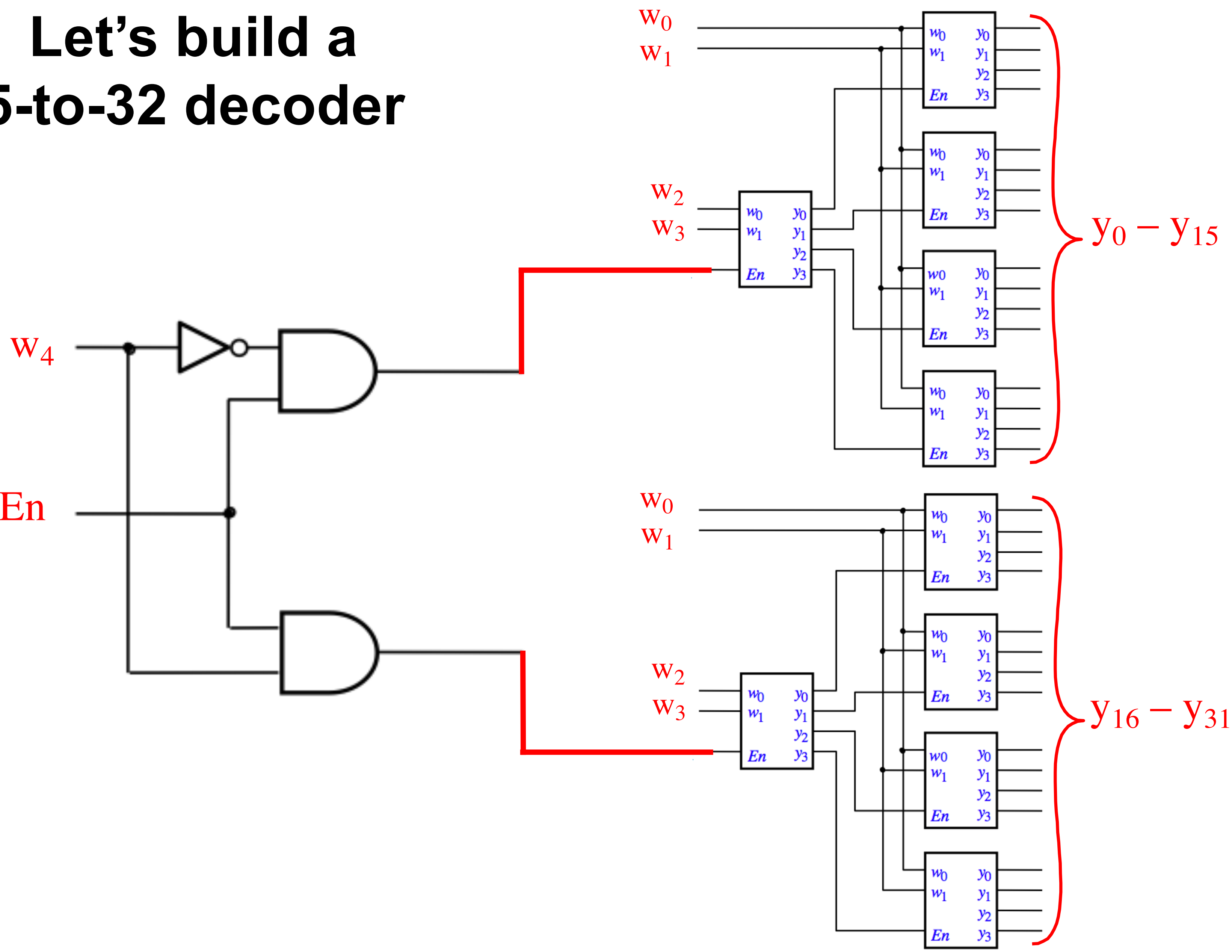




Let's build a 5-to-32 decoder



Let's build a 5-to-32 decoder



Demultiplexers

1-to-4 Demultiplexer (Definition)

- Has one data input line: D

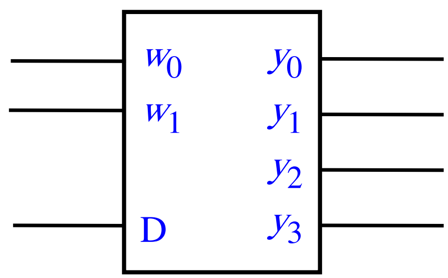
- Has two output select lines: w_1 and w_0

- Has four outputs: y_0 , y_1 , y_2 , and y_3

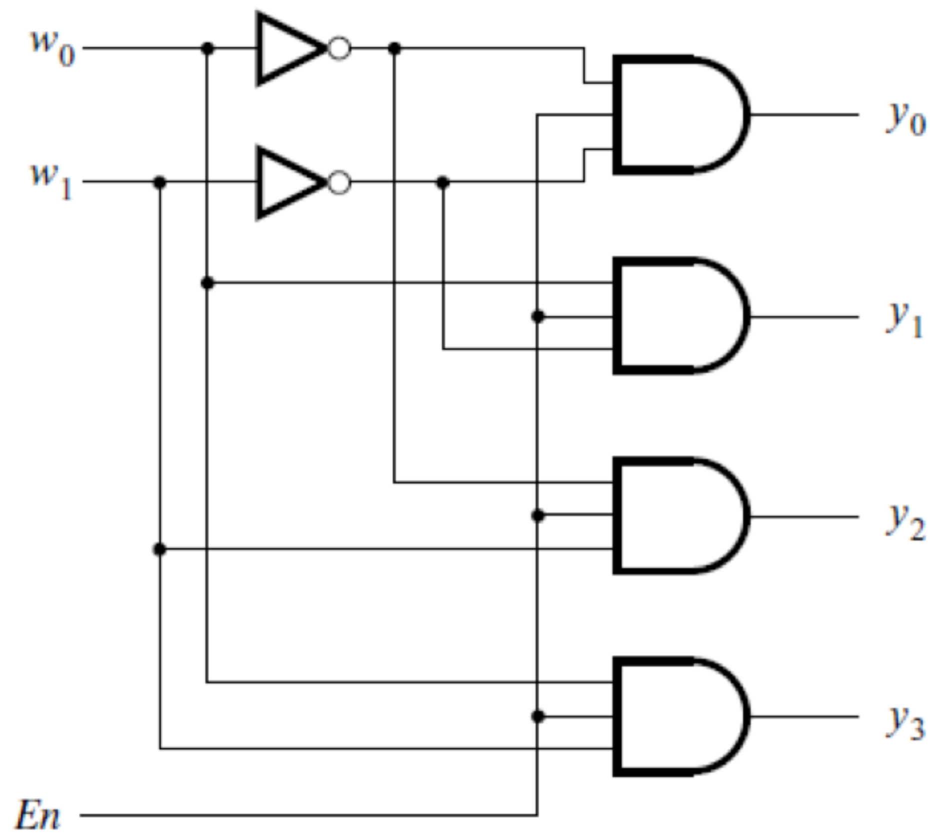
- If w_1=0 and w_0=0, then the output y_0 is set to D
- If w_1=0 and w_0=1, then the output y_1 is set to D
- If w_1=1 and w_0=0, then the output y_2 is set to D
- If w_1=1 and w_0=1, then the output y_3 is set to D

- Only one output is set to D. All others are set to 0.

Graphical Symbol for a 1-to-4 Demultiplexer



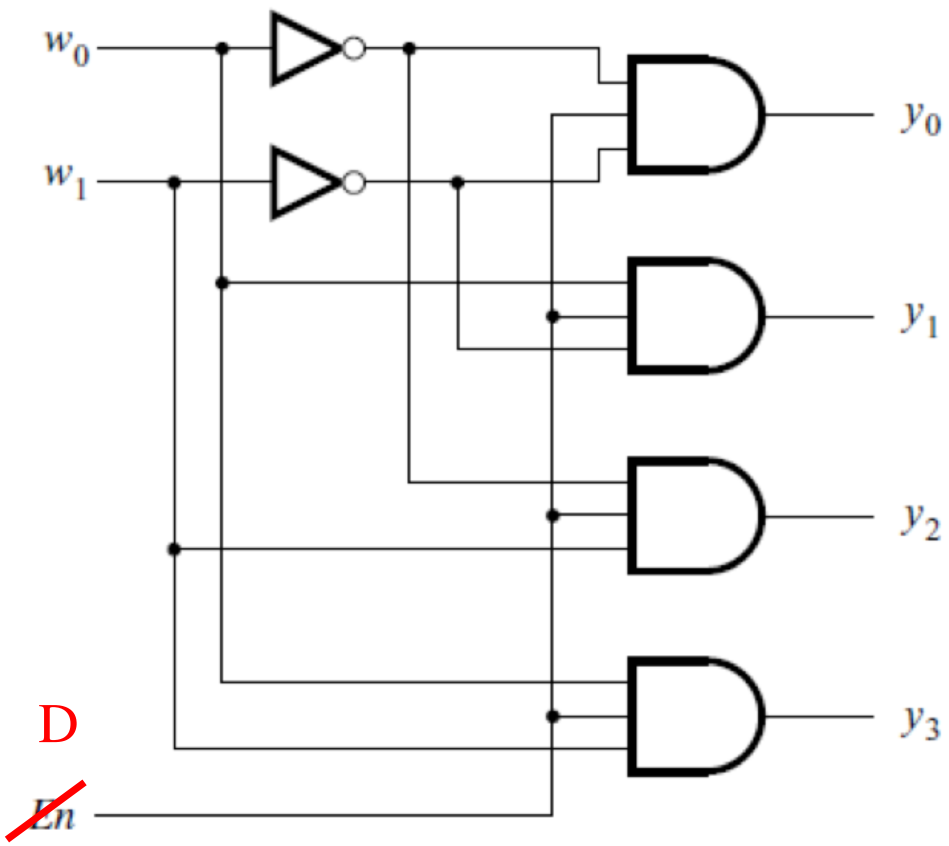
A 1-to-4 demultiplexer
built with a 2-to-4 decoder with enable



[Figure 4.14c from the textbook]

A 1-to-4 demultiplexer
built with a 2-to-4 decoder with enable

output select lines

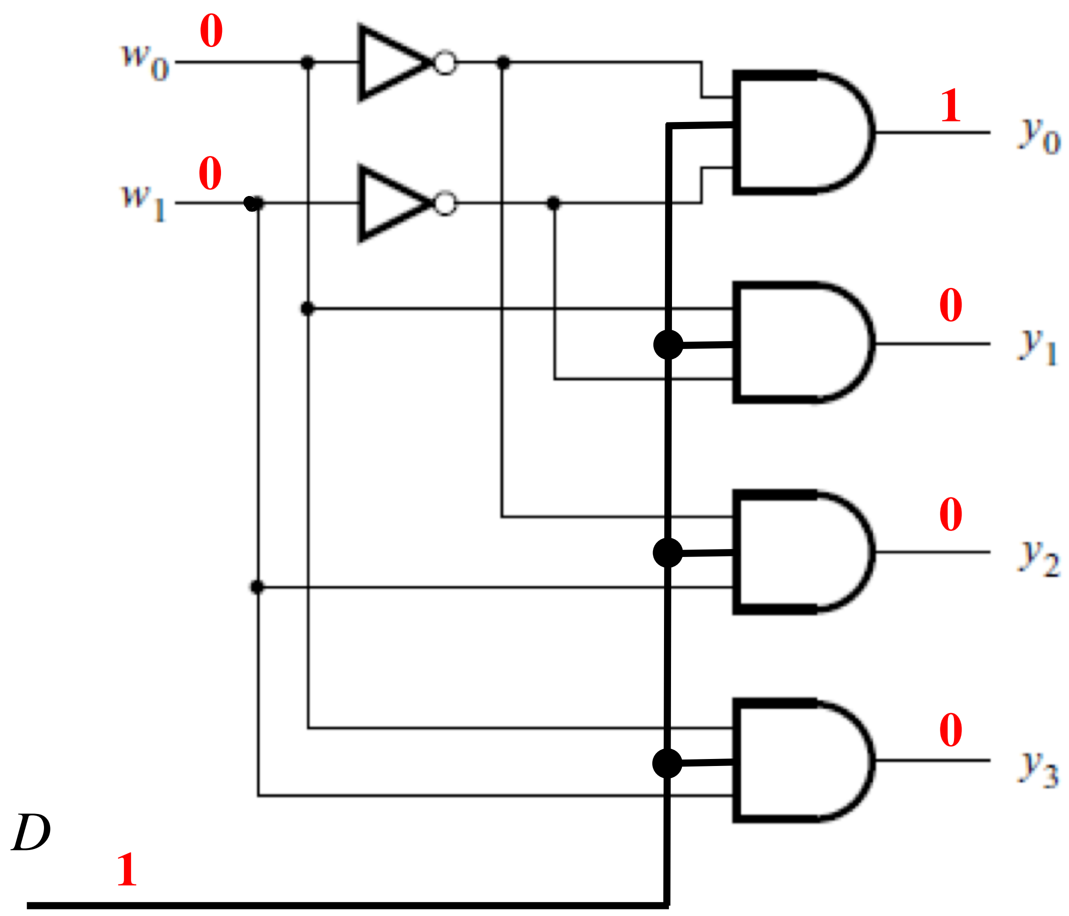


the four output lines

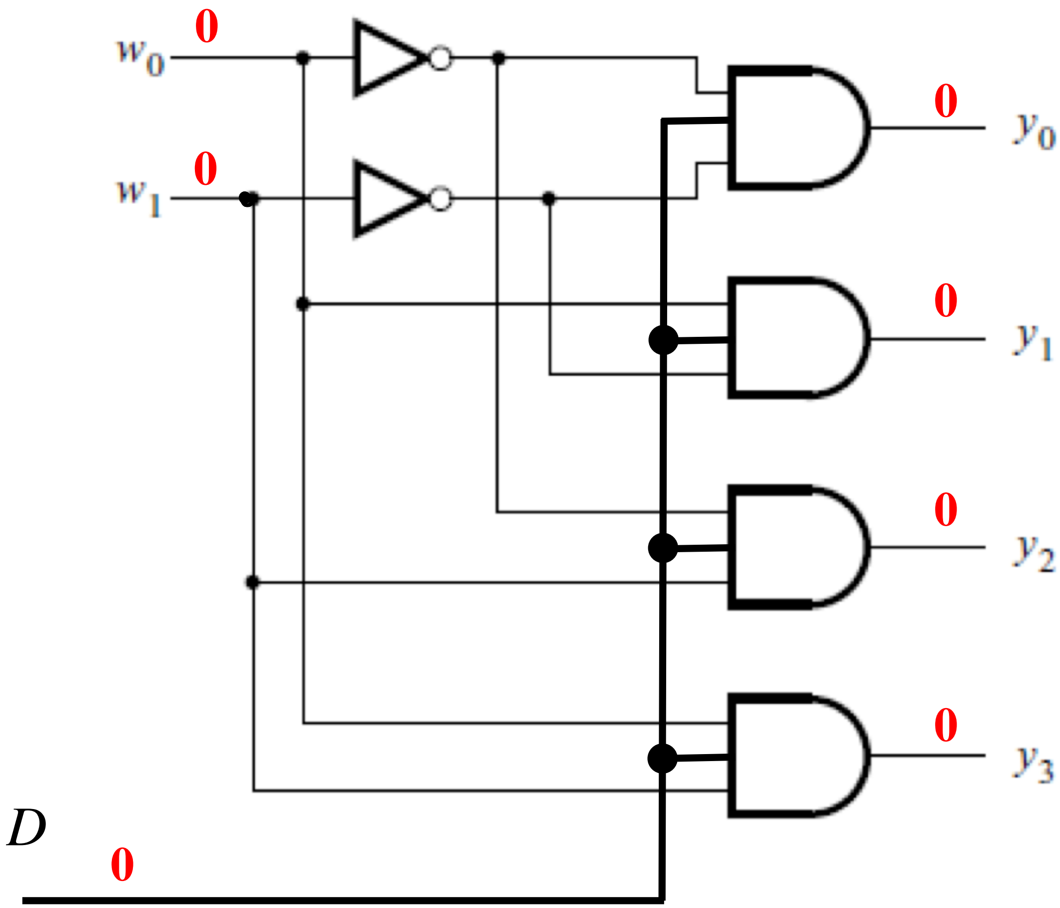
data input line

[Figure 4.14c from the textbook]

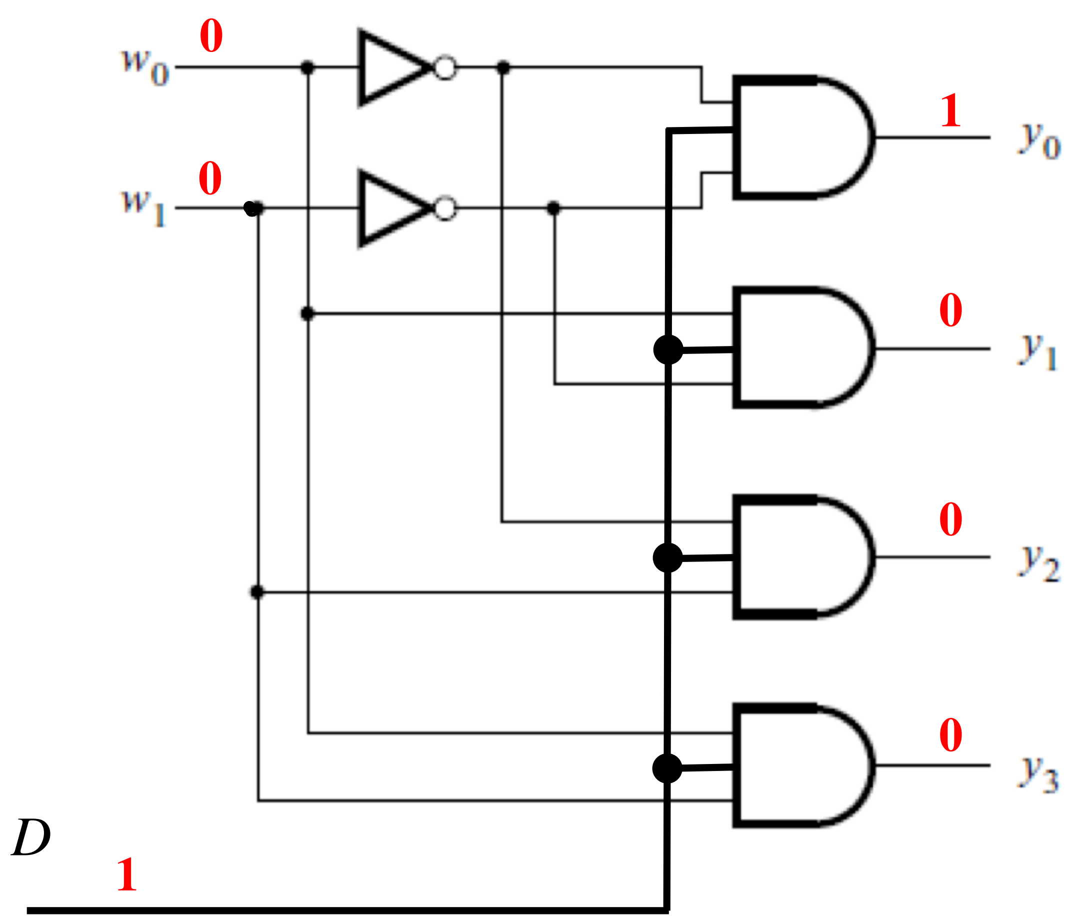
A 1-to-4 demultiplexer built with a 2-to-4 decoder with enable



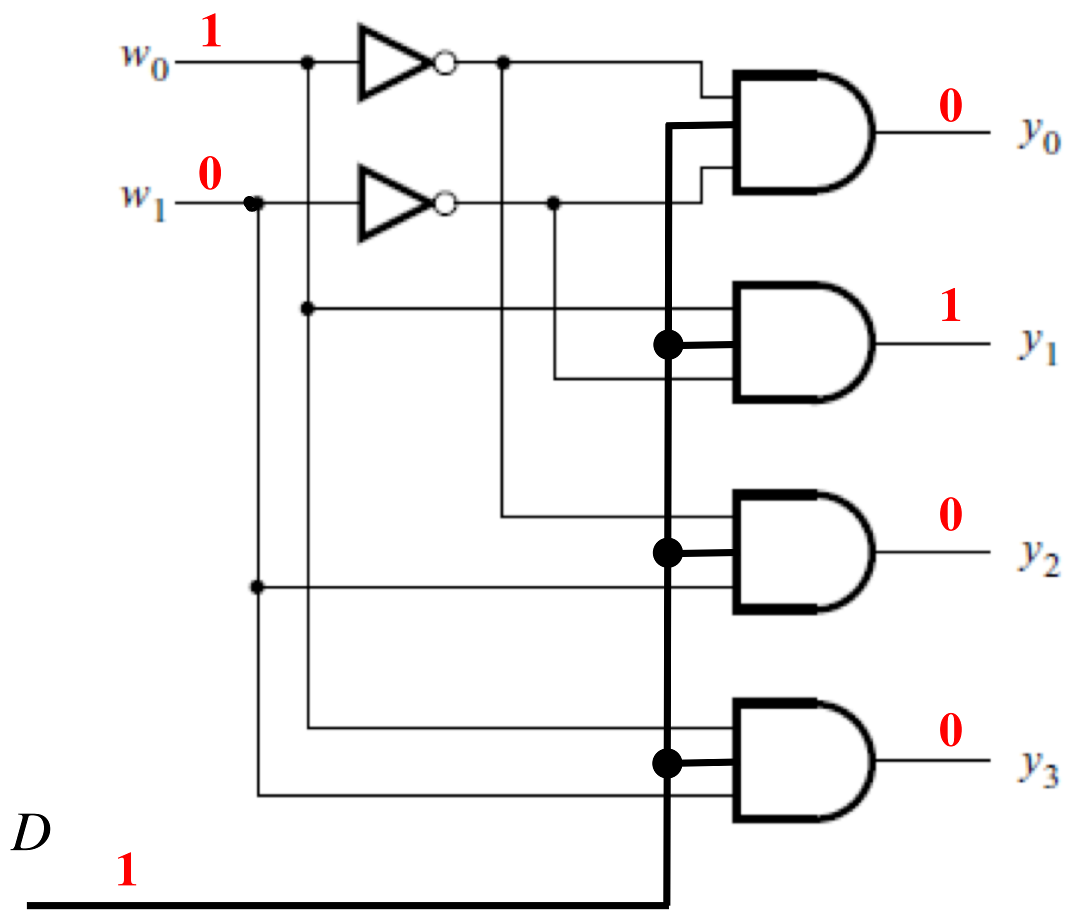
A 1-to-4 demultiplexer
built with a 2-to-4 decoder with enable



A 1-to-4 demultiplexer
built with a 2-to-4 decoder with enable

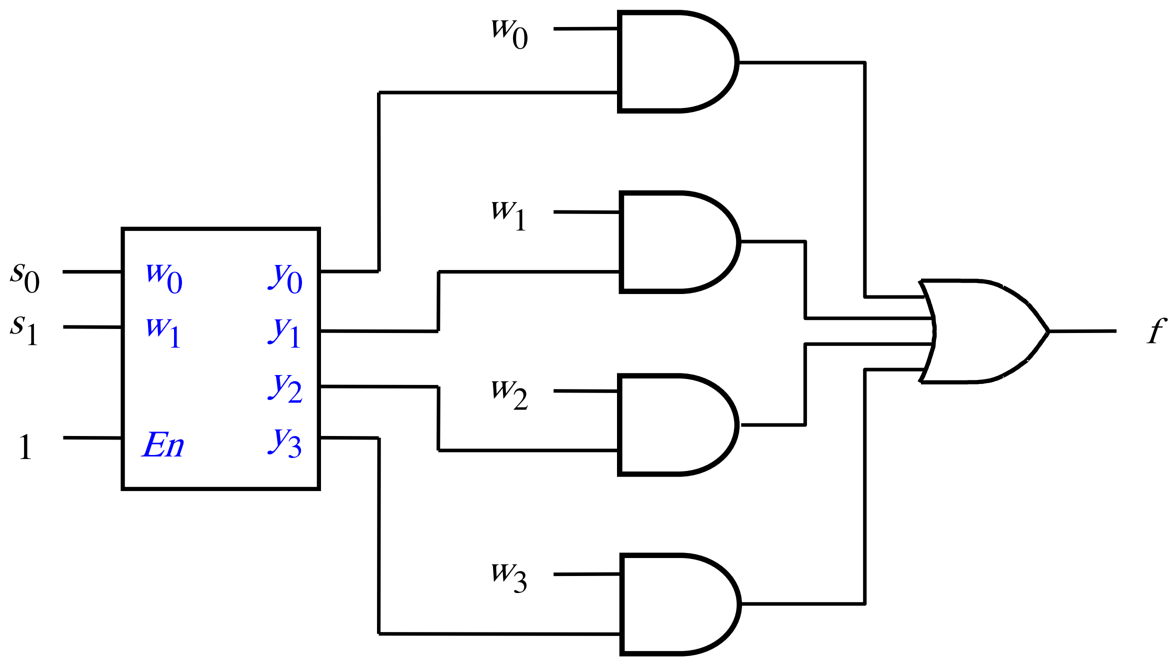


A 1-to-4 demultiplexer
built with a 2-to-4 decoder with enable



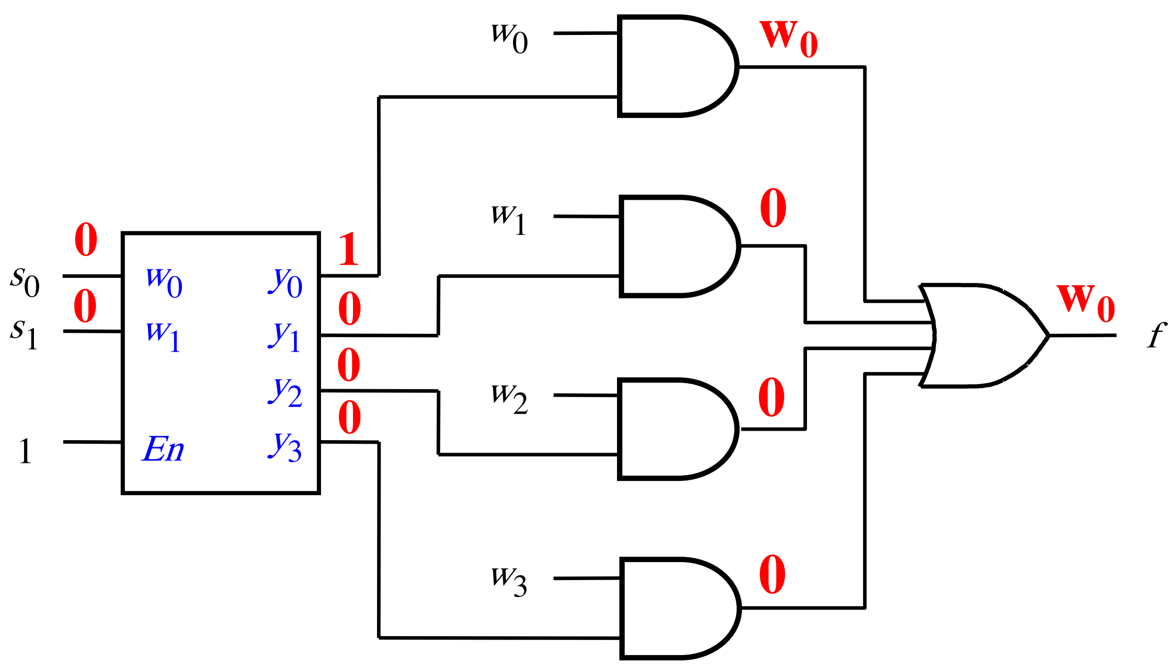
Multiplexers
(Implemented with Decoders)

A 4-to-1 multiplexer
built using a 2-to-4 decoder



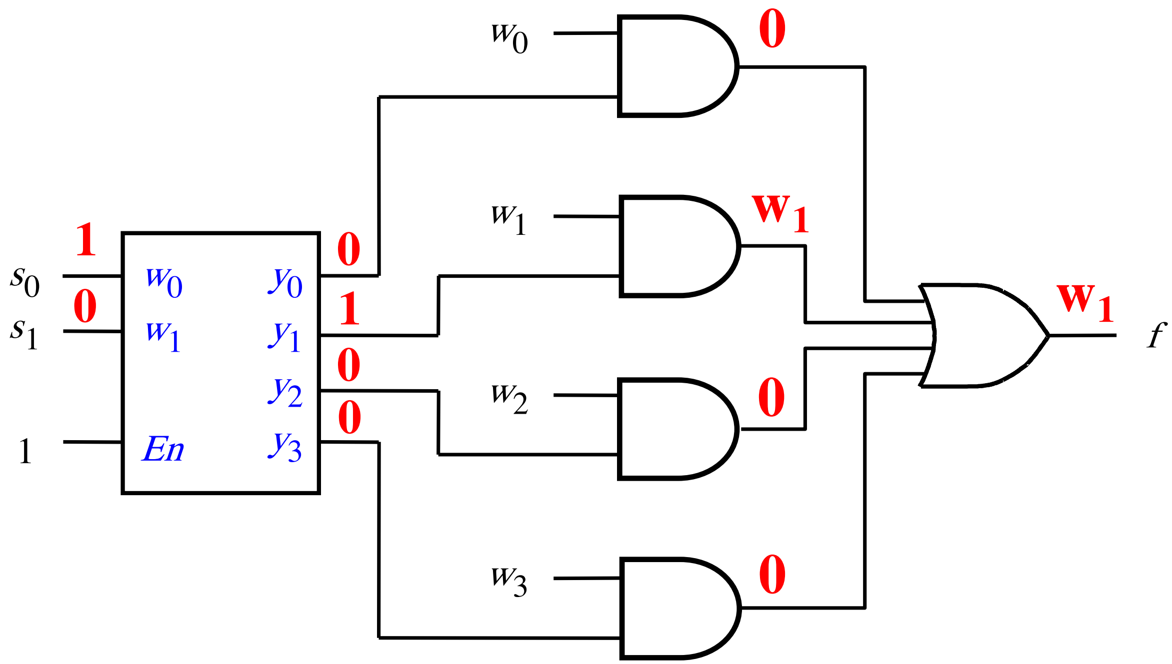
[Figure 4.17 from the textbook]

A 4-to-1 multiplexer built using a 2-to-4 decoder



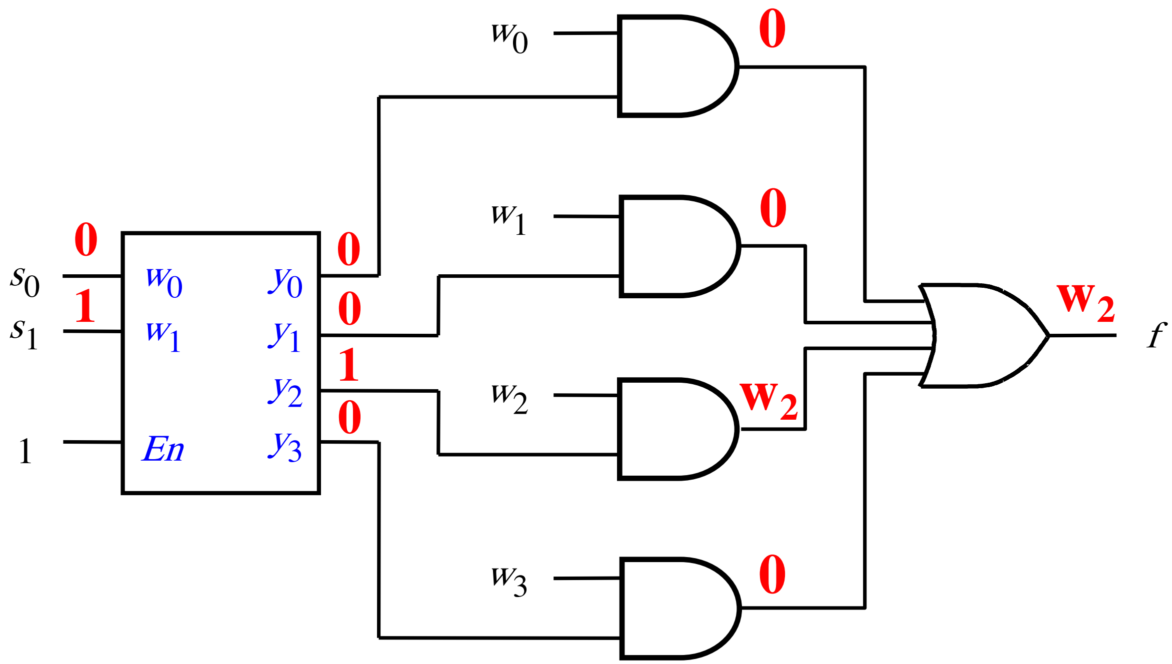
[Figure 4.17 from the textbook]

A 4-to-1 multiplexer built using a 2-to-4 decoder



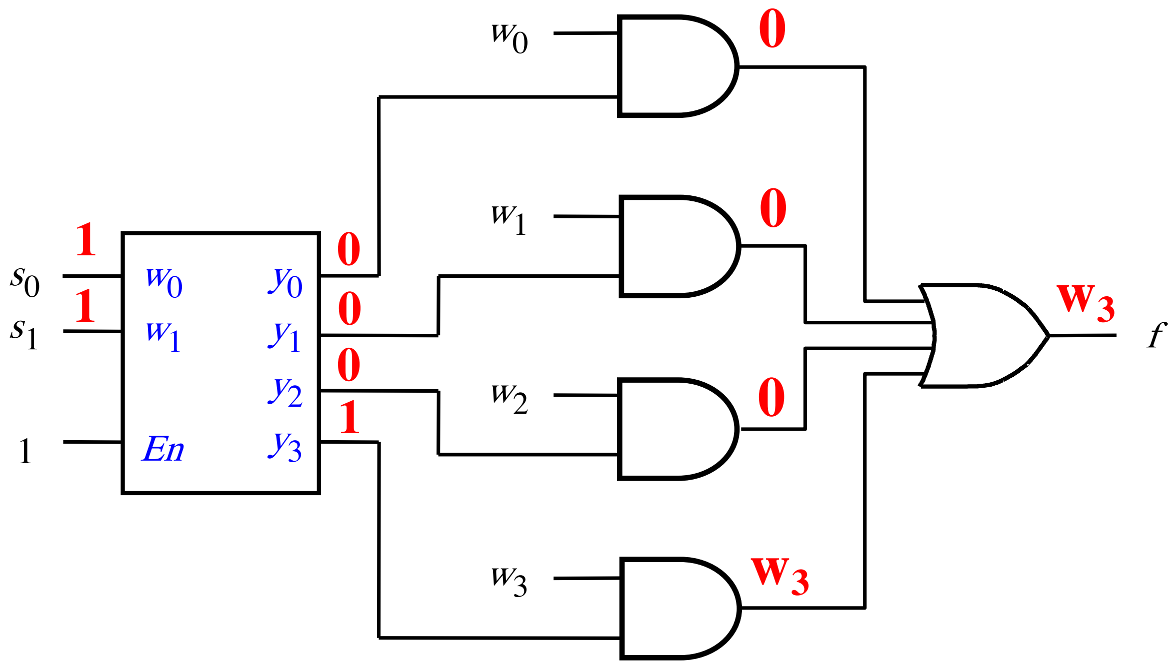
[Figure 4.17 from the textbook]

A 4-to-1 multiplexer built using a 2-to-4 decoder



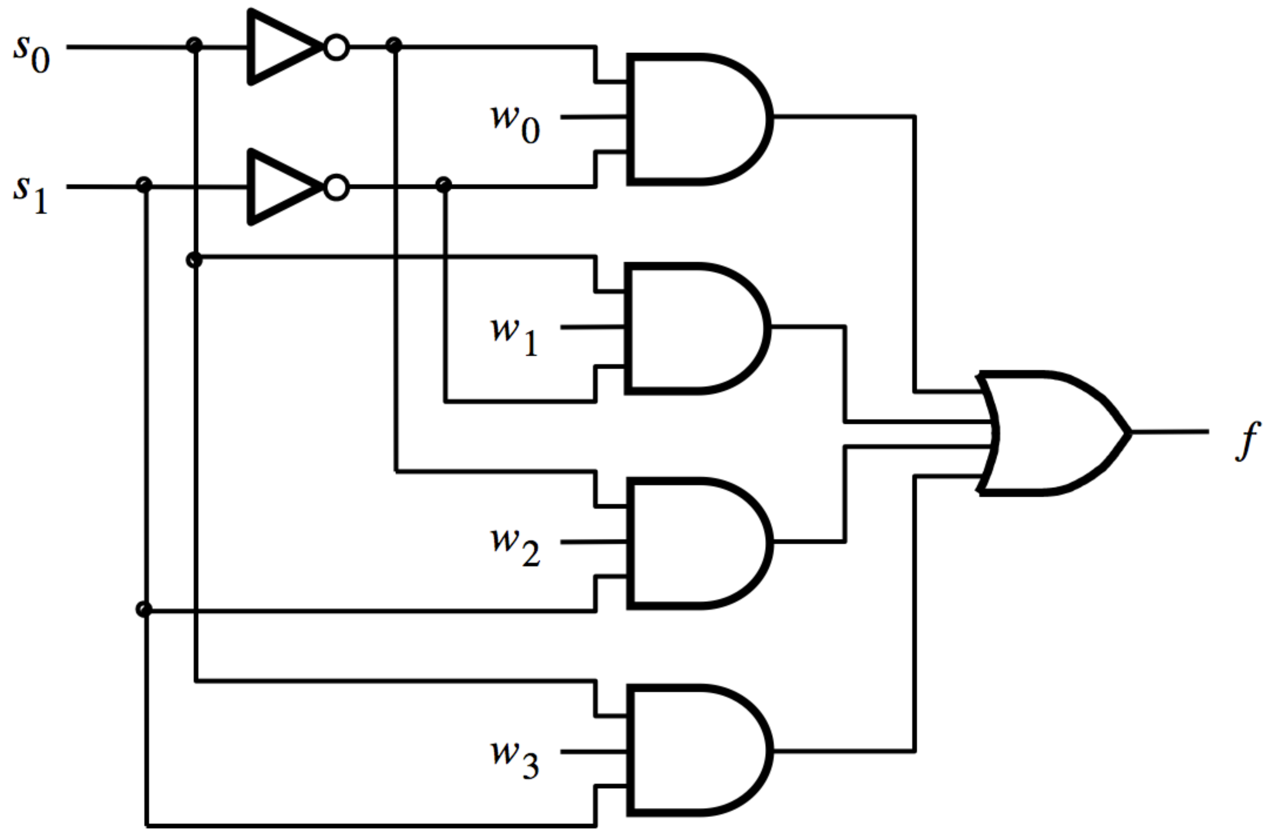
[Figure 4.17 from the textbook]

A 4-to-1 multiplexer built using a 2-to-4 decoder



[Figure 4.17 from the textbook]

4-to-1 Multiplexer (SOP circuit)



$$f = \overline{s_1}\,\overline{s_0}\,w_0 + \overline{s_1}\,s_0\,w_1 + s_1\,\overline{s_0}\,w_2 + s_1\,s_0\,w_3$$

[Figure 4.2c from the textbook]

Encoders

Encoders
(there are several types)

Binary Encoders

4-to-2 Binary Encoder (Definition)

- Has four inputs: w_3 , w_2 , w_1 , and w_0
- Has two outputs: y_1 and y_0
- Only one input is set to 1 ("one-hot" encoded). All others are set to 0.

- If $w_0=1$ then $y_1=0$ and $y_0=0$
- If $w_1=1$ then $y_1=0$ and $y_0=1$
- If $w_2=1$ then $y_1=1$ and $y_0=0$
- If $w_3=1$ then $y_1=1$ and $y_0=1$

Truth table for a 4-to-2 binary encoder

w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

[Figure 4.19 from the textbook]

Truth table for a 4-to-2 binary encoder

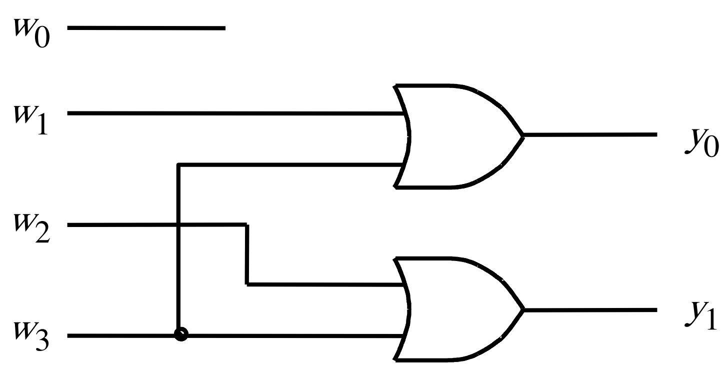
w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

The inputs are “one-hot” encoded

[Figure 4.19 from the textbook]

Circuit for a 4-to-2 binary encoder

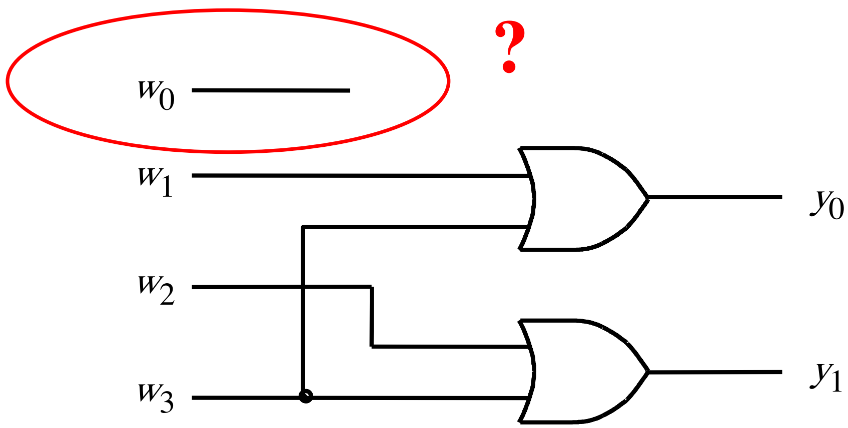
w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1



[Figure 4.19 from the textbook]

Circuit for a 4-to-2 binary encoder

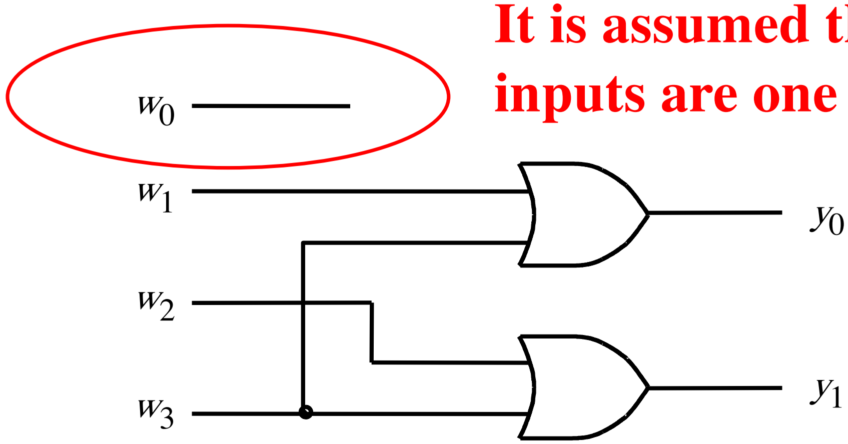
w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1



[Figure 4.19 from the textbook]

Circuit for a 4-to-2 binary encoder

w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

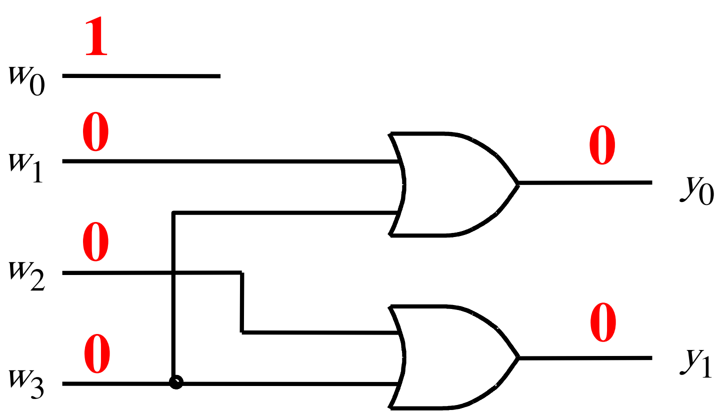


It is assumed that the inputs are one hot encoded

[Figure 4.19 from the textbook]

Circuit for a 4-to-2 binary encoder

w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

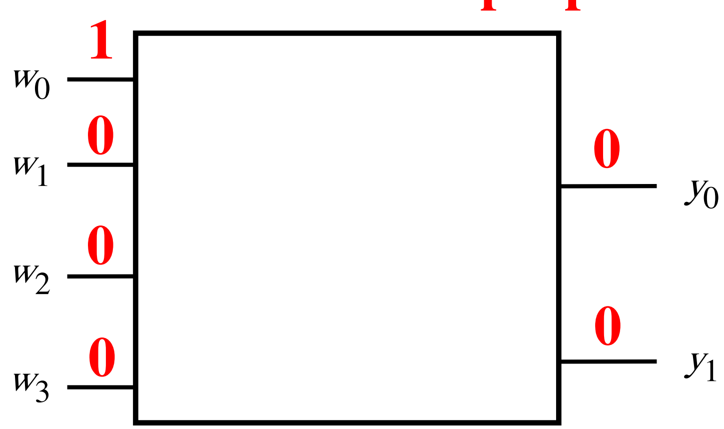


[Figure 4.19 from the textbook]

Circuit for a 4-to-2 binary encoder

w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

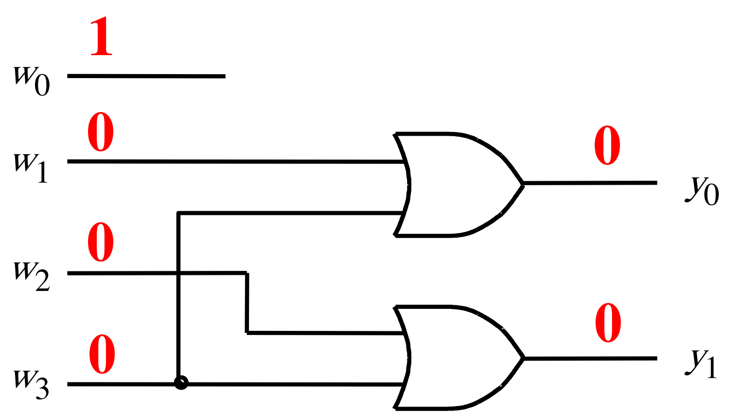
As this level of abstraction we need that w_0 input for this to be a proper 4-to-2 binary encoder.



[Figure 4.19 from the textbook]

Circuit for a 4-to-2 binary encoder

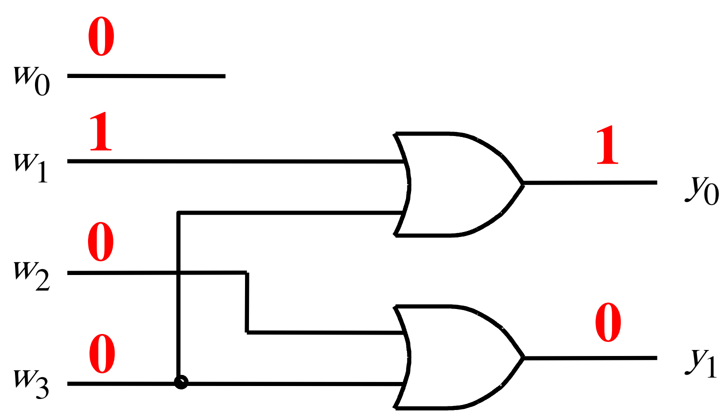
w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1



[Figure 4.19 from the textbook]

Circuit for a 4-to-2 binary encoder

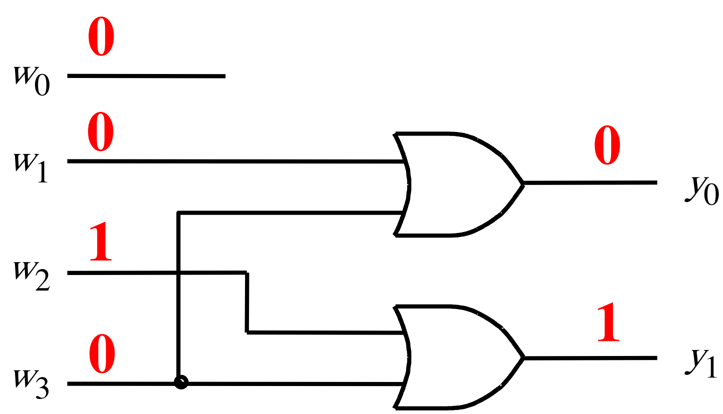
w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1



[Figure 4.19 from the textbook]

Circuit for a 4-to-2 binary encoder

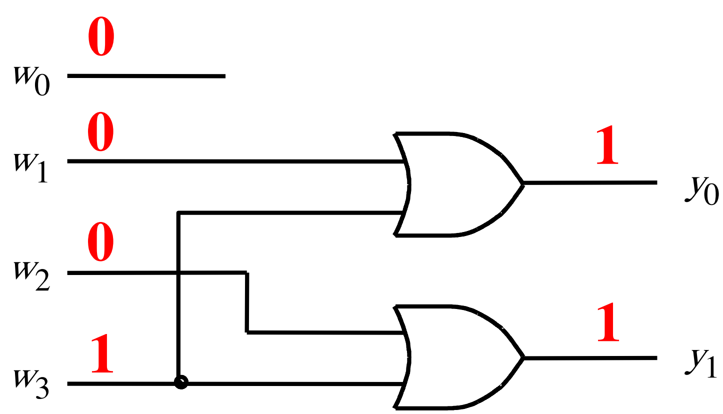
w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1



[Figure 4.19 from the textbook]

Circuit for a 4-to-2 binary encoder

w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1



[Figure 4.19 from the textbook]

Expressions for 4-to-2 binary encoder

w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	0		
0	0	0	1	0	0
0	0	1	0	0	1
0	0	1	1		
0	1	0	0	1	0
0	1	0	1		
0	1	1	0		
0	1	1	1		
1	0	0	0	1	1
1	0	0	1		
1	0	1	0		
1	0	1	1		
1	1	0	0		
1	1	0	1		
1	1	1	0		
1	1	1	1		

w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

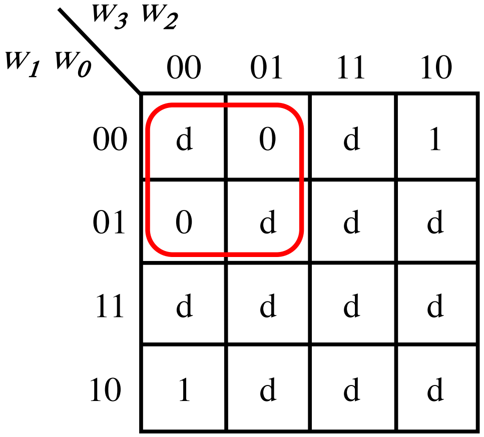
Expressions for 4-to-2 binary encoder

w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	0	d	d
0	0	0	1	0	0
0	0	1	0	0	1
0	0	1	1	d	d
0	1	0	0	1	0
0	1	0	1	d	d
0	1	1	0	d	d
0	1	1	1	d	d
1	0	0	0	1	1
1	0	0	1	d	d
1	0	1	0	d	d
1	0	1	1	d	d
1	1	0	0	d	d
1	1	0	1	d	d
1	1	1	0	d	d
1	1	1	1	d	d

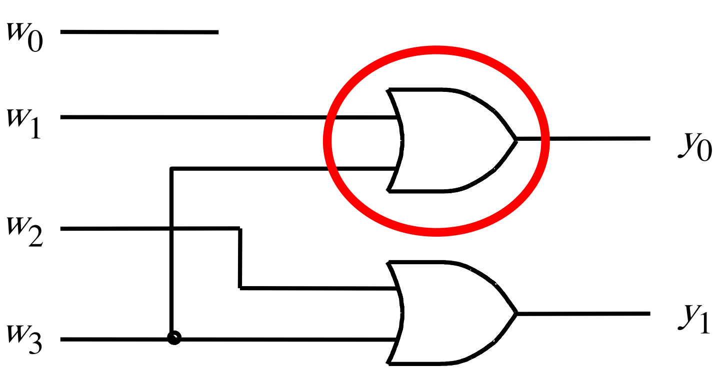
w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

Expressions for 4-to-2 binary encoder

w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	0	d	d
0	0	0	1	0	0
0	0	1	0	0	1
0	0	1	1	d	d
0	1	0	0	1	0
0	1	0	1	d	d
0	1	1	0	d	d
0	1	1	1	d	d
1	0	0	0	1	1
1	0	0	1	d	d
1	0	1	0	d	d
1	0	1	1	d	d
1	1	0	0	d	d
1	1	0	1	d	d
1	1	1	0	d	d
1	1	1	1	d	d



$$y_0 = (w_1 + w_3)$$

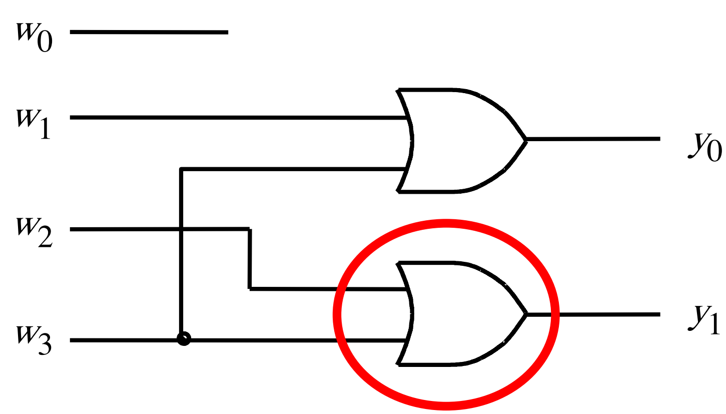


Expressions for 4-to-2 binary encoder

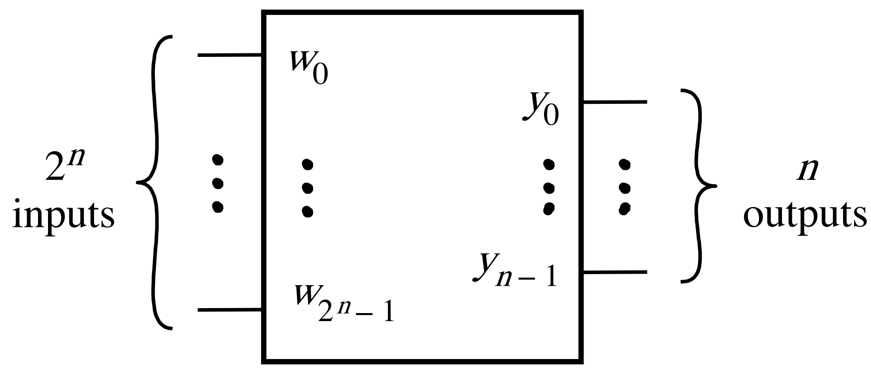
w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	0	d	d
0	0	0	1	0	0
0	0	1	0	0	1
0	0	1	1	d	d
0	1	0	0	1	0
0	1	0	1	d	d
0	1	1	0	d	d
0	1	1	1	d	d
1	0	0	0	1	1
1	0	0	1	d	d
1	0	1	0	d	d
1	0	1	1	d	d
1	1	0	0	d	d
1	1	0	1	d	d
1	1	1	0	d	d
1	1	1	1	d	d

$w_1 w_0$ \ $w_3 w_2$	00	01	11	10
00	d	1	d	1
01	0	d	d	d
11	d	d	d	d
10	0	d	d	d

$$y_1 = (w_3 + w_2)$$



The Most General Case: 2^n-to-n binary encoder



[Figure 4.18 from the textbook]

Priority Encoders

4-to-2 Priority Encoder (Definition)

- Has four inputs: w_3 , w_2 , w_1 , and w_0
- Has two primary outputs: y_1 and y_0
- Has one other output: z

- The inputs are NOT "one-hot" encoded.
- More than one input can be set to 1 but they have priorities associated with them: w_3 – highest priority and w_0 – lowest priority.

- $y_1=0$ and $y_0=0$ (if $w_0=1$ and $w_3=w_2=w_1=0$)
- $y_1=0$ and $y_0=1$ (if $w_1=1$ and $w_3=w_2=0$)
- $y_1=1$ and $y_0=0$ (if $w_2=1$ and $w_3=0$)
- $y_1=1$ and $y_0=1$ (if $w_3=1$)

4-to-2 Priority Encoder (Definition)

- Has four inputs: w_3 , w_2 , w_1 , and w_0
- Has two primary outputs: y_1 and y_0
- Has one other output: z

- The inputs are NOT "one-hot" encoded.
- More than one input can be set to 1 but they have priorities associated with them: w_3 – highest priority and w_0 – lowest priority.

- y_1=0 and y_0=0 (if w_0=1 and w_3=w_2=w_1=0)
- y_1=0 and y_0=1 (if w_1=1 and w_3=w_2=0) w_0
- y_1=1 and y_0=0 (if w_2=1 and w_3=0) w_0 , w_1
- y_1=1 and y_0=1 (if w_3=1) w_0 , w_1 , w_2

these have lower priorities and can be either 0 or 1.

4-to-2 Priority Encoder (Definition)

- **Has four inputs: w_3 , w_2 , w_1 , and w_0**
- **Has two primary outputs: y_1 and y_0**
- **Has one other output: z**

- **The inputs are NOT "one-hot" encoded.**
- **More than one input can be set to 1 but they have priorities associated with them: w_3 – highest priority and w_0 – lowest priority.**

- **y_1=0 and y_0=0 (if w_0=1 and w_3=w_2=w_1=0)**
- **y_1=0 and y_0=1 (if w_1=1 and w_3=w_2=0)**
- **y_1=1 and y_0=0 (if w_2=1 and w_3=0)**
- **y_1=1 and y_0=1 (if w_3=1)**

- **z =0 if w_3=w_2=w_1=w_0=0; otherwise z=1.**

Truth table for a 4-to-2 priority encoder
(abbreviated version)

w_3	w_2	w_1	w_0	y_1	y_0	z
0	0	0	0	d	d	0
0	0	0	1	0	0	1
0	0	1	x	0	1	1
0	1	x	x	1	0	1
1	x	x	x	1	1	1

[Figure 4.20 from the textbook]

Truth table for a 4-to-2 priority encoder (abbreviated version)

w_3	w_2	w_1	w_0	y_1	y_0	z
0	0	0	0	d	d	0
0	0	0	1	0	0	1
0	0	1	x	0	1	1
0	1	x	x	1	0	1
1	x	x	x	1	1	1

[Figure 4.20 from the textbook]

Truth table for a 4-to-2 priority encoder

- 0 0 0 0
- 0 0 0 1
- 0 0 1 x
- 0 1 x x
- 1 x x x

w_3	w_2	w_1	w_0	y_1	y_0	z
0	0	0	0	d	d	0
0	0	0	1	0	0	1
0	0	1	0	0	1	1
0	0	1	1	0	1	1
0	1	0	0	1	0	1
0	1	0	1	1	0	1
0	1	1	0	1	0	1
0	1	1	1	1	0	1
1	0	0	0	1	1	1
1	0	0	1	1	1	1
1	0	1	0	1	1	1
1	0	1	1	1	1	1
1	1	0	0	1	1	1
1	1	0	1	1	1	1
1	1	1	0	1	1	1
1	1	1	1	1	1	1

Expressions for 4-to-2 priority encoder

w_3	w_2	w_1	w_0	y_1	y_0	z
0	0	0	0	d	d	0
0	0	0	1	0	0	1
0	0	1	0	0	1	1
0	0	1	1	0	1	1
0	1	0	0	1	0	1
0	1	0	1	1	0	1
0	1	1	0	1	0	1
0	1	1	1	1	0	1
1	0	0	0	1	1	1
1	0	0	1	1	1	1
1	0	1	0	1	1	1
1	0	1	1	1	1	1
1	1	0	0	1	1	1
1	1	0	1	1	1	1
1	1	1	0	1	1	1
1	1	1	1	1	1	1

$w_1 w_0$ \ $w_3 w_2$	00	01	11	10
00	d	1	1	1
01	0	1	1	1
11	0	1	1	1
10	0	1	1	1

$$y_1 = w_3 + w_2$$

Expressions for 4-to-2 priority encoder

w_3	w_2	w_1	w_0	y_1	y_0	z
0	0	0	0	d	d	0
0	0	0	1	0	0	1
0	0	1	0	0	1	1
0	0	1	1	0	1	1
0	1	0	0	1	0	1
0	1	0	1	1	0	1
0	1	1	0	1	0	1
0	1	1	1	1	0	1
1	0	0	0	1	1	1
1	0	0	1	1	1	1
1	0	1	0	1	1	1
1	0	1	1	1	1	1
1	1	0	0	1	1	1
1	1	0	1	1	1	1
1	1	1	0	1	1	1
1	1	1	1	1	1	1

$w_1 w_0$ \ $w_3 w_2$	00	01	11	10
00	d	0	1	1
01	0	0	1	1
11	1	0	1	1
10	1	0	1	1

$$y_0 = w_3 + w_1 \overline{w_2}$$

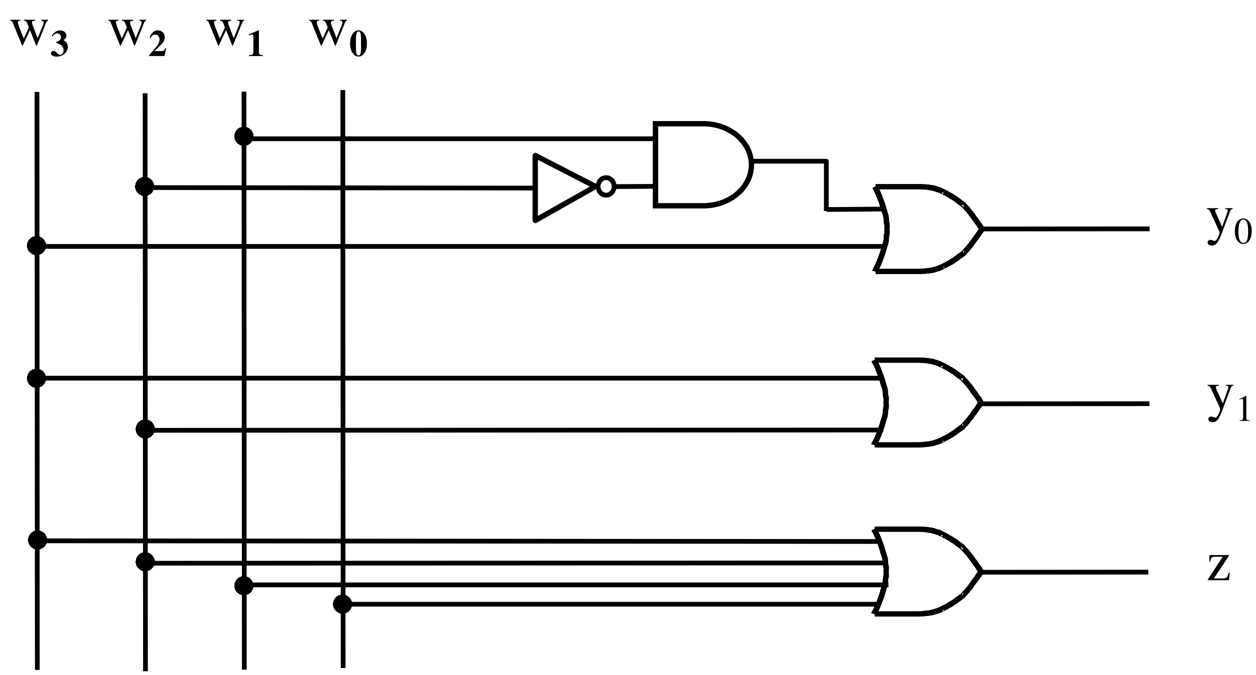
Expressions for 4-to-2 priority encoder

w_3	w_2	w_1	w_0	y_1	y_0	z
0	0	0	0	d	d	0
0	0	0	1	0	0	1
0	0	1	0	0	1	1
0	0	1	1	0	1	1
0	1	0	0	1	0	1
0	1	0	1	1	0	1
0	1	1	0	1	0	1
0	1	1	1	1	0	1
1	0	0	0	1	1	1
1	0	0	1	1	1	1
1	0	1	0	1	1	1
1	0	1	1	1	1	1
1	1	0	0	1	1	1
1	1	0	1	1	1	1
1	1	1	0	1	1	1
1	1	1	1	1	1	1

$w_1 w_0$ \ $w_3 w_2$	00	01	11	10
00	0	1	1	1
01	1	1	1	1
11	1	1	1	1
10	1	1	1	1

$$z = w_3 + w_2 + w_1 + w_0$$

Circuit for the 4-to-2 priority encoder



The textbook derives a different circuit for the 4-to-2 priority encoder using a 4-to-2 binary encoder

w_3	w_2	w_1	w_0	y_1	y_0	z
0	0	0	0	d	d	0
0	0	0	1	0	0	1
0	0	1	x	0	1	1
0	1	x	x	1	0	1
1	x	x	x	1	1	1

$$i_0 = \overline{w}_3\overline{w}_2\overline{w}_1 w_0$$
$$i_1 = \overline{w}_3\overline{w}_2 w_1$$
$$i_2 = \overline{w}_3 w_2$$
$$i_3 = w_3$$

$$y_0 = i_1 + i_3$$
$$y_1 = i_2 + i_3$$

$$z = i_0 + i_1 + i_2 + i_3$$

The textbook derives a different circuit for the 4-to-2 priority encoder using a 4-to-2 binary encoder

w_3	w_2	w_1	w_0	y_1	y_0	z
0	0	0	0	d	d	0
0	0	0	1	0	0	1
0	0	1	x	0	1	1
0	1	x	x	1	0	1
1	x	x	x	1	1	1

$$i_0 = \overline{w}_3\overline{w}_2\overline{w}_1 w_0$$

$$i_1 = \overline{w}_3\overline{w}_2 w_1$$

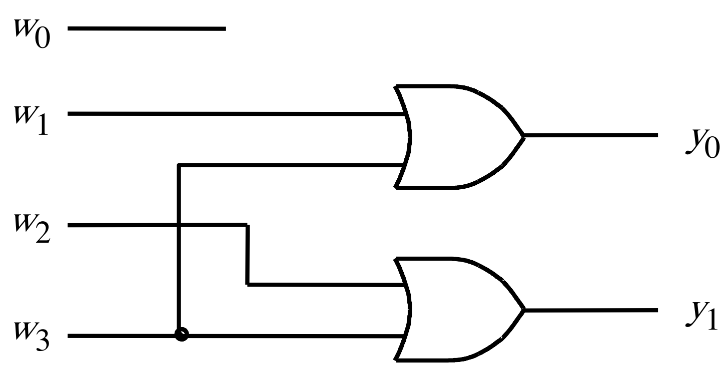
$$i_2 = \overline{w}_3 w_2$$

$$i_3 = w_3$$

$$y_0 = i_1 + i_3$$

$$y_1 = i_2 + i_3$$

$$z = i_0 + i_1 + i_2 + i_3$$



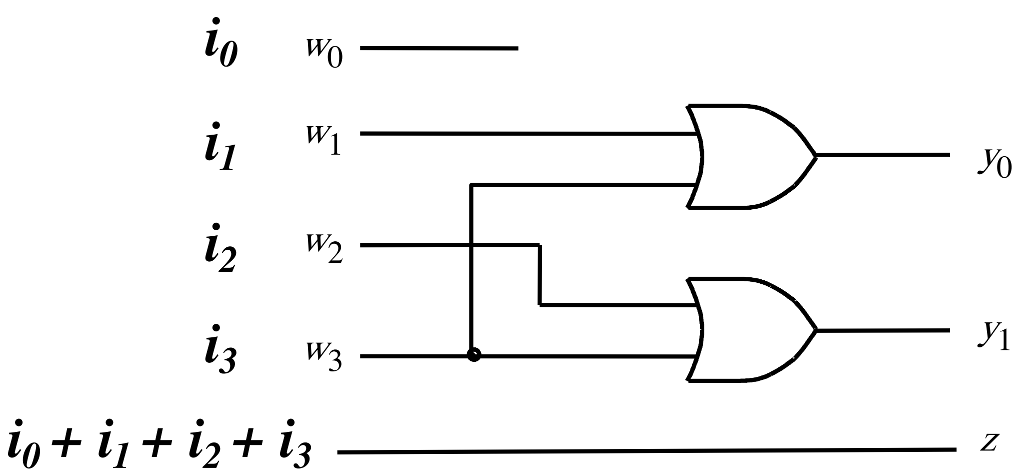
The textbook derives a different circuit for the 4-to-2 priority encoder using a 4-to-2 binary encoder

w_3	w_2	w_1	w_0	y_1	y_0	z
0	0	0	0	d	d	0
0	0	0	1	0	0	1
0	0	1	x	0	1	1
0	1	x	x	1	0	1
1	x	x	x	1	1	1

$$i_0 = \overline{w}_3\overline{w}_2\overline{w}_1 w_0$$
$$i_1 = \overline{w}_3\overline{w}_2 w_1$$
$$i_2 = \overline{w}_3 w_2$$
$$i_3 = w_3$$

$$y_0 = i_1 + i_3$$
$$y_1 = i_2 + i_3$$

$$z = i_0 + i_1 + i_2 + i_3$$



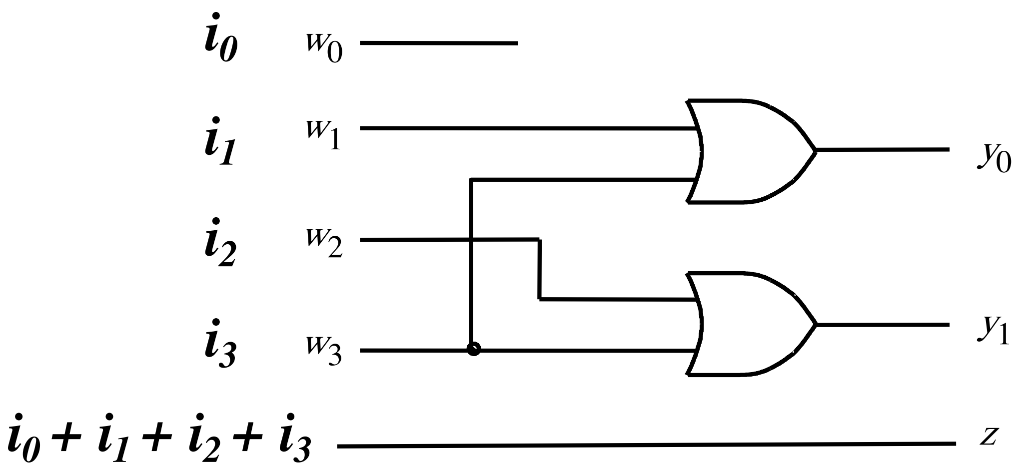
The textbook derives a different circuit for the 4-to-2 priority encoder using a 4-to-2 binary encoder

w_3	w_2	w_1	w_0	y_1	y_0	z
0	0	0	0	d	d	0
0	0	0	1	0	0	1
0	0	1	x	0	1	1
0	1	x	x	1	0	1
1	x	x	x	1	1	1

$$i_0 = \overline{w}_3\overline{w}_2\overline{w}_1 w_0$$
$$i_1 = \overline{w}_3\overline{w}_2 w_1$$
$$i_2 = \overline{w}_3 w_2$$
$$i_3 = w_3$$

$$y_0 = i_1 + i_3$$
$$y_1 = i_2 + i_3$$

$$z = i_0 + i_1 + i_2 + i_3$$



Try to prove that this is equivalent to the circuit that was derived above.

Let's prove this for z

w_3	w_2	w_1	w_0	y_1	y_0	z
0	0	0	0	d	d	0
0	0	0	1	0	0	1
0	0	1	x	0	1	1
0	1	x	x	1	0	1
1	x	x	x	1	1	1

$$i_0 = \overline{w}_3\overline{w}_2\overline{w}_1 w_0$$

$$i_1 = \overline{w}_3\overline{w}_2 w_1$$

$$i_2 = \overline{w}_3 w_2$$

$$i_3 = w_3$$

$$y_0 = i_1 + i_3$$

$$y_1 = i_2 + i_3$$

$$z = i_0 + i_1 + i_2 + i_3$$

$w_1 w_0$ \ $w_3 w_2$	00	01	11	10
00	0	1	1	1
01	1	1	1	1
11	1	1	1	1
10	1	1	1	1

z = ?

Let's prove this for z

w_3	w_2	w_1	w_0	y_1	y_0	z
0	0	0	0	d	d	0
0	0	0	1	0	0	1
0	0	1	x	0	1	1
0	1	x	x	1	0	1
1	x	x	x	1	1	1

$$i_0 = \overline{w}_3\overline{w}_2\overline{w}_1 w_0$$

$$i_1 = \overline{w}_3\overline{w}_2 w_1$$

$$i_2 = \overline{w}_3 w_2$$

$$i_3 = w_3$$

$$y_0 = i_1 + i_3$$

$$y_1 = i_2 + i_3$$

$$z = i_0 + i_1 + i_2 + i_3$$

$w_1 w_0$ \ $w_3 w_2$	00	01	11	10
00	0	1	1	1
01	1	1	1	1
11	1	1	1	1
10	1	1	1	1

$$z = (w_0 + w_1 + w_2 + w_3)$$

Let's prove this for y_0

w_3	w_2	w_1	w_0	y_1	y_0	z
0	0	0	0	d	d	0
0	0	0	1	0	0	1
0	0	1	x	0	1	1
0	1	x	x	1	0	1
1	x	x	x	1	1	1

$$i_0 = \overline{w}_3\overline{w}_2\overline{w}_1 w_0$$
$$i_1 = \overline{w}_3\overline{w}_2 w_1$$
$$i_2 = \overline{w}_3 w_2$$
$$i_3 = w_3$$

$$y_0 = i_1 + i_3$$
$$y_1 = i_2 + i_3$$

$$z = i_0 + i_1 + i_2 + i_3$$

$w_1 w_0$ \ $w_3 w_2$	00	01	11	10
00	0	0	1	1
01	0	0	1	1
11	1	0	1	1
10	1	0	1	1

$y_0 = ?$

Let's prove this for y_0

w_3	w_2	w_1	w_0	y_1	y_0	z
0	0	0	0	d	d	0
0	0	0	1	0	0	1
0	0	1	x	0	1	1
0	1	x	x	1	0	1
1	x	x	x	1	1	1

$$i_0 = \overline{w}_3\overline{w}_2\overline{w}_1 w_0$$
$$i_1 = \overline{w}_3\overline{w}_2 w_1$$
$$i_2 = \overline{w}_3 w_2$$
$$i_3 = w_3$$

$$y_0 = i_1 + i_3$$
$$y_1 = i_2 + i_3$$

$$z = i_0 + i_1 + i_2 + i_3$$

$w_1 w_0$ \ $w_3 w_2$	00	01	11	10
00	0	0	1	1
01	0	0	1	1
11	1	0	1	1
10	1	0	1	1

$$y_0 = w_3 + w_1\overline{w_2}$$

Let's prove this for y_1

w_3	w_2	w_1	w_0	y_1	y_0	z
0	0	0	0	d	d	0
0	0	0	1	0	0	1
0	0	1	x	0	1	1
0	1	x	x	1	0	1
1	x	x	x	1	1	1

$$i_0 = \overline{w}_3\overline{w}_2\overline{w}_1 w_0$$

$$i_1 = \overline{w}_3\overline{w}_2 w_1$$

$$i_2 = \overline{w}_3 w_2$$

$$i_3 = w_3$$

$$y_0 = i_1 + i_3$$

$$y_1 = i_2 + i_3$$

$$z = i_0 + i_1 + i_2 + i_3$$

$w_1 w_0$ \ $w_3 w_2$	00	01	11	10
00	0	1	1	1
01	0	1	1	1
11	0	1	1	1
10	0	1	1	1

$y_1 = ?$

Let's prove this for y_1

w_3	w_2	w_1	w_0	y_1	y_0	z
0	0	0	0	d	d	0
0	0	0	1	0	0	1
0	0	1	x	0	1	1
0	1	x	x	1	0	1
1	x	x	x	1	1	1

$$i_0 = \overline{w}_3\overline{w}_2\overline{w}_1 w_0$$
$$i_1 = \overline{w}_3\overline{w}_2 w_1$$
$$i_2 = \overline{w}_3 w_2$$
$$i_3 = w_3$$

$$y_0 = i_1 + i_3$$
$$y_1 = i_2 + i_3$$

$$z = i_0 + i_1 + i_2 + i_3$$

$w_1 w_0$ \ $w_3 w_2$	00	01	11	10
00	0	1	1	1
01	0	1	1	1
11	0	1	1	1
10	0	1	1	1

$$y_1 = w_3 + w_2$$

Therefore, this circuit for the 4-to-2 priority encoder is equivalent to …

w_3	w_2	w_1	w_0	y_1	y_0	z
0	0	0	0	d	d	0
0	0	0	1	0	0	1
0	0	1	x	0	1	1
0	1	x	x	1	0	1
1	x	x	x	1	1	1

$$i_0 = \overline{w}_3\overline{w}_2\overline{w}_1 w_0$$

$$i_1 = \overline{w}_3\overline{w}_2 w_1$$

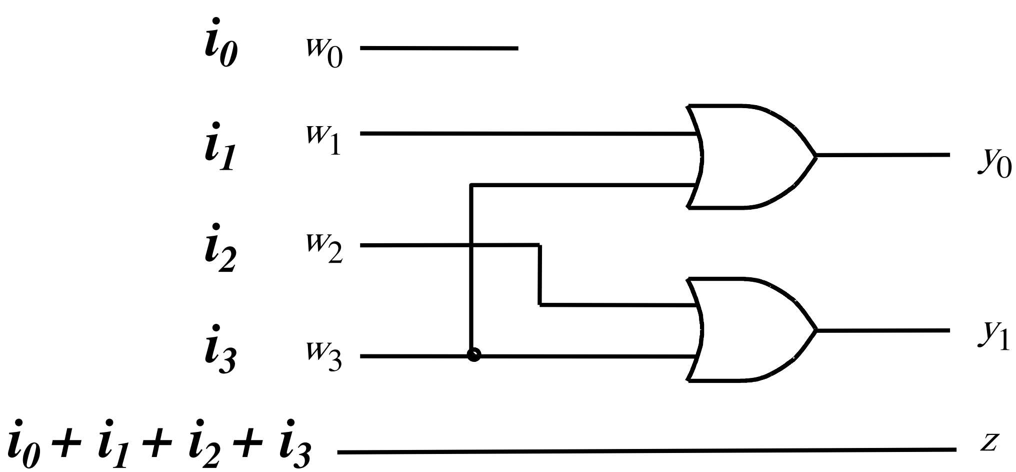
$$i_2 = \overline{w}_3 w_2$$

$$i_3 = w_3$$

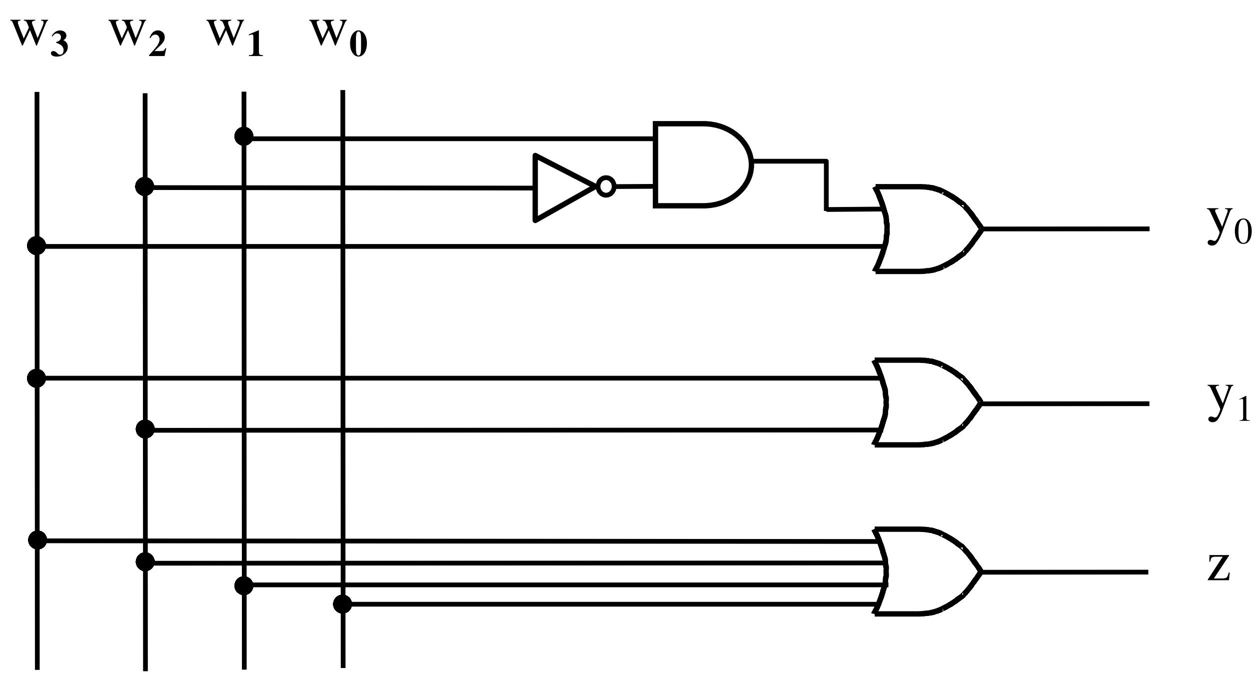
$$y_0 = i_1 + i_3$$

$$y_1 = i_2 + i_3$$

$$z = i_0 + i_1 + i_2 + i_3$$



… this circuit for the 4-to-2 priority encoder



Questions?

THE END