

CprE 281: Digital Logic

Instructor: Alexander Stoytchev

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

Code Converters

CprE 281: Digital Logic
Iowa State University, Ames, IA
Copyright © Alexander Stoytchev

Administrative Stuff

- **HW 7 is out**
- **It is due next Monday (Oct 15) @ 4pm**

Administrative Stuff

- **The second midterm is in 2 weeks.**

Administrative Stuff

- **Midterm Exam #2**
- **When: Friday October 26 @ 4pm.**
- **Where: This classroom**
- **What: Chapters 1, 2, 3, 4 and 5.1-5.8**
- **The exam will be open book and open notes (you can bring up to 3 pages of handwritten/typed notes).**

Midterm 2: Format

- **The exam will be out of 130 points**
- **You need 95 points to get an A on the exam**
- **It will be great if you can score more than 100 points.**
 - **but you can't roll over your extra points 😞**

Quick Review

Decoders

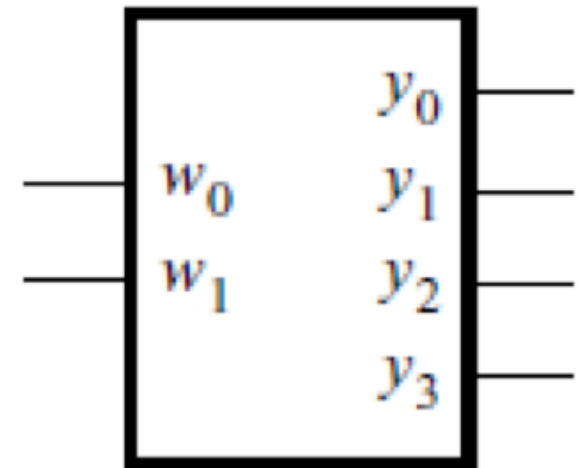
2-to-4 Decoder (Definition)

- 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



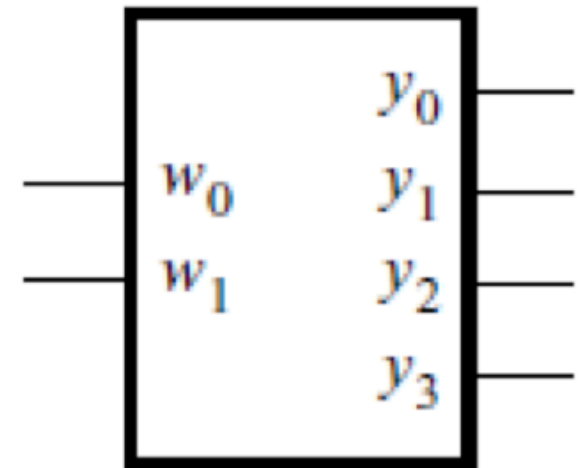
(b) Graphical symbol

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

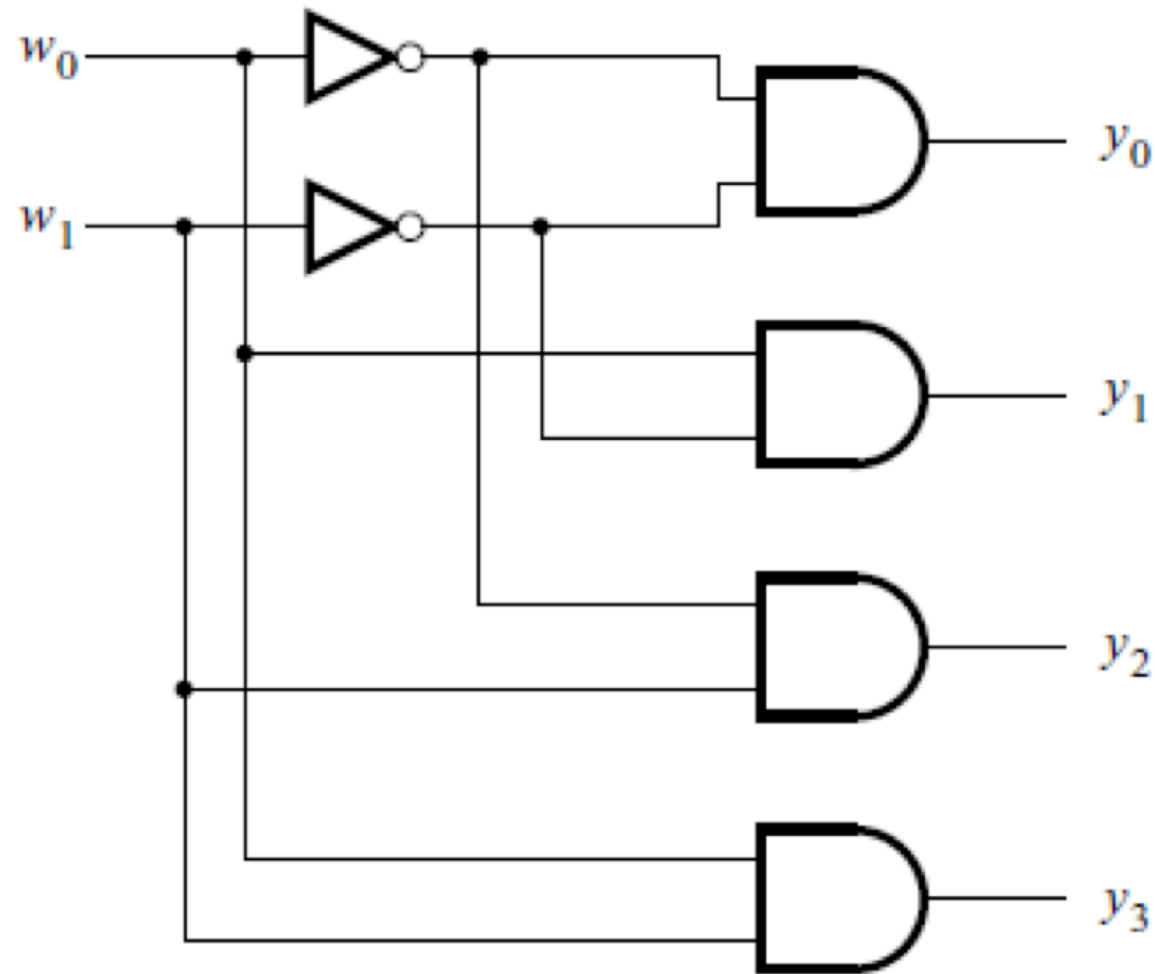
The outputs are “one-hot” encoded

(a) Truth table



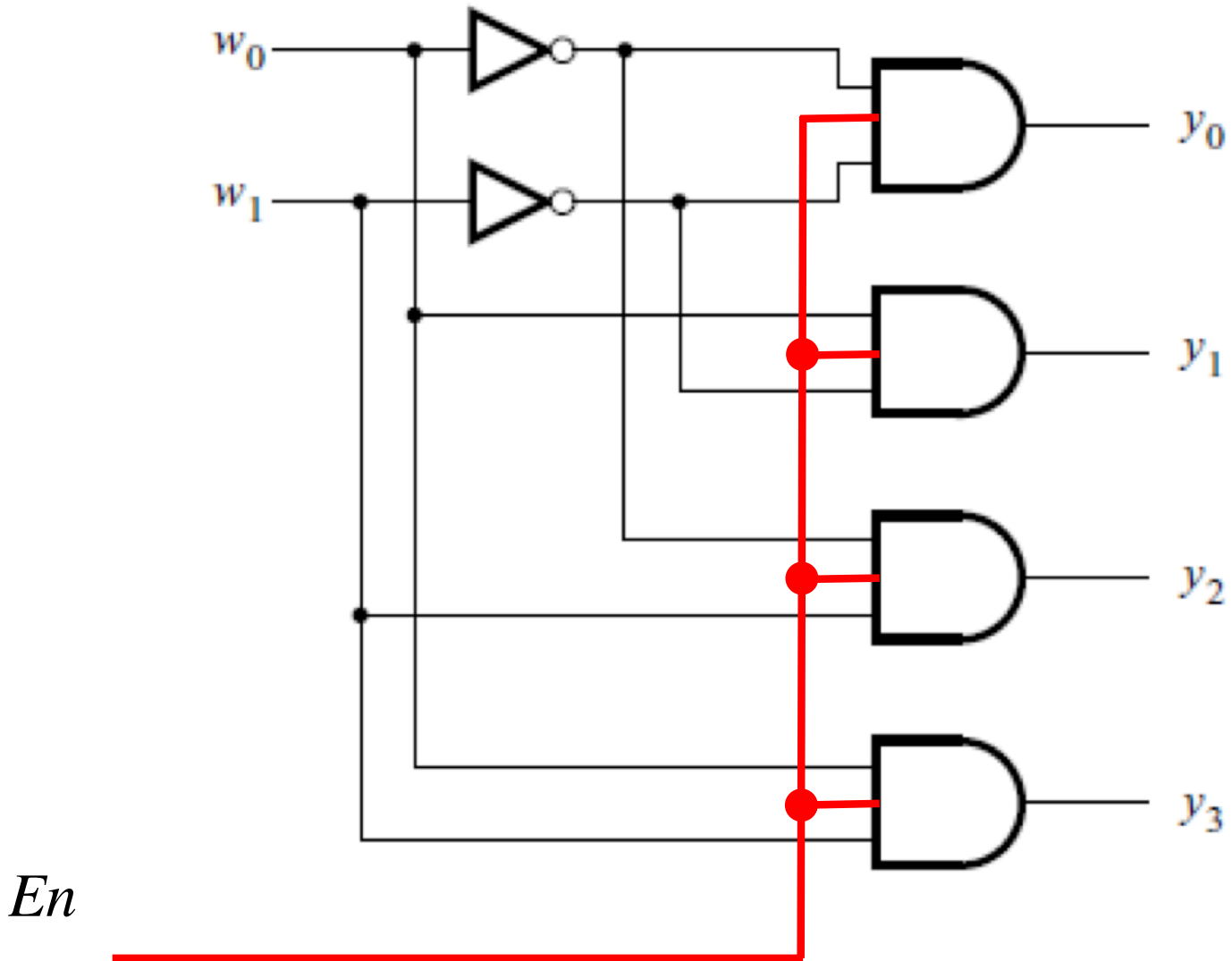
(b) Graphical symbol

Truth Logic Circuit for a 2-to-4 Decoder



[Figure 4.13c from the textbook]

Adding an Enable Input

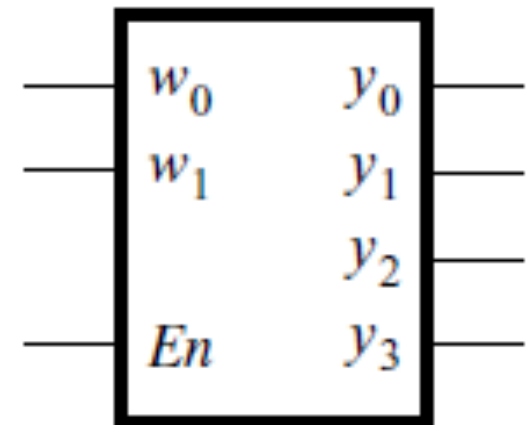


[Figure 4.13c from the textbook]

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

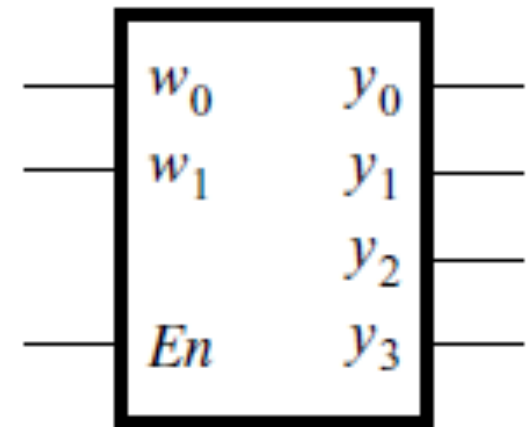


(b) Graphical symbol

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

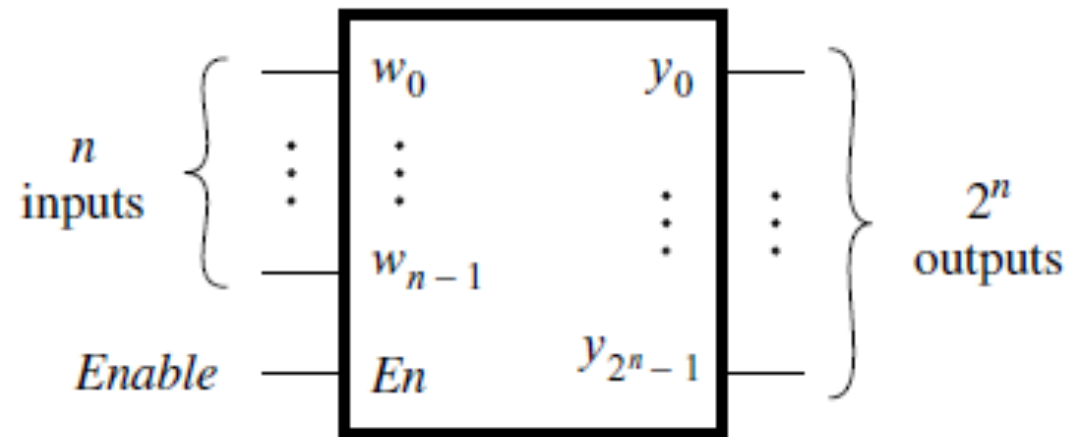
(a) Truth table



(b) Graphical symbol

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

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

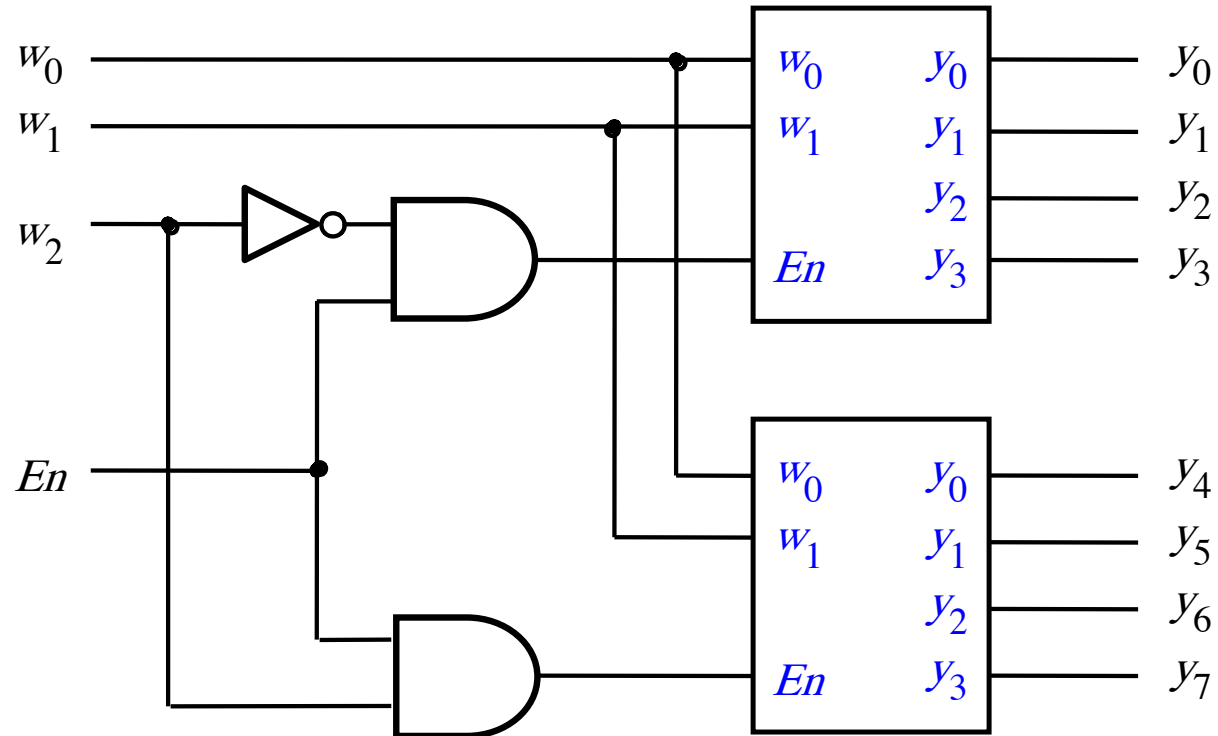


(d) An n -to- 2^n decoder

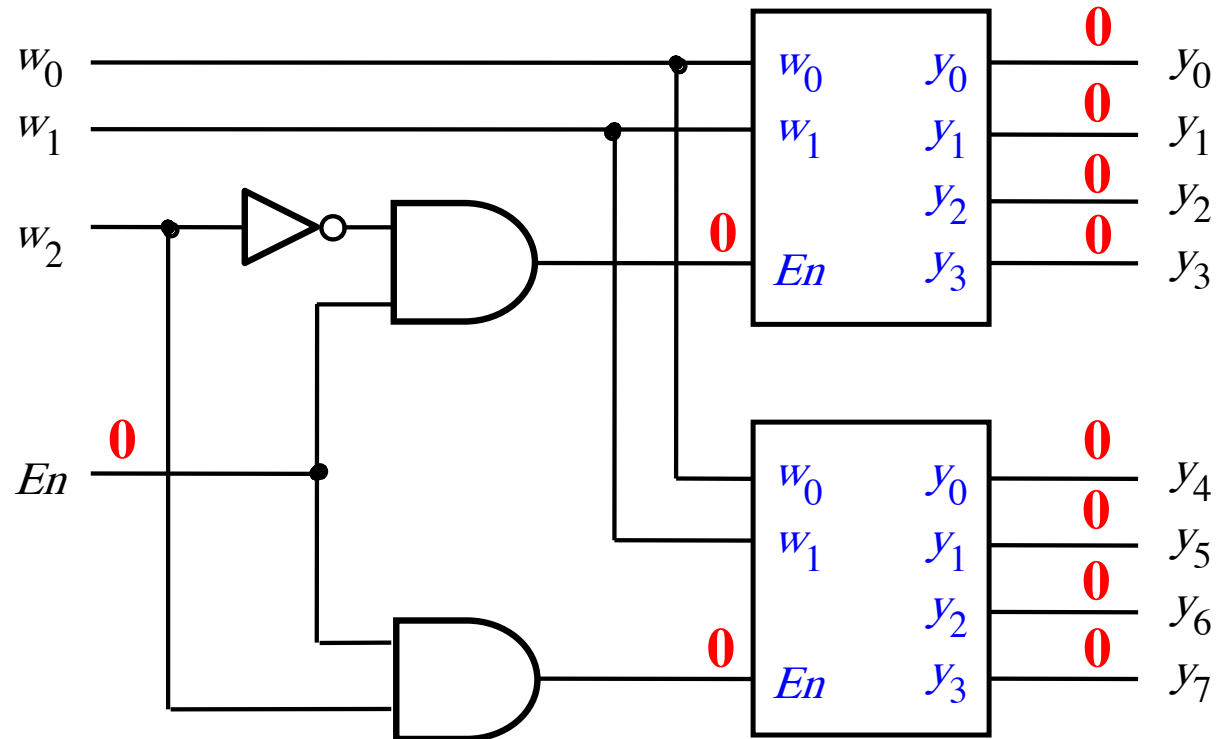
A binary decoder with n inputs has 2^n outputs

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*.

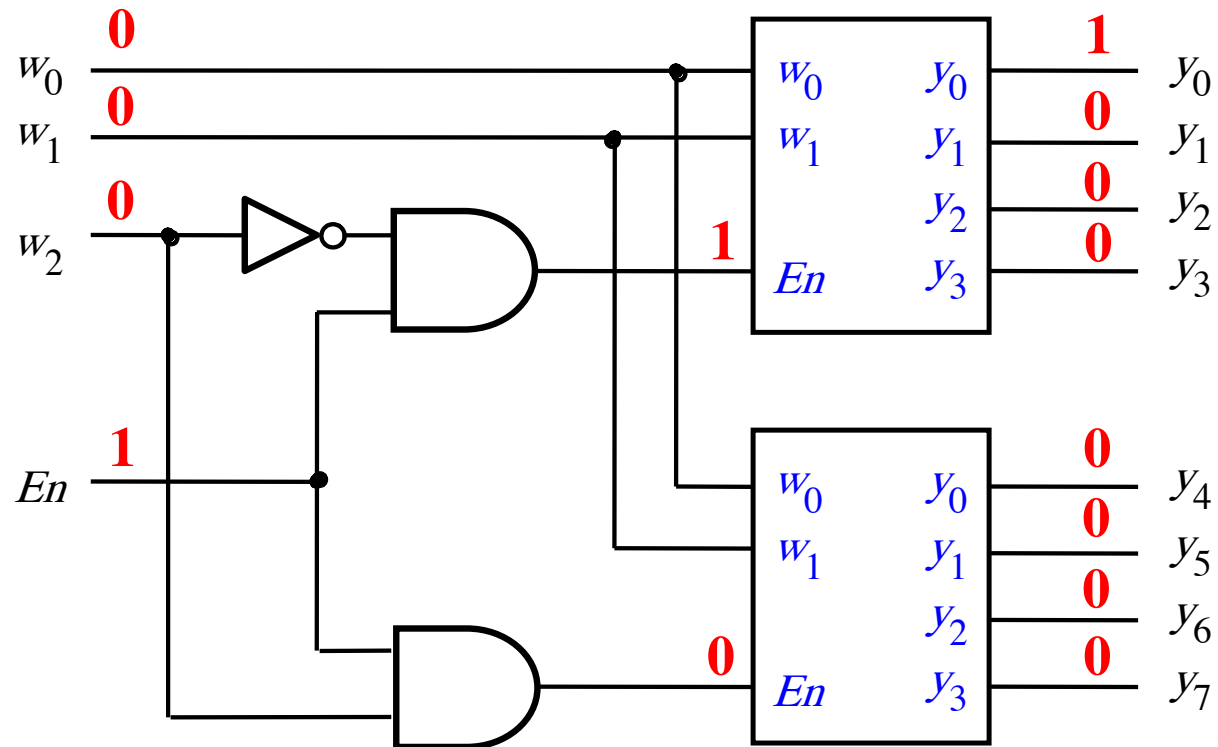
A 3-to-8 decoder using two 2-to-4 decoders



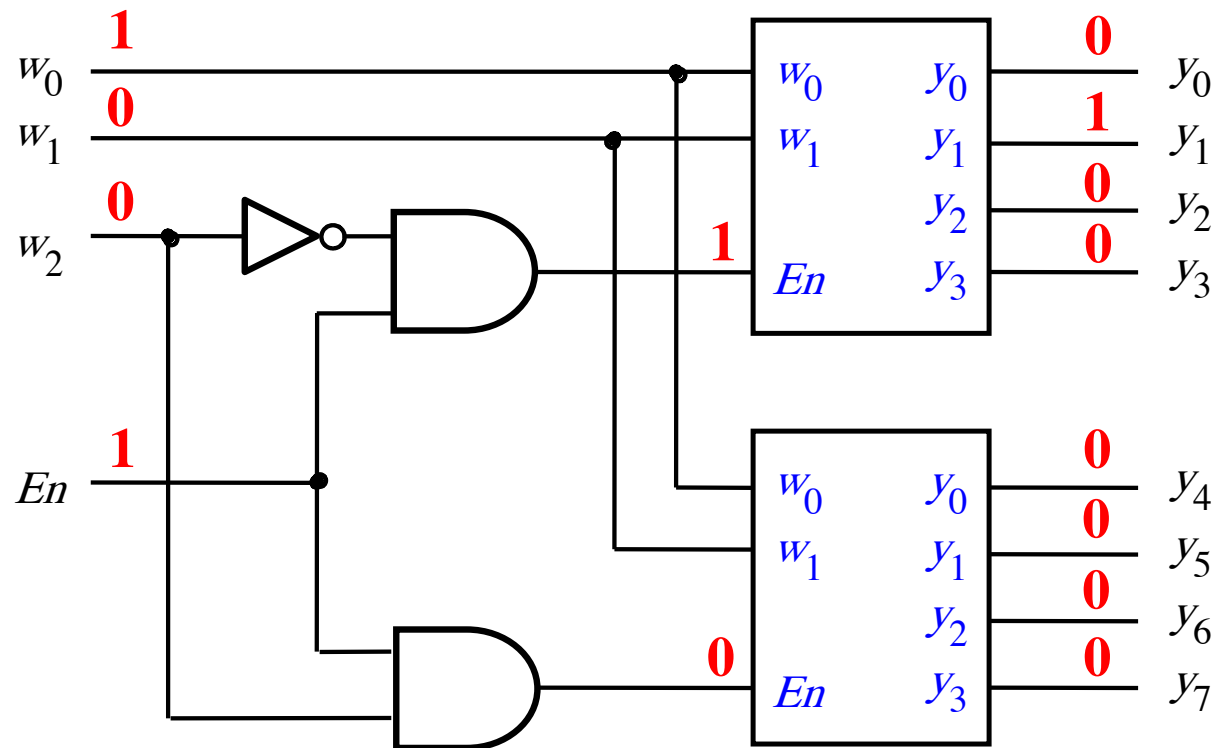
A 3-to-8 decoder using two 2-to-4 decoders



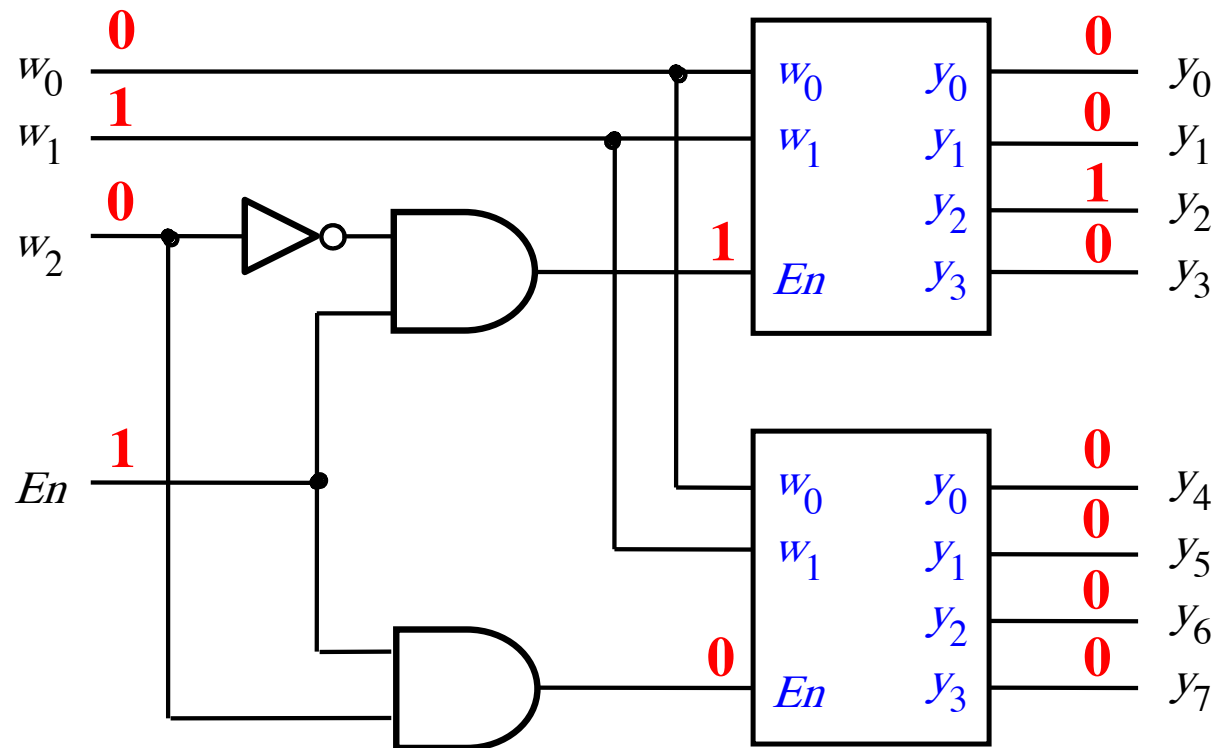
A 3-to-8 decoder using two 2-to-4 decoders



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

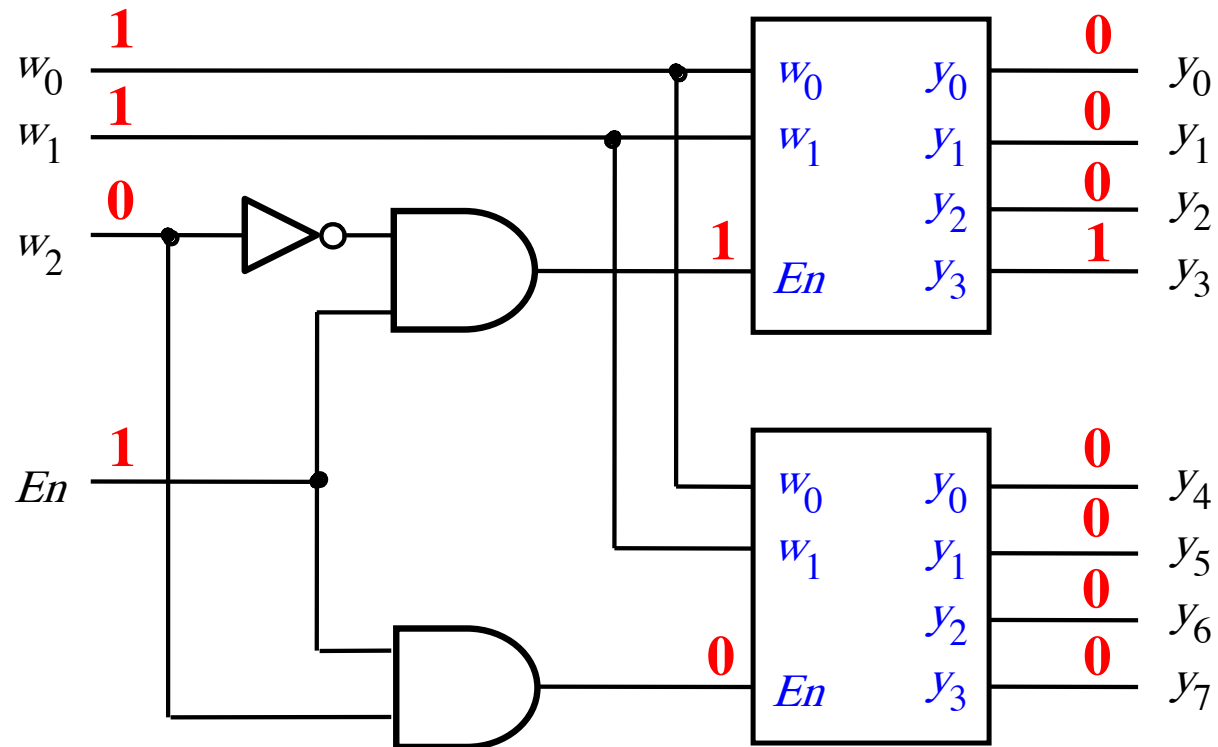


A 3-to-8 decoder using two 2-to-4 decoders

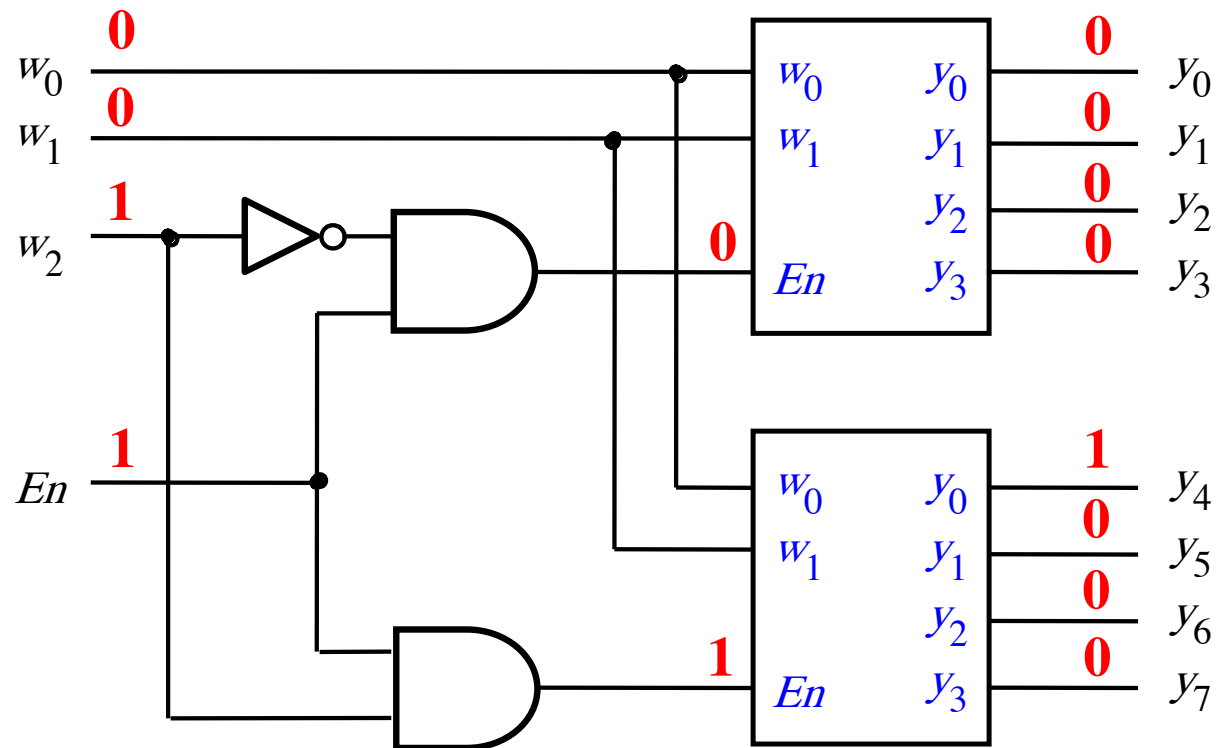


[Figure 4.15 from the textbook]

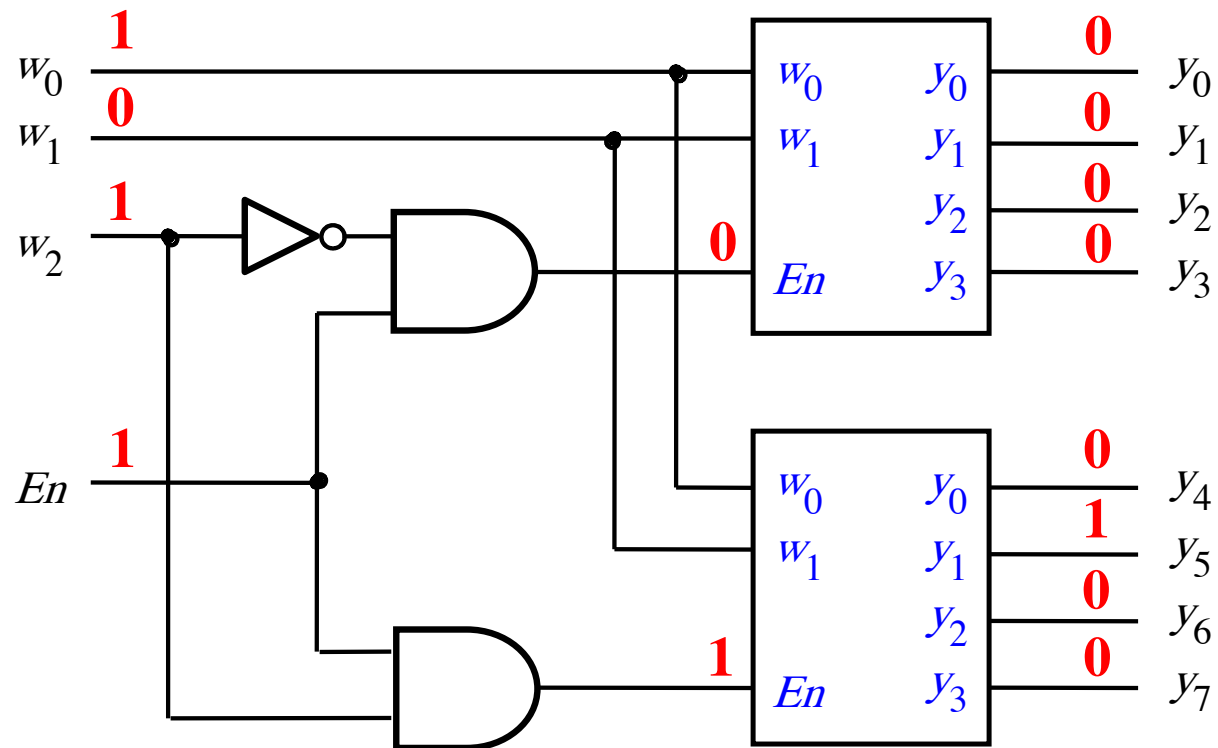
A 3-to-8 decoder using two 2-to-4 decoders



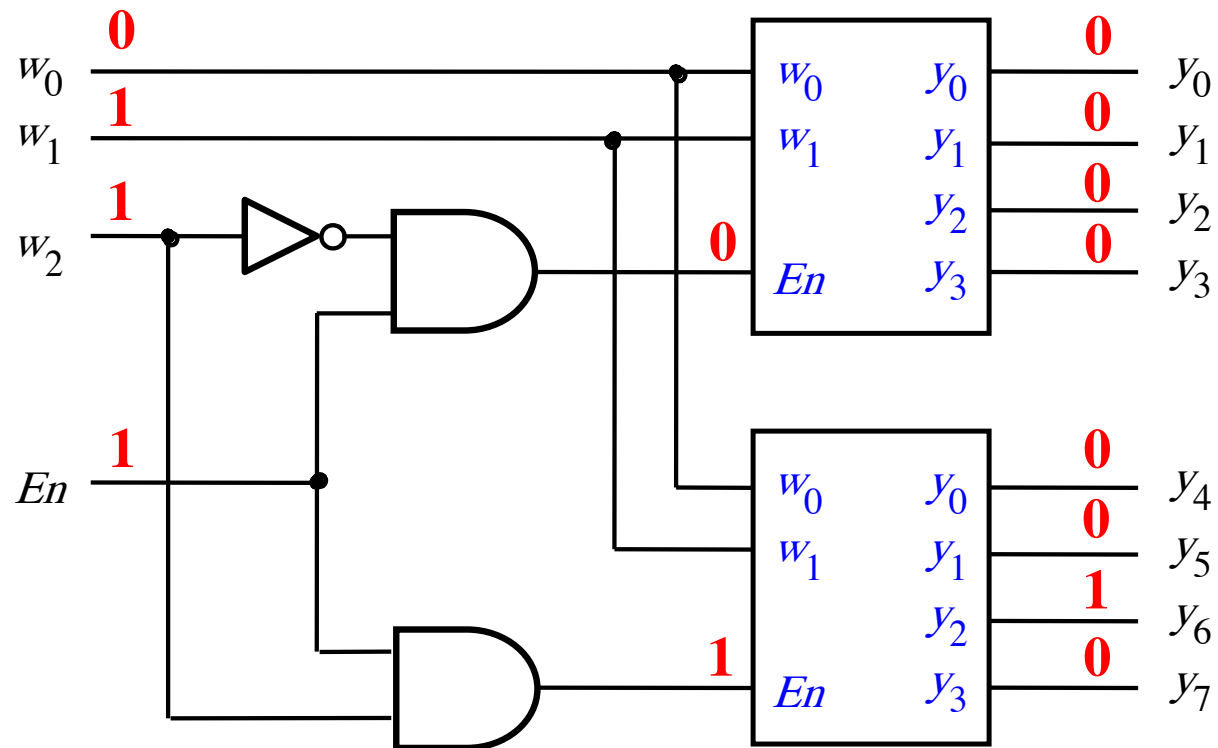
A 3-to-8 decoder using two 2-to-4 decoders



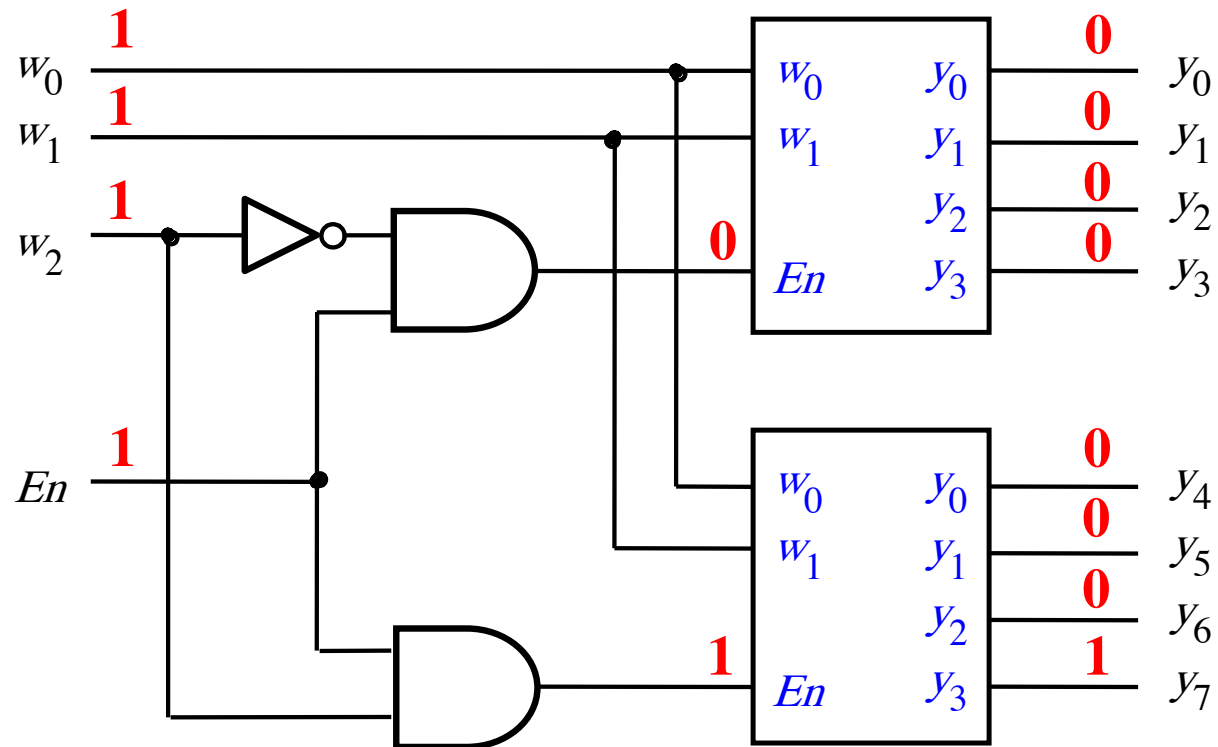
A 3-to-8 decoder using two 2-to-4 decoders



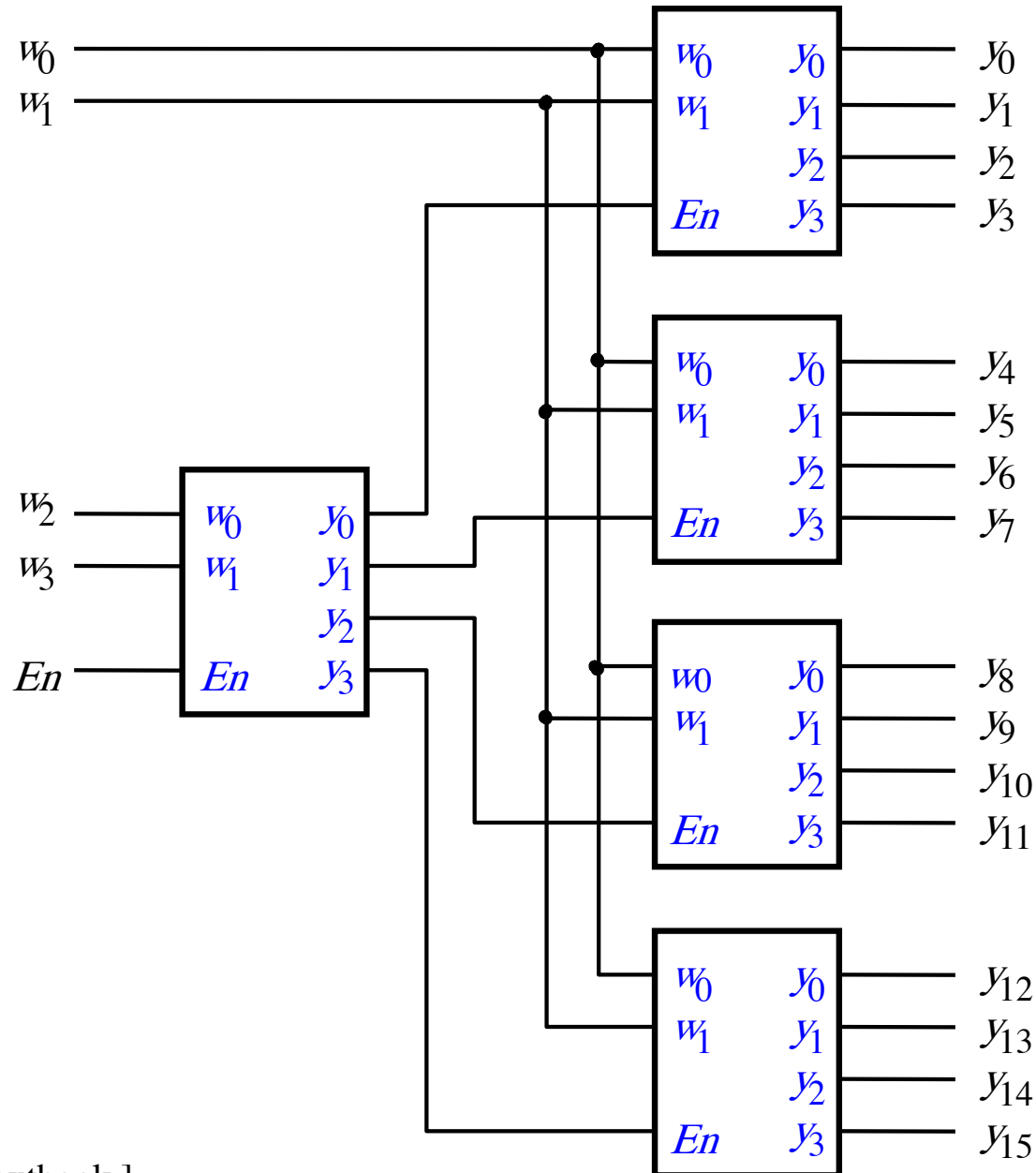
A 3-to-8 decoder using two 2-to-4 decoders



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

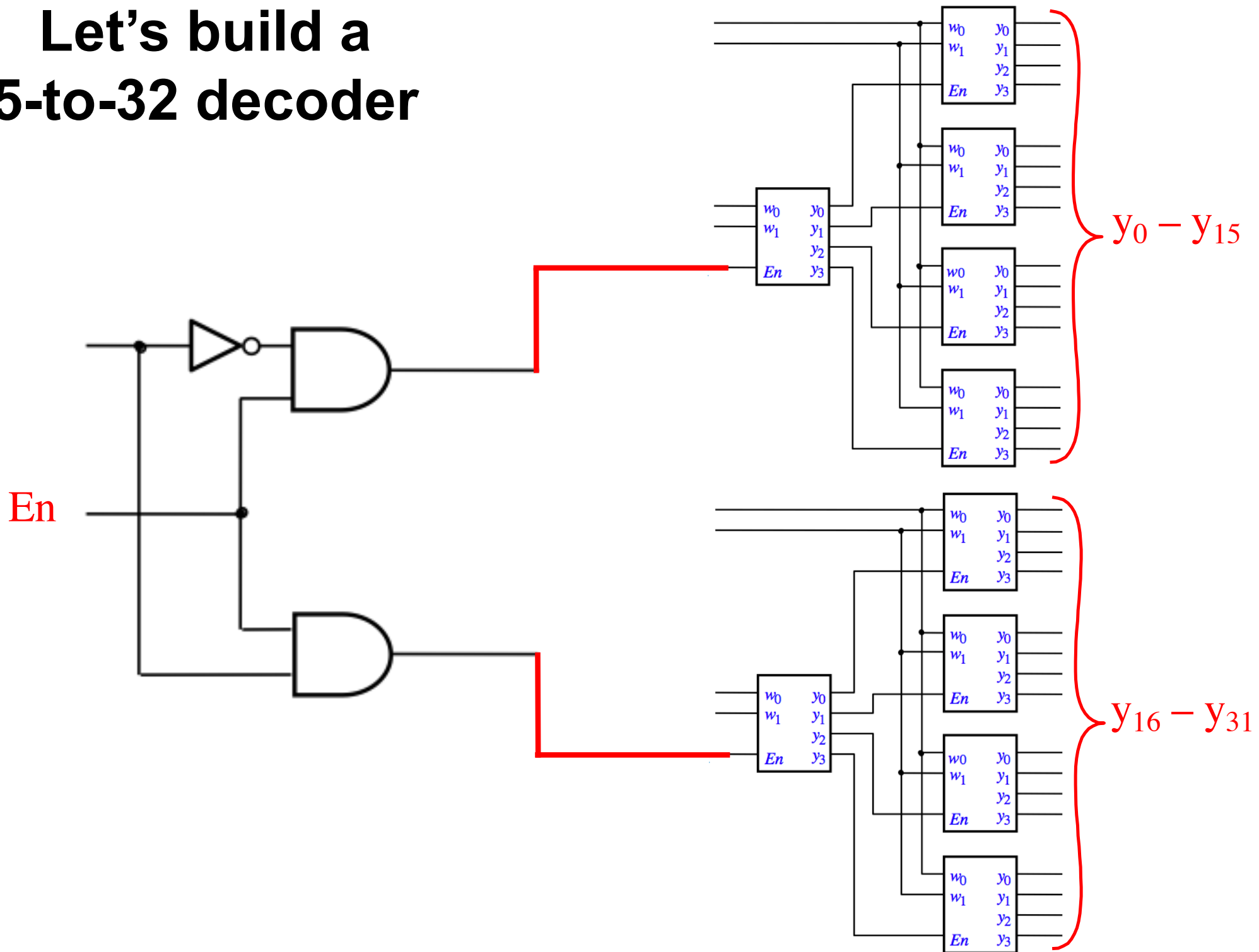


A 4-to-16 decoder built using a decoder tree

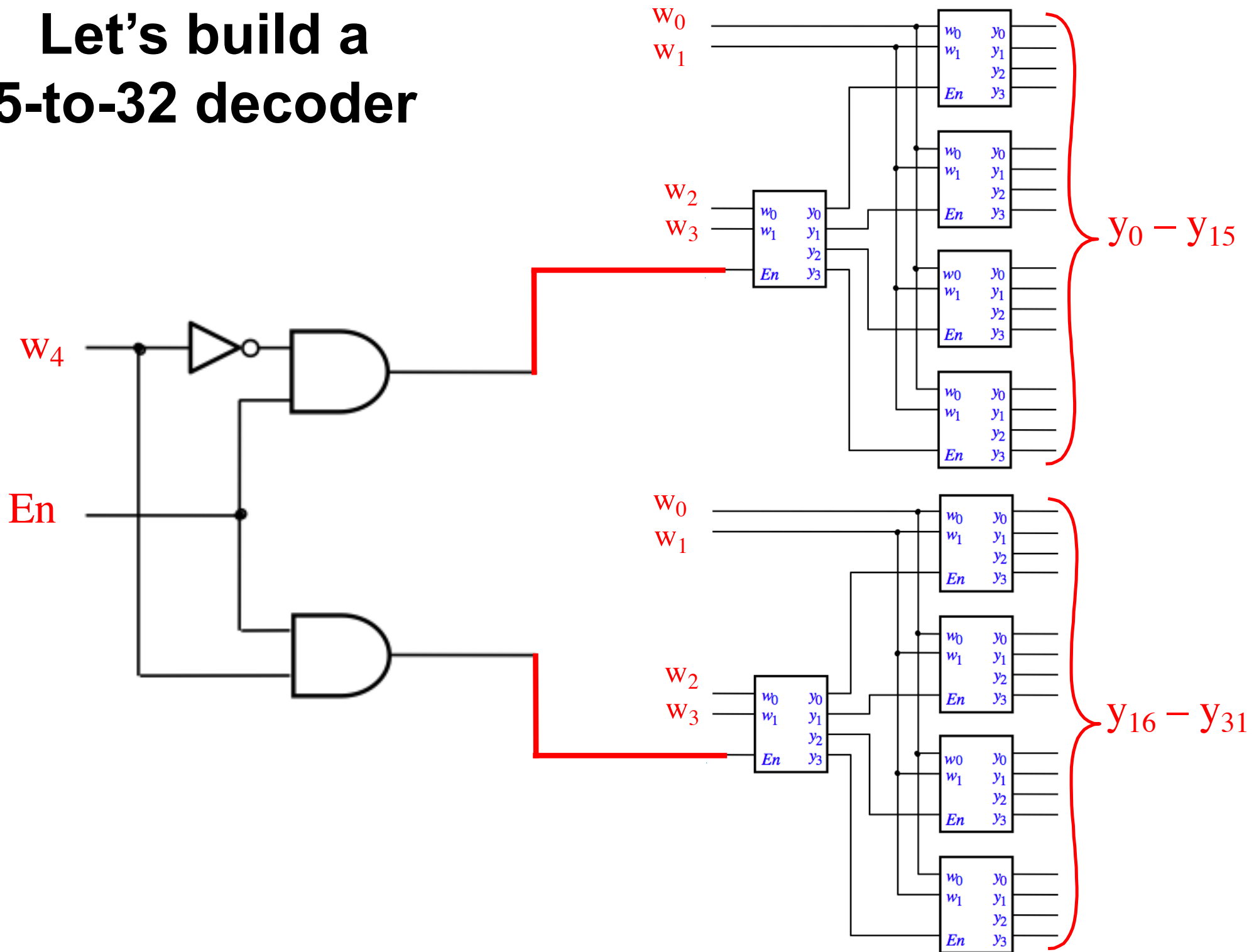


[Figure 4.16 from the textbook]

Let's build a 5-to-32 decoder



Let's build a 5-to-32 decoder

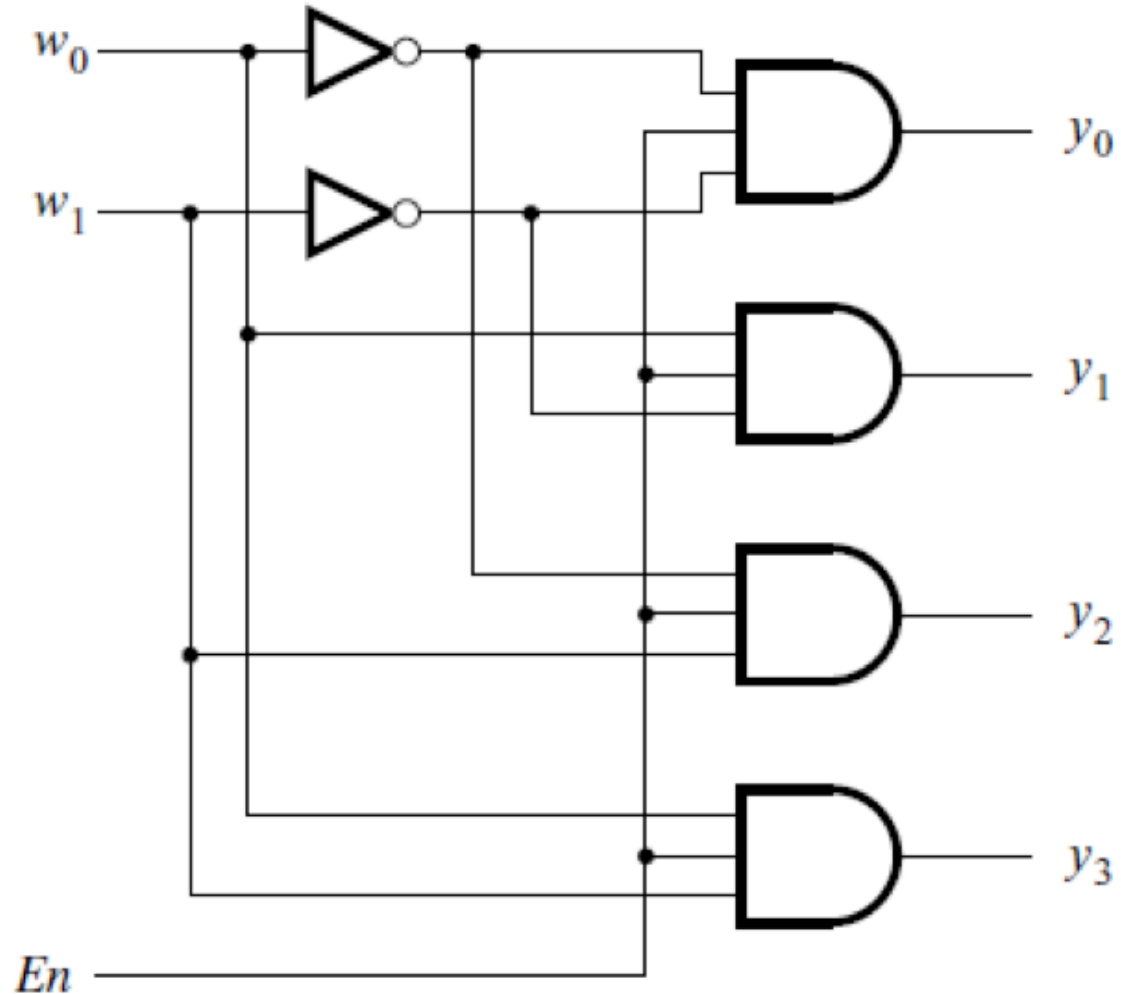


Demultiplexers

1-to-4 Demultiplexer (Definition)

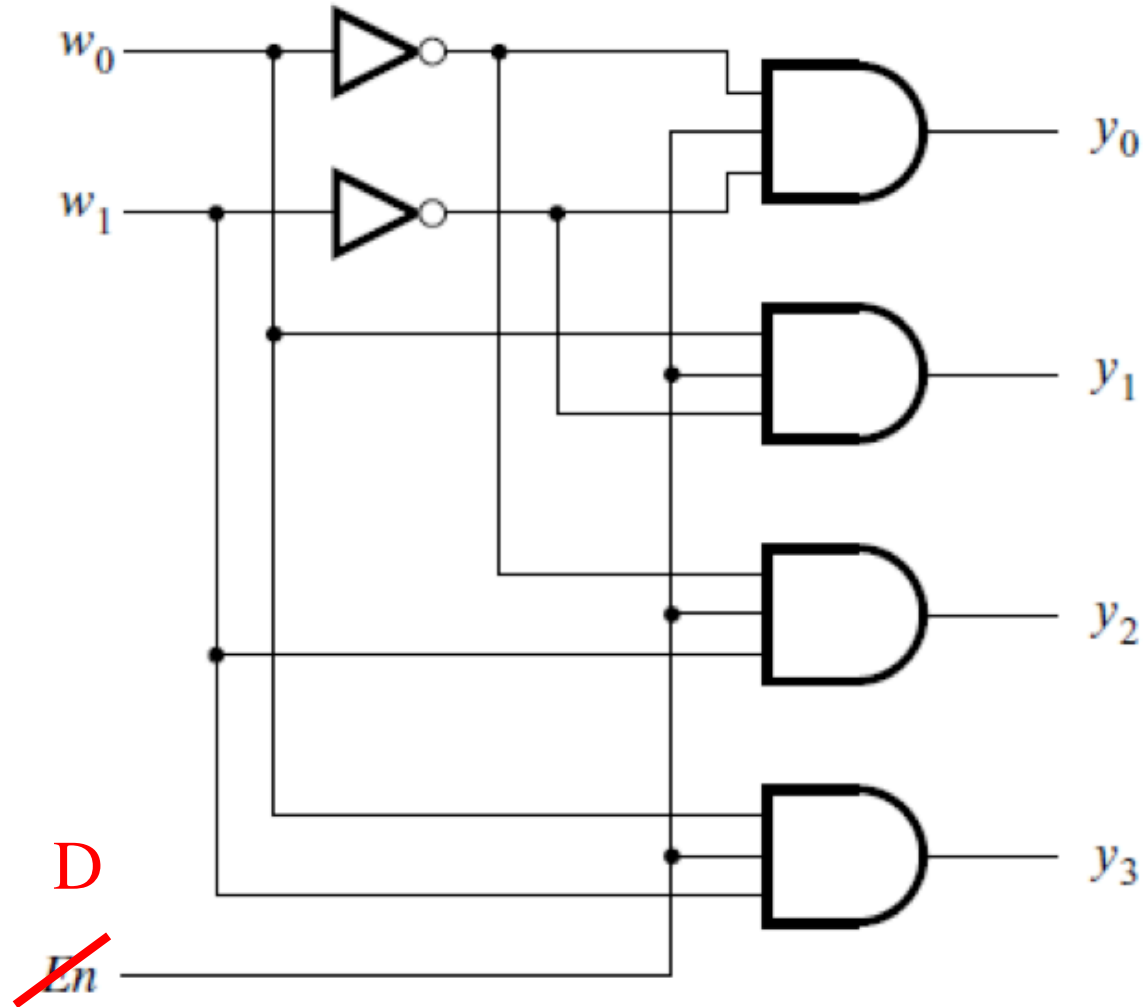
- Has one data input line: D
- Has two output select lines: w_1 and w_0
- Has four outputs: y_0 , y_1 , y_2 , and y_3
- If $w_1=0$ and $w_0=0$, then the output y_0 is set to D
- If $w_1=0$ and $w_0=1$, then the output y_1 is set to D
- If $w_1=1$ and $w_0=0$, then the output y_2 is set to D
- If $w_1=1$ 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.

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



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

output
select
lines

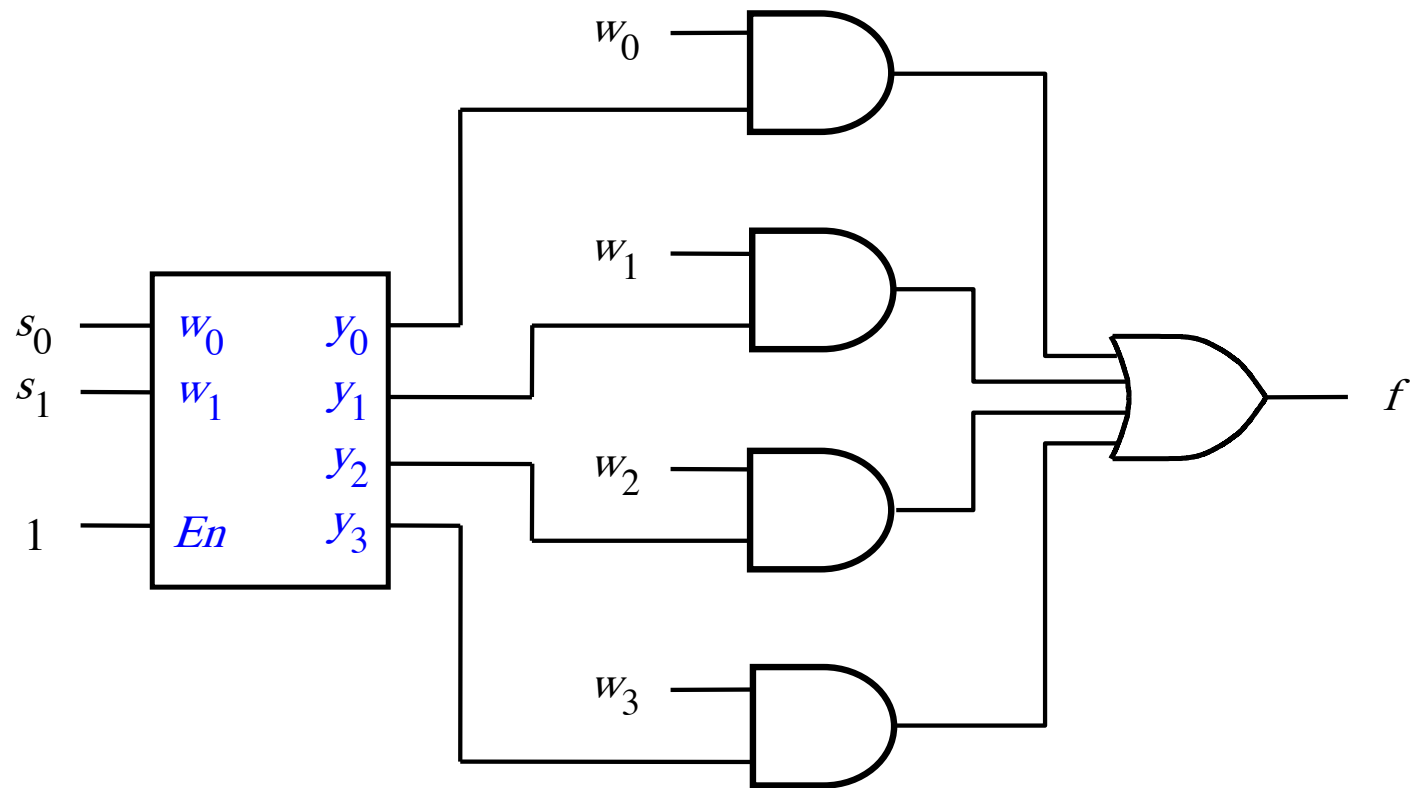


the
four
output
lines

data
input
line

Multiplexers (Implemented with Decoders)

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



Encoders

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

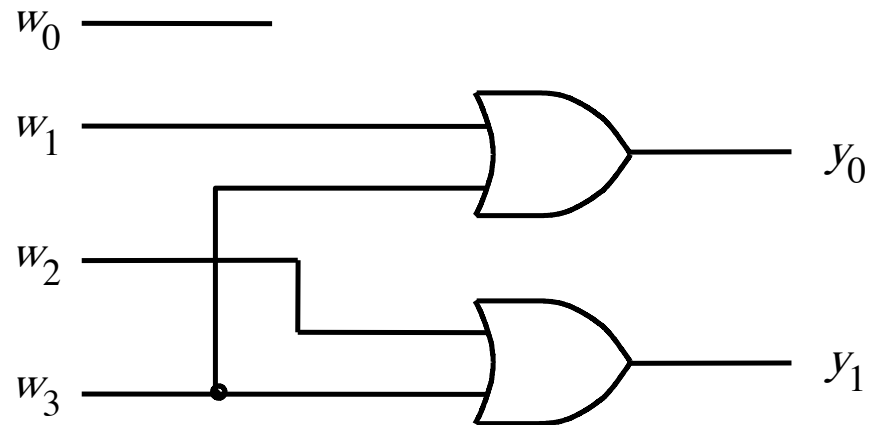
Truth table for a 4-to-2 binary encoder

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

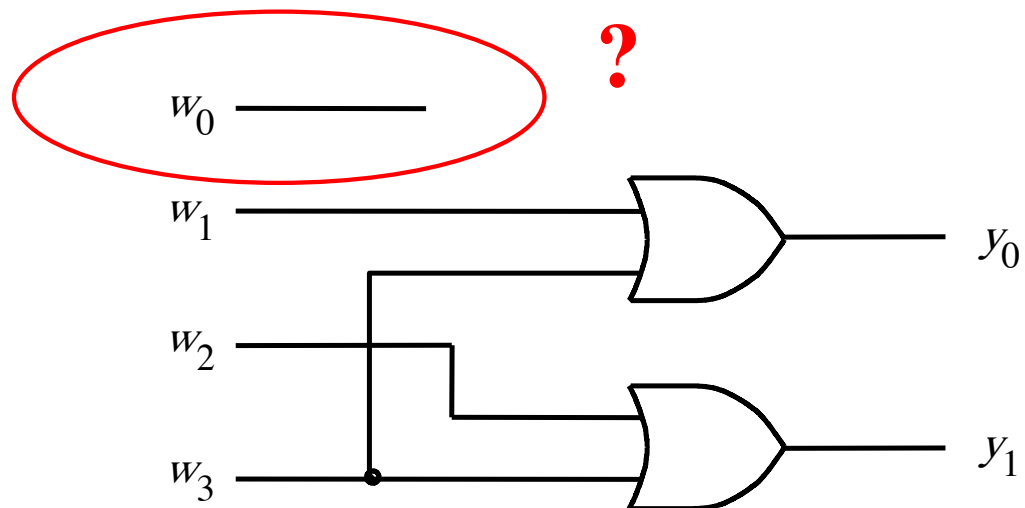
Circuit for a 4-to-2 binary encoder

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



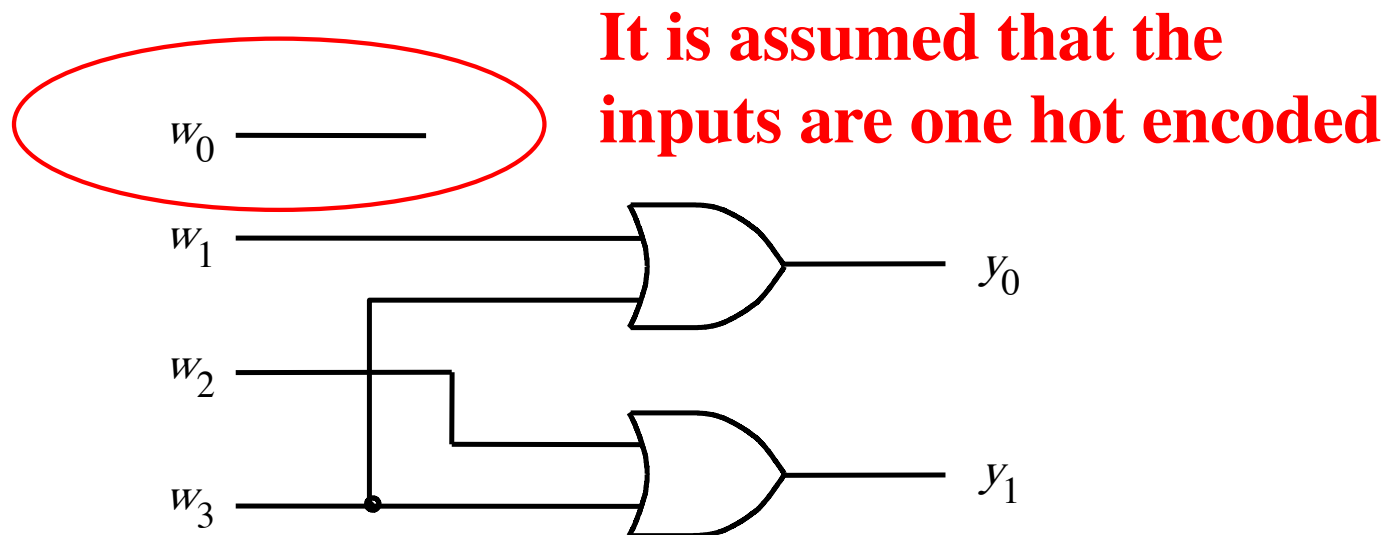
Circuit for a 4-to-2 binary encoder

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



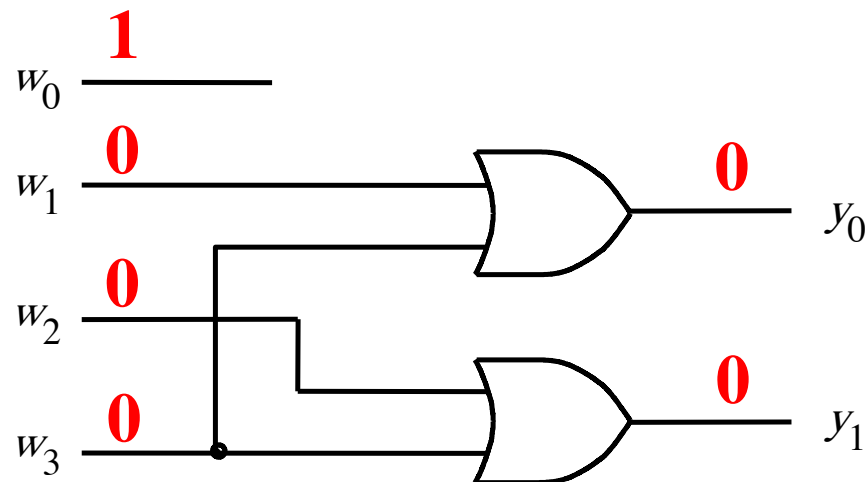
Circuit for a 4-to-2 binary encoder

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



Circuit for a 4-to-2 binary encoder

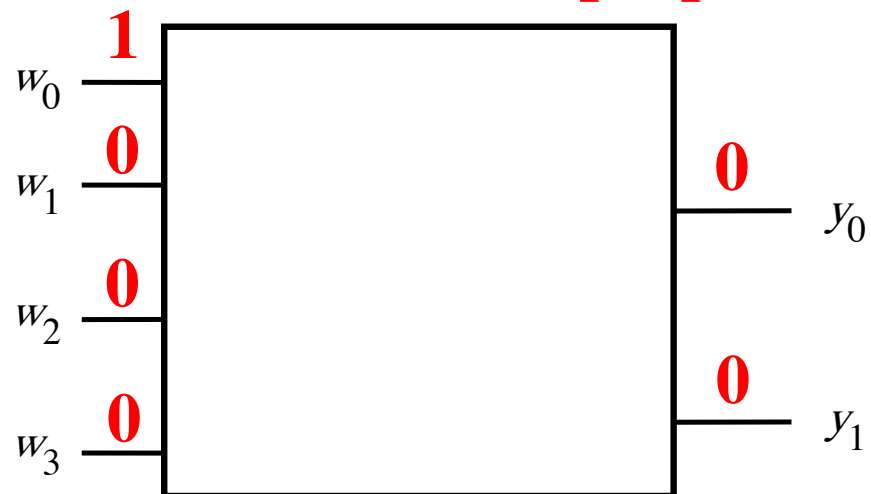
w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1



Circuit for a 4-to-2 binary encoder

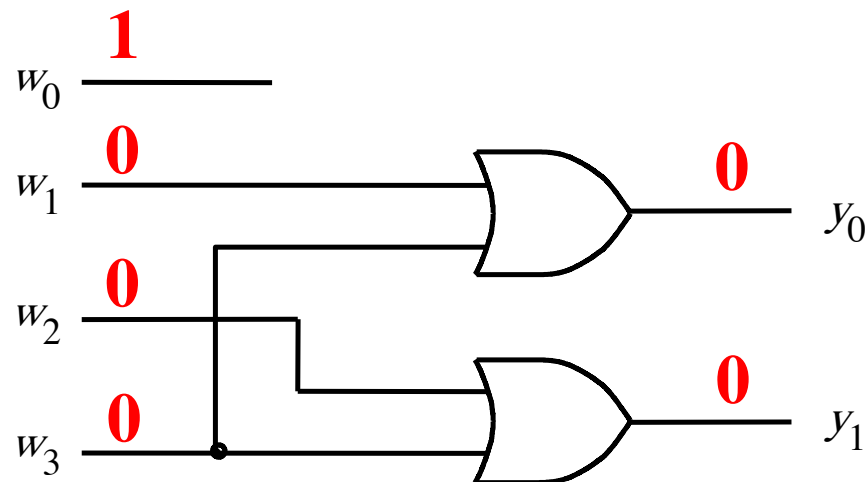
w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

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



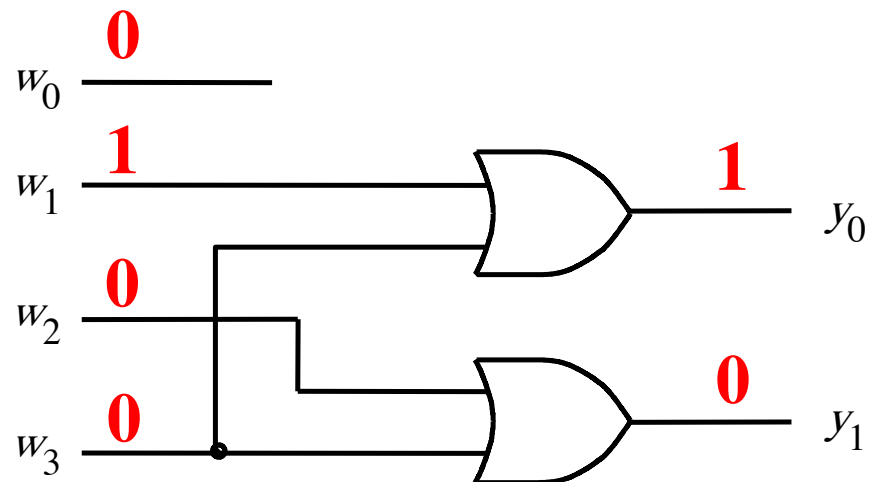
Circuit for a 4-to-2 binary encoder

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



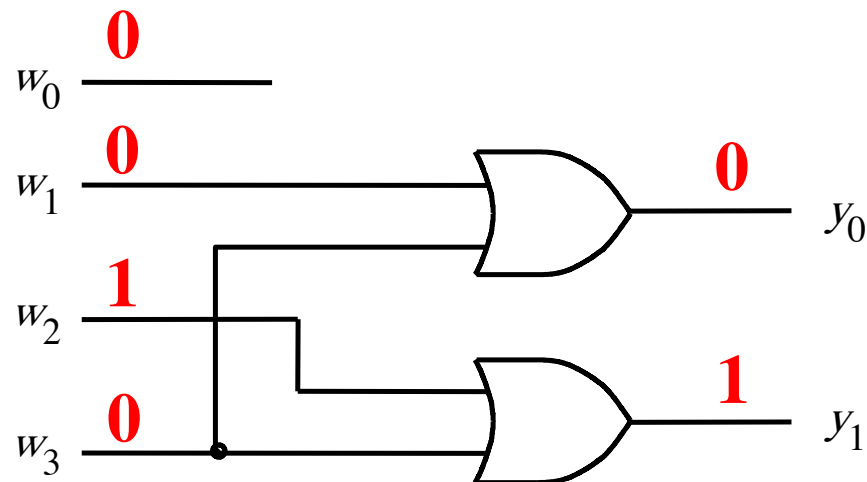
Circuit for a 4-to-2 binary encoder

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



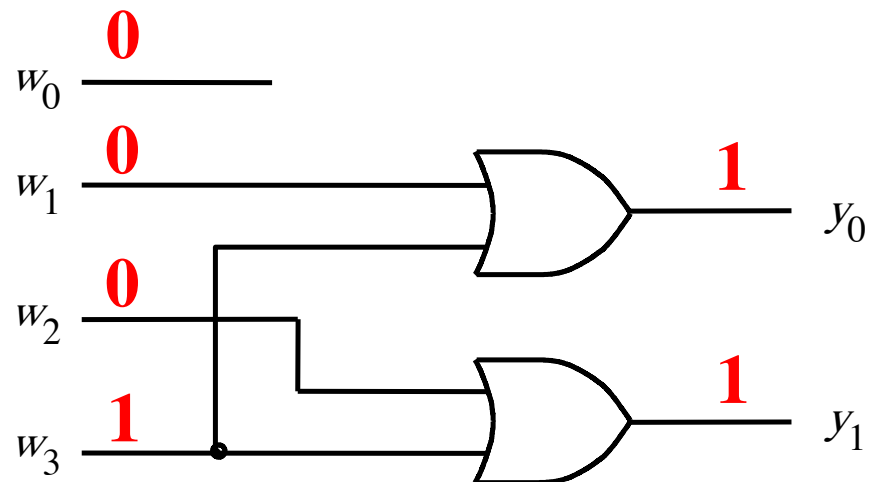
Circuit for a 4-to-2 binary encoder

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



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



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
<hr/>					
0	1	0	0	1	0
0	1	0	1	d	d
0	1	1	0	d	d
0	1	1	1	d	d
<hr/>					
1	0	0	0	1	1
1	0	0	1	d	d
1	0	1	0	d	d
1	0	1	1	d	d
<hr/>					
1	1	0	0	d	d
1	1	0	1	d	d
1	1	1	0	d	d
1	1	1	1	d	d

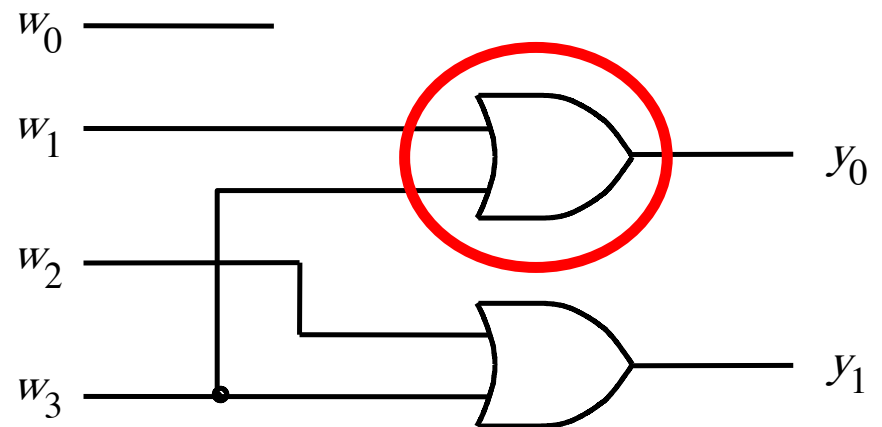
w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

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

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

$$y_0 = (w_1 + w_3)$$

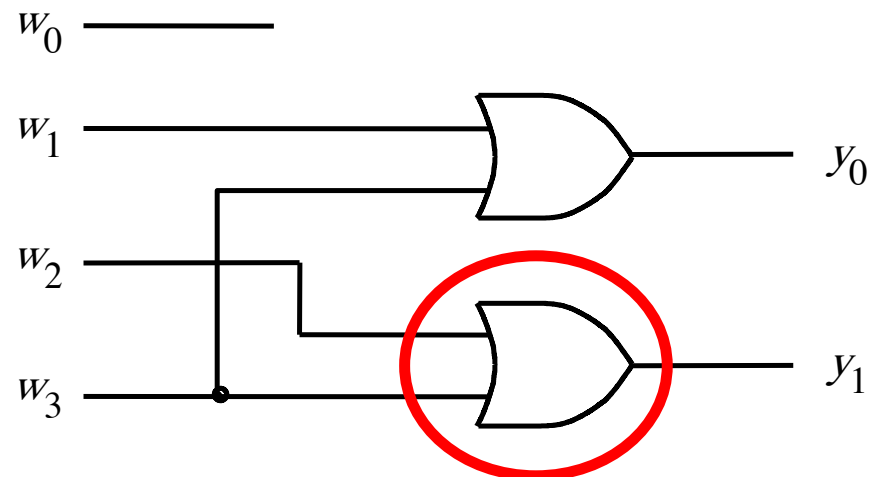


Expressions for 4-to-2 binary encoder

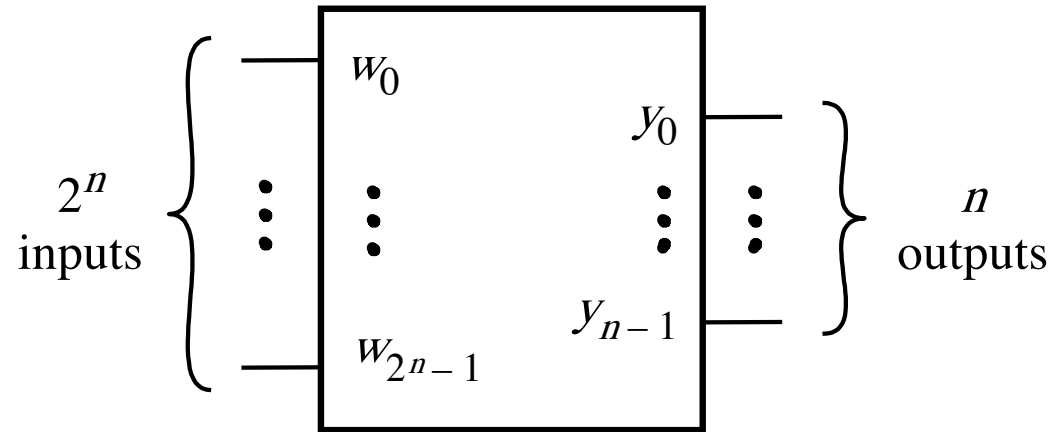
w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	0	d	d
0	0	0	1	0	0
0	0	1	0	0	1
0	0	1	1	d	d
0	1	0	0	1	0
0	1	0	1	d	d
0	1	1	0	d	d
0	1	1	1	d	d
1	0	0	0	1	1
1	0	0	1	d	d
1	0	1	0	d	d
1	0	1	1	d	d
1	1	0	0	d	d
1	1	0	1	d	d
1	1	1	0	d	d
1	1	1	1	d	d

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

$$y_1 = (w_3 + w_2)$$



A 2^n -to- n binary encoder



Priority Encoders

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

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

Truth table for a 4-to-2 priority encoder

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

		$w_3 w_2$			
$w_1 w_0$		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

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

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

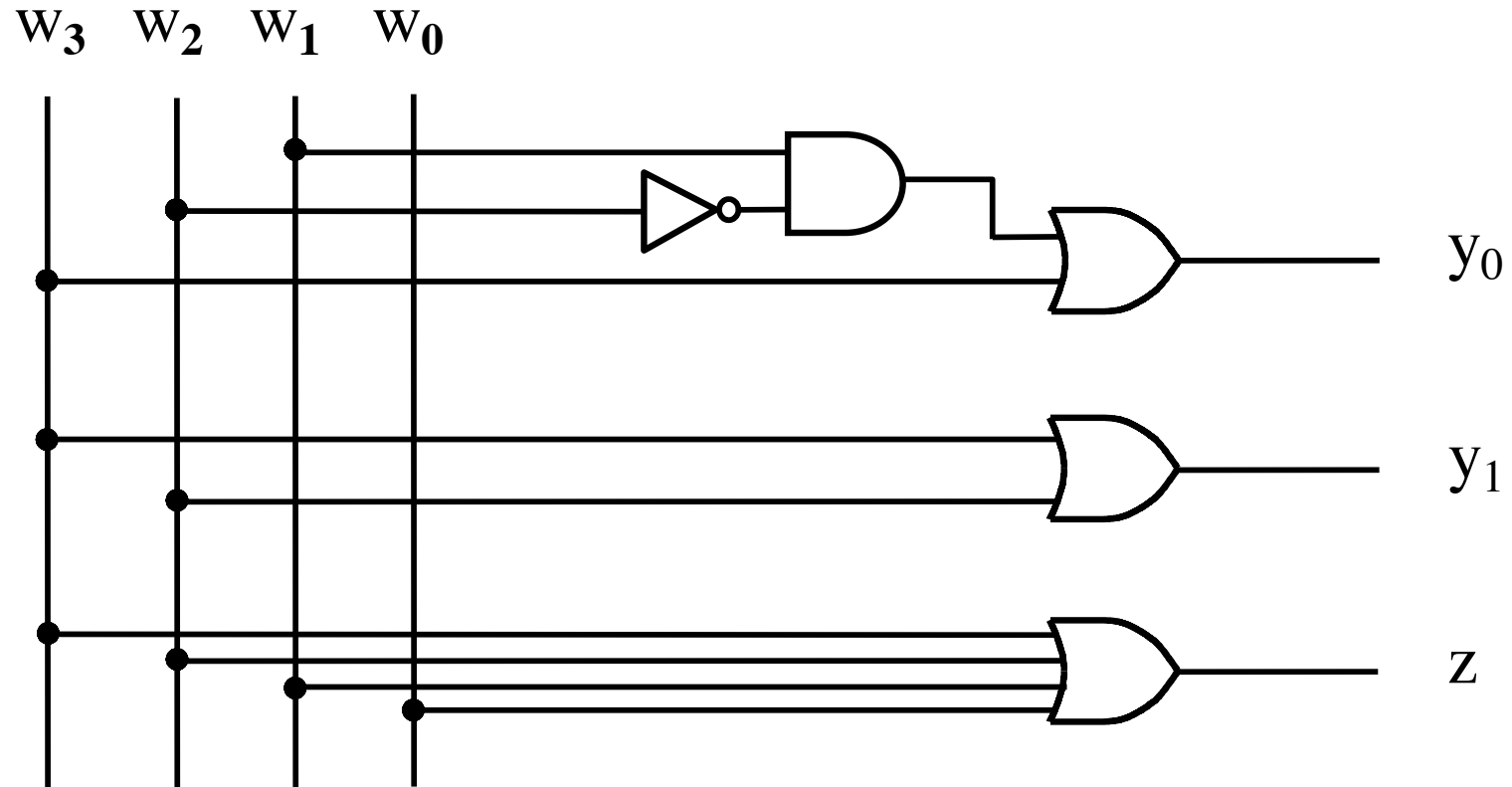
Expressions for 4-to-2 priority encoder

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

		$w_3 w_2$			
		00	01	11	10
$w_1 w_0$	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

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

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

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

$$i_3 = w_3$$

$$y_0 = i_1 + i_3$$

$$y_1 = i_2 + i_3$$

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

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

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

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

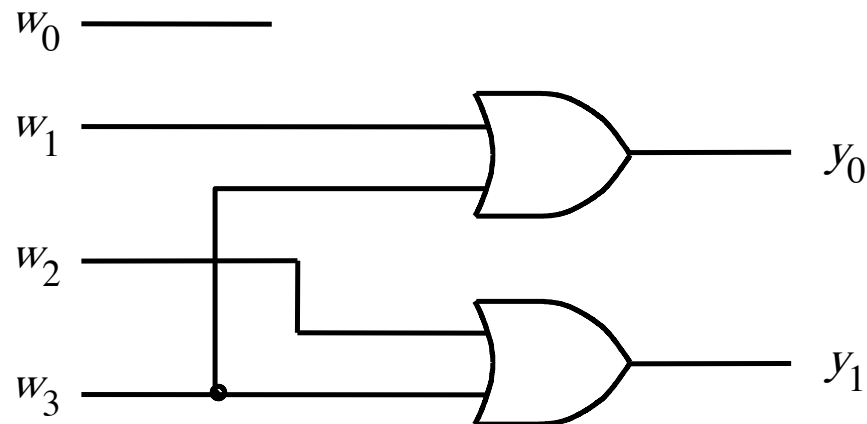
$$i_2 = \bar{w}_3 w_2$$

$$i_3 = w_3$$

$$y_0 = i_1 + i_3$$

$$y_1 = i_2 + i_3$$

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



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

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

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

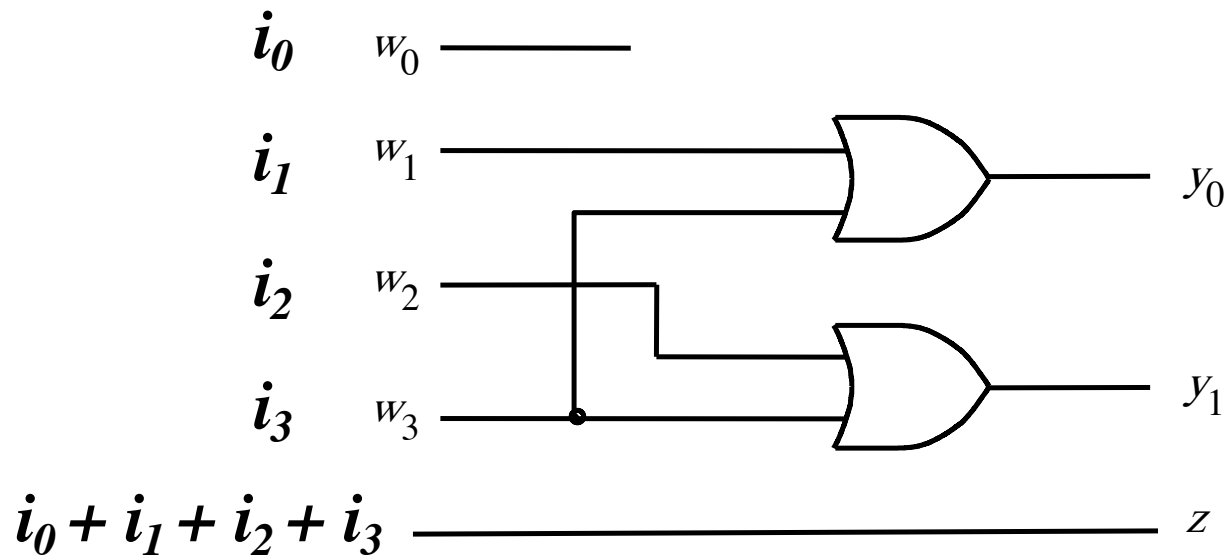
$$i_2 = \bar{w}_3 w_2$$

$$i_3 = w_3$$

$$y_0 = i_1 + i_3$$

$$y_1 = i_2 + i_3$$

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



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

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

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

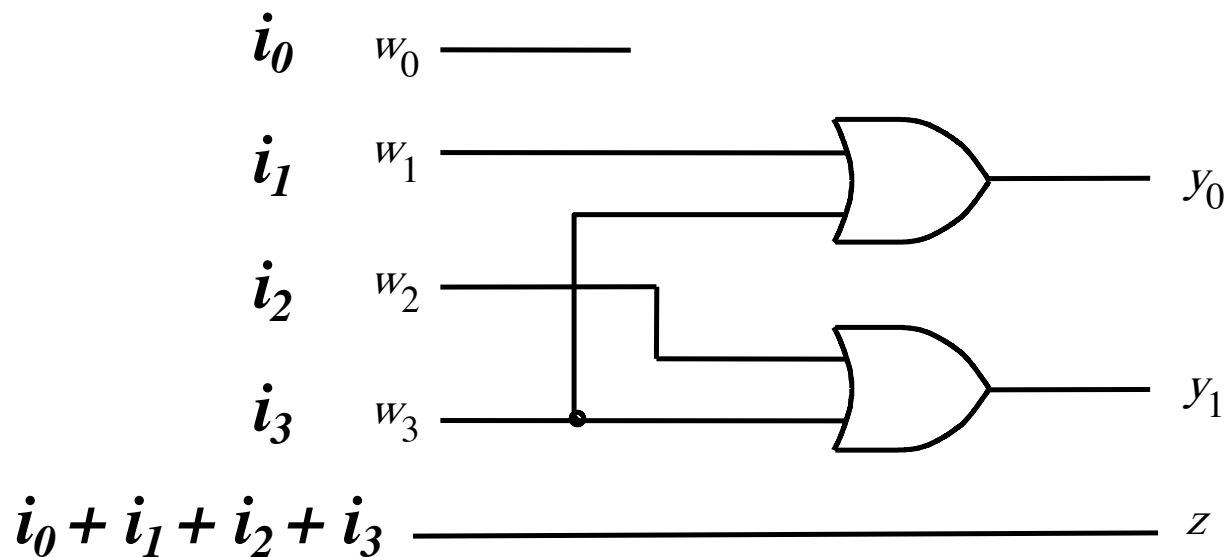
$$i_2 = \bar{w}_3 w_2$$

$$i_3 = w_3$$

$$y_0 = i_1 + i_3$$

$$y_1 = i_2 + i_3$$

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



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

Code Converters

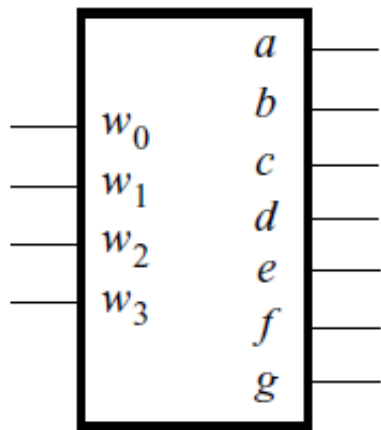
Code Converter (Definition)

- **Converts from one type of input encoding to a different type of output encoding.**

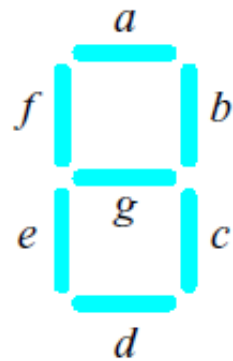
Code Converter (Definition)

- **Converts from one type of input encoding to a different type of output encoding.**
- **A decoder does that as well, but its outputs are always one-hot encoded so the output code is really only one type of output code.**
- **A binary encoder does that as well but its inputs are always one-hot encoded so the input code is really only one type of input code.**

A hex-to-7-segment display code converter



(a) Code converter

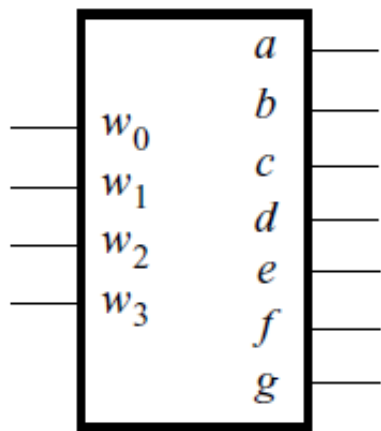


(b) 7-segment display

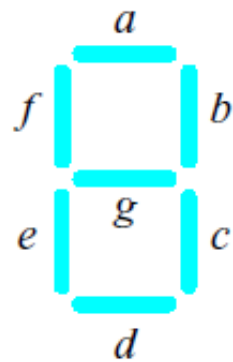
w_3	w_2	w_1	w_0	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1
1	0	1	0	1	1	1	0	1	1	1
1	0	1	1	0	0	1	1	1	1	1
1	1	0	0	1	0	0	1	1	1	0
1	1	0	1	0	1	1	1	1	0	1
1	1	1	0	1	0	0	1	1	1	1
1	1	1	1	1	0	0	0	1	1	1

(c) Truth table

A hex-to-7-segment display code converter



(a) Code converter

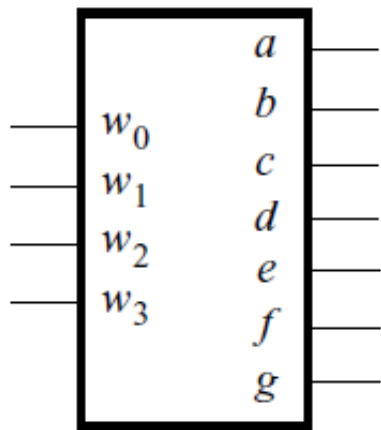


(b) 7-segment display

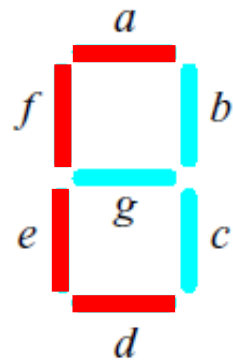
w_3	w_2	w_1	w_0	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1
1	0	1	0	1	1	1	0	1	1	1
1	0	1	1	0	0	1	1	1	1	1
1	1	0	0	1	0	0	1	1	1	0
1	1	0	1	0	1	1	1	1	0	1
1	1	1	0	1	0	0	1	1	1	1
1	1	1	1	1	0	0	0	1	1	1

(c) Truth table

A hex-to-7-segment display code converter



(a) Code converter

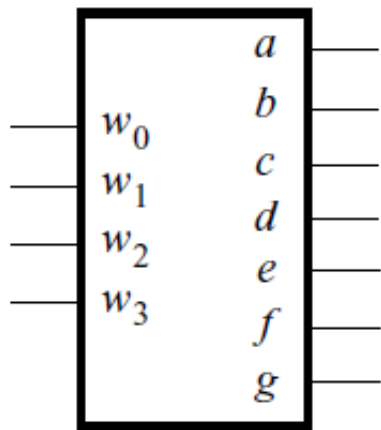


(b) 7-segment display

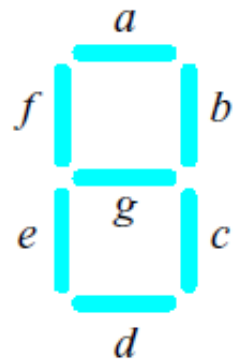
w_3	w_2	w_1	w_0	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1
1	0	1	0	1	1	1	0	1	1	1
1	0	1	1	0	0	1	1	1	1	1
1	1	0	0	1	0	0	1	1	1	0
1	1	0	1	0	1	1	1	1	0	1
1	1	1	0	1	0	0	1	1	1	1
1	1	1	1	1	0	0	0	1	1	1

(c) Truth table

A hex-to-7-segment display code converter



(a) Code converter

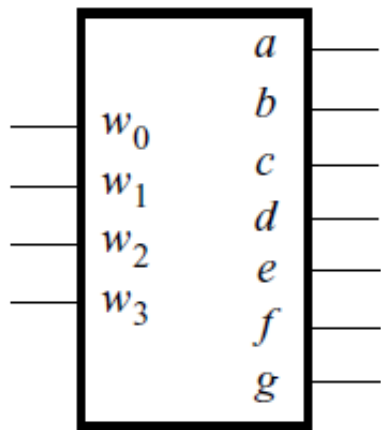


(b) 7-segment display

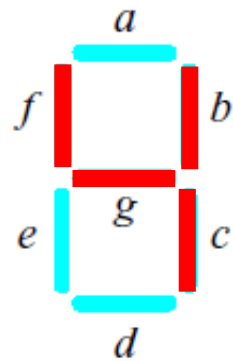
w_3	w_2	w_1	w_0	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1
1	0	1	0	1	1	1	0	1	1	1
1	0	1	1	0	0	1	1	1	1	1
1	1	0	0	1	0	0	1	1	1	0
1	1	0	1	0	1	1	1	1	0	1
1	1	1	0	1	0	0	1	1	1	1
1	1	1	1	1	0	0	0	1	1	1

(c) Truth table

A hex-to-7-segment display code converter



(a) Code converter



(b) 7-segment display

w_3	w_2	w_1	w_0	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1
1	0	1	0	1	1	1	0	1	1	1
1	0	1	1	0	0	1	1	1	1	1
1	1	0	0	1	0	0	1	1	1	0
1	1	0	1	0	1	1	1	1	0	1
1	1	1	0	1	0	0	1	1	1	1
1	1	1	1	1	0	0	0	1	1	1

(c) Truth table

What is the circuit for this code converter?

x_3	x_2	x_1	x_0	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1
1	0	1	0	1	1	1	0	1	1	1
1	0	1	1	0	0	1	1	1	1	1
1	1	0	0	1	0	0	1	1	1	0
1	1	0	1	0	1	1	1	1	0	1
1	1	1	0	1	0	0	1	1	1	1
1	1	1	1	1	0	0	0	1	1	1

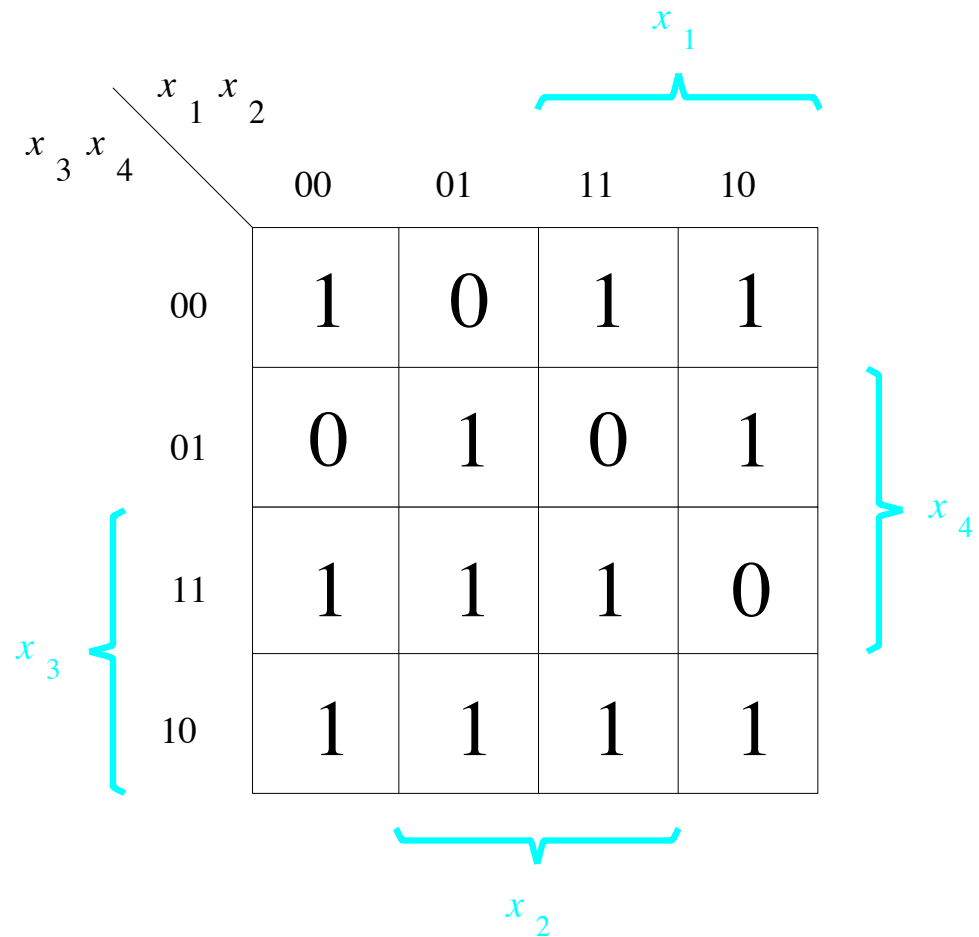
What is the circuit for this code converter?

x_3	x_2	x_1	x_0	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1
1	0	1	0	1	1	1	0	1	1	1
1	0	1	1	0	0	1	1	1	1	1
1	1	0	0	1	0	0	1	1	1	0
1	1	0	1	0	1	1	1	1	0	1
1	1	1	0	1	0	0	1	1	1	1
1	1	1	1	1	0	0	0	1	1	1

$$f(x_1, x_2, x_3, x_4) = \sum m(0, 2, 3, 5, 6, 7, 8, 9, 10, 12, 14, 15)$$

What is the circuit for this code converter?

x_3	x_2	x_1	x_0	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1
1	0	1	0	1	1	1	0	1	1	1
1	0	1	1	0	0	1	1	1	1	1
1	1	0	0	1	0	0	1	1	1	0
1	1	0	1	0	1	1	1	1	0	1
1	1	1	0	1	0	0	1	1	1	1
1	1	1	1	1	0	0	0	1	1	1



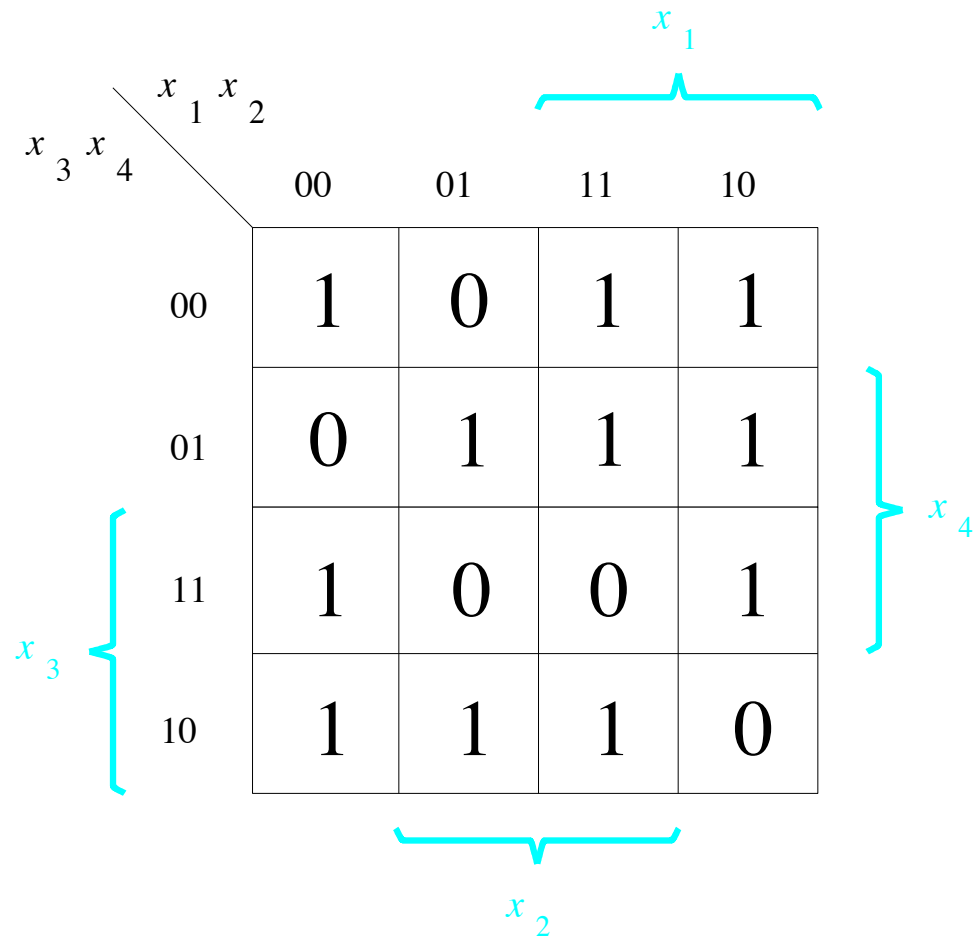
$$f(x_1, x_2, x_3, x_4) = \sum m(0, 2, 3, 5, 6, 7, 8, 9, 10, 12, 14, 15)$$

What is the circuit for this code converter?

x_3	x_2	x_1	x_0	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1
1	0	1	0	1	1	1	0	1	1	1
1	0	1	1	0	0	1	1	1	1	1
1	1	0	0	1	0	0	1	1	1	0
1	1	0	1	0	1	1	1	1	0	1
1	1	1	0	1	0	0	1	1	1	1
1	1	1	1	1	0	0	0	1	1	1

What is the circuit for this code converter?

x_3	x_2	x_1	x_0	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1
1	0	1	0	1	1	1	0	1	1	1
1	0	1	1	0	0	1	1	1	1	1
1	1	0	0	1	0	0	1	1	1	0
1	1	0	1	0	1	1	1	1	0	1
1	1	1	0	1	0	0	1	1	1	1
1	1	1	1	1	0	0	0	1	1	1



$$f(x_1, x_2, x_3, x_4) = \sum m(0, 2, 3, 5, 6, 8, 9, 11, 12, 13, 14)$$

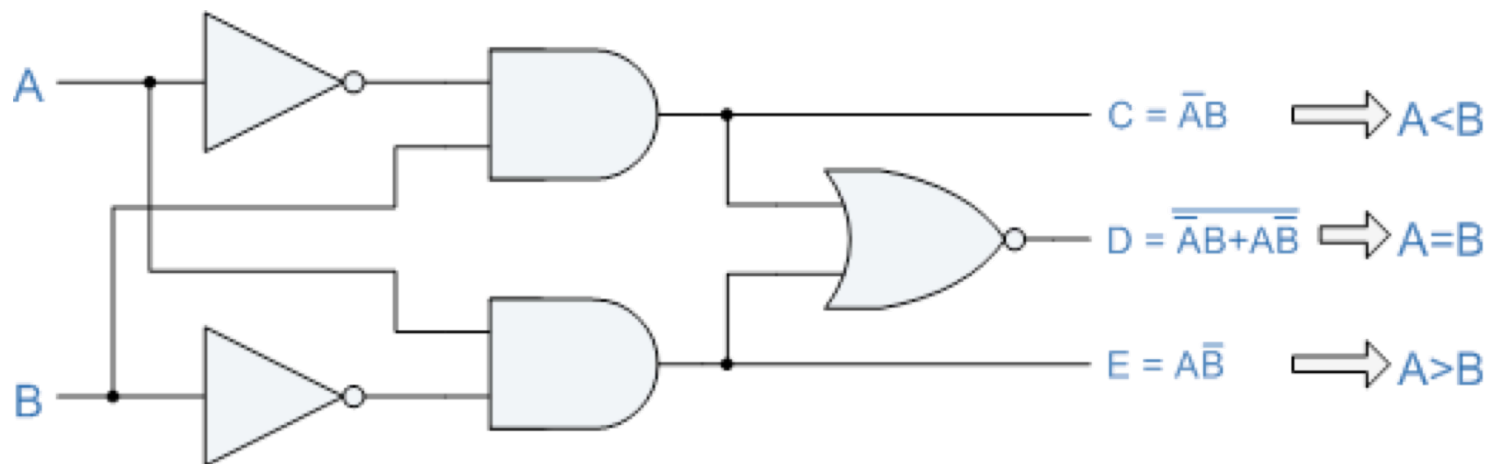
Arithmetic Comparison Circuits

Truth table for a one-bit digital comparator

Inputs		Outputs		
A	B	$A > B$	$A = B$	$A < B$
0	0	0	1	0
0	1	0	0	1
1	0	1	0	0
1	1	0	1	0

A one-bit digital comparator circuit

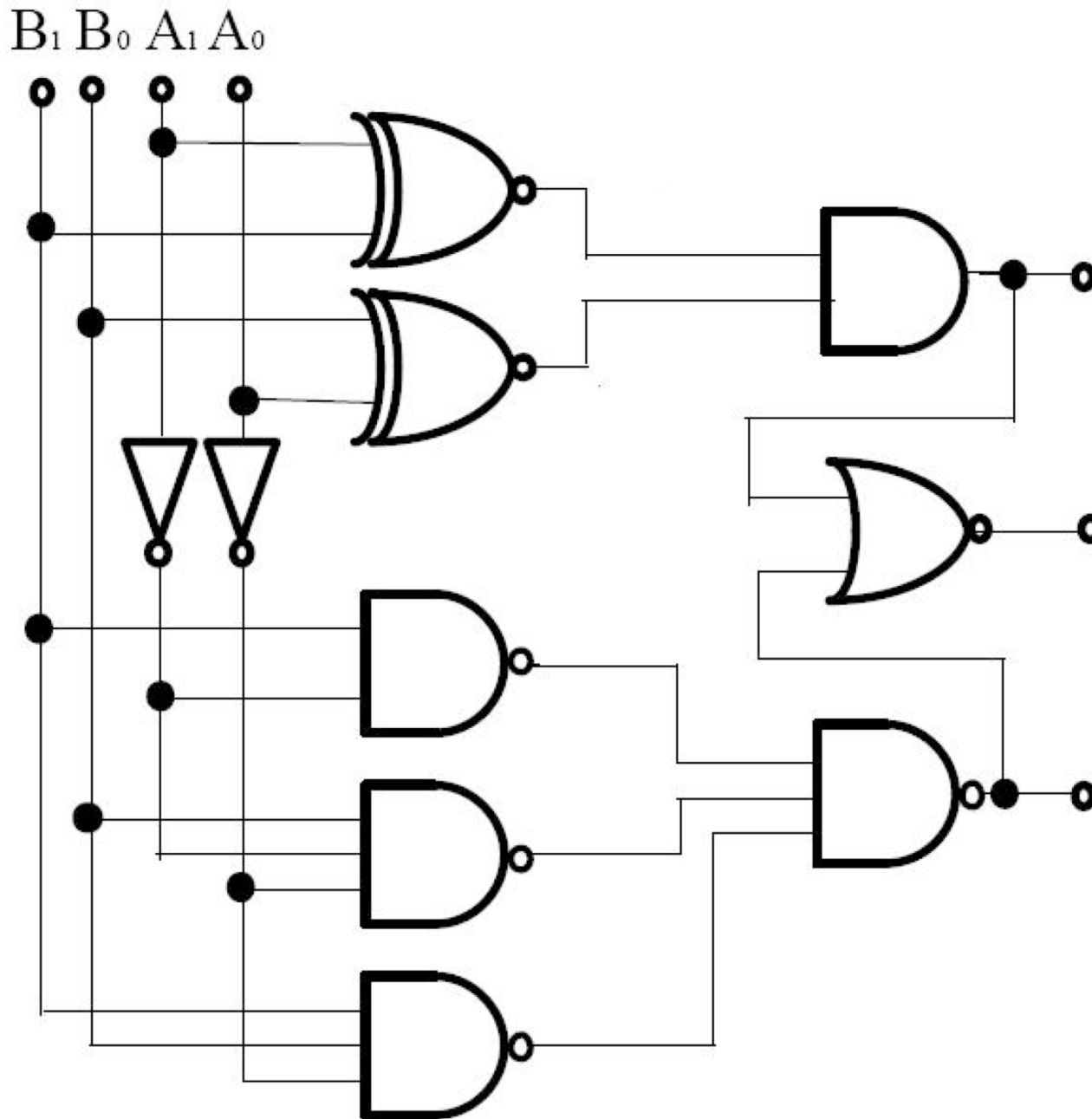
Inputs		Outputs		
A	B	$A > B$	$A = B$	$A < B$
0	0	0	1	0
0	1	0	0	1
1	0	1	0	0
1	1	0	1	0



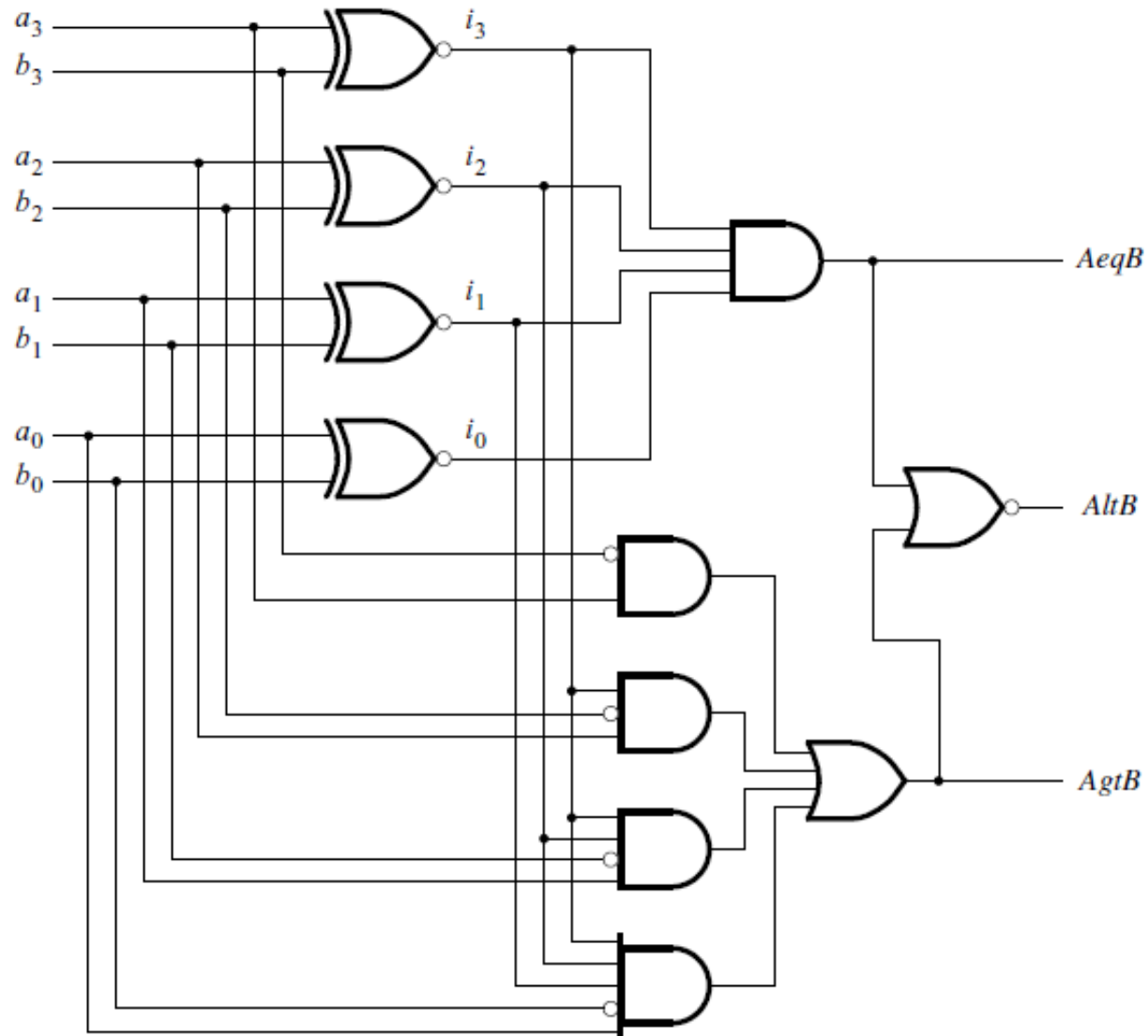
Truth table for a two-bit digital comparator

Inputs				Outputs		
A_1	A_0	B_1	B_0	$A < B$	$A = B$	$A > B$
0	0	0	0	0	1	0
0	0	0	1	1	0	0
0	0	1	0	1	0	0
0	0	1	1	1	0	0
0	1	0	0	0	0	1
0	1	0	1	0	1	0
0	1	1	0	1	0	0
0	1	1	1	1	0	0
1	0	0	0	0	0	1
1	0	0	1	0	0	1
1	0	1	0	0	1	0
1	0	1	1	1	0	0
1	1	0	0	0	0	1
1	1	0	1	0	0	1
1	1	1	0	0	0	1
1	1	1	1	0	1	0

A two-bit digital comparator circuit



A four-bit comparator circuit



[Figure 4.22 from the textbook]

Example Problems from Chapter 4

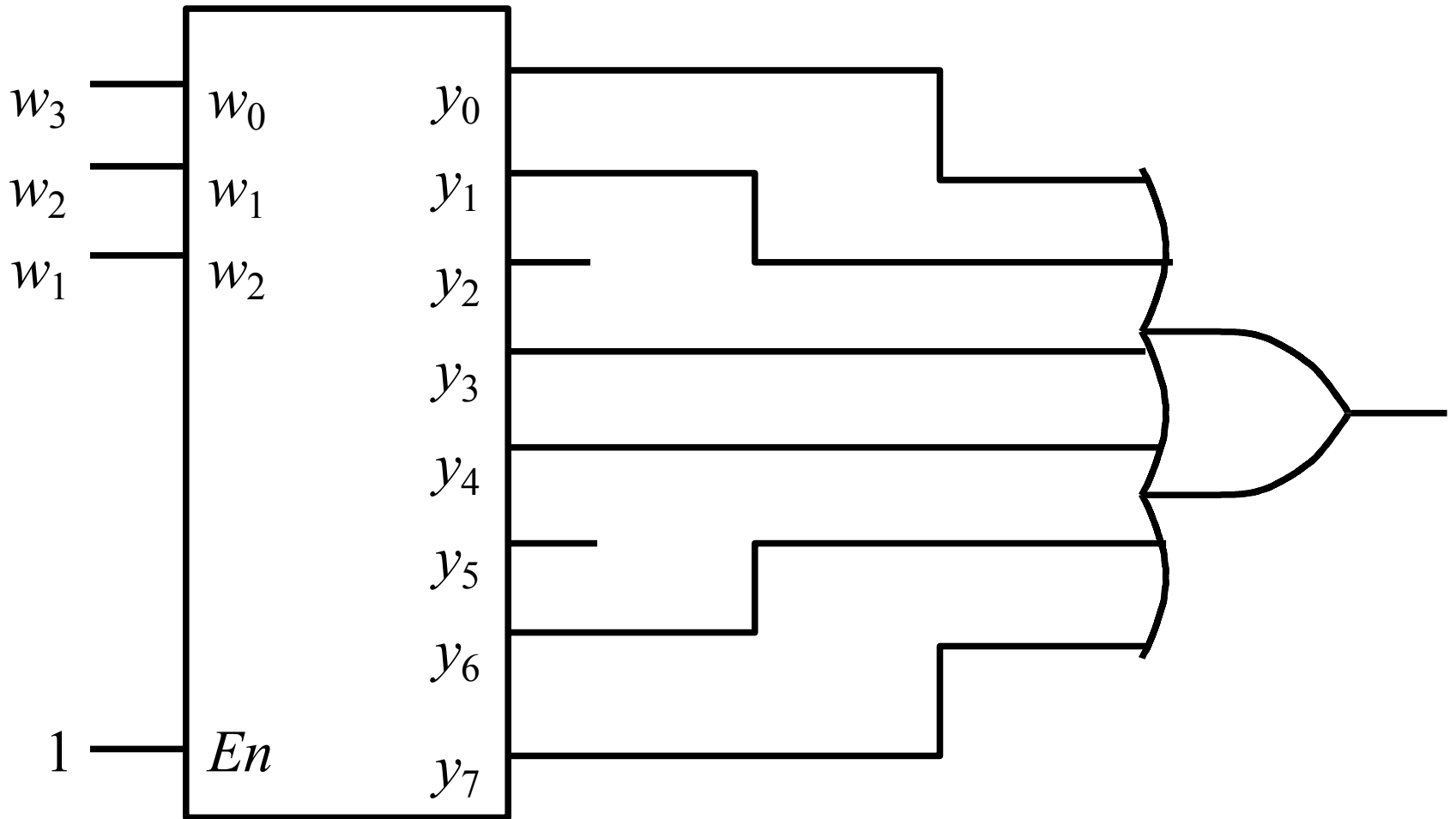
Example 1: SOP vs Decoders

Implement the function

$$f(w_1, w_2, w_3) = \Sigma m(0, 1, 3, 4, 6, 7)$$

by using a 3-to-8 binary decoder and one OR gate.

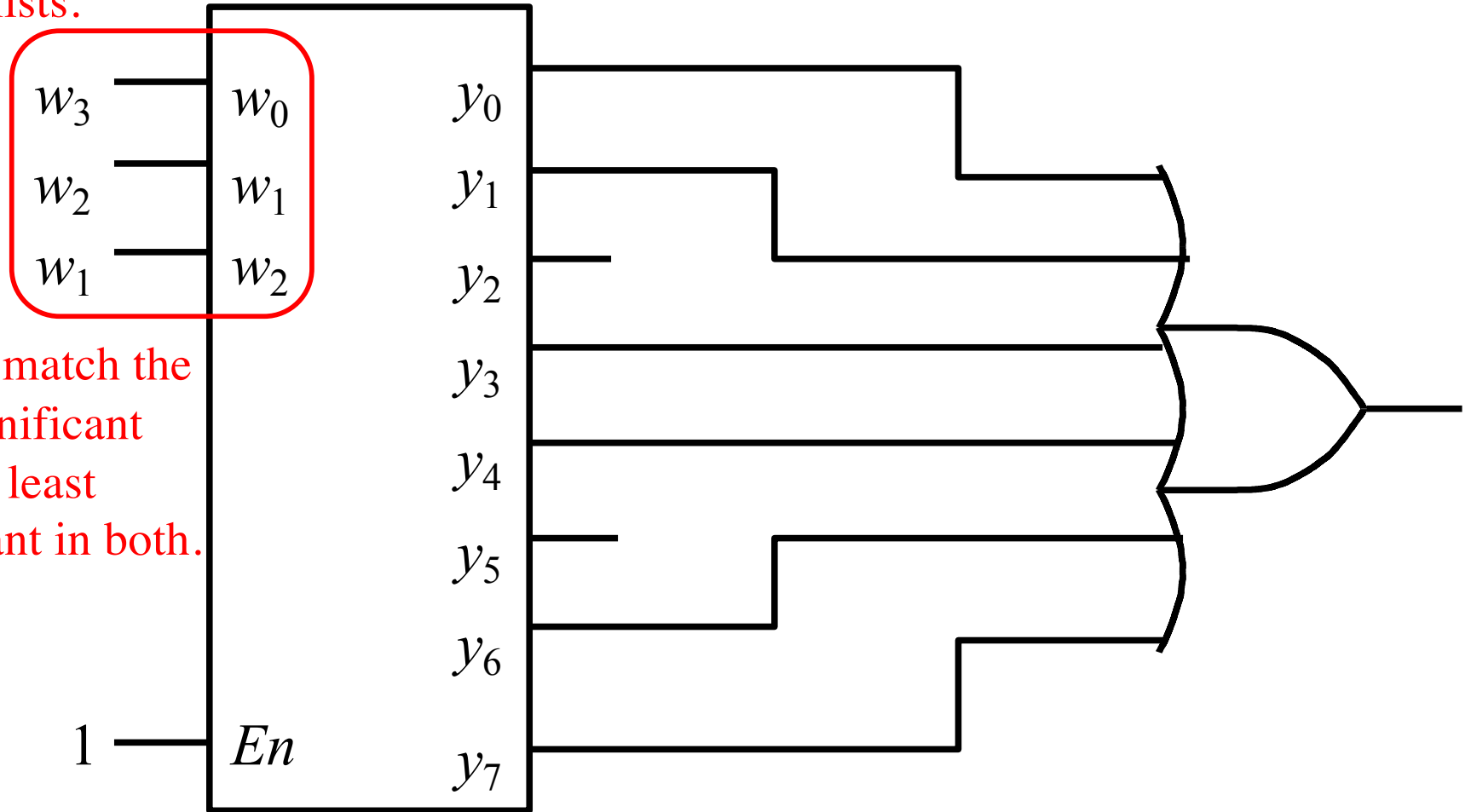
Solution Circuit



$$f(w_1, w_2, w_3) = \sum m(0, 1, 3, 4, 6, 7)$$

Solution Circuit

Notice this swap
of variables in
the two lists.



Need to match the
least significant
with the least
significant in both.

$$f(w_1, w_2, w_3) = \sum m(0, 1, 3, 4, 6, 7)$$

Example 2: Implement an 8-to-3 binary encoder

w_7	w_6	w_5	w_4	w_3	w_2	w_1	w_0	y_2	y_1	y_0
0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	1	0	0	0	1	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	0
0	0	1	0	0	0	0	0	1	0	1
0	1	0	0	0	0	0	0	1	1	0
1	0	0	0	0	0	0	0	1	1	1

Example 2: Implement an 8-to-3 binary encoder

w_7	w_6	w_5	w_4	w_3	w_2	w_1	w_0	y_2	y_1	y_0
0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	1	0	0	0	1	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	0
0	0	1	0	0	0	0	0	1	0	1
0	1	0	0	0	0	0	0	1	1	0
1	0	0	0	0	0	0	0	1	1	1

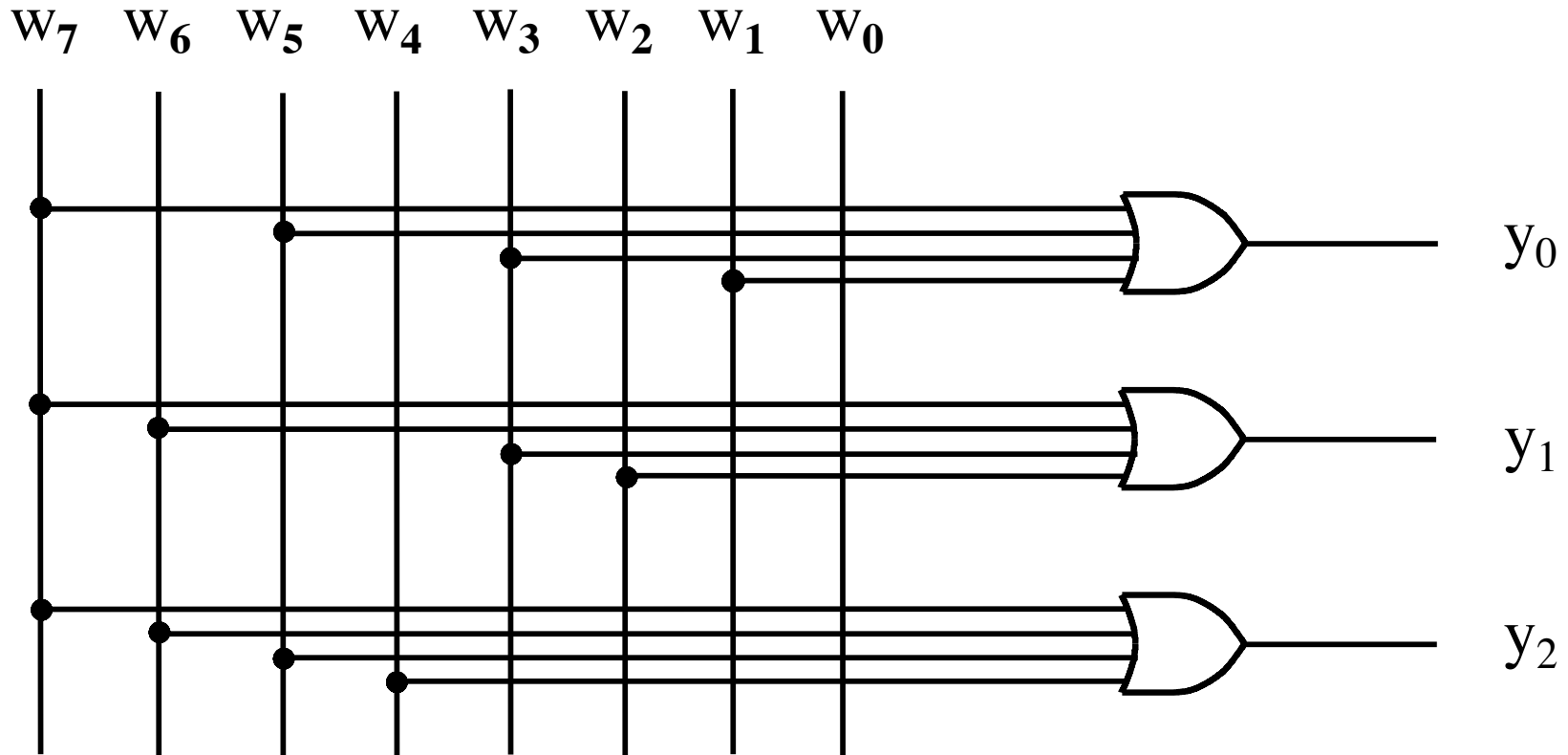
$$y_0 = w_1 + w_3 + w_5 + w_7$$

$$y_1 = w_2 + w_3 + w_6 + w_7$$

$$y_2 = w_4 + w_5 + w_6 + w_7$$

[Figure 4.45 from the textbook]

Circuit for the 8-to-3 binary encoder



$$y_0 = w_1 + w_3 + w_5 + w_7$$

$$y_1 = w_2 + w_3 + w_6 + w_7$$

$$y_2 = w_4 + w_5 + w_6 + w_7$$

Example 3: Implement an 8-to-3 priority encoder

w_7	w_6	w_5	w_4	w_3	w_2	w_1	w_0	y_2	y_1	y_0
0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	1	0	0	0	1	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	0
0	0	1	0	0	0	0	0	1	0	1
0	1	0	0	0	0	0	0	1	1	0
1	0	0	0	0	0	0	0	1	1	1

Example 3: Implement an 8-to-3 priority encoder

w_7	w_6	w_5	w_4	w_3	w_2	w_1	w_0	y_2	y_1	y_0
0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	X	0	0	1
0	0	0	0	0	1	X	X	0	1	0
0	0	0	0	1	X	X	X	0	1	1
0	0	0	1	X	X	X	X	1	0	0
0	0	1	X	X	X	X	X	1	0	1
0	1	X	X	X	X	X	X	1	1	0
1	X	X	X	X	X	X	X	1	1	1

Example 3: Implement an 8-to-3 priority encoder

w_7	w_6	w_5	w_4	w_3	w_2	w_1	w_0	y_2	y_1	y_0	z
0	0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	0	1	X	0	0	1	1
0	0	0	0	0	1	X	X	0	1	0	1
0	0	0	0	1	X	X	X	0	1	1	1
0	0	0	1	X	X	X	X	1	0	0	1
0	0	1	X	X	X	X	X	1	0	1	1
0	1	X	X	X	X	X	X	1	1	0	1
1	X	X	X	X	X	X	X	1	1	1	1
0	0	0	0	0	0	0	0	d	d	d	0

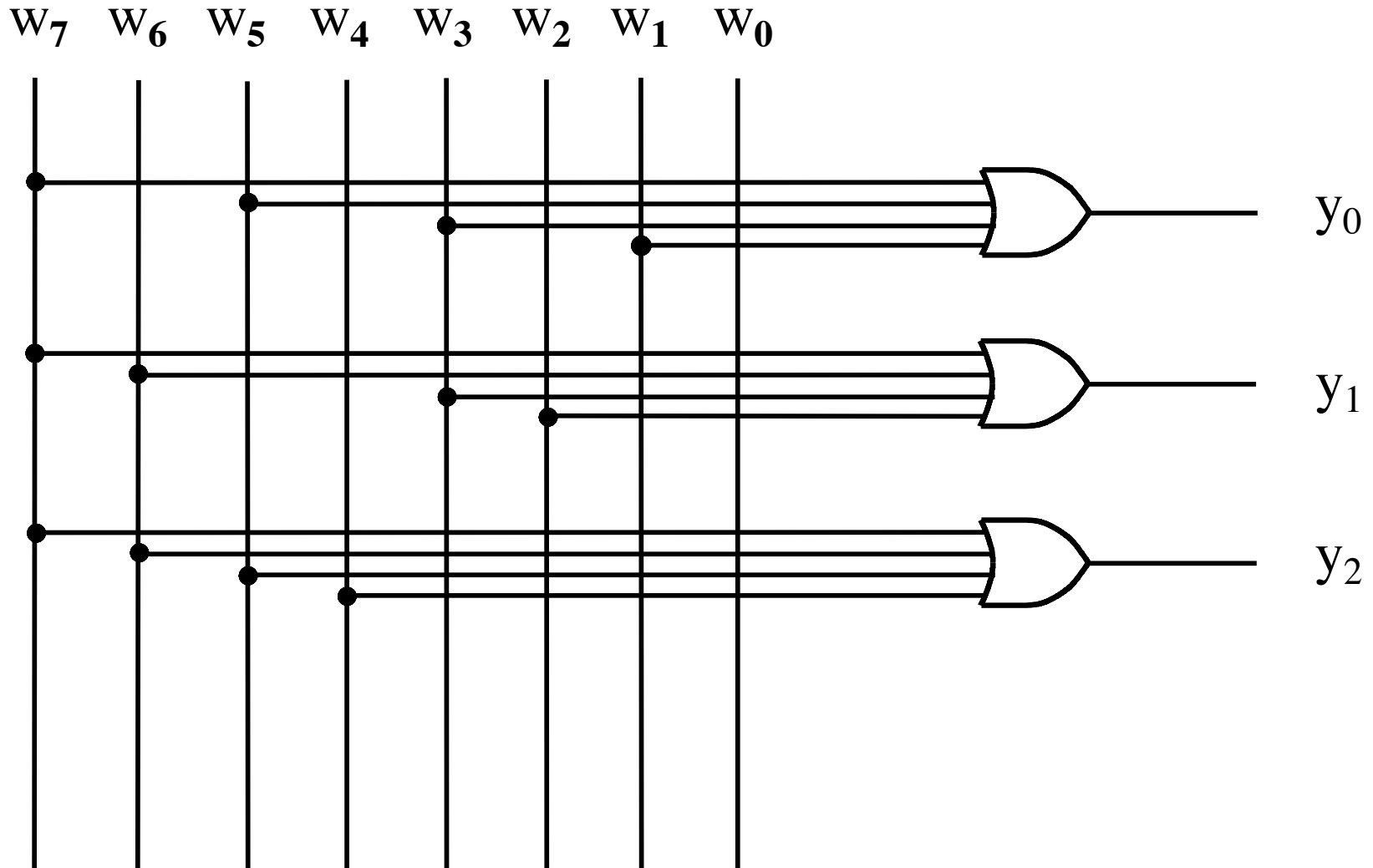
Example 3: Implement an 8-to-3 priority encoder

w_7	w_6	w_5	w_4	w_3	w_2	w_1	w_0	y_2	y_1	y_0	z
0	0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	0	1	X	0	0	1	1
0	0	0	0	0	1	X	X	0	1	0	1
0	0	0	0	1	X	X	X	0	1	1	1
0	0	0	1	X	X	X	X	1	0	0	1
0	0	1	X	X	X	X	X	1	0	1	1
0	1	X	X	X	X	X	X	1	1	0	1
1	X	X	X	X	X	X	X	1	1	1	1
0	0	0	0	0	0	0	0	d	d	d	0

Example 3: Implement an 8-to-3 priority encoder

	w_7	w_6	w_5	w_4	w_3	w_2	w_1	w_0	y_2	y_1	y_0	z
$i_0 = \overline{w_7} \overline{w_6} \overline{w_5} \overline{w_4} \overline{w_3} \overline{w_2} \overline{w_1} w_0$	0	0	0	0	0	0	0	1	0	0	0	1
$i_1 = \overline{w_7} \overline{w_6} \overline{w_5} \overline{w_4} \overline{w_3} \overline{w_2} w_1$	0	0	0	0	0	0	1	X	0	0	1	1
$i_2 = \overline{w_7} \overline{w_6} \overline{w_5} \overline{w_4} \overline{w_3} w_2$	0	0	0	0	0	1	X	X	0	1	0	1
$i_3 = \overline{w_7} \overline{w_6} \overline{w_5} \overline{w_4} w_3$	0	0	0	0	1	X	X	X	0	1	1	1
$i_4 = \overline{w_7} \overline{w_6} \overline{w_5} w_4$	0	0	0	1	X	X	X	X	1	0	0	1
$i_5 = \overline{w_7} \overline{w_6} w_5$	0	0	1	X	X	X	X	X	1	0	1	1
$i_6 = \overline{w_7} w_6$	0	1	X	X	X	X	X	X	1	1	0	1
$i_7 = w_7$	1	X	X	X	X	X	X	X	1	1	1	1
$z = i_0 + i_1 + i_2 + i_3 + i_4 + i_5 + i_6 + i_7$	0	0	0	0	0	0	0	0	d	d	d	0

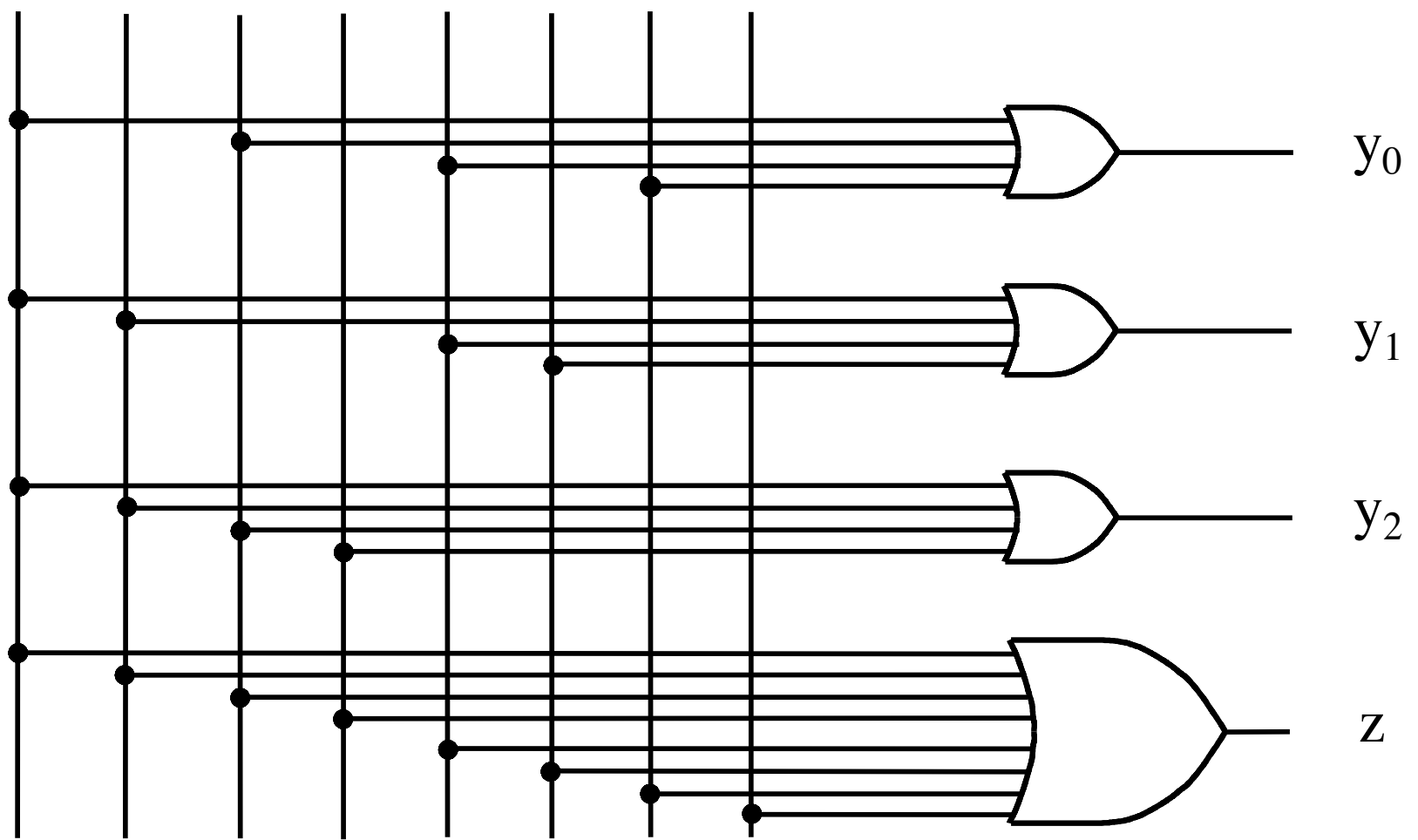
Circuit for the 8-to-3 binary encoder



Circuit for the 8-to-3 priority encoder

i_7 i_6 i_5 i_4 i_3 i_2 i_1 i_0

w_7 w_6 w_5 w_4 w_3 w_2 w_1 w_0



Example 4: Circuit implementation using a multiplexer

Implement the function

$$f(w_1, w_2, w_3, w_4, w_5) = \bar{w}_1 \bar{w}_2 \bar{w}_4 \bar{w}_5 + w_1 w_2 + w_1 w_3 + w_1 w_4 + w_3 w_4 w_5$$

using a 4-to-1 multiplexer

Some Boolean Algebra Leads To

$$\overline{w_1} \overline{w_2} \overline{w_4} \overline{w_5} + w_1 w_2 + w_1 w_3 + w_1 w_4 + w_3 w_4 w_5$$

$$\overline{w_1} \overline{w_4} (\overline{w_5} \overline{w_2}) + w_4 (w_3 w_5) + w_1 (w_2 + w_3) + w_1 w_4 (1)$$

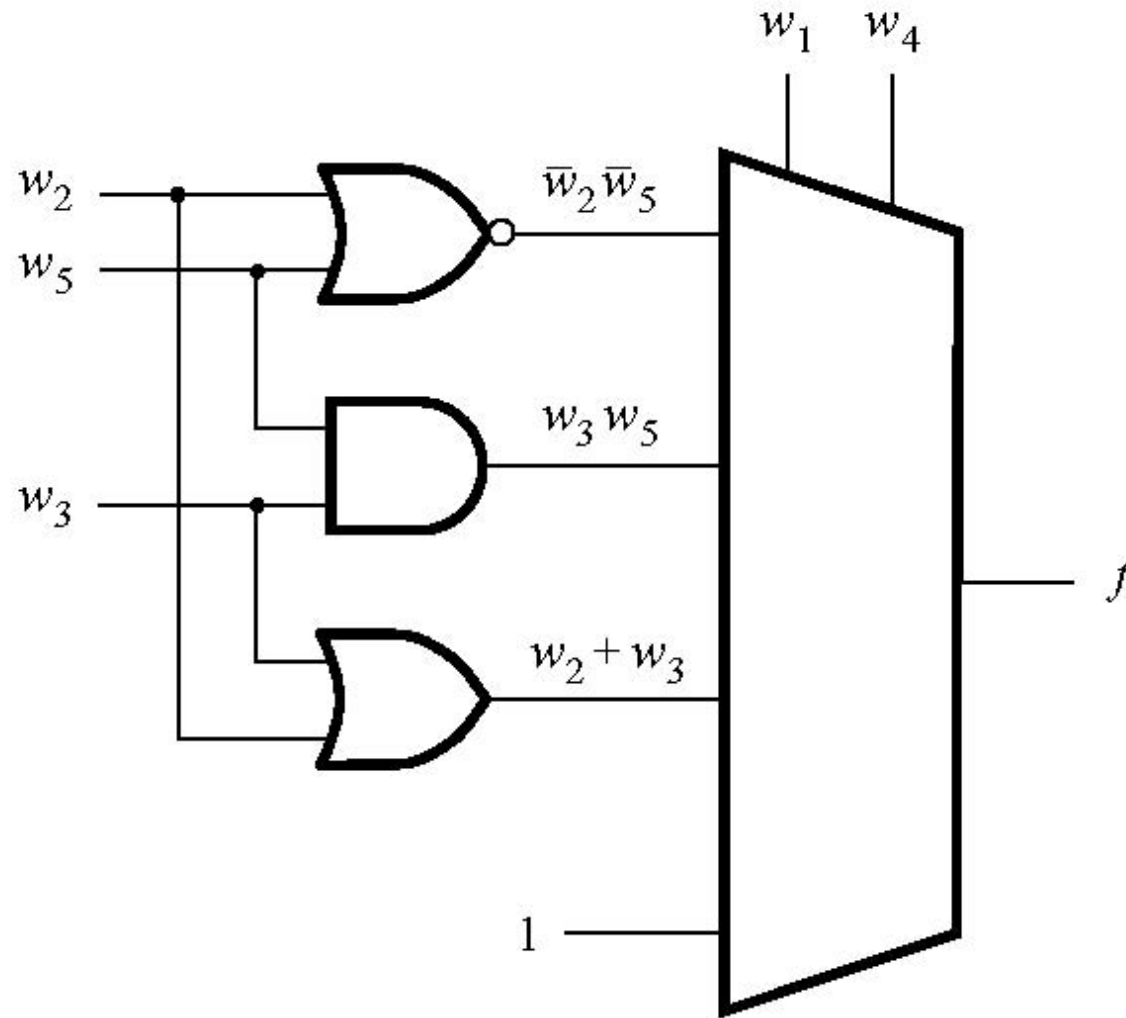
$$\overline{w_1} \overline{w_4} (\overline{w_5} \overline{w_2}) + (\overline{w_1} + w_1) w_4 (w_3 w_5) + w_1 (\overline{w_4} + w_4) (w_2 + w_3) + w_1 w_4 (1)$$

$$\overline{w_1} \overline{w_4} (\overline{w_5} \overline{w_2}) + \overline{w_1} w_4 (w_3 w_5) + w_1 \overline{w_4} (w_2 + w_3) + w_1 w_4 (w_3 w_5 + (w_2 + w_3) + 1)$$

$$\overline{w_1} \overline{w_4} (\overline{w_5} \overline{w_2}) + \overline{w_1} w_4 (w_3 w_5) + w_1 \overline{w_4} (w_2 + w_3) + w_1 w_4 (1)$$

Note that the split is by w_1 and w_4 , not w_1 and w_2

Solution Circuit

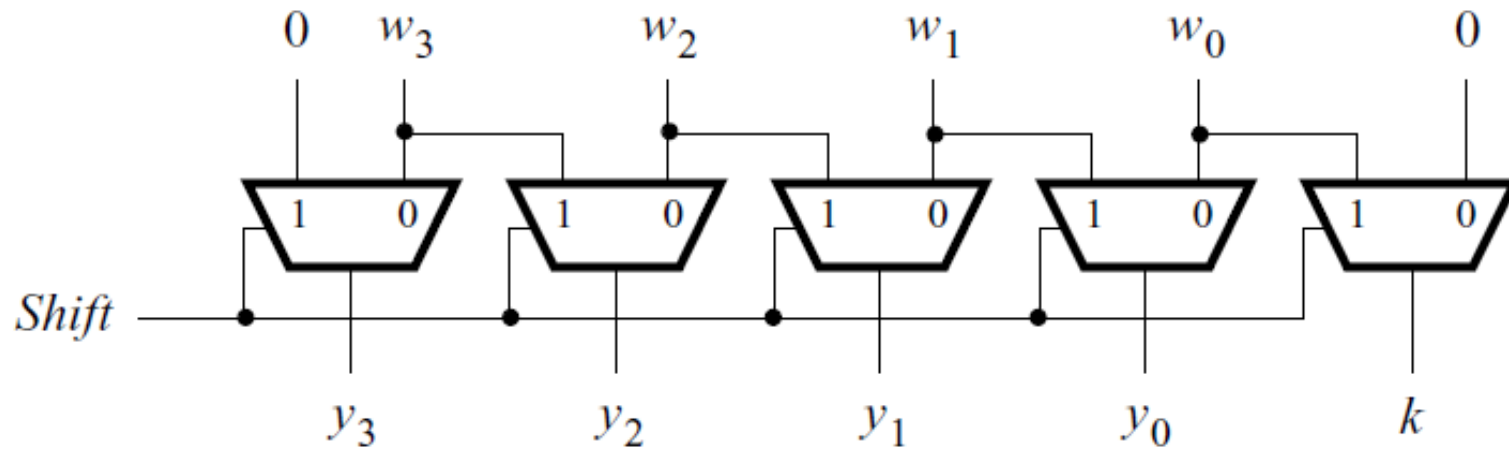


$$\bar{w}_1 \bar{w}_4 (\bar{w}_5 \bar{w}_2) + \bar{w}_1 w_4 (w_3 w_5) + w_1 \bar{w}_4 (w_2 + w_3) + w_1 w_4 (1)$$

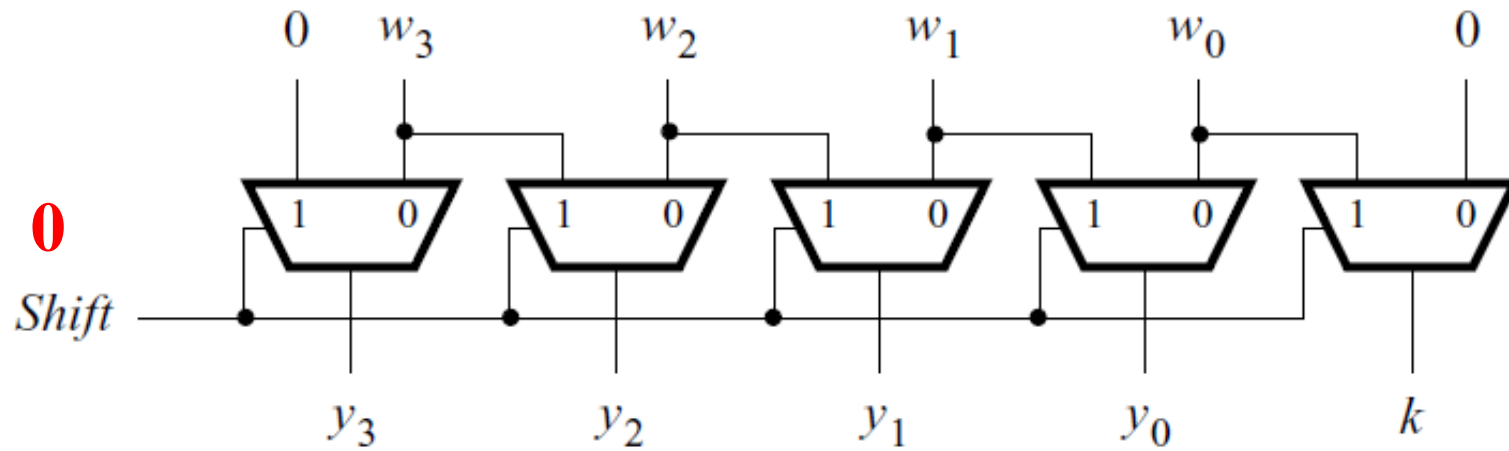
[Figure 4.46 from the textbook]

Some Final Things from Chapter 4

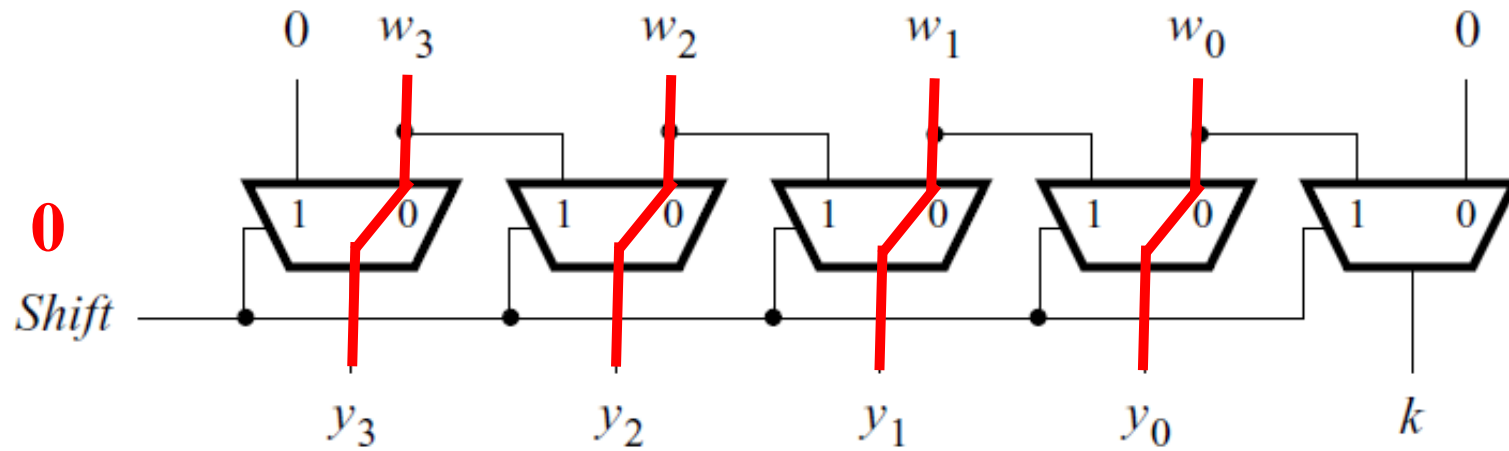
A shifter circuit



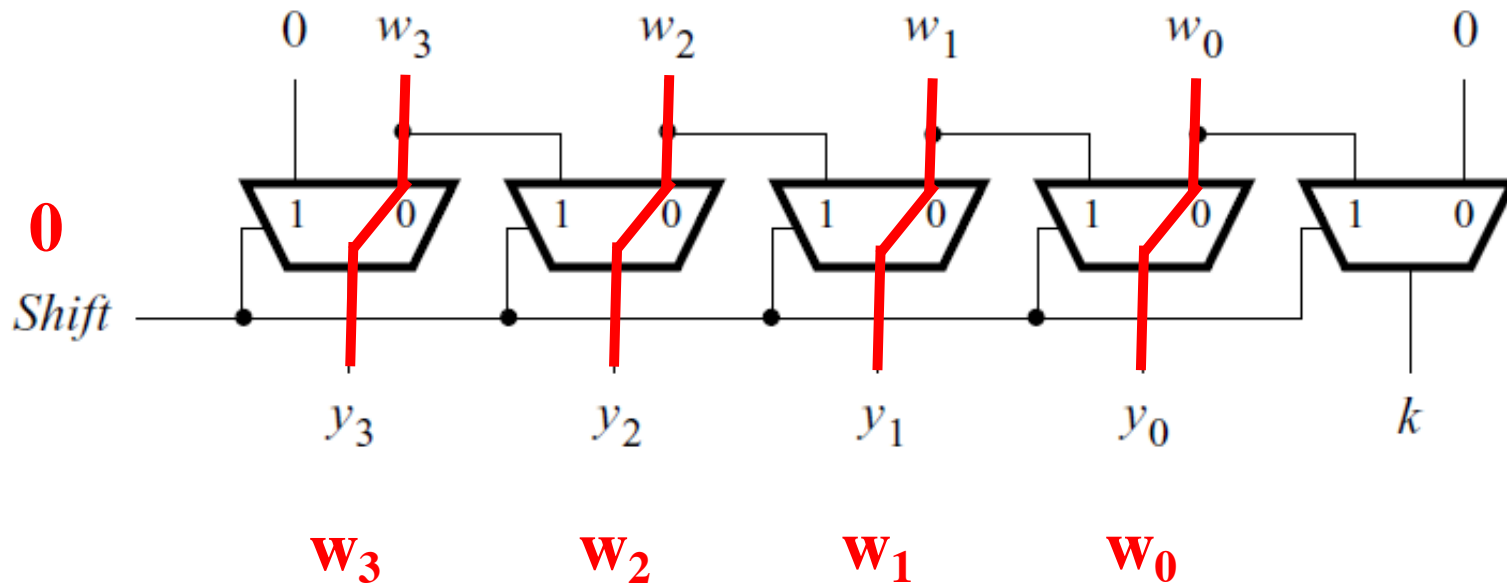
A shifter circuit



A shifter circuit

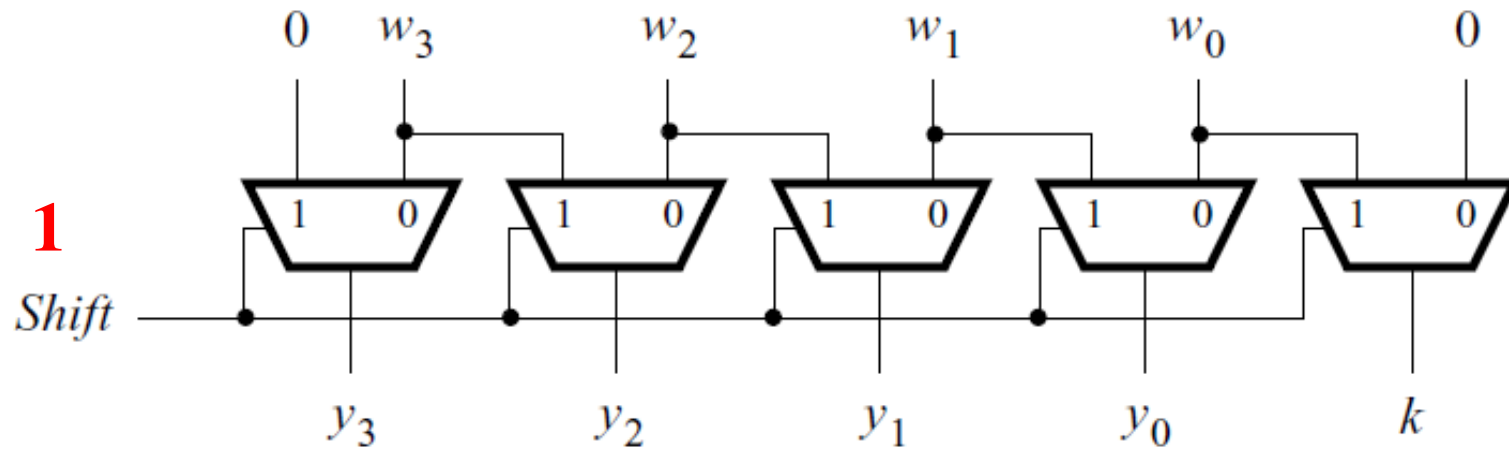


A shifter circuit

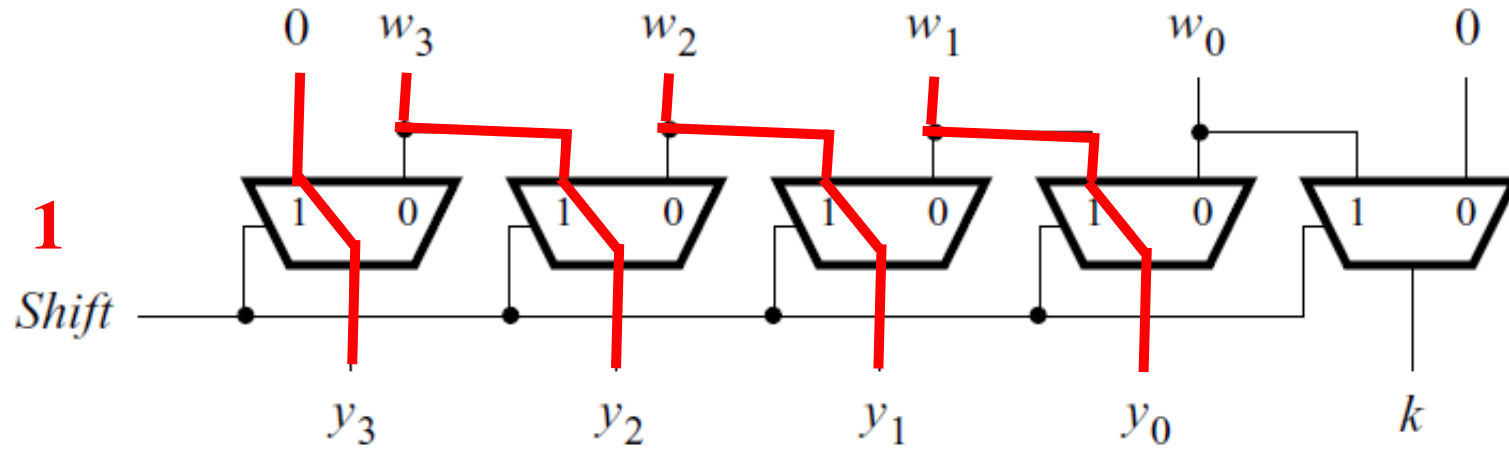


No shift in this case.

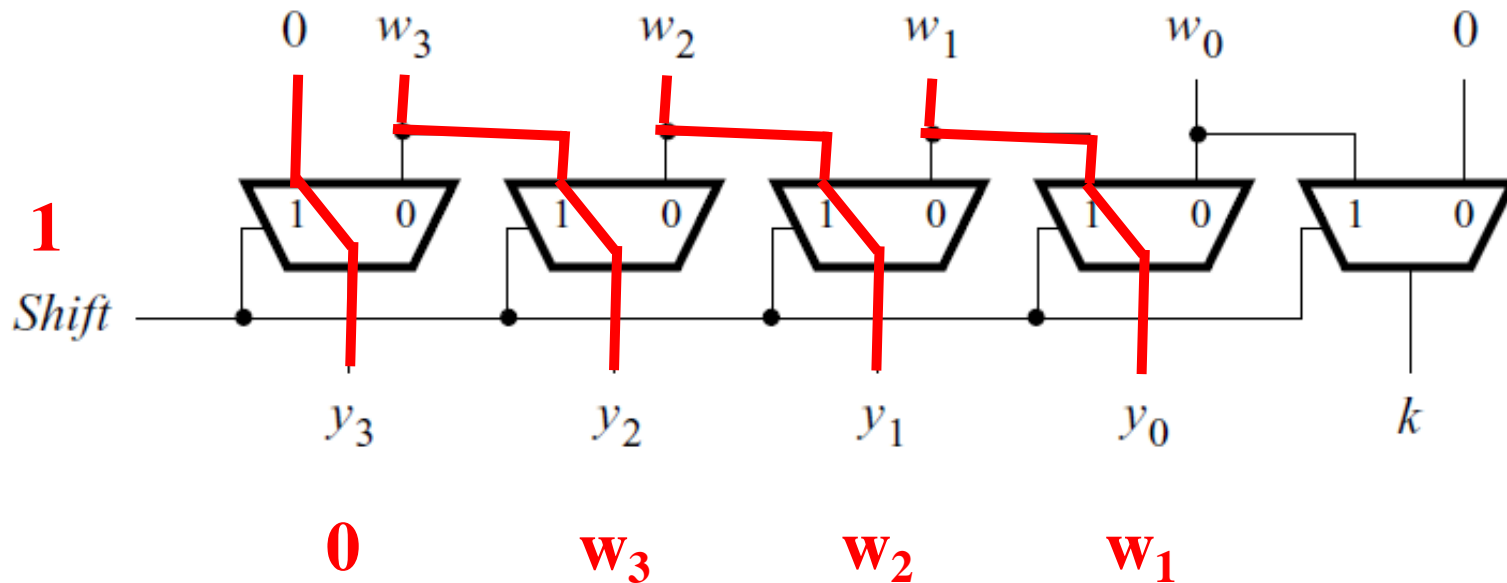
A shifter circuit



A shifter circuit



A shifter circuit

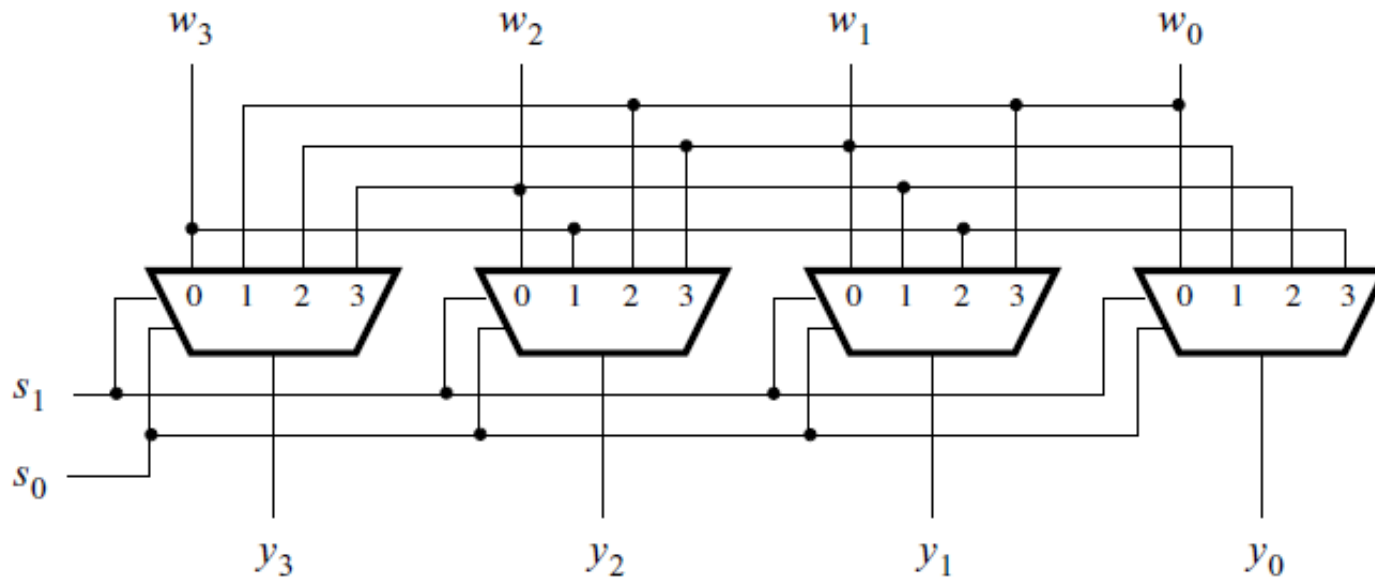


Shift to the right by 1 bit

A barrel shifter circuit

s_1	s_0	y_3	y_2	y_1	y_0
0	0	w_3	w_2	w_1	w_0
0	1	w_0	w_3	w_2	w_1
1	0	w_1	w_0	w_3	w_2
1	1	w_2	w_1	w_0	w_3

(a) Truth table



(b) Circuit

[Figure 4.51 from the textbook]

Questions?

THE END